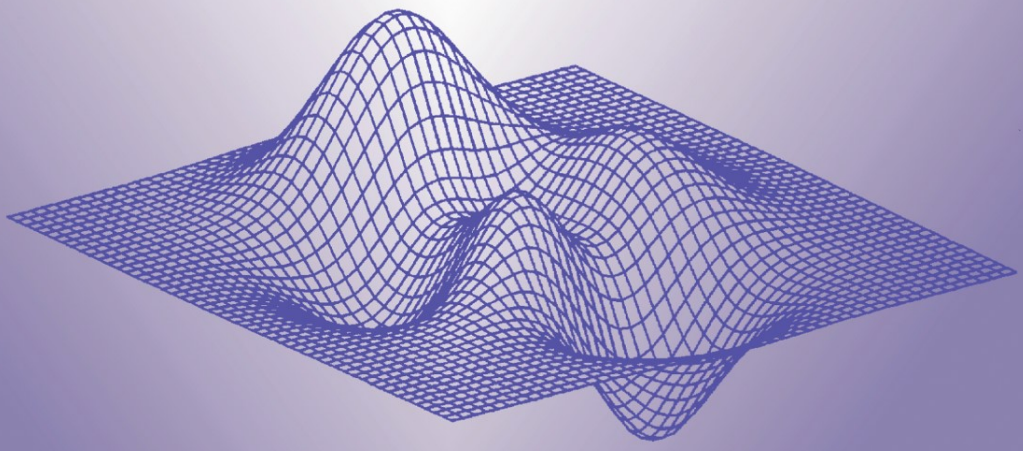
NONCONVEX OPTIMIZATION AND ITS APPLICATIONS

Large-Scale Nonlinear Optimization

Edited by Gianni Di Pillo and Massimo Roma



Springer

LARGE-SCALE NONLINEAR OPTIMIZATION

Nonconvex Optimization and Its Applications

VOLUME 83

LARGE-SCALE NONLINEAR OPTIMIZATION

Edited by

G. DI PILLO
University of Rome "La Sapienza," Italy

M. ROMA
University of Rome "La Sapienza," Italy

Library of Congress Control Number: 2005935078

ISBN-10: 0-387-30063-5 e-ISBN: 0-387-30065-1

ISBN-13: 978-0387-30063-4

Printed on acid-free paper.

AMS Subject Classifications: 90C30, 90C06, 65K05, 65K10

Printed in the United States of America.

9 8 7 6 5 4 3 2 1

springeronline.com

Contents

Preface

This volume contains refereed papers presented at the Workshop on *Large Scale Nonlinear Optimization* held in Erice, Italy, at the "G. Stampacchia" International School of Mathematics of the "E. Majorana" Centre for Scientific Culture, during June 22–July 1, 2004. The Workshop was the fourth in a series of Workshops on Nonlinear Optimization held in Erice; the three previous ones were held in 1995, 1998, 2001, respectively.

In the tradition of these meetings, the purpose of the Workshop was to review and discuss recent advances and promising research trends in the field of Nonlinear Optimization and its applications, with a main focus on the large dimensional case, currently at the forefront of the research effort.

The meeting was attended by 71 people from 18 different countries. Besides the lectures, several formal and informal discussions took place. The outcome was a wide and deep knowledge of the present research achievements and tendencies in the field. We wish to express our appreciation for the active contribution of all the participants in the meeting. By editing this volume we aim at enlarging the community of researchers, professionals and students who can benefit from this valuable knowledge.

The 16 papers included in this volume represent a significant selection of recent developments in Nonlinear Optimization theory and practice. They show that there are plenty of exciting ideas and new applications which give evidence of a fast evolution in the field. Moreover, they give an updated overview from different and complementary standpoints: theoretical analysis, algorithmic development, implementation issues, real world applications.

In particular, as concerns unconstrained optimization, the paper by Lukšan and Vlček is an up–to–date survey of efficient methods for large scale problems, while Powell gives an accurate description of the derivative–free NEWUOA software. Large space in the volume is devoted to constrained optimization, from both a theoretical and algorithmic point of view. In fact, the paper by Bertsekas deals with sensitivity properties of Lagrange multipliers. Dostál reviews recently proposed algorithms for large quadratic programming prob-

lems. Byrd et al., in their paper, give a detailed description of the KNITRO package for nonlinear programming. This package is designed for solving large scale problems; various algorithmic options, including two interior point methods and an active–set method, are considered. A new class of preconditioners is proposed in the paper by Dollar et al.: implicit–factorization constraint preconditioners are considered for the iterative solution of symmetric linear systems arising from saddle–point problems. Preconditioning in *one–step one–shot* design optimization is considered in the paper by Griewank: in particular, the problem of minimizing an objective function subject to a very large dimensional state equation with a separate design space is tackled, addressing the issue of selecting a suitable preconditioner for the approximate reduced gradient. The use of exact penalty functions for solving generalized Nash equilibrium problems is proposed by Facchinei and Pang; a broad algorithmic scheme for the solution of such problems is described too. In the paper by Mastroeni a variational model for traffic network problems is studied; in particular, the author considers a generalized minimum cost flow problem formulated by means of a variational inequality. The paper by Bérend et al. deals with the use of dedicated algebra solvers and an interior point algorithm for the efficient solution of the linear systems arising when solving an optimal control problem by a Runge–Kutta discretization scheme. Vector optimization is also treated: in the paper by Gutiérrez et al. approximate solutions of vector optimization problems are studied by means of a concept of approximate efficiency.

The paper by Evtushenko et al. deals with algorithms for constructing a family of parallel hyperplanes that separate two disjoint polyedra given by systems of linear inequalities; it is well known how the capability of finding such hyperplanes plays an important role in solving some practical problems. Another interesting problem is considered in the paper by Burdakov et al.: the problem, known as *isotonic regression*, has important applications in Operations Research and Statistics, and it is often characterized by large dimensionality. In order to solve such problems the authors introduce a new algorithm which exhibits both low computational complexity and high accuracy.

Finally, important real world applications are considered in the papers by Griesse and Volkwein, in the paper by Pesch et al. and in the paper by Peri et al. The first one considers boundary optimal control problems for a nonlinear reaction–diffusion equation system in three spatial dimensions. The second one presents a mathematical model for the dynamical behaviour of a molten carbonate fuel cell, which yields a large scale partial differential algebraic equation constrained optimization problem. The third one deals with optimum ship design problems and proposes the use of multi–objective formulations and global optimization strategies for their solution.

We are indebted to many anonymous referees who took care to review all the papers submitted for publication in this volume.

The Workshop was organized within the framework of the activities of the Research Project FIRB RBNE01WBBB on "*Large Scale Nonlinear Optimization*", funded by the Italian Ministry of Education, University and Research.

We are grateful to Professor Franco Giannessi, Director of the International School of Mathematics "G. Stampacchia", for promoting the Workshop and contributing valuable advice.

We are grateful to the "E. Majorana" Centre in Erice, which offered its facilities and rewarding environment: its staff was certainly instrumental for the success of the Workshop.

We are also grateful to the University of Rome "La Sapienza", to the University of Calabria, to the University of Pisa, to the Italian National Research Council through its "Research Center on Parallel Computing and Supercomputers", for their financial support.

Finally, we wish to thank Springer for publishing this volume.

August 2005

Gianni Di Pillo
Massimo Roma
Università di Roma "La Sapienza", Roma, Italy

Fast Linear Algebra for Multiarc Trajectory Optimization

Nicolas Bérend[1], J. Frédéric Bonnans[2], Julien Laurent-Varin[3], Mounir Haddou[4], and Christophe Talbot[5]

[1] Long-Term Design & System Integration Department, ONERA, BP 72, 29 av. division Leclerc, 92322 Châtillon, France (Nicolas.Berend@onera.fr)
[2] Projet Sydoco INRIA, B.P. 105, 78153 Le Chesnay, France (Frederic.Bonnans@inria.fr)
[3] Long-Term Design & System Integration Department, ONERA, and Projet Sydoco, INRIA, BP 105, 78153 Le Chesnay, France (Julien.Laurent-Varin@inria.fr)
[4] UMR 6628, MAPMO, BP 6759, 45067 Orléans Cedex 2, France (haddou@labomath.univ-orleans.fr)
[5] CNES, Launcher Directorate, Rond-Point de l'Espace, 91023 Evry Cedex, France (Christophe.Talbot@cnes.fr)

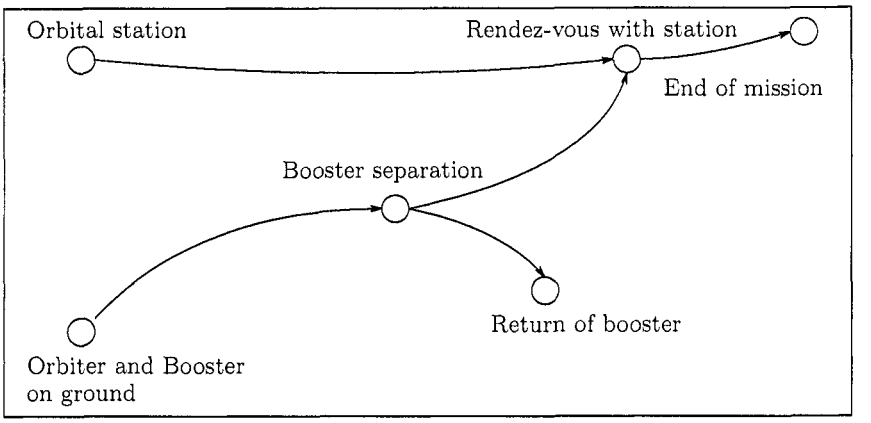
Summary. This paper presents some methods for solving in a fast and reliable way the linear systems arising when solving an optimal control problem by a Runge-Kutta discretization scheme, combined with an interior-point algorithm. Our analysis holds for a multiarc problem, i.e., when several arcs, each of them associated with a dynamics and integral cost, are linked by junction points, called nodes; with the latter are associated junction conditions and a cost function.

Our main result is that a sparse QR band factorization combined with a specific elimination procedure for arcs and nodes allows to factorize the Jacobian of the discrete optimality system in a small number of operations. Combined with an "optimal" refinement procedure, this gives an efficient method that we illustrate on Goddard's problem.

Key words: Optimal control, differential equations, Runge-Kutta and symplectic schemes, sparses algebra, band matrices, QR factorization.

1 Introduction

This paper discusses numerical methods for solving multi arc optimal control problems. Let us give an example of a possible mission to be optimized.



Here a separation between booster and orbiter occurs at some point; the orbiter meets a spatial station and both remain linked for some time. The return trajectory of the booster is to be optimized, as well as the one of the space station. While arcs are connected here through separation or rendez-vous, one may think also of other arcs corresponding to possible events, for instance of failure, that requires to change trajectories.

We concentrate on direct methods, which solve approximately a time discretized version of the optimal control problem, and are able to refine discretization. See [4,11,18,19] for an overview of numerical methods for optimal control problems, and [5,7] for specific direct methods.

The paper is organized as follows. Section 2 deals with single arc problems. We discuss the error analysis in subsection 2.1, and linear algebra issues in subsection 2.2. An "optimal" refinement technique is presented in 2.4, and 2.5 discusses the monitoring of interior-point algorithms. Section 3 is devoted to multiarc problems. Their structure is presented in subsection 3.1, and the linear algebra is analysed in subsection 3.2. Since our software is still under construction, we present only numerical results for the well known (single arc) Goddard problem in section 4.

2 Single arc problem

2.1 Framework and discretization

We start by describing an optimal control problem with a single arc and, to begin with, without constraints. We may assume without loss of generality (adding an additional state variable if necessary) that there is no integral cost, and we take into account only a final cost:

$$\text{Min } \Phi(y(T)); \quad \dot{y}(t) = f(y(t), u(t)), \quad t \in [0, T]; \quad y(0) = y^0. \tag{1}$$

Here f and Φ are C^∞ functions $\mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}^n$ and $\mathbb{R}^n \to \mathbb{R}$, respectively. Introduce now a Runge-Kutta type discretization, as done in Hager [14]:

$$\begin{cases} \text{Min } \Phi(y_N); \\ y_{k+1} = y_k + h_k \sum_{i=1}^{s} b_i f(y_{ki}, u_{ki}),\ k = 0, \ldots, N-1, \\ y_{ki} = y_k + h_k \sum_{j=1}^{s} a_{ij} f(y_{kj}, u_{kj}),\ i = 1, \ldots, s, \\ y_0 = y^0. \end{cases} \tag{2}$$

The positive time steps h_k are such that $\sum_{k=0}^{N-1} h_k = T$. Note that with each "inner state" y_{ki} is associated an inner control u_{ki}. The Runge-Kutta scheme is parameterized by its coefficients (a, b), an $s \times s$ matrix and a vector of $\mathbb{R}^n$ respectively. Assuming all coefficients b_i to be nonzero, it was shown by Hager [14] that the optimality system can be written in the following form:

$$\begin{cases} y_{k+1} = y_k + h_k \sum_{i=1}^{s} b_i f(y_{ki}, u_{ki}), & k = 0, \ldots, N-1, \\ y_{ki} = y_k + h_k \sum_{j=1}^{s} a_{ij} f(y_{kj}, u_{kj}), & i = 1, \ldots, s, \\ p_{k+1} = p_k - h_k \sum_{i=1}^{s} \hat{b}_i f_y(y_{ki}, u_{ki})^\top p_{ki}, & k = 0, \ldots, N-1, \\ p_{ki} = p_k - h_k \sum_{j=1}^{s} \hat{a}_{ij} f_y(y_{kj}, u_{kj})^\top p_{kj}, & i = 1, \ldots, s, \\ 0 = f_u(y_k, u_k)^\top p_k, & k = 0, \ldots, N-1, \\ 0 = f_u(y_{ki}, u_{ki})^\top p_{ki}, & i = 1, \ldots, s, \\ y_0 = y^0, \quad p_N = \Phi'(y_N), \end{cases} \tag{3}$$

where

$$\hat{b} := b \quad \text{and} \quad \hat{a}_{ij} := (b_i b_j - b_j a_{ij})/b_i, \quad \text{for all } i \text{ and } j. \tag{4}$$

The above system may be interpreted as a partitioned Kunge-Kutta discretization scheme for the optimality conditions of problem (1) stated below:

$$\begin{cases} \dot{y}(t) = f(y(t), u(t)), & t \in [0, T], \\ \dot{p}(t) = -f_y(y(t), u(t))^\top p(t), & t \in [0, T], \\ p(T) = \Phi'(y(T)), \quad y(0) = y^0, \\ 0 = f_u(y(t), u(t))^\top p(t), & t \in [0, T]. \end{cases} \tag{5}$$

If the Hamiltonian function $H(y, u, p) := p \cdot f(u, y)$ is, in the neighborhood of the optimal trajectory, a strongly convex function of u then we can eliminate the control from the above algebraic constraint (thanks to the implicit function theorem) so that (5) reduces to a two points boundary value problem. In view of (4), this system is symplectic [15–17]. These references present the theory of order conditions for partitioned Kunge-Kutta schemes. One may find in [8] a simplification of order conditions for symplectic schemes, that allows to state them for order up to 6 (they were obtained in [14] for order up to 4). Yet these results apply only for unconstrained problems with strongly convex Hamiltonians; some results for for constrained optimal control problems may be found in [12, 13].

2.2 Linear algebra: problems without design variables

Here we present a fast and reliable approach to the numerical resolution of the linear systems arising when solving the discrete optimality conditions (3), in the case when there is no design variable. The starting point is that the Jacobian matrix denoted A is, when variables are ordered by increasing values of time, a band matrix. Let q denote the band size, i.e., such that $A_{ij} = 0$ if $|i-j| > q$ (for instance, a diagonal matrix has band size 0). Remind that n (resp. m) denotes the number of states (resp. controls). Our implementation achieves a band size of $q = (s+1)(2n+m)+n$, where s is the number of inner stages in the Runke Kutta scheme. For a problem with n_g distributed constraints (upper and lower bounds), dealt with by the interior-point approach discussed later, we have a band size of $q = (s+1)(2n+m+2n_g)+n-1$. Note that at each time step there are $(s+1)(2n+m+2n_g)$ variables (state, costate, control and Lagrange multipliers) and there are dynamic constraints with the previous and next state and costate.

Lemma 1. *The number of multiplications for a QR factorization based on Givens rotations, applied to a matrix M of band size q_M, and having N_M lines, with N_M large w.r.t. q_M, is at most $6q_M^2 N_M + O(q_M^2)$ multiplications. The number of multiplications for solving the linear system (after factorization) is $6q_M N_M$.*

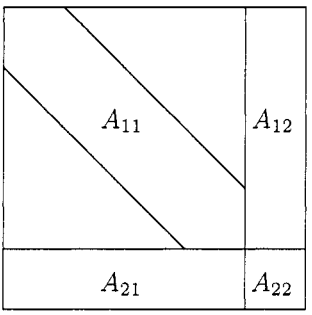
Proof. Row i eliminates element of index $(i+k, i)$ with k varying from 1 to q_M. This results in a fill-in from row i only. Therefore, when elimination of element $(i+k, i)$ occurs, row i of R has only $q+k$ elements, and hence, $4(q_M+k)+O(1)$ operations are performed. Since $\sum_{k=1}^{q_M}(q_M+k) = 3q_M^2/2 + O(q_M)$, the result follows.

Solving needs the resolutions of two linear systems with matrices R and Q. For the one with matrix R, we compute the components of the solution backwards (starting from the last one). Computing component i knowing all components of index $j > i$ needs $2q_M$ multiplications, except for the $2q_M$ last rows, that is a total of $2q_M N_M + O(q_M^2)$ multiplications. For the one with matrix Q, we have to apply the inverse $q_M N_M + O(q_M^2)$ Givens rotations to the right-hand-side, each of them needing 4 multiplications. The conclusion follows. ■

Remark 1. (i) Usually m and n_g are of the order of n, and in this case the cost of factorization (resp. resolution) of the Jacobian A is of order s^3n^3N (resp. s^2n^2N).
(ii) It would be interesting to have an idea of the least possible number of operations for an orthogonal factorization of a band matrix. Our procedure seems to be quite effective, but we do not know how far it is from this lower bound.
(iii) The ratio between factorization and solve is, under the assumptions of the lemma, essentially sn. Typical values of sn will be larger than 10. This means that factorization is by far the most expensive part of resolution.

2.3 Linear algebra: problems with design variables

We start our analysis with a general result on the factorization by elimination. Let A be a $n \times n$ invertible matrix, where $n = n_1 + n_2$, n_1 and n_2 being positive integers. Let A_{ij}, $i, j = 1, 2$, be its decomposition by blocks of size $n_i \times n_j$, as on the figure below (corresponding to the case when A_{11} is a band matrix):



Let (x, y) be solution of the following linear system:

$$A_{11}x + A_{12}y = b_1; \quad A_{21}x + A_{22}y = b_2. \tag{6}$$

Assuming A_{11} to be invertible, we may eliminate x from the first equation, and express (x, y) as

$$y = (A_{22} - A_{21}A_{11}^{-1}A_{12})^{-1}(b_2 - A_{21}A_{11}^{-1}b_1); \quad x = A_{11}^{-1}(b_1 - A_{12}y). \tag{7}$$

The *factorization* step consists in factorizing A_{11}, solving n_2 linear systems of type $A_{11}x = c$, where c is an arbitrary column of A_{12}, and computing the *reduced matrix* $A_R := (A_{22} - A_{21}A_{11}^{-1}A_{12})^{-1}$, and finally factorizing A_R. The *resolution step* needs the resolution of two linear systems with matrix A_{11}, and one with matrix A_R. We obtain the following complexity result:

Lemma 2. *The resolution of (6) based on reduction (7) requires* (i) *for factorization: the factorization of A_{11}, n_2 resolutions with matrix A_{11}, and the factorization of A_R, and* (ii) *for each resolution (after factorization): to solve two linear systems with matrix A_{11} and one with matrix A_R.*

We apply this result to the case of a single arc problem with n_s static parameters, the time dependant variables being eliminated first, using the number of operations estimated in lemma 1. Then static parameters are computed using a full matrix factorization. Lemma 2 implies the following result:

Corollary 1. *A single arc problem with n_s static parameters, using the above elimination procedure, and the QR factorization for time dependant variable, needs $O(n_s q^2 N + q^3 N) + O(n_s^3)$ operations for factorization, and $O(Nq^2 + n_s^2)$ for each resolution (after factorization).*

2.4 Mesh Refinement

For a Cauchy problem (integration of an ordinary differential equation with given intial value) the control of the size of time step is done once for all for each given time step. For a two points boundary value problem, the situation is different. Although the error estimate is of the same nature, one needs to solve a given discretization grid accurately enough before obtaining the local error estimates that allow to predict the needed additional grid points. Some refinement schemes are based on primal errors only, see [5,6]. We show here how to compute an "optimal" refinement for the state and costate equations.

The theory of error estimates tells us that the local error on step k is of the form $e_k = C_k h_k^{p+1}$, where the order p is, in principle, known. The error estimate may be obtained by comparing the local variation of the differential variable with the one computed by a scheme of higher order. Dividing the step h_k into q_k smaller steps of size h_k/q_k, where q_k is a positive integer, we reduce the error to $C_k(h_k/q_k)^{p+1}$ on each smaller step, i.e. a total of e_k/q_k^p on the q_k steps. This is valid, of course, provided the step is small enough. The problem of reaching a specific error estimate by adding the smallest number of new grid points can therefore be formulated as follows:

$$\underset{q\in\mathbb{N}^N}{\text{Min}} \sum_{k=1}^{N} q_k; \quad \sum_{k=1}^{N} \frac{e_k}{q_k^p} \leq E. \tag{INP}$$

This nonlinear integer programming problem appears to be easily solvable, by adding iteratively points at steps for which the local gain, obtain by incrementing q_k by one, is maximal. See algorithm 1 below, and [3] for details.

Algorithm 1 *Primal Algorithm*

For $k = 1, \ldots, N$ *do* $q_k := 1$. **End for**
While $\sum_{k=1}^{N} e_k/q_k^p > E$ *do*
  Compute $k_g \in \text{argmax}_k \{e_k\,(1/q_k^p - 1/(q_k+1)^p)\}$
  $q_{k_g} := q_{k_g} + 1$.
End While

2.5 Interior-point algorithms

Consider now a constrained optimal control problem of the following form:

$$\begin{cases} \text{Min} \displaystyle\int_0^T \ell(y(t),u(t))\mathrm{d}t + \Phi(y(T)); \\ \dot{y}(t) = f(y(t),u(t)), \quad t \in [0,T]; \\ 0 \leq g(y(t),u(t)), \quad t \in [0,T]; \\ y(0) = y^0. \end{cases} \tag{8}$$

Introducing a nonnegative slack variable for inequality constraints, a using a logarithmic penalty for the nonnegativity constraint on the slack variable,

and setting $\ell_\varepsilon(y,u) := \ell(y,u) - \varepsilon \sum_{i=1}^{q} g_i(y,u)$, where $\varepsilon > 0$, we obtain the following approximation:

$$\begin{cases} \text{Min} \displaystyle\int_0^T \ell_\varepsilon(y(t),u(t))\mathrm{d}t + \Phi(y(T)); \\ \dot{y}(t) = f(y(t),u(t)), \quad t \in [0,T], \\ y(0) = y^0. \end{cases} \tag{9}$$

The penalized problem (9) is unconstrained with an integral and final cost. Adding an additional state variable allows to reduce it to a problem with final cost only, that can be discretized by the Runge-Kutta schemes discussed in subsection 2. For a given value of ε the error analysis of that section applies when the steps vanish. The constants, however, will depend on ε.

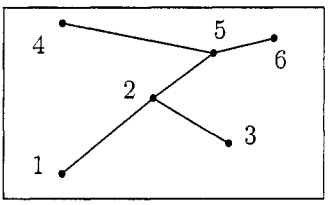
An interesting open problem is to obtain error estimates for given parameter ε and discretization steps. This is obviously difficult, since without logarithmic penalty, one may encounter a reduction order when state constraints are active (it is known that in general a Runge-Kutta scheme applied to an algebraic differential system suffers from order reduction). In addition junction points for constraints (times when the constraint begins or stops to be active) need a specific analysis.

Related to this is the question of the choice of a path of convergence, i.e., at which speed should the parameter ε and the discretization steps to 0. Again this is essentially an open problem. In our present implementation we simply require that the error estimate is not more than a constant times ε.

3 Multiarc problems

3.1 Framework

With a multiarc problem is associated a *scenario graph* whose edges are the arcs of the optimal control problem, i.e. time intervals with a given dynamics and integral cost, and vertices are junction points corresponding to starting or ending points, separation, or rendez-vous. For instance, with the mission example stated in the introduction is associated the following scenario graph:



Formally, the graph is a pair (V,E) where V is the finite set of vertices, and $E \subset V \times V$ is the set of edges. With edge $e = (i,j)$, where i and j are vertices, are associated the following variables and data:

$$
\begin{array}{ll}
y^e(t) \in I\!R^{n_e},\ u^e(t) \in I\!R^{m_e}: & \text{state and control,} \\
\pi^e \in I\!R^{n_{s,e}}: & \text{static optimization parameters, size } n_{s,e}, \\
\ell_e(t, y^e(t), u^e(t), \pi^e): & \text{distributed cost,} \\
f_e(t, y^e(t), u^e(t), \pi^e): & \text{state dynamics,} \\
g_e(t, y^e(t), u^e(t), \pi^e): & \text{distributed constraints.}
\end{array} \tag{10}
$$

Static optimization parameters are parameters involved at each time step, such as a mass of vehicle or the engine power. Since we reduce the possible variable time (spent on the arc) to a fixed unit time through the usual change of variables, the durations also appear as static parameters in our formulation.

With a vertex $j \in V$ are associated the following variables and data:

$$
\begin{array}{ll}
z^j: & \text{variables at vertex, size } n_j, \\
\Phi_i(z^j): & \text{cost at vertex,} \\
g_i(t_j, z^j): & \text{distributed constraints.}
\end{array} \tag{11}
$$

Variables z^j include the date t_i associated with the vertex, a copy of either initial or terminal states of arcs connected to this vertex, and as a copy of all static optimization parameters of these arcs. The multiarc problem may be formulated as follows:

$$
\left\{
\begin{array}{l}
\text{Min} \displaystyle\sum_{e=(i,j)\in E} \int_{t_i}^{t_j} \ell_e(t, y^e(t), u^e(t))\mathrm{d}t + \sum_{j\in V} \Phi_j(z^j)); \\
\dot{y}^e(t) = f_e(t, y^e(t), u^e(t), \pi^e), \quad t \in [t_i, t_j], \quad \text{for all } e = (i,j) \in E, \\
0 \le g_e(t, y^e(t), u^e(t), \pi^e), \quad \text{for all } e \in E, \\
0 \le g_i(t_j, z^j), \quad \text{for all } j \in V, \\
0 = h_j(t_j, (y^{(\cdot,j)}(t_j), \pi^{(\cdot,j)}), (y^{(j,\cdot)}(t_j), \pi^{(j,\cdot)}), z^j), \quad \text{for all } j \in V.
\end{array}
\right.
$$

The equality constraint at the nodes h_j includes the above mentioned copies of initial or terminal states and static parameters, as well as equality constraints on initial or terminal states, if any.

3.2 Linear algebra

Here is the main result of this section.

Theorem 2. *Let (V, E) be the scenario graph of the multiarc optimal control problem. Let N_e denote the number of time steps of arc e. Denote by s_e the number of inner steps in the Runge Kutta formula used for arc e, and by $n_{g,e}$ the number of distributed constraints on arc e. Let $n_{s,e}$ and n_v denote the number of static parameters on edge e, and the number of vertices variables on vertex v, respectively.*

Then the band size q_e on arc e satisfies $q_e = s_e O(n_e + m_e + n_{g,e})$, and a procedure (to be detailed below) allows to obtain a complexity for factorization and resolution of the Jacobian of discretized optimality system of order

$$\sum_{e\in E}\left[N_e(q_e^3+n_{s,e}q_e^2)+n_{s,e}^3\right]+\sum_{e\in E}(n_{s,e}^2+q_e^2N_e)(n_{\underline{e}}+n_{\overline{e}})+\sum_{v\in V}n_v^3. \quad (12)$$

Here $\underline{e}$ and $\overline{e}$ denote the starting and ending vertices, respectively, of edge e.

Remark 2. (i) The first sum in (12) represents the order of number of operations needed for the factorization of parts of the Jacobian corresponding to each arc. Therefore the total cost is proportional to the number of vertices and edges (weighted by coefficients depending on the number of associated variables) and the overall complexity (12) seems to be quite low.
(ii) For most real world optimal control problems, we may expect that the dominant term in (12) will be $\sum_{e\in E}q_e^3N_e$, itself of order $\sum_{e\in E}s_e^3n_e^3N_e$.

The elimination procedure is as follows.

Algorithm 3 *Elimination*

1) Factorize the parts of Jacobian corresponding to each arc
2) Eliminate all arc variables
3) Eliminate recursively all node variables, starting from leaves

Proof of theorem 2. That $q_e = s_eO(n_e+m_e+n_{g,e})$ results from the discussion before lemma 1. In view of that lemma, the number of multiplications for step 1 is of order $\sum_{e\in E}[N_e(q_e^3+n_{s,e}q_e^2)+n_{s,e}^3]$. Step 2 needs at most $(n_{\underline{e}}+n_{\overline{e}})$ resolutions of the four blocs matrix and each resolution need $(n_{s,e}^2+q_e^2N_e)$ operations. In step 3, elimination of leaf v connected to vertex w does not interplay with other vertices, and need, by lemma 2, the factorization of matrices of size n_i and n_j, plus solving n_j linear systems of size n_i, which means a number of operations of order $n_i^3+n_j^3+n_jn_i^2$. Since $n_jn_i^2 \leq \max(n_i^3,n_j^3)$ this is of order $n_i^3+n_j^3$. The third step being recursive, the conclusion follows. ■

Remark 3. Our implementation of the elimination procedure in C describes matrices as data structures plus a routines allowing factorization, so that sparse matrices of different types may be involved. We have presently three types of matrices: band, full and identity. Only the single arc minimization is implemented presently, but it uses already this general elimination procedure. What lacks for the multiarc problem is the construction of the optimality conditions and its Jacobian, starting from the description of subsection 3.1.

4 Application to Goddard's problem

Goddard's problem, stated in 1919 (see e.g. [20]) consists in maximizing the final altitude of a rocket with vertical ascent. The model involves three state variables: altitude r, speed v and mass m, and the thrust F as control variable.

The objective is to maximize the final altitude, the final time being free. The Dynamics is

$$\begin{cases} \dot{r}(t) = v(t), \\ \dot{v}(t) = (F(t) - D(v(t), r(t)))/m(t) - g(r(t)), \\ \dot{m}(t) = -F(t)/c \end{cases} \tag{13}$$

where D is the drag function, with arguments speed and altitude, $g(r)$ is the gravity field, and c is the speed of ejection of gas. The final mass is given, and we have a bound on thrust: $0 \leq F(t) \leq F_{\max}$.

Starting from a speed close to zero, it happens that the optimal thrust is not (as numerical experiments show) to set the thrust at $F_{\max}$ as long as the final mass is not attained. The reason is that aerodynamic forces would, in that case, tend to reduce significantly the speed. Therefore (although this is apparently not proved yet) the optimal law is to set the thrust at $F_{\max}$ for a certain time, then to set it to values in $(0, F_{\max})$ (this is the singular arc) and finally to set it to zero. The term singular arc comes from the fact that, since the control appears linearly in the dynamics (and not in the cost function) it cannot be computed by minimizing the Hamiltonian function. There exists a nice piece of theory that allows to obtain the control law during the singular arc (see [10,20]), but of course we do not use it in our numerical experiments.

Although our code may use arbitrary Runge-Kutta coefficients, we have used only the Gauss-Legendre of order 2, so that we may see how the code bevahes when there are many time steps. The results are synthetized in Table 1. Each major iteration corresponds to the resolution of the penalized problem for a given value of ε. We display the values of the major iteration It, the size N of the mesh, the number of points to be added (predicted by the refinement procedure), the current value of ε and the threshold E on error estimates. Observe the great accuracy of the refinement procedure, that essentially predicts in one shot the points to be added. The number of inner iterations remains small. It is fair to say, however, that our reduction procedure for ε is quite conservative (division by 2). Stronger reductions will be the subject of future research.

The optimal control and states are displayed on figures 1-4, for each value of the parameter ε. We observe the barrier effect for large ε, and the convergence to a three arcs structure, one of them being singular.

The density of mesh, after each major iteration is displayed in figure 5. The small numbers on the vertical axis are 0,4,8,10,14,16. The original mesh has 100 equal steps. We can observe on the final mesh a high density in the region corresponding to the singular arc.

5 Conclusion

The main features of our approach are the use of dedicated algebra solvers that exploit the band structure of sparse matrices, and the optimal refinement

It	N	n_{it}	Points	ε	E
1	100	13	+43	1.0e-3	5.0e-4
	143	3	+1		
	144	3	+0		
2	144	6	+63	5.0e-4	2.5e-4
	207	3	+1		
	208	3	+0		
3	208	6	+102	2.5e-4	1.25e-4
	310	3	+0		
4	310	6	+163	1.25e-4	6.25e-5
	473	3	+0		
5	473	5	+283	6.25e-5	3.125e-5
	756	3	+0		
6	756	5	+484	3.125e-5	1.5625e-5
	1240	3	+0		
7	1240	5	+862	1.5625e-5	7.8125e-6
	2102	2	+0		
8	2102	5	+1458	7.8125e-6	3.90625e-6
	3560	2	+0		
9	3560	5	+2663	3.90625e-6	1.95313e-6
	6223	2	+0		

Table 1: Results

scheme. The interior-point methodology avoids the need for large number of iterations as in sequential quadratic programming algorithms [5, 7], but it also gives flexibility for refinement that can be performed at any iteration. Of course shooting methods are always less expensive for a given precision, but they are not always stable and frequently need a priori knowledge of the sequence of active constraints.

We apply our methodology to reentry problems in [1] (that paper contains also a detailed analysis of the refinement problem). Numerical computations for multiarc problems will be presented in J. Laurent-Varin's Ph.D. Thesis.

Much remains to be done in several directions. (i) In the case of state constraints there is a reduction in error orders that should be taken into account in the analysis. (ii) We do not have any theory telling what should be the order of parameter ε for a given value of discretization errors (i.e., along which path these two parameters should go to zero). (iii) Convergence could be improved (especially for avoiding stationary points that are not local minima) by using step decomposition, see [9, Part III].

Finally heuristics for having good starting points are of course essential and are a research subject by themselves.

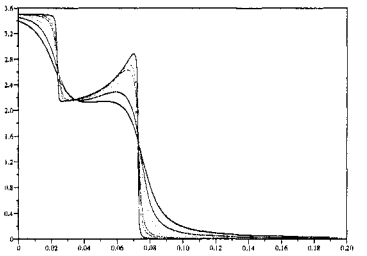


Fig. 1: Thrust law

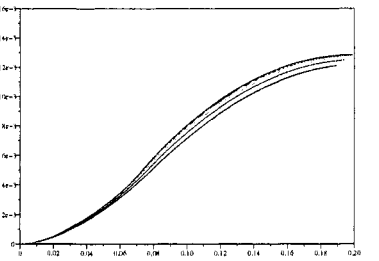


Fig. 2: Altitude

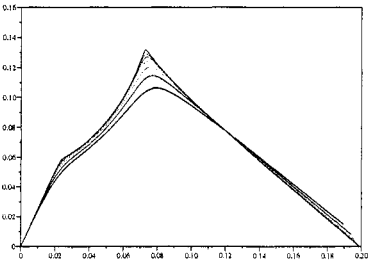


Fig. 3: Velocity

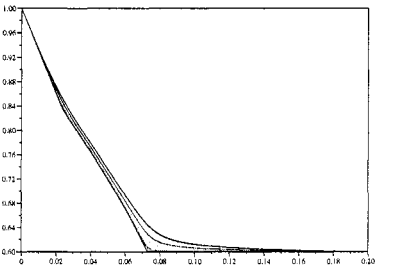


Fig. 4: Mass

References

1. Bérend, N., Bonnans, F., Haddou, M., Laurent-Varin, J., Talbot, C.: An Interior-Point Approach to Trajectory Optimization. INRIA Research Report RR-5613, www.inria.fr/rrrt/rr-5613.html (2005)
2. Bérend, N., Bonnans, F., Haddou, M., Laurent-Varin, J., Talbot, C.: A Preliminary Interior Point Algorithm For Solving Optimal Control Problems, 5^{th} International Conference on Launcher Technology. Madrid, Spain (2003)
3. Bérend, N., Bonnans, F., Haddou, M., Laurent-Varin, J., Talbot, C. On the refinement of discretization for optimal control problems. 16^{th} IFAC SYMPOSIUM Automatic Control in Aerospace. St. Petersburg, Russia (2004)
4. Betts, J.T.: Survey of Numerical Methods for Trajectory Optimization. AIAA J. of Guidance, Control and Dynamics, **21**, 193–207 (1998)
5. Betts, J.T.: Practical methods for optimal control using nonlinear programming. Society for Industrial and Applied Mathematics (SIAM), Philadelphia (2001)
6. Betts, J.T., Huffman, W.P.: Mesh refinement in direct transcription methods for optimal control. Optimal Control Applications & Methods, **19**, 1–21 (1998)
7. Bonnans, J.F., Launay, G.: Large Scale Direct Optimal Control Applied to a Re-Entry Problem. AIAA J. of Guidance, Control and Dynamics, **21**, 996–1000 (1998)

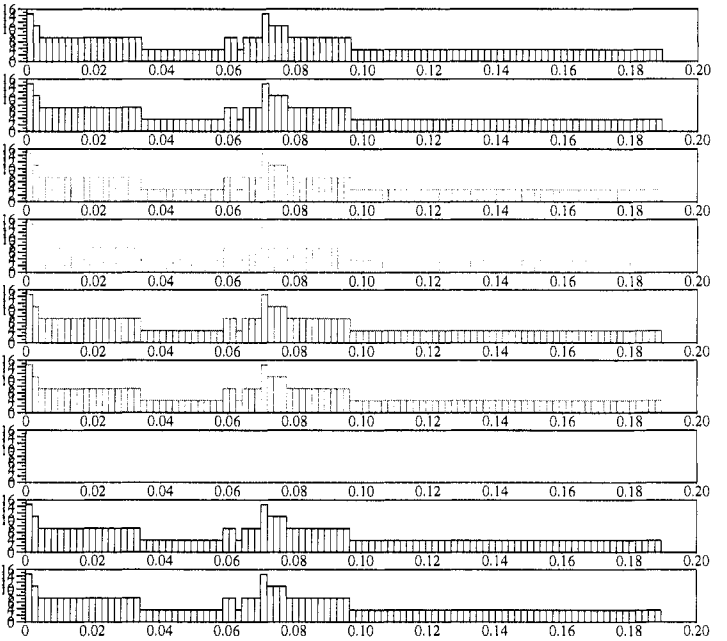


Fig. 5: Mesh

8. Bonnans, J.F., Laurent-Varin, J.: Computation of order conditions for symplectic partitioned Runge-Kutta schemes with application to optimal control. INRIA Research report RR-5398, www.inria.fr/rrrt/rr-5398.html (2004)
9. Bonnans, J.F., Gilbert, J.Ch., Lemaréchal, C., Sagastizábal, C.: Numerical Optimization: theoretical and numerical aspects. Springer-Verlag, Berlin (2003)
10. Bryson, A. E., Ho., Y.-C.: Applied optimal control. Hemisphere Publishing, New-York (1975)
11. Bulirsch, R., Nerz, E., Pesch, H. J., von Stryk, O.: Combining direct and indirect methods in optimal control: range maximization of a hang glider. In "Optimal control", Birkhäuser, Basel, 273–288 (1993)
12. Dontchev, A. L., Hager, W. W.: The Euler approximation in state constrained optimal control, Mathematics of Computation, **70**, 173–203 (2001)
13. Dontchev, A. L., Hager, W. W., Veliov, V. M.: Second-order Runge-Kutta approximations in control constrained optimal control. SIAM Journal on Numerical Analysis, **38**, 202–226 (2000)
14. Hager, W.: Runge-Kutta methods in optimal control and the transformed adjoint system. Numerische Mathematik, **87**, 247–282 (2000)
15. Hairer, E., Lubich, C., Wanner, G.: Geometric numerical integration. Springer-Verlag, Berlin (2002)

16. Hairer, E., Nørsett, S. P., Wanner, G.: Solving ordinary differential equations I. Springer-Verlag, Berlin (1993)
17. Hairer, E., Wanner, G.: Solving ordinary differential equations II. Springer-Verlag, Berlin (1996)
18. Pesch, H. J.: A practical guide to the solution of real-life optimal control problems. Control and Cybernetics, **23**, 7–60 (1994)
19. von Stryk, O., Bulirsch, R.: Direct and indirect methods for trajectory optimization. Annals of Operations Research, **37**, 357–373 (1992)
20. Tsiotras, P., Kelley, H.J.: Drag-law effects in the Goddard problem. Automatica, **27**, 481–490 (1991)

Lagrange Multipliers with Optimal Sensitivity Properties in Constrained Optimization

Dimitri P. Bertsekas[1]

Dept. of Electrical Engineering and Computer Science, M.I.T., Cambridge, Mass., 02139, USA. (dimitrib@mit.edu)

Summary. We consider optimization problems with inequality and abstract set constraints, and we derive sensitivity properties of Lagrange multipliers under very weak conditions. In particular, we do not assume uniqueness of a Lagrange multiplier or continuity of the perturbation function. We show that the Lagrange multiplier of minimum norm defines the optimal rate of improvement of the cost per unit constraint violation.

Key words: constrained optimization, Lagrange multipliers, sensitivity.

1 Introduction

We consider the constrained optimization problem

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in X, \quad g_j(x) \leq 0, \quad j = 1, \ldots, r, \end{array} \tag{P}$$

where X is a nonempty subset of $\Re^n$, and $f : \Re^n \to \Re$ and $g_j : \Re^n \to \Re$ are smooth (continuously differentiable) functions.

In our notation, all vectors are viewed as column vectors, and a prime denotes transposition, so $x'y$ denotes the inner product of the vectors x and y. We will use throughout the standard Euclidean norm $\|x\| = (x'x)^{1/2}$. The gradient vector of a smooth function $h : \Re^n \mapsto \Re$ at a vector x is denoted by $\nabla h(x)$. The positive part of the constraint function $g_j(x)$ is denoted by

$$g_j^+(x) = \max\bigl\{0, g_j(x)\bigr\},$$

and we write

$$g(x) = \bigl(g_1(x), \ldots, g_r(x)\bigr), \qquad g^+(x) = \bigl(g_1^+(x), \ldots, g_r^+(x)\bigr).$$

[1] Research supported by NSF Grant ECS-0218328.

The *tangent cone* of X at a vector $x \in X$ is denoted by $T_X(x)$. It is the set of vectors y such that either $y = 0$ or there exists a sequence $\{x^k\} \subset X$ such that $x^k \neq x$ for all k and

$$x^k \to x, \qquad \frac{x^k - x}{\|x^k - x\|} \to \frac{y}{\|y\|}.$$

An equivalent definition often found in the literature (e.g., Bazaraa, Sherali, and Shetty [BSS93], Rockafellar and Wets [RoW98]) is that $T_X(x)$ is the set of vectors y such that that there exists a sequence $\{x^k\} \subset X$ with $x^k \to x$, and a positive sequence $\{\alpha^k\}$ such that $\alpha^k \to 0$ and $(x^k - x)/\alpha^k \to y$. Note that $T_X(x)$ is a closed cone, but it need not be convex (it is convex if X is convex, or more generally, if X is regular at x in the terminology of nonsmooth optimization; see [BNO03] or [RoW78]). For any cone N, we denote by N^* its polar cone ($N^* = \{z \mid z'y \leq 0, \forall\, y \in N\}$). This paper is related to research on optimality conditions of the Fritz John type and associated subjects, described in the papers by Bertsekas and Ozdaglar [BeO02], Bertsekas, Ozdaglar, and Tseng [BOT04], and the book [BNO03]. We generally use the terminology of these works.

A *Lagrange multiplier* associated with a local minimum x^* is a vector $\mu = (\mu_1, \ldots, \mu_r)$ such that

$$\left(\nabla f(x^*) + \sum_{j=1}^{r} \mu_j \nabla g_j(x^*) \right)' d \geq 0, \qquad \forall\, d \in T_X(x^*), \tag{1.1}$$

$$\mu_j \geq 0, \quad \forall\, j = 1, \ldots, r, \qquad \mu_j = 0, \quad \forall\, j \notin A(x^*), \tag{1.2}$$

where $A(x^*) = \{j \mid g_j(x^*) = 0\}$ is the index set of inequality constraints that are active at x^*. The set of Lagrange multipliers corresponding to x^* is a (possibly empty) closed and convex set. Conditions for existence of at least one Lagrange multiplier are given in many sources, including the books [BSS93], [Ber99], and [BNO03], and the survey [Roc93].

We will show the following sensitivity result. The proof is given in the next section.

Proposition 1. *Let x^* be a local minimum of problem (P), assume that the set of Lagrange multipliers is nonempty, and let μ^* be the vector of minimum norm on this set. Then for every sequence $\{x^k\} \subset X$ of infeasible vectors such that $x^k \to x^*$, we have*

$$f(x^*) - f(x^k) \leq \|\mu^*\| \|g^+(x^k)\| + o\big(\|x^k - x^*\|\big). \tag{1.3}$$

Furthermore, if $\mu^* \neq 0$ and $T_X(x^*)$ is convex, the preceding inequality is sharp in the sense that there exists a sequence of infeasible vectors $\{x^k\} \subset X$ such that $x^k \to x^*$ and

$$\lim_{k\to\infty} \frac{f(x^*) - f(x^k)}{\|g^+(x^k)\|} = \|\mu^*\|. \tag{1.4}$$

For this sequence, we have

$$\lim_{k\to\infty} \frac{g_j^+(x^k)}{\|g^+(x^k)\|} = \frac{\mu_j^*}{\|\mu^*\|}, \qquad j = 1, \ldots, r. \tag{1.5}$$

A sensitivity result of this type was first given by Bertsekas, Ozdaglar, and Tseng [BOT04], for the case of a convex, possibly nondifferentiable problem. In that paper, X was assumed convex, and the functions f and g_j were assumed convex over X (rather than smooth). Using the definition of the dual function $[q(\mu) = \inf_{x\in X}\{f(x) + \mu' g(x)\}]$, it can be seen that

$$q^* - f(x) = q(\mu^*) - f(x) \le f(x) + \mu^{*\prime} g(x) - f(x) = \mu^{*\prime} g(x) \le \|\mu^*\|\, \|g^+(x)\|,$$

$\forall\, x \in X$, where q^* is the dual optimal value (assumed finite), and μ^* is the dual optimal solution of minimum norm (assuming a dual optimal solution exists). The inequality was shown to be sharp, assuming that $\mu^* \neq 0$, in the sense that there exists a sequence of infeasible vectors $\{x^k\} \subset X$ such that

$$\lim_{k\to\infty} \frac{q^* - f(x^k)}{\|g^+(x^k)\|} = \|\mu^*\|.$$

This result is consistent with Prop. 1. However, the line of analysis of the present paper is different, and in fact simpler, because it relies on the machinery of differentiable calculus rather than convex analysis (there is a connection with convex analysis, but it is embodied in Lemma 1, given in the next section).

Note that Prop. 1 establishes the optimal rate of cost improvement with respect to infeasible constraint perturbations, under much weaker assumptions than earlier results for nonconvex problems. For example, classical sensitivity results, include second order sufficiency assumptions guaranteeing that the Lagrange multiplier is unique and that the *perturbation function*

$$p(u) = \inf_{x\in X,\, g(x)\le u} f(x)$$

is differentiable (see e.g., [Ber99]). More recent analyses (see, e.g., Bonnans and Shapiro [BoS00], Section 5.2) also require considerably stronger conditions that ours.

Note also that under our weak assumptions, a sensitivity analysis based on the directional derivative of the perturbation function p is not appropriate. The reason is that our assumptions do not preclude the possibility that p has discontinuous directional derivative at $u = 0$, as illustrated by the following example, first discussed in [BOT04].

Example 1. Consider the two-dimensional problem,

$$\begin{aligned} &\text{minimize} \quad -x_2 \\ &\text{subject to} \quad x \in X = \{x \mid x_2^2 \leq x_1\}, \quad g_1(x) = x_1 \leq 0, \quad g_2(x) = x_2 \leq 0, \end{aligned}$$

we have

$$p(u) = \begin{cases} -u_2 & \text{if } u_2^2 \leq u_1, \\ -\sqrt{u_1} & \text{if } u_1 \leq u_2^2, u_1 \geq 0, u_2 \geq 0, \\ \infty & \text{otherwise.} \end{cases}$$

It can be verified that $x^* = 0$ is the global minimum (in fact the unique feasible solution) and that the set of Lagrange multipliers is

$$\{\mu \geq 0 \mid \mu_2 = 1\}.$$

Consistently with the preceding proposition, for the sequence $x^k = (1/k^2, 1/k)$, we have

$$\lim_{k \to \infty} \frac{f(x^*) - f(x^k)}{\|g^+(x^k)\|} = \|\mu^*\| = 1.$$

However, $\mu^* = (0, 1)$, is not a direction of steepest descent, since starting at $u = 0$ and going along the direction $(0, 1)$, $p(u)$ is equal to 0, so

$$p'(0; \mu^*) = 0.$$

In fact p has no direction of steepest descent at $u = 0$, because $p'(0; \cdot)$ is not continuous or even lower semicontinuous. However, one may achieve the optimal improvement rate of $\|\mu^*\|$ by using constraint perturbations that lie on the curved boundary of X.

Finally, let us illustrate with an example how our sensitivity result fails when the convexity assumption on $T_X(x^*)$ is violated. In this connection, it is worth noting that nonconvexity of $T_X(x^*)$ implies that X is not regular at x^* (in the terminology of nonsmooth analysis - see [BNO03] and [RoW78]), and this is a major source of exceptional behavior in relation to Lagrange multipliers (see [BNO03], Chapter 5).

Example 2. In this 2-dimensional example, there are two linear constraints

$$g_1(x) = x_1 + x_2 \leq 0, \qquad g_2(x) = -x_1 + x_2 \leq 0,$$

and the set X is the (nonconvex) cone

$$X = \{x \mid (x_1 + x_2)(x_1 - x_2) = 0\}$$

(see Fig. 1). Let the cost function be

$$f(x_1, x_2) = x_1^2 + (x_2 - 1)^2.$$

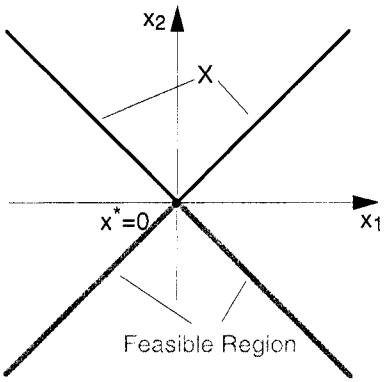


Fig. 1: Constraints of Example 2. We have $T_X(x^*) = X = \{x \mid (x_1+x_2(x_1-x_2) = 0\}$. The set X consists of the two lines shown, but the feasible region is the lower portion where $x_2 \leq 0$.

Then the vector $x^* = (0,0)$ is a local minimum, and we have $T_X(x^*) = X$, so $T_X(x^*)$ is not convex.

A Lagrange multiplier is a nonnegative vector (μ_1^*, μ_2^*) such that

$$\big(\nabla f(x^*) + \mu_1^* \nabla g_1(x^*) + \mu_2^* \nabla g_2(x^*)\big)' d \geq 0, \qquad \forall\, d \in T_X(x^*),$$

from which, since $T_X(x^*)$ contains the vectors $(1,0)$, $(-1,0)$, $(0,1)$, and $(0,-1)$, we obtain

$$\nabla f(x^*) + \mu_1^* \nabla g_1(x^*) + \mu_2^* \nabla g_2(x^*) = 0,$$

or

$$\begin{pmatrix} 0 \\ -2 \end{pmatrix} + \mu_1^* \begin{pmatrix} 1 \\ 1 \end{pmatrix} + \mu_2^* \begin{pmatrix} -1 \\ 1 \end{pmatrix} = 0.$$

Thus the unique Lagrange multiplier vector is $\mu^* = (1,1)$. There are two types of sequences $\{x_k\} \subset X$ (and mixtures of these two) that are infeasible and converge to x^*: those that approach x^* along the boundary of the constraint $x_1 + x_2 \leq 0$ [these have the form $(-\xi^k, \xi^k)$, where $\xi^k > 0$ and $\xi^k \to 0$], and those that approach x^* along the boundary of the constraint $-x_1 + x_2 \leq 0$ [these have the form (ξ^k, ξ^k), where $\xi^k > 0$ and $\xi^k \to 0$]. For any of these sequences, we have $f(x^k) = (\xi^k)^2 + (\xi^k - 1)^2$ and $\|g^+(x^k)\| = 2\xi^k$, so

$$\lim_{k\to\infty} \frac{f(x^*) - f(x^k)}{\|g^+(x^k)\|} = \lim_{k\to\infty} \frac{1 - (\xi^k)^2 - (\xi^k - 1)^2}{2\xi^k} = 1 < \sqrt{2} = \|\mu^*\|.$$

Thus $\|\mu^*\|$ is strictly larger than the optimal rate of cost improvement, and the conclusion of Prop. 1 fails.

2 Proof

Let $\{x^k\} \subset X$ be a sequence of infeasible vectors such that $x^k \to x^*$. We will show the bound (1.3). The sequence $\{(x^k - x^*)/\|x^k - x^*\|\}$ is bounded and each of its limit points belongs to $T_X(x^*)$. Without loss of generality, we assume that $\{(x^k - x^*)/\|x^k - x^*\|\}$ converges to a vector $d \in T_X(x^*)$. Then for the minimum norm Lagrange multiplier μ^*, we have

$$\left(\nabla f(x^*) + \sum_{j=1}^{r} \mu_j^* \nabla g_j(x^*)\right)' d \geq 0. \tag{2.1}$$

Denote

$$\xi^k = \frac{x^k - x^*}{\|x^k - x^*\|} - d.$$

We have

$$\left(\nabla f(x^*) + \sum_{j=1}^{r} \mu_j^* \nabla g_j(x^*)\right)' (x^k - x^*)$$

$$= \left(\nabla f(x^*) + \sum_{j=1}^{r} \mu_j^* \nabla g_j(x^*)\right)' (d + \xi^k)\,\|x^k - x^*\|,$$

so using Eq. (2.1) and the fact $\xi^k \to 0$, we have

$$\left(\nabla f(x^*) + \sum_{j=1}^{r} \mu_j^* \nabla g_j(x^*)\right)' (x^k - x^*) \geq o\big(\|x^k - x^*\|\big). \tag{2.2}$$

Using Eq. (2.2), a Taylor expansion, and the fact $\mu^{*\prime} g(x^*) = 0$, we have

$$\begin{aligned} f(x^k) + \mu^{*\prime} g(x^k) &= f(x^*) + \mu^{*\prime} g(x^*) + \\ &\quad + \left(\nabla f(x^*) + \sum_{j=1}^{r} \mu_j^* \nabla g_j(x^*)\right)' (x^k - x^*) + o\big(\|x^k - x^*\|\big) \\ &\geq f(x^*) + o\big(\|x^k - x^*\|\big). \end{aligned}$$

We thus obtain, using the fact $\mu \geq 0$,

$$f(x^*) - f(x^k) \leq \mu^{*\prime} g(x^k) + o\big(\|x^k - x^*\|\big) \leq \mu^{*\prime} g^+(x^k) + o\big(\|x^k - x^*\|\big),$$

and using the Cauchy-Schwarz inequality,

$$f(x^*) - f(x^k) \leq \|\mu^*\|\,\|g^+(x^k)\| + o\big(\|x^k - x^*\|\big),$$

which is the desired bound (1.3).

For the proof that the bound is sharp, we will need the following lemma first given in Bertsekas and Ozdaglar [BeO02] (see also [BNO03], Lemma 5.3.1).

Lemma 1. *Let N be a closed convex cone in $\Re^n$, and let $a_0, \ldots, a_r$ be given vectors in $\Re^n$. Suppose that the set*

$$M = \left\{ \mu \geq 0 \;\middle|\; -\left(a_0 + \sum_{j=1}^{r} \mu_j a_j \right) \in N \right\}$$

is nonempty, and let μ^* be the vector of minimum norm in M. Then, there exists a sequence $\{d^k\} \subset N^*$ such that

$$a_0' d^k \to -\|\mu^*\|^2, \qquad (a_j' d^k)^+ \to \mu_j^*, \quad j = 1, \ldots, r. \tag{2.3}$$

For simplicity, we assume that all the constraints are active at x^*. Inactive inequality constraints can be neglected since the subsequent analysis focuses in a small neighborhood of x^*, within which these constraints remain inactive. We will use Lemma 1 with the following identifications:

$$N = T_X(x^*)^*, \qquad a_0 = \nabla f(x^*), \qquad a_j = \nabla g_j(x^*), \quad j = 1, \ldots, r,$$

$$M = \text{ set of Lagrange multipliers},$$

$$\mu^* = \text{ Lagrange multiplier of minimum norm}.$$

Since $T_X(x^*)$ is closed and is assumed convex, we have $N^* = T_X(x^*)$, so Lemma 1 yields a sequence $\{d^k\} \subset T_X(x^*)$ such that

$$\nabla f(x^*)' d^k \to -\|\mu^*\|^2, \qquad \nabla g_j(x^*)' d^k \to \mu_j^*, \quad j = 1, \ldots, r.$$

Since $d^k \in T_X(x^*)$, for each k we can select a sequence $\{x^{k,t}\} \subset X$ such that $x^{k,t} \neq x^*$ for all t and

$$\lim_{t \to \infty} x^{k,t} = x^*, \qquad \lim_{t \to \infty} \frac{x^{k,t} - x^*}{\|x^{k,t} - x^*\|} = d^k. \tag{2.4}$$

Denote

$$\xi^{k,t} = \frac{x^{k,t} - x^*}{\|x^{k,t} - x^*\|} - d^k.$$

For each k, we select t_k sufficiently large so that

$$\lim_{k \to \infty} \xi^{k,t_k} = 0, \qquad \lim_{k \to \infty} x^{k,t_k} = x^*,$$

and we denote

$$x^k = x^{k,t_k}, \qquad \xi^k = \xi^{k,t_k}.$$

Thus, we have

$$x^k \to x^*, \qquad \xi^k = \frac{x^k - x^*}{\|x^k - x^*\|} - d^k \to 0.$$

Using a first order expansion for the cost function f, we have for each k and t,

$$\begin{aligned} f(x^k) - f(x^*) &= \nabla f(x^*)'(x^k - x^*) + o(\|x^k - x^*\|) \\ &= \nabla f(x^*)' \left(d^k + \xi^k\right) \|x^k - x^*\| + o(\|x^k - x^*\|) \\ &= \|x^k - x^*\| \left(\nabla f(x^*)' d^k + \nabla f(x^*)' \xi^k + \frac{o(\|x^k - x^*\|)}{\|x^k - x^*\|}\right), \end{aligned}$$

and, since $\xi^k \to 0$ and $\nabla f(x^*)' d^k \to -\|\mu^*\|^2$,

$$f(x^k) - f(x^*) = -\|x^k - x^*\| \cdot \|\mu^*\|^2 + o(\|x^k - x^*\|). \tag{2.5}$$

Similarly, using also the fact $g_j(x^*) = 0$, we have for each k and t,

$$g_j(x^k) = \|x^k - x^*\| \mu_j^* + o(\|x^k - x^*\|), \qquad j = 1, \ldots, r,$$

from which we also have

$$g_j^+(x^k) = \|x^k - x^*\| \mu_j^* + o(\|x^k - x^*\|), \qquad j = 1, \ldots, r. \tag{2.6}$$

We thus obtain

$$\|g^+(x^k)\| = \|x^k - x^*\| \cdot \|\mu^*\| + o(\|x^k - x^*\|), \tag{2.7}$$

which, since $\|\mu^*\| \neq 0$, also shows that $\|g^+(x^k)\| \neq 0$ for all sufficiently large k. Without loss of generality, we assume that

$$\|g^+(x^k)\| \neq 0, \qquad k = 0, 1, \ldots \tag{2.8}$$

By multiplying Eq. (2.7) with $\|\mu^*\|$, we see that

$$\|\mu^*\| \cdot \|g^+(x^k)\| = \|x^k - x^*\| \|\mu^*\|^2 + o(\|x^k - x^*\|). \tag{2.9}$$

Combining Eqs. (2.5) and (2.9), we obtain

$$f(x^*) - f(x^k) = \|\mu^*\| \cdot \|g^+(x^k)\| + o(\|x^k - x^*\|),$$

which together with Eqs. (2.7) and (2.8), shows that

$$\frac{f(x^*) - f(x^k)}{\|g^+(x^k)\|} = \|\mu^*\| + \frac{o(\|x^k - x^*\|)}{\|x^k - x^*\| \cdot \|\mu^*\| + o(\|x^k - x^*\|)}.$$

Taking the limit as $k \to \infty$ and using the fact $\|\mu^*\| \neq 0$, we obtain

$$\lim_{k \to \infty} \frac{f(x^*) - f(x^k)}{\|g^+(x^k)\|} = \|\mu^*\|.$$

Finally, from Eqs. (2.6) and (2.7), we see that

$$\frac{g_j^+(x^k)}{\|g^+(x^k)\|} = \frac{\mu_j^*}{\|\mu^*\|} + \frac{o(\|x^k - x^*\|)}{\|x^k - x^*\|}, \qquad j = 1, \ldots, r,$$

from which Eq. (1.5) follows. □

References

[BSS93] Bazaraa, M. S., Sherali, H. D., and Shetty, C. M., Nonlinear Programming Theory and Algorithms, (2nd Ed.), Wiley, N. Y (1993).

[BNO03] Bertsekas, D. P., with Nedić, A., and Ozdaglar, A. E., Convex Analysis and Optimization, Athena Scientific, Belmont, MA (2003).

[BOT04] Bertsekas, D. P., Ozdaglar, A. E., and Tseng, P., Enhanced Fritz John Conditions for Convex Programming, Lab. for Information and Decision Systems Report 2631, MIT (2004). To appear in SIAM J. on Optimization.

[BeO02] Bertsekas, D. P., and Ozdaglar, A. E. Pseudonormality and a Lagrange Multiplier Theory for Constrained Optimization, J. Opt. Theory Appl., Vol. 114, pp. 287-343 (2002)

[Ber99] Bertsekas, D. P., Nonlinear Programming: 2nd Edition, Athena Scientific, Belmont, MA (1999).

[BoS00] Bonnans, J. F., and Shapiro, A., Perturbation Analysis of Optimization Problems, Springer-Verlag, N. Y. (2000).

[RoW98] Rockafellar, R. T., and Wets, R. J.-B., Variational Analysis, Springer-Verlag, Berlin (1998).

[Roc93] Rockafellar, R. T., Lagrange Multipliers and Optimality, SIAM Review, Vol. 35, pp. 183-238 (1993).

An $O(n^2)$ Algorithm for Isotonic Regression

Oleg Burdakov, Oleg Sysoev, Anders Grimvall, and Mohamed Hussian

Department of Mathematics, Linköping University, SE-58183 Linköping, Sweden
({olbur,olsys,angri,mohus}@mai.liu.se)

Summary. We consider the problem of minimizing the distance from a given n-dimensional vector to a set defined by constraints of the form $x_i \leq x_j$. Such constraints induce a partial order of the components x_i, which can be illustrated by an acyclic directed graph. This problem is also known as the isotonic regression (IR) problem. IR has important applications in statistics, operations research and signal processing, with most of them characterized by a very large value of n. For such large-scale problems, it is of great practical importance to develop algorithms whose complexity does not rise with n too rapidly. The existing optimization-based algorithms and statistical IR algorithms have either too high computational complexity or too low accuracy of the approximation to the optimal solution they generate. We introduce a new IR algorithm, which can be viewed as a generalization of the Pool-Adjacent-Violator (PAV) algorithm from completely to partially ordered data. Our algorithm combines both low computational complexity $O(n^2)$ and high accuracy. This allows us to obtain sufficiently accurate solutions to IR problems with thousands of observations.

Key words: quadratic programming, large scale optimization, least distance problem, isotonic regression, pool-adjacent-violators algorithm.

1 Introduction

We consider the isotonic regression problem (IR) in the following least distance setting. Given a vector $a \in R^n$, a strictly positive vector of weights $w \in R^n$ and a directed acyclic graph $G(N, E)$ with the set of nodes $N = \{1, 2, \ldots, n\}$, find $x^* \in R^n$ that solves the problem:

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} w_i (x_i - a_i)^2 \\ \text{s.t. } & x_i \leq x_j \quad \forall (i,j) \in E. \end{aligned} \tag{1}$$

Since this is a strictly convex quadratic programming problem, its solution x^* is unique. The optimality conditions and an analysis of the typical block (or cluster) structure of x^* can be found in [BC90,Lee83,PX99]. The monotonicity constraints defined by the acyclic graph $G(N,E)$ imply a partial order of the components x_i, $i=1,\dots,n$.

A special case of IR problem arises when there is a complete order of the components. This problem, referred to as IRC problem, is defined by a directed path $G(N,E)$ and is formulated as follows:

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} w_i(x_i - a_i)^2 \\ \text{s.t.} \quad & x_1 \le x_2 \le \cdots \le x_n. \end{aligned} \tag{2}$$

IR problem has numerous important applications, for instance in statistics [BBBB72,DR82,Lee83], operations research [MM85,R86] and signal processing [AB98, RB93]. These applications are characterized by a very large value of n. For such large-scale problems, it is of great practical importance to develop algorithms whose complexity does not rise with n too rapidly. The existing optimization-based algorithms [DMT01] and statistical IR algorithms have either high computational complexity or low accuracy of the approximation to the optimal solution they generate. In this connection, let us quote a recent paper by Strand [Str03]:

> Unfortunately, in the case where $m > 1$ and at least some of the explanatory variables are continuous (i.e. typical multiple regression data), *there is no practical algorithm* to determine LSIR ... estimates

where m stands for the number of explanatory variables. The case $m > 1$ in the least squares isotonic regression (LSIR) corresponds to problem (1), while the case $m = 1$ corresponds to (2).

The most widely used method for solving IRC problem (2) is the so-called pool-adjacent-violators (PAV) algorithm [ABERS55,BBBB72,HPW73]. This algorithm is of computational complexity $O(n)$ (see [BC90,PX99]). The PAV algorithm has been extended by Pardalos and Xue [PX99] to the special case of IR problem (1), in which the partial order of the components is presented by a directed tree. Several other special cases, in which the PAV algorithm is applied repeatedly to different subsets of the components, are considered in [BDPR84,DR82,SS97,Str03]. In [BC90], it was shown that some algorithms for problems (1) and (2) may be viewed as active set quadratic programming methods. The minimum lower set algorithm [Br55] is known to be the first algorithm proposed for the general IR problem. It was shown in [BC90] that this algorithm, being applied to IRC problem, is of complexity $O(n^2)$. Perhaps, the most widely used approaches for solving applied IR problems are based on simple averaging techniques [M88,MS94,Str03]. They can be easily implemented and enjoy a relatively low computational burden, but the quality of their approximations to x^* are very case-dependent and can be far from

optimal. The best known computational complexity for the general case of IR problem is $O(n^4)$, and it refers to an algorithm introduced in [MM85, R86]. This algorithm provides the exact solution to problem (1) by solving $O(n)$ minimal flow problems.

The aim of this paper is to present our generalization of the PAV algorithm to the case of IR problem (1). It is referred to as the GPAV algorithm. Our numerical experiments shows that it enjoys both low computational burden and high accuracy. The computational complexity of our algorithm is $O(n^2)$. It must be emphasized here that, for graphs of some special structure, the number of non-redundant monotonicity constraints in (1) grows in proportion to n^2.

In the case of the partial order presented by a directed tree, the GPAV algorithm is closely related to the algorithm of Pardalos and Xue [PX99]. These two algorithms provide the exact optimal solution, and they are of complexity $O(n \log n)$.

The article is organized as follows. Section 2 describes the GPAV algorithm. Its $O(n^2)$ complexity is discussed in Section 3. Section 4 presents the results of our numerical experiments, which indicate that the GPAV algorithm provides much higher accuracy and has lower computational burden in comparison to the simple averaging algorithm. The numerical results complement those we reported in [BGH04, HGBS05] for some other test and applied problems.

2 Generalization of PAV algorithm

We say that the component x_i is adjacent to the component x_j, if $(i, j) \in E$. According to [BC90, Lee83, PX99], the optimal solution to IR problem (1) is characterized by groups (blocks) of several adjacent components x_i^* with one and the same value equal to the average a-value over the block they belong to.

The GPAV algorithm generates some splitting of the set of nodes N into disjoint blocks of nodes. Since the values of components within one block are the same, each block is presented in the subsequent computations by one element called the head element. The block with the head element $i \in N$ is denoted by B_i. The set of head elements is denoted by H. Obviously, $H \subseteq N$, and moreover,

- $B_i \subseteq N, \quad \forall i \in H,$
- $\cup_{i \in H} B_i = N,$
- $B_i \cap B_j = \emptyset, \quad \forall i, j \in H.$

The components of x associated with each block B_i, $i \in H$, have the following values

$$x_j = \frac{\sum_{k \in B_i} w_k a_k}{W_i}, \quad \forall j \in B_i,$$

where

$$W_i = \sum_{k \in B_i} w_k$$

is the weight of block B_i.

For node $i \in N$, denote the set of its immediate predecessors $\{j \in N : (j,i) \in E\}$ by i^-. The block B_i is said to be adjacent to block B_j, or called immediate predecessor for B_j, if there exist $k \in B_i$ and $l \in B_j$ such that $k \in l^-$. Let B_i^- denote the set of all adjacent blocks for B_i. The GPAV algorithm sets initially $B_i = \{i\}$ and $B_i^- = i^-$ for all the nodes $i \in N$, and subsequently operates with the blocks only.

The GPAV algorithm deals with a reduced acyclic graph of blocks, which is initially identical to $G(N,E)$, and which subsequently can shrink from step to step. This is because one block can absorb, under certain conditions, its adjacent block. This operation reduces the set of blocks and edges connecting them. The operation of absorbing $B_i \in B_j^-$ by B_j can be presented as follows.

Algorithm ABSORB (i,j).

1. Set $H = H \setminus \{i\}$.
2. Set $B_j^- = B_i^- \cup B_j^- \setminus \{i\}$.
3. Set $x_j = (W_j x_j + W_i x_i) \;/\; (W_j + W_i)$.
4. Set $B_j = B_j \cup B_i$ and $W_j = W_j + W_i$.
5. For all $k \in H$ such that $i \in B_k^-$, set $B_k^- = B_k^- \setminus \{i\} \cup \{j\}$.

The GPAV algorithm returns a feasible approximation $\hat{x}$ to the optimal solution x^*. Our numerical experiments show that the accuracy of this approximation is sufficiently high. The GPAV algorithm begins with the set of untreated blocks $U = N$, and then treats the blocks in U one by one. This takes n steps. After each treatment step, the values assigned to the components x_i, $i \in H \setminus U$, ensure the fulfillment of those of the original monotonicity constraints, which involve only the components composing the blocks B_i, $i \in H \setminus U$. It is typical in practice that these values are optimal solutions to problem (1) reduced to the variables $x_i \in N \setminus U$. If not optimal, the values of these components are reasonably close to the optimal ones.

We call B_j a lower block of U, if the set $\{i \in U : \; i \in B_j^-\}$ is empty, i.e., if U contains no blocks adjacent to $B_j \in U$. The block to be treated, say B_j, is chosen in the GPAV algorithm among the lower elements of the partially ordered set U. Then B_j can be enlarged by absorbing, one by one, some of its adjacent blocks in $H \setminus U$. If the common value of the components of x in the new block violates any of the monotonicity constraints involving its adjacent blocks, the new block absorbs the one corresponding to the mostly violated constraint. This operation is repeated until all the constraints involving the new block and its adjacent ones are satisfied. Our generalization of the PAV algorithm can be presented as follows.

Algorithm GPAV.

1. Set $H = N$ and $U = N$.
2. For all $i \in N$, set $B_i = \{i\}$, $B_i^- = i^-$, $x_i = a_i$ and $W_i = w_i$.
3. While $U \neq \emptyset$, do:
4. Find any of the lower elements of U, say, j. Set $U = U \setminus \{j\}$.
5. While there exists $k \in B_j^-$ such that $x_k \geq x_j$, do:
6. Find $i \in B_j^-$ such that $x_i = \max\{x_k : k \in B_j^-\}$.
7. ABSORB (i, j).
8. For all $k \in H$ and $i \in B_k$, set $\hat{x}_i = x_k$.

This algorithm can be viewed as a generalization of the PAV algorithm, because they coincide for IRC problem (2). Moreover, the GPAV algorithm is closely related to the algorithm of Pardalos and Xue [PX99], when the graph $G(N, E)$ in IR problem (1) is a directed tree.

3 Complexity

A detailed proof of the fact that the GPAV algorithm is of computational complexity $O(n^2)$ will be presented in a separate paper. Here we just outline the basic properties of our algorithm which will be used in the proof.

Obviously, Steps 1, 2 and 8 of Algorithm GPAV take $O(n)$ time. Step 4 is repeated n times, and each time the complexity of finding a lower element of U does not exceed $O(|U|)$. Thus, all the operations related to Step 4 are of complexity $O(n^2)$. Algorithm ABSORB can be implemented in $O(n)$ time. The number of times that it is called on Step 7 of Algorithm GPAV does not exceed n, the total number of nodes. The while-loop presented by Steps 5-7 can not be repeated more than $2n$ times, because every computation of the while-condition either terminates (n times at most) the treatment of the new block, or allows (n times at most) the new block to absorb one of its adjacent blocks. At Steps 5 and 6, each operation of finding the maximal value of the head components of the blocks adjacent to the new one takes $O(n)$ time. Therefore, the total number of operations associated with the while-loop can be estimated as $O(n^2)$.

The reasoning above explains why the GPAV algorithm is of computational complexity $O(n^2)$. Notice that the complexity is of the same order as the possible maximal number of non-redundant constraints in (1), which is also estimated as $O(n^2)$.

4 Numerical results

We used MATLAB for implementing the GPAV algorithm, the simple averaging algorithm described in [M88], and a modification of the GPAV algorithm. They are referred to as GPAV, SA and GPAV+, respectively.

In the modification mentioned above, we run GPAV twice. First, we run it on the original problem (1), which returns an approximation $\hat{x}^I$ to the optimal solution. Second, we run it on the problem

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} w_i(\bar{x}_i - \bar{a}_i)^2 \\ \text{s.t. } & \bar{x}_i \le \bar{x}_j \quad \forall (i,j) \in \bar{E}, \end{aligned}$$

which is equivalent to (1) for $\bar{a}_i = -a_i$, $\bar{x}_i = -x_i$ and for the same edges in $\bar{E}$ as in E, but inverted. Let $\hat{x}^{II}$ denote the resulting approximation to the optimal solution of (1). Denote the convex linear combination of these two approximations as $x(\lambda) = \lambda\hat{x}^I + (1-\lambda)\hat{x}^{II}$, where the scalar $\lambda \in [0,1]$. Since the feasible region in (1) is a convex set, $x(\lambda)$ does not violate the monotonicity constraints. Then GPAV+ returns an approximation to the optimal solution of problem (1) that corresponds to the optimal solution of the simple one-dimensional problem:

$$\min_{0 \le \lambda \le 1} \sum_{i=1}^{n} w_i(x_i(\lambda) - a_i)^2.$$

In our test IR problems we set $w_i = 1$, $i = 1, \ldots, n$. The test problems were based on the model

$$a = U_1 + U_2 + \epsilon,$$

where U_1, U_2 are two explanatory variables ($m = 2$) and ϵ is an error. Samples of $n = 80$, $n = 400$ and $n = 2000$ observations

$$U^i = \begin{pmatrix} U_1^i \\ U_2^i \end{pmatrix}, \quad a_i = U_1^i + U_2^i + \epsilon_i, \quad i = 1, \ldots, n,$$

were generated for normally distributed explanatory variables and for normally, exponentially or double-exponentially distributed error. Thereafter, for each $i, j \in N$ such that $U^i \le U^j$ component-wise, we generated the monotonicity constraint $x_i \le x_j$. In this way we constructed the set of edges

$$E = \{(i,j) : \ U^i \le U^j, \ i,j \in N\}, \tag{3}$$

that, along with $a_1, \ldots, a_n$, define problem (1). Then we reduced E by eliminating all the redundant constraints, i.e., we eliminated the edge (i,j), if there existed $k \in N$ such that $(i,k) \in E$ and $(k,j) \in E$. We also performed a topological sort [CLRS01] of our directed acyclic graph to assure that $(i,j) \in E$ implies $i < j$. The topological sort allowed to skip the search for a lower element of U at Step 4 of Algorithm GPAV; instead, the blocks were treated in the natural order $B_1, B_2, \ldots, B_n$.

For the future numerical experiments, we plan to generate in an efficient way the reduced set of edges directly from the data, without generating the

complete set of edges (3). This will produce simultaneously a topological sort of our graph.

The numerical results presented here were obtained on a PC running under Windows XP with a Pentium 4 processor (2.8 GHz). We compare the objective function value of problem (1), obtained by GPAV, GPAV+ and SA, with the optimal objective function value obtained by MATLAB solver lsqlin. Tables 1, 2 and 3 summarize the obtained performance data, where we use the notation

N/A = lsqlin failed to solve problem within given time limits,
sum = objective function value (sum of squares),
cpu = CPU time (seconds) until termination,
r.e.% = relative error (%) calculated as follows,

$$\frac{\text{sum}_{\text{A}} - \text{sum}_{\text{ lsqlin}}}{\text{sum}_{\text{ lsqlin}}} \cdot 100\%,$$

where A stands for GPAV, GPAV+ or SA.

Table 1: Normally distributed errors.

n	80			400			2000	
	sum	cpu	r.e.%	sum	cpu	r.e.%	sum	cpu
GPAV	41.966	0.016	4.14	279.819	0.172	1.16	1670.699	6.532
GPAV+	41.315	0.048	2.52	279.496	0.562	1.05	1657.814	19.579
SA	52.722	0.047	30.83	356.515	0.875	28.89	2132.418	18.875
lsqlin	40.297	0.657	0.00	276.589	263.969	0.00	N/A	N/A

Table 2: Exponentially distributed errors.

n	80			400			2000	
	sum	cpu	r.e.%	sum	cpu	r.e.%	sum	cpu
GPAV	19.475	0.032	2.52	161.177	0.171	0.74	1445.026	6.375
GPAV+	19.365	0.062	1.94	160.631	0.655	0.40	1439.493	19.359
SA	23.601	0.047	24.24	222.578	0.891	39.12	1726.989	18.750
lsqlin	18.995	0.390	0.00	159.983	233.766	0.00	N/A	N/A

The three tables show that GPAV required less CPU time than SA for producing much more accurate approximation to the optimal solution. Moreover, GPAV+ always improved the relative error produced by GPAV, requiring for this about three times more CPU time. The linear combination of the approximations generated by GPAV+ and SA did not result in any appreciable improvement of the accuracy.

Table 3: Double-exponentially distributed errors.

n	80			400			2000	
	sum	cpu	r.e.%	sum	cpu	r.e.%	sum	cpu
GPAV	51.190	0.015	0.93	604.085	0.266	1.56	3344.406	6.266
GPAV+	51.105	0.047	0.76	598.177	0.702	0.56	3317.172	18.219
SA	70.043	0.031	38.10	1406.101	0.844	136.40	7488.342	18.250
lsqlin	50.716	0.484	0.00	594.787	365.000	0.00	N/A	N/A

All these observations are in good agreement with the numerical results reported in [BGH04, HGBS05] for some other test and applied problems.

5 Conclusions

The presented generalization of the PAV algorithm allows us now to obtain sufficiently accurate solutions to the isotonic regression problems with thousands of observations. The lack of the complete order in the data is not that limiting now as it was previously.

In our future work, we plan to improve the presented version of the GPAV algorithm in the following way. When the algorithm is running, it is not difficult to check whether the x-components of the new block produced on Steps 5-7 are suspected to be nonoptimal. Then such block can be split in smaller ones in order to make the corresponding values of the x-components more close, or even equal, to the optimal values.

6 Acknowledgements

The authors are grateful for financial support from the Swedish Research Council.

References

[AB98] Acton, S.A., Bovik, A.C.: Nonlinear image estimation using piecewise and local image models. IEEE Transactions on Image Processing, **7**, 979–991 (1998).

[ABERS55] Ayer, M., Brunk, H.D., Ewing, G.M., Reid, W.T., Silverman, E.: An empirical distribution function for sampling with incomplete information. The Annals of Mathematical Statistics, **26**, 641–647 (1955).

[BBBB72] Barlow, R.E., Bartholomew, D.J., Bremner, J.M., Brunk, H.D.: Statistical inference under order restrictions. Wiley, New York (1972).

[BC90] Best, M.J., Chakravarti, N.: Active set algorithms for isotonic regression: a unifying framework. Mathematical Programming, **47**, 425–439 (1990).

[BDPR84] Bril, G., Dykstra, R., Pillers, C., Robertson, T.: Algorithm AS 206, isotonic regression in two independent variables. Applied Statistics, **33**, 352–357 (1984).

[Br55] Brunk, H.D.: Maximumlikelihood estimates of monotone parameters. Annals of Mathematical Statistics, **26**, 607–616 (1955).

[BGH04] Burdakov, O., Grimvall, A., Hussian, M.: A generalised PAV algorithm for monotonic regression in several variables. In: Antoch, J. (ed.) COMPSTAT, Proceedings in Computational Statistics, 16th Symposium Held in Prague, Czech Republic. Physica-Verlag, A Springer Company, Heidelberg New York, 761–767 (2004).

[CLRS01] Cormen, T. H., Leiserson, C.E., Rivest, R. L., Stein, C.: Introduction to Algorithms. The MIT Press, Cambridge MA (2001).

[DMT01] De Simone, V., Marino, M., Toraldo, G.: Isotonic regression problems. In: Floudas, C.A., Pardalos, P.M. (eds) Encyclopedia of Optimization. Kluwer Academic Publishers, Dordrecht London Boston (2001).

[DR82] Dykstra, R., Robertson, T.: An algorithm for isotonic regression for two or more independent variables. he Annals of Statistics, **10**, 708–716 (1982).

[HPW73] Hanson, D.L., Pledger, G., Wright, F.T.: On consistency in monotonic regression. The Annals of Statistics, **1**, 401–421 (1973).

[HGBS05] Hussian, M., Grimvall, A., Burdakov, O., Sysoev, O.: Monotonic regression for the detection of temporal trends in environmental quality data. MATCH Commun. Math. Comput. Chem., **54**, 535–550 (2005).

[Lee83] Lee, C.I.C.: The min-max algorithm and isotonic regression. The Annals of Statistics, **11**, 467–477 (1983).

[MM85] Maxwell, W.L., Muchstadt, J.A.: Establishing consistent and realistic reorder intervals in production-distribution systems. Operations Research, **33**, 1316–1341 (1985).

[M88] Mukarjee, H.: Monotone nonparametric regression. The Annals of Statistics, **16**, 741–750 (1988).

[MS94] Mukarjee, H., Stern, H.: Feasible nonparametric estimation of multiargument monotone functions. Journal of the American Statistical Association, **425**, 77–80 (1994).

[PX99] Pardalos, P.M., Xue, G.: Algorithms for a class of isotonic regression problems. Algorithmica, **23**, 211–222 (1999).

[RB93] Restrepo, A., Bovik, A.C.: Locally monotonic regression. IEEE Transactions on Signal Processing, **41**, 2796–2810 (1993).

[R86] Roundy, R.: A 98% effective lot-sizing rule for a multiproduct multistage production/inventory system. Mathematics of Operations Research, **11**, 699–727 (1986).

[SS97] Schell, M.J., Singh, B.: The reduced monotonic regression method. Journal of the American Statistical Association, **92**, 128–135 (1997).

[Str03] Strand, M.: Comparison of methods for monotone nonparametric multiple regression. Communications in Statistics - Simulation and Computation, **32**, 165–178 (2003).

KNITRO: An Integrated Package for Nonlinear Optimization

Richard H. Byrd[1], Jorge Nocedal[2], and Richard A. Waltz[2]

[1] Department of Computer Science, University of Colorado, Boulder, CO 80309, USA. (richard@cs.colorado.edu)
[2] Department of Electrical and Computer Engineering, Northwestern University, Evanston, IL, 60208-3118, USA. ({nocedal,rwaltz}@ece.northwestern.edu.)

Summary. This paper describes KNITRO 5.0, a C-package for nonlinear optimization that combines complementary approaches to nonlinear optimization to achieve robust performance over a wide range of application requirements. The package is designed for solving large-scale, smooth nonlinear programming problems, and it is also effective for the following special cases: unconstrained optimization, nonlinear systems of equations, least squares, and linear and quadratic programming. Various algorithmic options are available, including two interior methods and an active-set method. The package provides crossover techniques between algorithmic options as well as automatic selection of options and settings.

Key words: nonlinear optimization, optimization software, interior-point, SLQP.

1 Introduction

Nonlinear programming problems are often difficult to solve. In spite of the rapid pace of algorithmic improvements, the most efficient algorithms available at present provide no guarantees of success or of fast performance over a range of applications. To complicate matters, the search for improved methods has led researchers to propose a variety of algorithms, each of which is typically implemented in a separate software package. To overcome the numerous difficulties that arise in practice, software developers have included a variety of options and heuristics to improve the chances of success. These packages are, however, constrained by the underlying algorithm, and as is

[1] This author was supported by Army Research Office Grants DAAG55-98-1-0176 and DAAD19-02-1-0407, and NSF grants CCR-0219190 and CHE-0205170.
[2] These authors were supported by National Science Foundation grants CCR-0219438 and DMI-0422132, and Department of Energy grant DE-FG02-87ER25047-A004.

well known, no single approach is uniformly successful in nonlinear optimization. The prospective user is thus faced with a difficult choice. Each code is unique in many ways: input and output formats, options and conventions. Thus there is a steep learning curve in trying to achieve the most effective use of a package. The availability of many codes through the NEOS Server `http://www-neos.mcs.anl.gov/` addresses only some of these issues.

The KNITRO software package aims to achieve greater flexibility and robustness through an integration of two powerful and complementary algorithmic approaches for nonlinear optimization: the interior-point approach and the active-set approach. The impressive success of an integrated approach of this sort for linear and integer programming, particularly over the past decade [21, 23], argues for a similar approach to be taken in nonlinear optimization. KNITRO is capable of applying features of an interior-point method or an active-set method — or possibly both — depending on problem characteristics. Within the interior-point approach, KNITRO provides two algorithms implementing distinct barrier approaches. One of the main challenges in the development of KNITRO has been the effective integration of the interior and active-set algorithms into a unified package, and the development of tools that exploit the power of our integrated approach.

The nonlinear programming formulation considered in this paper is:

$$\min_x f(x) \tag{1.1a}$$

$$\text{subject to } c_E(x) = 0 \tag{1.1b}$$

$$c_I(x) \geq 0, \tag{1.1c}$$

where $f : \mathbb{R}^n \to \mathbb{R}$, $c_E : \mathbb{R}^n \to \mathbb{R}^l$ and $c_I : \mathbb{R}^n \to \mathbb{R}^m$ are twice continuously differentiable functions. Problem (1.1) includes as special cases unconstrained optimization, systems of nonlinear equations, least squares problems, linear programs and quadratic programs. An important feature of the algorithms implemented in KNITRO is that they automatically reduce to effective algorithms for each of the simpler problem classes.

The quality and diversity of nonlinear optimization software has greatly improved during the last 10 years. Some of the established packages have matured, and new packages have emerged. SNOPT [18] and FILTERSQP [15] implement active-set sequential quadratic programming (SQP) methods. SNOPT uses a line search approach, and in its default setting, employs quasi-Newton approximations to the Hessian. FILTERSQP follows a trust region approach, with filter globalization, and makes use of second-derivative information. The MINOS [29] and LANCELOT [12] packages, which were the first widely available codes capable of solving problems with tens of thousands of variables and constraints, implement augmented Lagrangian methods. Another well established package is CONOPT [14], which offers reduced Hessian and SQP methods.

Most of the new packages are based on the interior-point approach. LOQO [33] implements a line search primal-dual algorithm that can be viewed as a direct extension of interior methods for linear and quadratic programming.

The first release of KNITRO [6] offered a trust region interior-point algorithm employing a conjugate gradient iteration in the step computation; the second release added a line search interior algorithm that is safeguarded by the trust region approach [38]. BARNLP [2] and IPOPT [36] implement line search interior-point approaches; IPOPT uses a filter globalization and includes a feasibility restoration phase. MOSEK [1] is a primal-dual interior-point method for *convex* optimization, and PENNON [25] follows an augmented Lagrangian approach.

New active-set methods based on Sequential Linear-Quadratic Programming (SLQP) have recently been studied by Chin and Fletcher [9] and Byrd et al. [5]. Unlike SQP methods, which combine the active-set identification and the step computation in one quadratic subproblem, SLQP methods decouple these tasks into two subproblems. The active-set algorithm in KNITRO, implements the SLQP method described in [5].

Interior-point and active-set methods offer competing state-of-the-art approaches for solving nonlinear optimization problems — each with its own set of advantages. Benchmarking studies [13,28] have tried to identify the classes of problems for which each approach is best suited, but the rapid pace of software development makes it difficult to arrive at concrete conclusions at this time. We take the view that interior-point and active-set methods will both be needed in the years to come.

2 Overview of the Package

KNITRO 5.0 is a C-package for solving nonlinear optimization problems. It is designed for large-scale applications, but it is also effective on small and medium scale problems. A great deal of attention has been given to the performance of the KNITRO algorithms on simpler classes of problems such as systems of nonlinear equations and unconstrained problems because these tasks are crucial in the solution of nonlinear programming problems. We have also ensured that the algorithms are fast and reliable on linear and quadratic programming problems. A schematic view of the KNITRO package is given in Figure 1.

In Figure 1 the nomenclature CG reflects the fact that the algorithmic step is computed using an iterative conjugate gradient approach, while DIRECT implies that the step is (usually) computed via a direct factorization of a linear system. As the figure suggests, the software design will enable the addition of future options in the package such as a DIRECT version of the active-set algorithm. Throughout the remainder of this paper we will refer to the implementations of the CG and direct interior-point algorithms in KNITRO as INTERIOR/CG and INTERIOR/DIRECT, and the active-set algorithm implementation will be called ACTIVE.

In the following sections we give an outline of the algorithms implemented in KNITRO. The descriptions are inevitably incomplete, since many additional

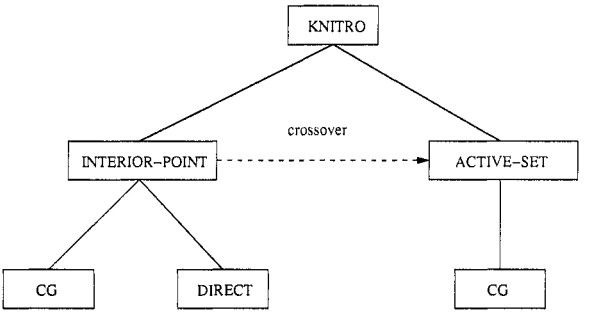


Fig. 1: The main algorithmic options in the KNITRO 5.0 package.

features (such as second-order corrections, iterative refinement steps, resetting of parameters, and regularization procedures) are needed to achieve efficiency and robustness over a range of problems. Nevertheless, our outlines highlight the main features of the algorithms.

3 Interior-Point Methods

The interior (or barrier) methods implemented in KNITRO associate with (1.1) the barrier problem

$$\min_{x,s} \quad f(x) - \mu \sum_{i=1}^{m} \log s_i \tag{3.1a}$$

$$\text{subject to} \quad c_E(x) = 0 \tag{3.1b}$$

$$c_I(x) - s = 0, \tag{3.1c}$$

where s is a vector of slack variables and $\mu > 0$. The interior approach consists of finding (approximate) solutions of the barrier problem (3.1) for a sequence of positive barrier parameters $\{\mu_k\}$ that converges to zero.

The KKT conditions for (3.1) can be written as

$$\nabla f(x) - A_E{}^T(x)y - A_I{}^T(x)z = 0 \tag{3.2a}$$

$$-\mu e + Sz = 0 \tag{3.2b}$$

$$c_E(x) = 0 \tag{3.2c}$$

$$c_I(x) - s = 0, \tag{3.2d}$$

where $e = (1, ..., 1)^T$, $S = \text{diag}(s_1, ..., s_m)$, A_E and A_I are the Jacobian matrices corresponding to the equality and inequality constraint vectors respectively, and y and z represent vectors of Lagrange multipliers. We also must have that $s, z \geq 0$. In the line search approach, we apply Newton's method to (3.2), backtracking if necessary so that the variables s, z remain positive,

and so that the merit function is sufficiently reduced. In the trust region approach, we associate a quadratic program with (3.1) and let the step of the algorithm be an approximate solution of this quadratic subproblem. These two approaches are implemented, respectively, in the INTERIOR/DIRECT and INTERIOR/CG algorithms, and are described in more detail below.

The other major ingredient in interior methods is the procedure for choosing the sequence of barrier parameters $\{\mu_k\}$. Several options are provided in KNITRO. In the *Fiacco-McCormick/monotone approach*, the barrier parameter μ is held fixed for a series of iterations until the KKT conditions (3.2) are satisfied to some accuracy. An alternative is to use an *adaptive strategy* in which the barrier parameter is updated at every iteration. We have implemented the following adaptive update options: (i) the rule implemented in LOQO [33] based on the deviation of the minimum complementarity pair from the average; (ii) a probing strategy that uses Mehrotra's predictor step to select a target value for μ; (iii) a so-called quality-function approach; (iv) variants of option (ii) which possibly utilize safeguarded corrector steps. These rules are described and tested in Nocedal, Wächter and Waltz [30]. Since it is not known at present which one is the most effective in practice, KNITRO allows the user to experiment with the barrier update strategies just mentioned.

To control the quality of the steps, both interior algorithms make use of the non-differentiable merit function

$$\phi_\nu(x,s) = f(x) - \mu \sum_{i=1}^{m} \log s_i + \nu \|c_E(x)\|_2 + \nu \|c_I(x) - s\|_2, \qquad (3.3)$$

where $\nu > 0$. A step is acceptable only if it provides a sufficient decrease in ϕ_ν. Although it has been reported in the literature [22, 34] that merit functions of this type can interfere with rapid progress of the iteration, our experience indicates that the implementation described in Section 3.3 overcomes these difficulties. These observations are consistent with the results reported in Table 2 of Wächter and Biegler [36], which suggest that this merit function approach is as tolerant as a filter mechanism.

3.1 Algorithm I: KNITRO-INTERIOR/DIRECT

In this algorithm a typical iteration first computes a (primary) line search step using direct linear algebra. In order to obtain global convergence in the presence of non-convexity and Hessian or Jacobian singularities, the primary step may be replaced, under certain circumstances, by a safeguarding trust region step. INTERIOR-DIRECT is an implementation of the algorithm described in [38].

We begin by describing the (primary) line search step. Applying Newton's method to (3.2), in the variables x, s, y, z, gives

$$\begin{bmatrix} \nabla^2_{xx}\mathcal{L} & 0 & -A_E{}^T(x) & -A_I{}^T(x) \\ 0 & Z & 0 & S \\ A_E(x) & 0 & 0 & 0 \\ A_I(x) & -I & 0 & 0 \end{bmatrix} \begin{bmatrix} d_x \\ d_s \\ d_y \\ d_z \end{bmatrix} = - \begin{bmatrix} \nabla f(x) - A_E{}^T(x)y - A_I{}^T(x)z \\ Sz - \mu e \\ c_E(x) \\ c_I(x) - s \end{bmatrix}, \tag{3.4}$$

where $\mathcal{L}$ denotes the Lagrangian

$$\mathcal{L}(x, s, y, z) = f(x) - y^T c_E(x) - z^T(c_I(x) - s). \tag{3.5}$$

If the inertia of the matrix in (3.4) is

$$(n + m, l + m, 0), \tag{3.6}$$

then the step d determined from (3.4) can be guaranteed to be a descent direction for the merit function (3.3). In this case, we compute the scalars

$$\alpha_s^{\max} = \max\{\alpha \in (0, 1] : s + \alpha d_s \geq (1 - \tau)s\}, \tag{3.7a}$$

$$\alpha_z^{\max} = \max\{\alpha \in (0, 1] : z + \alpha d_z \geq (1 - \tau)z\}, \tag{3.7b}$$

with $\tau = 0.995$. If $\min(\alpha_s^{\max}, \alpha_z^{\max})$ is not too small, we perform a backtracking line search that computes the steplengths

$$\alpha_s \in (0, \alpha_s^{\max}], \qquad \alpha_z \in (0, \alpha_z^{\max}], \tag{3.8}$$

providing sufficient decrease of the merit function (3.3). The new iterate is then defined as

$$x^+ = x + \alpha_s d_x, s^+ = s + \alpha_s d_s, \tag{3.9a}$$

$$y^+ = y + \alpha_z d_y, z^+ = z + \alpha_z d_z. \tag{3.9b}$$

On the other hand, if the inertia is not given by (3.6) or if the steplength α_s or α_z is less than a given threshold α_{min}, then the primary step d is rejected. In this case the algorithm reverts to the trust region method implemented in the INTERIOR/CG algorithm (see the next section) which is guaranteed to provide a successful step even in the presence of negative curvature or singularity.

The use of this safeguarding trust region step makes the KNITRO-INTERIOR/DIRECT algorithm distinct from other line search interior-point algorithms, such as BARNLP, IPOPT and LOQO, which modify the Hessian $\nabla^2_{xx}\mathcal{L}$ whenever the inertia condition (3.6) is not satisfied. We prefer to revert to a trust region iteration because this permits us to compute a step using a null-space approach, without modifying the Hessian $\nabla^2_{xx}\mathcal{L}$. An additional benefit of invoking the trust region step is that it guarantees progress in cases when the line search approach can fail [7, 35]. Since it is known that, when line search iterations converge to non-stationary points, the steplengths α_s or α_z in (3.9) converge to zero, we monitor these steplengths. If one of them is smaller than

a given threshold, we discard the line search iteration (3.4)-(3.9) and replace it with the trust region step.

We outline the method in Algorithm 3.1. Here $D\phi_\nu(x, s; d)$ denotes the directional derivative of the merit function ϕ_ν along a direction d. The algorithm maintains a trust region radius Δ_k at every iteration, in case it needs to revert to the trust region approach.

Algorithm 3.1: KNITRO-INTERIOR/DIRECT

Choose x_0, $s_0 > 0$, and the parameters $0 < \eta$, and $0 < \alpha_{min} < 1$.
Compute initial values for the multipliers y_0, $z_0 > 0$, the trust-region radius $\Delta_0 > 0$, and the barrier parameter $\mu > 0$. Set $k = 0$.
Repeat until a stopping test for the nonlinear program (1.1) is satisfied:
  Repeat until the perturbed KKT conditions (3.2) are approximately satisfied:
    Factor the primal-dual system (3.4) and record the number `neig` of negative eigenvalues of its coefficient matrix.
    Set `LineSearch = False`.
    If `neig` $\leq l + m$
      Solve (3.4) to obtain the search direction $d = (d_x, d_s, d_y, d_z)$.
      Define $w = (x_k, s_k)$ and $d_w = (d_x, d_s)$.
      Compute $\alpha_s^{\max}, \alpha_z^{\max}$ by (3.7).
      If $\min\{\alpha_s^{\max}, \alpha_z^{\max}\} > \alpha_{min}$,
        Update the penalty parameter ν_k (see Section 3.3).
        Compute a steplength $\alpha_s = \bar{\alpha}\alpha_s^{\max}, \bar{\alpha} \in (0, 1]$ such that $\phi_\nu(w + \alpha_s d_w) \leq \phi_\nu(w) + \eta\alpha_s D\phi_\nu(w; d_w)$.
        If $\alpha_s > \alpha_{min}$,
          Set $\alpha_z = \bar{\alpha}\alpha_z^{\max}$.
          Set $(x_{k+1}, s_{k+1}, y_{k+1}, z_{k+1})$ by (3.9).
          Set `LineSearch = True`.
        Endif
      Endif
    Endif
    If `LineSearch == False`,
      Compute $(x_{k+1}, s_{k+1}, y_{k+1}, z_{k+1})$ using the INTERIOR/CG algorithm of Section 3.2.
    Endif
    Compute Δ_{k+1}.
    Set $k \leftarrow k + 1$.
  End
  Choose a smaller value for the barrier parameter μ.
End

The initial multipliers y_0, z_0 are computed as the least-squares solution of the system (3.2*a*)-(3.2*b*). When the line search step is discarded (the last If-Endif block in Algorithm 3.1) we compute one or more INTERIOR/CG steps (described in the following section) until one of them provides sufficient reduction in the merit function.

We assume in Algorithm 3.1 that we are using the Fiacco-McCormick/monotone approach for updating the barrier parameter μ. However, this algorithm is easily modified to implement the adaptive barrier update strategies discussed at the beginning of Section 3. In this case, there is no barrier stop test and the barrier parameter μ is updated at every iteration using some adaptive rule (which could cause μ to increase or decrease).

3.2 Algorithmic Option II: KNITRO-INTERIOR/CG

The second interior algorithm implemented in KNITRO computes steps by using a quadratic model and trust regions. This formulation allows great freedom in the choice of the Hessian and provides a mechanism for coping with Jacobian and Hessian singularities. The price for this flexibility is a more complex iteration than in the line search approach. INTERIOR/CG is an implementation of the algorithm described in [6], which is based on the approach described and analyzed in [3].

To motivate the INTERIOR/CG algorithm, we first note that the barrier problem (3.1) is an equality-constrained optimization problem and can be solved by using a sequential quadratic programming method with trust regions. A straightforward application of SQP techniques to the barrier problem leads, however, to inefficient steps that tend to violate the positivity of the slack variables and are frequently cut short by the trust-region constraint. To overcome this problem, we design the following SQP method specifically tailored to the barrier problem.

At the current iterate (x_k, s_k), and for a given barrier parameter μ, we first compute Lagrange multiplier estimates (y_k, z_k) and then compute a step $d = (d_x, d_s)$ that aims to solve the subproblem,

$$\min_{d_x, d_s} \nabla f(x_k)^T d_x + \frac{1}{2} d_x^T \nabla^2_{xx} \mathcal{L}(x_k, s_k, y_k, z_k) d_x - \mu e^T S_k^{-1} d_s + \frac{1}{2} d_s^T \Sigma_k d_s \tag{3.10a}$$

$$\text{subject to} \quad A_E(x_k) d_x + c_E(x_k) = r_E \tag{3.10b}$$

$$A_I(x_k) d_x - d_s + c_I(x_k) - s_k = r_I \tag{3.10c}$$

$$\|d_x, S_k^{-1} d_s\|_2 \leq \Delta_k \tag{3.10d}$$

$$d_s \geq -\tau s, \tag{3.10e}$$

where $\Sigma_k = S_k^{-1} Z_k$ and $\tau = 0.995$. Ideally, we would like to set $r = (r_E, r_I) = 0$, but since this could cause the constraints to be incompatible or produce poor steps, we choose r as the smallest vector such that (3.10*b*)-(3.10*d*) are

consistent (with some margin). This computation is described in more detail below.

We can motivate the choice of the objective (3.10*a*) by noting that the first-order optimality conditions of (3.10*a*)-(3.10*c*) are given by (3.2) (with the second block of equations scaled by S^{-1}). The steps computed by using (3.10) are thus related to those of the line search algorithm described in the previous section. The trust-region constraint (3.10*d*) guarantees that (3.10) has a finite solution even when $\nabla^2_{xx}\mathcal{L}(x_k, s_k, y_k, z_k)$ is not positive definite. Therefore the Hessian need never be modified. The scaling S_k^{-1} in the trust-region constraint is crucial; its effect on the iteration will be discussed later on.

Step Computation

The subproblem (3.10) is difficult to minimize exactly because of the presence of the nonlinear constraint (3.10*d*) and the bounds (3.10*e*). We can, however, compute useful inexact solutions, at a moderate cost. To do so, KNITRO follows a null-space approach in which the step d is the sum of a *normal step* v that attempts to satisfy the linear constraints (3.10*b*)-(3.10*c*) (with $r = 0$) as well as possible subject to a trust region, and a *tangential step* that lies on the tangent space of the constraints and that tries to achieve optimality.

To compute the normal step $v = (v_x, v_s)$, we formulate the following subproblem:

$$\min_v \|A_E v_x + c_E\|_2^2 + \|A_I v_x - v_s + c_I - s\|_2^2 \tag{3.11a}$$

$$\text{subject to } \|(v_x, S^{-1}v_s)\|_2 \leq 0.8\Delta. \tag{3.11b}$$

(Here and below we omit the arguments of the functions for simplicity.) We compute an inexact solution of this problem using a dogleg approach, which minimizes (3.11*a*) along a piecewise linear path composed of a steepest descent step in the norm used in (3.11*b*) and a minimum-norm Newton step with respect to the same norm. The scaling $S^{-1}v_s$ in the norm tends to limit the extent to which the bounds on the slack variables are violated.

Once the normal step v is computed, we define the vectors r_E and r_I in (3.10*b*)-(3.10*c*) as the residuals in the normal step computation, namely,

$$r_E = A_E v_x + c_E, \qquad r_I = A_I v_x - v_s + (c_I - s).$$

Having computed the normal step (v_x, v_s), the subproblem (3.10) can therefore be written as

$$\min_{d_x, d_s} \nabla f^T d_x - \mu e^T S^{-1} d_s + \frac{1}{2}(d_x^T \nabla^2_{xx}\mathcal{L} d_x + d_s^T \Sigma d_s) \tag{3.12a}$$

$$\text{subject to } A_E d_x = A_E v_x \tag{3.12b}$$

$$A_I d_x - d_s = A_I v_x - v_s \tag{3.12c}$$

$$\|(d_x, S^{-1}d_s)\|_2 \leq \Delta, \tag{3.12d}$$

which we refer to as the tangential subproblem. To find an approximate solution d of (3.12), we first introduce the scaling

$$\tilde{d}_s \leftarrow S^{-1} d_s, \tag{3.13}$$

which transforms (3.12d) into a sphere. Then we apply the projected conjugate gradient (CG) method of Section 5 to the transformed quadratic program, iterating in the linear manifold defined by (3.12b)-(3.12c). During the solution by CG, we use a Steihaug strategy, monitoring the satisfaction of the trust-region constraint (3.12d), and stopping if the boundary of this region is reached or if negative curvature is detected. Finally, we truncate the step d if necessary in order to satisfy (3.10e).

We outline this interior method in Algorithm 3.2. Here

$$\text{ared}(d) = \phi_\nu(x, s) - \phi_\nu(x + d_x, s + d_s) \tag{3.14}$$

is the actual reduction in the merit function, and the predicted reduction, $\text{pred}(d)$, is defined by (3.15),(3.16).

Algorithm 3.2: KNITRO-INTERIOR/CG

Choose parameter $\eta > 0$. Choose initial values for $\mu > 0$, x_0, $s_0 > 0$ and $\Delta_0 > 0$.
Set $k = 0$.
Repeat until a stopping test for the nonlinear program (1.1) is satisfied:
  Repeat until the perturbed KKT conditions (3.2) are approximately satisfied:
    Compute the normal step $v_k = (v_x, v_s)$.
    Compute Lagrange multipliers $y_k, z_k > 0$.
    Compute the total step d_k by applying the projected CG method to (3.12a)-(3.12c) (see Section 5).
    Update the penalty parameter ν_k (see Section 3.3).
    Compute $\text{ared}_k(d_k)$ by (3.14) and $\text{pred}_k(d_k)$ by (3.16).
    If $\text{ared}_k(d_k) \geq \eta \text{pred}_k(d_k)$
      Set $x_{k+1} = x_k + d_x$, $s_{k+1} = s_k + d_s$, and update Δ_{k+1}.
    Else
      Set $x_{k+1} = x_k$, $s_{k+1} = s_k$, and choose $\Delta_{k+1} < \Delta_k$.
    Endif
    Set $k \leftarrow k + 1$.
  End
  Choose a smaller value for the barrier parameter μ .
End

The multiplier estimates (y_k, z_k) are computed by a least squares approximation to the equations (3.2a)-(3.2b) at x_k, and shifted to ensure positivity

of z_k. The barrier stop tolerance can be defined as $\epsilon_\mu = \mu$. As with the INTERIOR/DIRECT algorithm, this algorithm is easily modified to implement adaptive barrier update strategies.

The interior-point method outlined in Algorithm 3.2 is asymptotically equivalent to standard line search interior methods, but it is significantly different in two respects. First, it is not a fully primal-dual method in the sense that multipliers are computed as a function of the primal variables (x, s) — as opposed to the formulation (3.4) in which primal and dual variables are computed simultaneously from their previous values. Second, the trust-region method uses a scaling of the variables that discourages moves toward the boundary of the feasible region. This causes the algorithm to generate steps that can be very different from those produced by a line search method.

3.3 Merit Function

The role of the merit function (3.3) is to determine whether a step is productive and should be accepted. Our numerical experience has shown that the choice of the merit parameter ν plays a crucial role in the efficiency of the algorithm. Both interior-point methods in KNITRO choose ν at every iteration so that the decrease in a quadratic model of the merit function produced by a step d is proportional to the product of ν times the decrease in linearized constraints.

To be more specific, suppose that either the INTERIOR/DIRECT or INTERIOR/CG algorithm has produced a step d. We define the following linear/quadratic model of the merit function ϕ_ν:

$$Q_\nu(d) = \nabla f^T d_x - \mu e^T S^{-1} d_s + \frac{\sigma}{2}\left(d_x^T \nabla_{xx}^2 \mathcal{L} d_x + d_s^T \Sigma d_s\right) + \nu\left(m(0) - m(d)\right), \tag{3.15}$$

where

$$m(d) = \left\| \begin{bmatrix} A_E d_x + c_E \\ A_I d_x - d_s + c_I - s \end{bmatrix} \right\|_2,$$

denotes the first-order violation of the constraints, and σ is a parameter to be discussed below. We also define the predicted decrease in the merit function as

$$\text{pred}(d) = Q_\nu(0) - Q_\nu(d). \tag{3.16}$$

In all cases we choose the penalty parameter ν large enough such that

$$\text{pred}(d) \geq \rho\nu[m(0) - m(d)], \tag{3.17}$$

for some parameter $0 < \rho < 1$ (e.g. $\rho = 0.1$). If the value of ν from the previous iteration satisfies (3.17), it is left unchanged, otherwise ν is increased so that it satisfies this inequality with some margin. Condition (3.17) is standard for trust region methods, but not for line search methods, where it may require ν to be larger than is needed to simply provide a descent direction. As shown

in [38] this stronger condition can improve performance of the line search iteration.

For a trust region method, such as that implemented in INTERIOR/CG, we set $\sigma = 1$ in (3.15) because these methods can deal well with indefiniteness of the Hessian. A line search method, on the other hand, does not always produce a descent direction for the merit function if the model on which it is based is not convex. Therefore in the INTERIOR/DIRECT algorithm we define σ as

$$\sigma = \begin{cases} 1 \text{ if } d_x^T \nabla_{xx}^2 \mathcal{L} d_x + d_s^T \Sigma d_s > 0 \\ 0 \text{ otherwise.} \end{cases} \tag{3.18}$$

This choice of σ guarantees the directional derivative of ϕ_ν in the direction d is negative.

4 Active-set Sequential Linear-Quadratic Programming

The active-set method implemented in KNITRO does not follow an SQP approach because, in our view, the cost of solving generally constrained quadratic programming subproblems imposes a limitation on the size of problems that can be solved in practice. In addition, the incorporation of second derivative information in SQP methods has proved to be difficult.

We use, instead a sequential linear-quadratic programming (SLQP) method [5,9,16] that computes a step in two stages, each of which scales up well with the number of variables. First, a linear program (LP) is solved to identify a working set. This is followed by an equality constrained quadratic programming (EQP) phase in which the constraints in the working set $\mathcal{W}$ are imposed as equalities. The total step of the algorithm is a combination of the steps obtained in the linear programming and equality constrained phases.

To achieve progress on both feasibility and optimality, the algorithm is designed to reduce the ℓ_1 penalty function,

$$P(x;\nu) = f(x) + \nu \sum_{i \in \mathcal{E}} |c_i(x)| + \nu \sum_{i \in \mathcal{I}} (\max(0, -c_i(x)), \tag{4.1}$$

where c_i, $i \in \mathcal{E}$, denote the components of the vector c_E, and similarly for c_I. The penalty parameter ν is chosen by an adaptive procedure described below.

An appealing feature of the SLQP algorithm is that established techniques for solving large-scale versions of the LP and EQP subproblems are readily available. Modern LP software is capable of solving problems with more than a million variables and constraints, and the solution of an EQP can be performed efficiently using the projected conjugate gradient iteration discussed in Section 5. We now outline the SLQP approach implemented in KNITRO-ACTIVE. This algorithm is an implementation of the algorithm SLIQUE described in [5].

4.1 Algorithm III: KNITRO-ACTIVE

In the LP phase, given an estimate x_k of the solution of the nonlinear program (1.1), we would like to solve

$$\min_d \nabla f(x_k)^T d \tag{4.2a}$$

$$\text{subject to } c_i(x_k) + \nabla c_i(x_k)^T d = 0, \quad i \in \mathcal{E} \tag{4.2b}$$

$$c_i(x_k) + \nabla c_i(x_k)^T d \geq 0, \quad i \in \mathcal{I} \tag{4.2c}$$

$$\|d\|_\infty \leq \Delta_k^{\text{LP}}, \tag{4.2d}$$

with $\Delta_k^{\text{LP}} > 0$. (Note that (4.2) differs from the subproblem used in SQP methods only in that the latter include a term of the form $\frac{1}{2} d^T H d$ in (4.2*a*), where H is an approximation to the Hessian of the Lagrangian of the nonlinear program.) Since the constraints of (4.2) may be inconsistent, we solve instead the ℓ_1 penalty reformulation of (4.2) given by

$$\min_d l_\nu(d) \stackrel{\text{def}}{=} \nabla f(x_k)^T d + \nu_k \sum_{i \in \mathcal{E}} |c_i(x_k) + \nabla c_i(x_k)^T d|$$

$$+ \nu_k \sum_{i \in \mathcal{I}} \max(0, -c_i(x_k) - \nabla c_i(x_k)^T d) \tag{4.3a}$$

$$\text{subject to } \|d\|_\infty \leq \Delta_k^{\text{LP}}. \tag{4.3b}$$

The solution of this linear program, which we denote by d^{LP}, is computed by the simplex method so as to obtain an accurate estimate of the optimal active set.

Based on this solution, we define the working set $\mathcal{W}$ as some linearly independent subset of the active set $\mathcal{A}$ at the LP solution, which is defined as

$$\mathcal{A}(d^{\text{LP}}) = \{i \in \mathcal{E} | c_i(x_k) + \nabla c_i(x_k)^T d^{\text{LP}} = 0\} \cup \{i \in \mathcal{I} | c_i(x_k) + \nabla c_i(x_k)^T d^{\text{LP}} = 0\}.$$

Likewise, we define the set $\mathcal{V}$ of violated constraints as

$$\mathcal{V}(d^{\text{LP}}) = \{i \in \mathcal{E} | c_i(x_k) + \nabla c_i(x_k)^T d^{\text{LP}} \neq 0\} \cup \{i \in \mathcal{I} | c_i(x_k) + \nabla c_i(x_k)^T d^{\text{LP}} < 0\}.$$

To ensure that the algorithm makes progress on the penalty function P, we define the *Cauchy step*,

$$d^{\text{C}} = \alpha^{\text{LP}} d^{\text{LP}}, \tag{4.4}$$

where $\alpha^{\text{LP}} \in (0, 1]$ is a steplength that provides sufficient decrease in the following (piecewise) quadratic model of the penalty function $P(x; \nu)$:

$$q_k(d) = l_\nu(d) + \frac{1}{2} d^T H(x_k, \lambda_k) d. \tag{4.5}$$

Here H is the Hessian of the Lagrangian or an approximation to it, and $l_\nu(d)$ is defined in (4.3*a*).

Given the working set $\mathcal{W}_k$, we now solve an equality constrained quadratic program (EQP) treating the constraints in $\mathcal{W}_k$ as equalities and ignoring all other constraints. This gives the subproblem

$$\min_d \frac{1}{2} d^T H(x_k, \lambda_k) d + \left(\nabla f(x_k) + \nu_k \sum_{i \in \mathcal{V}} \gamma_i \nabla c_i(x_k) \right)^T d \quad (4.6a)$$

$$\text{subject to} \quad c_i(x_k) + \nabla c_i(x_k)^T d = 0, \quad i \in \mathcal{E} \cap \mathcal{W}_k \quad (4.6b)$$

$$c_i(x_k) + \nabla c_i(x_k)^T d = 0, \quad i \in \mathcal{I} \cap \mathcal{W}_k \quad (4.6c)$$

$$\|d\|_2 \leq \Delta_k, \quad (4.6d)$$

where γ_i is the algebraic sign of the violated i-th constraint. Note that the trust region (4.6d) is spherical, and is distinct from the trust region Δ^{LP} used in (4.2d). Problem (4.6) is solved for the vector d^{Q} by applying the projected conjugated gradient procedure described in Section 5. The total step d of the SLQP method is given by

$$d = d^{\text{C}} + \alpha^{\text{Q}}(d^{\text{Q}} - d^{\text{C}}),$$

where $\alpha^{\text{Q}} \in [0, 1]$ is a steplength that approximately minimizes the model function (4.5).

Algorithm 4.1: KNITRO-ACTIVE

Initial data: x_0, $\Delta_0 > 0$, $\Delta_0^{\text{LP}} > 0$, $0 < \eta < 1$. Set $k = 0$.

Repeat until a stopping test for the nonlinear program (1.1) is satisfied:

LP point. Update the penalty parameter ν_k and solve the LP (4.3) to obtain the step d_k^{LP}, and working set $\mathcal{W}_k$.

Cauchy point. Compute $\alpha_k^{\text{LP}} \in (0, 1]$ as an approximate minimizer of $q(\alpha d_k^{\text{LP}})$ such that $\alpha_k^{\text{LP}} \|d_k^{\text{LP}}\| \leq \Delta_k$. Set $d_k^{\text{C}} = \alpha_k^{\text{LP}} d_k^{\text{LP}}$.

EQP point. Compute d_k^{Q} by solving the EQP (4.6).
Define $d_k^{\text{CE}} = d_k^{\text{Q}} - d_k^{\text{C}}$ as the segment leading from the Cauchy point to the EQP point.

Trial point. Compute $\alpha_k^{\text{Q}} \in [0, 1]$ as an approximate minimizer of $q(d_k^{\text{C}} + \alpha d_k^{\text{CE}})$. Set $d_k = d_k^{\text{C}} + \alpha_k^{\text{Q}} d_k^{\text{CE}}$ and $x_{\text{T}} = x_k + d_k$.

Step Acceptance. Compute
$\rho_k = (P(x_k; \nu_k) - P(x_{\text{T}}; \nu_k))/(q_k(0) - q_k(d_k))$.
If $\rho_k \geq \eta$, set $x_{k+1} \leftarrow x_{\text{T}}$, otherwise set $x_{k+1} \leftarrow x_k$.
Update Δ_{k+1}^{LP} and Δ_{k+1}. Set $k \leftarrow k + 1$.

End

The trust region radius Δ_k for the EQP phase is updated based on the ratio ρ_k, following standard trust region update strategies. The choice of Δ_{k+1}^{LP}

is based on an effort to generate a good working set. In our implementation, Δ_{k+1}^{LP} is set to be a little larger than the total step d_k, subject to some other restrictions, as described in [5]. The multiplier estimates λ_k used in the Hessian are least squares estimates using the working set $\mathcal{W}_k$, and modified so that $\lambda_i \geq 0$ for $i \in \mathcal{I}$.

Penalty Parameter Update Strategy.

A novel feature of our SLQP algorithm is the procedure for updating the penalty parameter. Unlike most strategies proposed in the literature [11], which hold the penalty parameter ν fixed for a series of iterations and only update it if insufficient progress toward feasibility is made, our algorithm chooses an appropriate value of ν at each iteration. This selection takes place during the linear programming phase, as we now explain.

We define a (piecewise) linear model of constraint violation at a point x_k by

$$m_k(d) = \sum_{i \in \mathcal{E}} |c_i(x_k) + \nabla c_i(x_k)^T d| + \sum_{i \in \mathcal{I}} \max(0, -c_i(x_k) - \nabla c_i(x_k)^T d), \quad (4.7)$$

so that the objective (4.3) of the LP subproblem can be written as

$$l_\nu(d) = \nabla f(x_k)^T d + \nu_k \, m_k(d). \quad (4.8)$$

Given a value ν_k, we write the solution of the LP problem (4.3) as $d^{\text{LP}}(\nu_k)$ to stress its dependence on the penalty parameter. Likewise, $d^{\text{LP}}(\nu_\infty)$ denotes the minimizer of $m_k(d)$ subject to the trust region constraint (4.3*b*). The following algorithm describes the computation of the LP step d_k^{LP} and the penalty parameter ν_k.

Algorithm Penalty Update. LP Step and Penalty Update Strategy.
Initial data: $x_k, \nu_{k-1} > 0$, $\Delta_k^{\text{LP}} > 0$, and parameters $\epsilon_1, \epsilon_2 \in (0,1)$.
Solve the subproblem (4.3) with $\nu = \nu_{k-1}$ to obtain $d^{\text{LP}}(\nu_{k-1})$.
If $m_k(d^{\text{LP}}(\nu_{k-1})) = 0$
  Set $\nu^+ \leftarrow \nu_{k-1}$.
Else compute $d^{\text{LP}}(\nu_\infty)$
  If $m_k(d^{\text{LP}}(\nu_\infty)) = 0$
    Find $\nu^+ > \nu_{k-1}$ such that $m_k(d^{\text{LP}}(\nu^+)) = 0$.
  Else
    Find $\nu^+ \geq \nu_{k-1}$ such that
      $m_k(0) - m_k(d^{\text{LP}}(\nu^+)) \geq \epsilon_1 [m_k(0) - m_k(d^{\text{LP}}(\nu_\infty))]$.
  Endif
Endif
Increase ν^+ if necessary to satisfy
    $l_{\nu^+}(0) - l_{\nu^+}(d^{\text{LP}}(\nu^+)) \geq \epsilon_2 \nu^+ [m_k(0) - m_k(d^{\text{LP}}(\nu^+))]$.
Set $\nu_k \leftarrow \nu^+$ and $d_k^{\text{LP}} \leftarrow d^{\text{LP}}(\nu^+)$.

The selection of $\nu^+ > \nu_{k-1}$ is achieved in all cases by successively increasing the current trial value of ν by 10 and re-solving the linear program. The penalty update algorithm above guarantees that ν is chosen large enough to ensure convergence to a stationary point [4]. Although the procedure does require the solution of some additional linear programs, our experience is that it results in an overall savings in iterations (and total LP solves) by achieving a better penalty parameter value more quickly, compared with rules which update the penalty parameter based on monitoring progress in feasibility. In addition, the extra LP solves are typically very inexpensive requiring relatively few simplex iterations because of the effectiveness of warm starts when re-solving the LP with a different penalty parameter value.

5 Projected CG Iteration

One of the main modules shared by the algorithms implemented in KNITRO, is a projected conjugate gradient iteration. The tangential subproblem (3.12) in the INTERIOR/CG algorithm and the EQP phase (4.6) of the ACTIVE algorithm both require the solution of an equality constrained quadratic program. We solve these problems using a projected conjugate gradient iteration [10, 20, 24, 26, 32], which is well suited for large problems and can handle the negative curvature case without the need for Hessian modifications. We now outline this iteration and refer the reader to [20] for a more detailed derivation.

Consider the quadratic program

$$\min_x \frac{1}{2} x^T G x + h^T x \tag{5.9a}$$

$$\text{subject to } Ax = b, \tag{5.9b}$$

and assume that G is positive definite on the null space of A. One way to solve (5.9) is to eliminate the constraints (5.9*b*) and apply the conjugate gradient method to the reduced problem. An equivalent strategy is to apply a special form of the CG iteration to the KKT system of (5.9), which is given by

$$\begin{bmatrix} G & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix} = \begin{bmatrix} -h \\ b \end{bmatrix}. \tag{5.10}$$

Although the coefficient matrix is not positive definite, we can apply the CG method to (5.10), provided we precondition and project the CG method so that it effectively solves the positive definite reduced problem within the feasible manifold (5.9*b*). This algorithm is specified below. Here we denote the preconditioning/projection operator by P and give its precise definition later on.

Algorithm PCG. Preconditioned Projected CG Method.
Choose an initial point x_0 satisfying $Ax_0 = b$. Set $x \leftarrow x_0$, compute $r = Gx+h$, $z = Pr$ and $p = -z$.
Repeat the following steps, until $\|z\|$ is smaller than a given tolerance:

$$\alpha = r^T z / p^T G p \tag{5.11}$$
$$x \leftarrow x + \alpha p \tag{5.12}$$
$$r^+ = r + \alpha G p \tag{5.13}$$
$$z^+ = P r^+ \tag{5.14}$$
$$\beta = (r^+)^T z^+ / r^T z \tag{5.15}$$
$$p \leftarrow -z^+ + \beta p. \tag{5.16}$$
$$z \leftarrow z^+ \quad \text{and} \quad r \leftarrow r^+ \tag{5.17}$$

End

This iteration has exactly the same form as the (standard) preconditioned CG method for solving symmetric and positive definite systems; see e.g. [19]. The crucial difference is that normally P is a symmetric and positive definite matrix, whereas in our case it represents a projection and preconditioning matrix, which we define (indirectly) as follows. Given a vector r, we compute $z = Pr$ as the solution of the system

$$\begin{bmatrix} D & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} z \\ w \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}, \tag{5.18}$$

where D is a symmetric matrix that is positive definite on the null space of A, and w is an auxiliary vector. For (5.18) to be a practical preconditioning operation, D should be a sparse matrix, so that solving (5.18) is significantly less costly than solving (5.10).

By construction $z = Pr$ is in the null space of A, and so are all the search directions generated by Algorithm PCG. Since initially $Ax_0 = b$, all subsequent iterates x also satisfy the linear constraints. To view this iteration relative to the reduced CG method in which we eliminate the constraints (5.9*b*) and apply CG to a problem of dimension $n - l$, note that all iterates of Algorithm PCG may be expressed as $x = x_0 + Zu$, for some vector $u \in R^{n-l}$, and where the columns of the $n \times (n - l)$ matrix Z form a basis for the null space of A. In these null-space coordinates the solution of the quadratic program (5.9) is given by the vector u that solves

$$(Z^T G Z)u = Z^T(Gx_0 + h). \tag{5.19}$$

It can be shown (see e.g. [20]) that the iterates x generated by Algorithm PCG are given by $x = x_0 + Zu$, where u are the iterates of the preconditioned conjugate gradient method on the system (5.19), using the matrix $Z^T D Z$ as a preconditioner. Therefore, Algorithm PCG is a standard preconditioned CG iteration as long as G and D are positive definite on the null space of A.

There are two advantages of following the approach of Algorithm PCG over the reduced CG approach. First, there is no need to compute a null space basis and consequently no risk that ill-conditioning in Z will deteriorate the rate of convergence of the CG iteration. Moreover, in the INTERIOR/CG algorithm we first scale the slack variables by (3.13), so that the matrix A in (5.9) has the form

$$\begin{bmatrix} A_E & 0 \\ A_I & -S \end{bmatrix}. \tag{5.20}$$

Therefore there is no ill conditioning caused by some slack variables approaching 0. The second benefit is that the projection matrix in (5.18) can also be used to compute the normal step and Lagrange multipliers; thus the extra cost of each of these computations is only one back solve involving the factors of this projection matrix.

In the INTERIOR/CG and ACTIVE algorithms we solve quadratic programs of the form (5.9) subject to a trust region constraint $\|x\| \leq \Delta$; in addition, G may not be positive definite on the null space of A. We adapt Algorithm PCG to this case by following Steihaug's approach: we terminate Algorithm PCG if the trust region is crossed or if negative curvature is encountered.

KNITRO 5.0 sets $D = I$ in (5.18) so that the preconditioner removes only ill-conditioning associated with the constraint matrix A. (We have experimented with other choices of D and future releases of KNITRO will include banded and incomplete Cholesky preconditioners.)

Algorithm PCG assumes that an initial feasible point x_0 is provided. The factorization of the system in (5.18) allows us to compute such a point by solving

$$\begin{bmatrix} D & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} w \\ x_0 \end{bmatrix} = \begin{bmatrix} 0 \\ b \end{bmatrix},$$

which is in fact the minimum-norm solution in the norm weighted by D.

6 Special Algorithmic Features

The KNITRO package provides many algorithmic options and features that are listed comprehensibly in the documentation that accompanies the software [37]. Here we highlight some of these options and discuss their relationship to the algorithms presented in the previous sections.

Hessian Options

The user can supply first and second derivatives, which generally results in the greatest level of robustness and efficiency for the three algorithms in KNITRO. In some applications, however, the Hessian of the Lagrangian $\nabla^2_{xx}\mathcal{L}$ cannot be computed or is too large to store, but products of this Hessian times vectors can be obtained through automatic differentiation tools, adjoint codes or user-provided routines. For this case the INTERIOR/CG and ACTIVE

algorithms allow the user to provide these Hessian vector products at every iteration of the projected CG iteration. In a related option, KNITRO takes control of this process and approximates the Hessian-vector products by finite differences of gradients of the Lagrangian; in this case the user need only provide gradients of the objective and constraints.

Quasi-Newton options have also been implemented for the three algorithms in KNITRO. Here, the Hessian of the Lagrangian $\nabla^2_{xx}\mathcal{L}$ is replaced by a quasi-Newton approximation B_k, which is updated by the BFGS, SR1 or limited memory BFGS formulae. For example, for the interior-point methods, we define

$$\Delta l = \nabla_x \mathcal{L}(x^+, s^+, y^+, z^+) - \nabla_x \mathcal{L}(x, s^+, y^+, z^+), \qquad \Delta x = x^+ - x,$$

and substitute the correction pairs $(\Delta l, \Delta x)$ in the standard definition of the BFGS, SR1 or limited memory BFGS update formulae (see e.g. [31]). To ensure positive definiteness of the BFGS and L-BFGS updates the vector Δl is modified, if necessary, using Powell's damping procedure. SR1 updating is safeguarded to avoid unboundedness, but is allowed to generate indefinite approximations.

Feasible Mode.

In some applications, it is desirable for all of the iterates generated by the optimization algorithm to be feasible with respect to some or all of the *inequality* constraints. For example, the objective function may be defined only when some of the constraints are satisfied, making this feature absolutely necessary. Interior-point methods provide a natural framework for deriving feasible algorithms, and we have therefore developed versions of the INTERIOR/CG and INTERIOR/DIRECT algorithms that have this feature.

The adaptation is simple. If the current iterate x satisfies $c_I(x) > 0$, then after computing the step d, we let $x^+ = x + d_x$, redefine the slacks as

$$s^+ \leftarrow c_I(x^+), \tag{6.21}$$

and test whether the point (x^+, s^+) is acceptable for the merit function ϕ_ν. If so, we define this point to be the new iterate; otherwise we reject the step d and compute a new, shorter, trial step (in a line search algorithm we backtrack, and in a trust-region method we compute a new step with a smaller trust region). This strategy is justified by the fact that, if at a trial point we have that $c_i(x^+) \leq 0$ for some inequality constraint, the value of the merit function is $+\infty$, and we reject the trial point. This strategy also rejects steps $x + d_x$ that are too close to the boundary of the feasible region because such steps increase the barrier term $-\mu \sum_{i=1}^m \log(s_i)$ in the merit function (3.3). Apart from the reset (6.21), in the INTERIOR/CG algorithm we must introduce a slight modification [8] in the normal step computation to ensure that this step makes sufficient progress toward feasibility.

Initial Point Strategy.

As is well known, interior methods can perform poorly if the initial point is unfavorable. To overcome this problem, we have implemented several initial point strategies that work well for linear and quadratic programming and are also appropriate for nonlinear programs. At present, the initial point strategies are available only in the INTERIOR/DIRECT option. We now describe one of these strategies.

We first compute, at the user supplied initial point x_0, an affine scaling step $d^A = (d_x^A, d_s^A, d_y^A, d_z^A)$ by setting $\mu = 0$ in (3.4). Then we define

$$s_1 = \max(1, |s_0 + d_s^A|), \quad z_1 = \max(1, |z_0 + d_z^A|),$$

where the max and absolute values are applied component-wise. The primal variables x and the equality-constraint multipliers y are not altered, i.e., we define $(x_1, y_1) = (x_0, y_0)$. Finally we define the initial value of the barrier parameter as $\mu_1 = s_1^T z_1/m$.

The motivation for this strategy is to take care that the initial slacks and inequality multipliers are not too close to the feasible boundary which can lead to slow progress, and ideally to generate an initial point nearer to the central path. Furthermore, nonlinear programming algorithms compute only local minimizers and accept user-supplied initial estimates x_0 that often lie in the vicinity of a minimizer of interest. Therefore initial point strategies should either respect the user-supplied estimate x_0 or compute one that is not too distant from it. In addition, large initial values of the multipliers should be avoided in the Hessian since they may introduce unnecessary non-convexities in the problem. In particular if one of the components, say z_1^i, is large and the corresponding Hessian term $\nabla^2 c_1(x_1)$ is indefinite, the Hessian of the Lagrangian can become indefinite, slowing down the iteration. Therefore, when computing the first step of the interior algorithm from (x_1, s_1, y_1, z_1) we evaluate the Hessian of the Lagrangian using z_0 and not z_1, i.e., $\nabla^2_{xx}\mathcal{L}(x_0, s_0, y_0, z_0)$ (this Hessian is independent of s, so the choice of that variable is irrelevant). More details about the initial point strategies are given in [17].

Special Problem Classes

When the nonlinear program (1.1) has a special form, the algorithms in KNITRO often reduce to well-known special-purpose methods.

For unconstrained optimization problems, the INTERIOR/CG and ACTIVE algorithms (using second derivatives) reduce to an inexact Newton-CG method with trust regions. This is because, in the unconstrained case, these algorithms skip their respective first phases, and compute the step using, respectively, the tangential subproblem (3.12) and the EQP phase (4.6), which are identical in this case. In the INTERIOR/DIRECT option, the algorithm will attempt to compute the Cholesky factorization of the Hessian, and if it is positive definite a backtracking line search will be performed along the Newton direction. If the Hessian is not positive definite, the algorithm reverts to

the trust region INTERIOR/CG algorithm and therefore computes an inexact Newton-CG step.

If the problem (1.1) is a system of nonlinear equations, the algorithms in KNITRO implement a form of Newton's method (if second derivatives are provided). In INTERIOR/CG, only the normal step (3.11) is computed, and the resulting algorithm coincides with the Levenberg-Marquardt trust region method. The INTERIOR/DIRECT algorithm reduces to a line search Newton method in this case, using as a merit function the Euclidean norm of the residuals of the system of equations. If the Jacobian is singular, INTERIOR/DIRECT reverts to the Levenberg-Marquardt method.

KNITRO adapts itself automatically to the two classes of problems just discussed (unconstrained minimization and nonlinear equations). If the problem is a linear or quadratic program, the user must inform KNITRO, so that the algorithms can take full advantage of this fact. For LPs or QPs, INTERIOR/DIRECT is the recommended interior-point option and automatically enables the initial point strategy described above, as well as a more aggressive barrier update strategy. ACTIVE reduces to a simplex method in the LP case.

Infeasibility detection

It is not rare for users to generate optimization problems that do not have a feasible solution, and KNITRO includes heuristics to attempt to diagnose this situation. As is well known, however, infeasibility detection is a very difficult problem for nonlinear constraints, and the algorithms in KNITRO cannot distinguish between infeasible problems and convergence to an (infeasible) stationary point for a measure of feasibility.

In the interior point algorithms, our heuristics are based on the theory developed in [3]. It states that, if the interior point algorithm is not capable of finding a feasible point, then we have that $A_E(x_k)^T c_E(x_k) \to 0$, and $A_I(x_k)^T c_I^-(x_k) \to 0$, where $c_I^- = \max(0, -c_I)$. The KNITRO interior-point algorithms will terminate if these vectors are sufficiently small while $\|(c_E(x_k), c_I^-(x_k))\|$ stays above some level.

Since the algorithm implemented in ACTIVE is a penalty method, it can deal naturally with infeasibility. If a problem is infeasible then the penalty parameter will be driven to infinity. Moreover, if the algorithm is converging to a stationary point for our infeasibility measure, we have

$$m_k(0) - m_k(d^{\mathrm{LP}}(\nu_\infty)) \to 0,$$

during the penalty update procedure providing a clear indication of local infeasibility.

7 Crossover

Interior methods provide only an approximate estimate of the solution and the optimal active set. In many practical applications, however, it is useful to

know precisely which constraints are active because this corresponds to the presence or activity of certain constituents of the solution. In addition, it is often important to have accurate estimates of the Lagrange multipliers (or sensitivities). This can be done by switching from the interior to an active-set iteration, a process that is often called *crossover*. Although crossover techniques have received much attention in the context of linear programming [27], to the best of our knowledge, none of the nonlinear interior codes provide an option for it. We regard it as essential to have this facility in our integrated system, both for computational efficiency, and to return solutions in a form that is useful for applications.

In linear programming, crossover involves two stages: identifying active constraints, and moving from a nonbasic optimal solution to a nearby basic one. In nonlinear programming, of course, we cannot expect the set of active constraints to correspond to a basic solution. Instead, our crossover procedure seeks to identify a set of active constraints with linearly independent constraint gradients, and computes a solution at which those constraints are satisfied with near equality, and which satisfies Lagrangian stationarity using these constraints only.

This crossover procedure is implemented by internally switching to the ACTIVE algorithm after the INTERIOR/DIRECT or INTERIOR/CG algorithm has solved the problem to the requested tolerance. We first solve the EQP phase of ACTIVE using a tolerance-based active-set estimate, and minimize the model function (4.5) along the resulting step direction to generate a new solution estimate. If this step does not solve the problem immediately, we begin the full ACTIVE algorithm with an initial LP trust region based on that active-set estimate. The goal is to judiciously choose the initial LP trust region small enough to exclude all the inactive constraints, but large enough to include the active ones. Below is a basic description of the KNITRO crossover procedure.

Algorithm Crossover. KNITRO Crossover Procedure.

1. The interior-point iteration terminates with stopping tolerance ϵ_{TOL} at iterate (x_k, s_k, y_k, z_k).
2. Estimate the set of active constraints, $\mathcal{A}$, using a tolerance test based on primal-dual feasibility and complementarity.
3. Using this active-set estimate, generate a step by solving the EQP given by (4.6) for d^{Q} and perform a line search to compute the steplength α^{Q}. If $x_k + \alpha^{\text{Q}} d^{\text{Q}}$ satisfies the stopping tolerances, terminate with that value and the corresponding multipliers.
4. Otherwise determine the initial LP trust region Δ_0^{LP}, and penalty parameter ν_0 for the KNITRO-ACTIVE algorithm (Algorithm 4.1):

$$\Delta_0^{\text{LP}} = \min\{\frac{c_i(x_k, s_k)}{\|\nabla c_i(x_k, s_k)\|} : i \notin \mathcal{A}\}, \tag{7.22}$$

$$\nu_0 = 10\|(y_k, z_k)\|_\infty. \tag{7.23}$$

5. Start KNITRO-ACTIVE using initial point (x_k, s_k, y_k, z_k), Δ_0^{LP} and ν_0.

Initially in Step 3 of crossover, the active set is estimated using a tolerance test rather than by solving the LP (4.3). This is because, on some difficult problems, the cost of solving the LP subproblem can be non-trivial and we would like the cost of our crossover procedure in most cases to be a small part of the overall solution time. Therefore, if it is not necessary to solve the LP to identify the optimal active set, we seek to avoid doing this. In many cases, especially if strict complementarity holds at the solution, the initial estimate of the active set based on the simple tolerance test will be correct and the crossover will succeed in one iteration without solving any LPs.

The condition (7.22) used to initialize the initial LP trust region Δ^{LP} guarantees that if our active-set estimate is correct, the initial LP trust region will be small enough to exclude all inactive constraints. Motivated by the theory for the ℓ_1 exact penalty function, the penalty parameter ν is initialized to be a little larger than the Lagrange multiplier of largest magnitude at the interior-point solution.

Acknowledgment. The authors are grateful to several collaborators who have contributed to the development of KNITRO. Todd Plantenga and Mary Beth Hribar developed the software modules that led to the first version of the INTERIOR/CG algorithm in KNITRO. Guanghui Liu, Marcelo Marazzi, José Luis Morales and Dominique Orban contributed algorithmic improvements and performed extensive testing. Nick Gould collaborated with us on the development of the SLQP algorithm and its convergence analysis, and Jean-Charles Gilbert studied with us the convergence properties of the INTERIOR/CG algorithm.

References

1. E.D. Andersen and K.D. Andersen. The MOSEK interior point optimizer for linear programming: an implementation of the homogeneous algorithm. In H. Frenk, K. Roos, T. Terlaky, and S. Zhang, editors, *High Performance Optimization*, pages 197–232, Dordrecht, The Netherlands, 2000. Kluwer Academic Publishers.
2. J. Betts, S. K. Eldersveld, P. D. Frank, and J. G. Lewis. An interior-point nonlinear programming algorithm for large scale optimization. Technical report MCT TECH-003, Mathematics and Computing Technology, The Boeing Company, P.O. Box 3707, Seattle WA 98124-2207, 2000.
3. R. H. Byrd, J.-Ch. Gilbert, and J. Nocedal. A trust region method based on interior point techniques for nonlinear programming. *Mathematical Programming*, 89(1):149–185, 2000.
4. R. H. Byrd, N. I. M. Gould, J. Nocedal, and R. A. Waltz. On the convergence of successive linear-quadratic programming algorithms. Technical Report OTC 2002/5, Optimization Technology Center, Northwestern University, Evanston, IL, USA, 2002.

5. R. H. Byrd, N. I. M. Gould, J. Nocedal, and R. A. Waltz. An algorithm for nonlinear optimization using linear programming and equality constrained subproblems. *Mathematical Programming, Series B*, 100(1):27–48, 2004.
6. R. H. Byrd, M. E. Hribar, and J. Nocedal. An interior point algorithm for large scale nonlinear programming. *SIAM Journal on Optimization*, 9(4):877–900, 1999.
7. R. H. Byrd, M. Marazzi, and J. Nocedal. On the convergence of Newton iterations to non-stationary points. *Mathematical Programming, Series A*, 99:127–148, 2004.
8. R. H. Byrd, J. Nocedal, and R. A. Waltz. Feasible interior methods using slacks for nonlinear optimization. *Computational Optimization and Applications*, 26(1):35–61, 2003.
9. C. M. Chin and R. Fletcher. On the global convergence of an SLP-filter algorithm that takes EQP steps. *Mathematical Programming, Series A*, 96(1):161–177, 2003.
10. T. F. Coleman. Linearly constrained optimization and projected preconditioned conjugate gradients. In J. Lewis, editor, *Proceedings of the Fifth SIAM Conference on Applied Linear Algebra*, pages 118–122, Philadelphia, USA, 1994. SIAM.
11. A. R. Conn, N. I. M. Gould, and Ph. Toint. *Trust-region methods*. MPS-SIAM Series on Optimization. SIAM publications, Philadelphia, Pennsylvania, USA, 2000.
12. A. R. Conn, N. I. M. Gould, and Ph. L. Toint. LANCELOT*: a Fortran package for Large-scale Nonlinear Optimization (Release A)*. Springer Series in Computational Mathematics. Springer Verlag, Heidelberg, Berlin, New York, 1992.
13. E. D. Dolan, J. J. Moré, and T. S. Munson. Optimality measures for performance profiles. Technical report, Argonne National Laboratory, Argonne, IL, USA, 2004.
14. A. Drud. CONOPT: A GRG code for large sparse dynamic nonlinear optimization problems. *Mathematical Programming*, 31:153–191, 1985.
15. R. Fletcher and S. Leyffer. User manual for filtersqp. Technical Report NA/181, Dundee, Scotland, 1998.
16. R. Fletcher and E. Sainz de la Maza. Nonlinear programming and nonsmooth optimization by successive linear programming. *Mathematical Programming*, 43(3):235–256, 1989.
17. E. Michael Gertz, J. Nocedal, and A. Sartenaer. A starting-point strategy for nonlinear interior methods. *Applied Math Letters*, 5, 2004.
18. P. E. Gill, W. Murray, and M. A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Journal on Optimization*, 12:979–1006, 2002.
19. G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, second edition, 1989.
20. N. I. M. Gould, M. E. Hribar, and J. Nocedal. On the solution of equality constrained quadratic problems arising in optimization. *SIAM Journal on Scientific Computing*, 23(4):1375–1394, 2001.
21. Christelle Guéret, Christian Prins, and Marc Sevaux. *Applications of optimization with Xpress-MP*. Dash Optimization, 2002.
22. C. Gurwitz and M. L. Overton. Sequential quadratic programming methods based on approximating a projected hessian matrix. *SIAM Journal on Scientific and Statistical Computing*, 10(4):631–653, 1989.

23. ILOG CPLEX 8.0. *User's Manual.* ILOG SA, Gentilly, France, 2002.
24. C. Keller, N. I. M. Gould, and A. J. Wathen. Constraint preconditioning for indefinite linear systems. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1300–1317, 2000.
25. M. Kocvara. Results of NLP problems: performance profiles, 2003. `http://www2.am.uni-erlangen.de/~kocvara/pennon/ampl-nlp-pp.html`.
26. L. Lukšan and J. Vlček. Indefinitely preconditioned inexact Newton method for large sparse equality constrained nonlinear programming problems. *Numerical Linear Algebra with Applications*, 5(3):219–247, 1998.
27. N. Megiddo. Pathways to the optimal set in linear programming. In N. Megiddo, editor, *Progress in mathematical programming: Interior-point and related methods*, chapter 8, pages 131–158. Springer-Verlag, New York, 1989.
28. J. L. Morales, J. Nocedal, R. A. Waltz, G. Liu, and J. P. Goux. Assessing the potential of interior methods for nonlinear optimization. In L. T. Biegler, O. Ghattas, M. Heinkenschloss, and B. van Bloemen Waanders, editors, *Large-Scale PDE-Constrained Optimization*, volume 30, pages 167–183, Heidelberg, Berlin, New York, 2003. Springer Verlag. Lecture Notes in Computational Science and Engineering.
29. B. A. Murtagh and M. A. Saunders. MINOS 5.4 user's guide. Technical report, SOL 83-20R, Systems Optimization Laboratory, Stanford University, 1983. Revised 1995.
30. J. Nocedal, A. Wächter, and R. A. Waltz. Adaptive barrier strategies for nonlinear interior methods. Technical Report RC 23563, IBM Watson Research Center, Yorktown Heights, NY, USA, 2005.
31. J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer Series in Operations Research. Springer, 1999.
32. B. T. Polyak. The conjugate gradient method in extremal problems. *U.S.S.R. Computational Mathematics and Mathematical Physics*, 9:94–112, 1969.
33. R. J. Vanderbei and D. F. Shanno. An interior point algorithm for nonconvex nonlinear programming. *Computational Optimization and Applications*, 13:231–252, 1999.
34. A. Wächter. *An interior point algorithm for large-scale nonlinear optimization with applications in process engineering.* PhD thesis, Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA, 2002.
35. A. Wächter and L. T. Biegler. Failure of global convergence for a class of interior point methods for nonlinear programming. *Mathematical Programming*, 88(3):565–574, 2000.
36. A. Wächter and L. T. Biegler. On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. Technical Report RC 23149, IBM T. J. Watson Research Center, Yorktown Heights, NY, USA, March 2004.
37. R. A. Waltz. KNITRO 4.0 User's Manual. Technical report, Ziena Optimization, Inc., Evanston, IL, USA, October 2004.
38. R. A. Waltz, J. L. Morales, J. Nocedal, and D. Orban. An interior algorithm for nonlinear optimization that combines line search and trust region steps. Technical Report 2003-6, Optimization Technology Center, Northwestern University, Evanston, IL, USA, June 2003. To appear in Mathematical Programming A.

On implicit-factorization constraint preconditioners

H. Sue Dollar[1], Nicholas I. M. Gould[2], and Andrew J. Wathen[1]

[1] Oxford University Computing Laboratory, Numerical Analysis Group, Wolfson Building, Parks Road, Oxford, OX1 3QD, England.
({Sue.Dollar,Andy.Wathen}@comlab.ox.ac.uk)
[2] Computational Science and Engineering Department, Rutherford Appleton Laboratory, Chilton, Oxfordshire, OX11 0QX, England.
(n.i.m.gould@rl.ac.uk)

Summary. Recently Dollar and Wathen [14] proposed a class of incomplete factorizations for saddle-point problems, based upon earlier work by Schilders [40]. In this paper, we generalize this class of preconditioners, and examine the spectral implications of our approach. Numerical tests indicate the efficacy of our preconditioners.

Key words: saddle-point systems, constraint preconditioners.

1 Introduction

Given a symmetric n by n matrix H and a full-rank m ($\leq n$) by n matrix A, we are interested in solving structured linear systems of equations

$$\begin{pmatrix} H & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}. \tag{1.1}$$

Such "saddle-point" systems arise as stationarity (KKT) conditions for equality-constrained optimization [37, §18.1], in mixed finite-element approximation of elliptic problems [5], including in particular problems of elasticity [38] and incompressible flow [19], as well as other areas.

In this paper, we are particularly interested in solving (1.1) by iterative methods, in which so-called constraint preconditioners [33]

$$K_G = \begin{pmatrix} G & A^T \\ A & 0 \end{pmatrix} \tag{1.2}$$

are used to accelerate the iteration for some suitable symmetric G. In Section 2, we examine the spectral implications of such methods, and consider

how to choose G to give favourable eigenvalue distributions. In Section 3, we then extend ideas by Dollar, Schilders and Wathen [14, 40] to construct "implicit" constraint preconditioners for which we can apply the eigenvalue bounds from Section 2. We demonstrate the effectiveness of such an approach in Section 4 and make broad conclusions in Section 5.

Notation

Let I by the (appropriately-dimensioned) identity matrix. Given a symmetric matrix M with, respectively, m_+, m_- and m_0 positive, negative and zero eigenvalues, we denote its inertia by $\mathrm{In}(M) = (m_+, m_-, m_0)$.

2 Constraint preconditioners

2.1 General considerations

For K_G to be a meaningful preconditioner for certain Krylov-based methods [27], it is vital that its inertia satisfies

$$\mathrm{In}(K_G) = (n, m, 0). \tag{2.1}$$

A key result concerning the use of K_G as a preconditioner is as follows.

Theorem 1. [33, Thm. 2.1] or, for diagonal G, [34, Thm. 3.3]. *Suppose that K_H is the coefficient matrix of (1.1), and N is any (n by $n-m$) basis matrix for the null-space of A. Then $K_G^{-1}K_H$ has $2m$ unit eigenvalues, and the remaining $n-m$ eigenvalues are those of the generalized eigenproblem*

$$N^T H N v = \lambda N^T G N v. \tag{2.2}$$

The eigenvalues of (2.2) are real since (2.1) is equivalent to $N^T G N$ being positive definite [7, 26].

Although we are not expecting or requiring that G (or H) be positive definite, it is well-known that this is often not a significant handicap.

Theorem 2. [1, Cor. 12.9, or 12, for example]. *The inertial requirement (2.1) holds for a given G if and only if there exists a positive semi-definite matrix $\bar{D}$ such that $G + A^T D A$ is positive definite for all D for which $D - \bar{D}$ is positive semi-definite.*

Since any preconditioning system

$$\begin{pmatrix} G & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} r \\ s \end{pmatrix} \tag{2.3}$$

may equivalently be written as

$$\begin{pmatrix} G + A^T D A & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} u \\ w \end{pmatrix} = \begin{pmatrix} r \\ s \end{pmatrix} \tag{2.4}$$

where $w = v - DAu$, there is little to be lost (save sparsity in G) in using (2.4), with its positive-definite leading block, rather than (2.3). This observation has allowed Golub, Greif and Varah [25, 31] to suggest[3] a variety of methods for solving (1.1) in the case that H is positive semi-definite, although the scope of their suggestions does not appear fundamentally to be limited to this case. Lukšan and Vlček [34] make related suggestions for more general G.

Note, however, that although Theorem 2 implies the existence of a suitable D, it alas does not provide a suitable value. In [31], the authors propose heuristics to use as few nonzero components of D as possible (on sparsity grounds) when G is positive semi-definite, but it is unclear how this extends for general G. Golub, Greif and Varah's methods aim particularly to produce well-conditioned $G + A^T D A$. Notice, though, that perturbations of this form do not change the eigenvalue distribution alluded to in Theorem 1, since if $H(D_H) = H + A^T D_H A$ and $G(D_G) = G + A^T D_G A$, for (possibly different) D_H and D_G,

$$N^T H(D_H) N v = N^T H N v = \lambda N^T G N v = \lambda N^T G(D_G) N v.$$

and thus the generalized eigen-problem (2.2), and hence eigenvalues of $K_{G(D_G)}^{-1} K_{H(D_H)}$, are unaltered.

2.2 Improved eigenvalue bounds with the reduced-space basis

In this paper, we shall suppose that we may partition the columns of A so that

$$A = (A_1 \;\; A_2),$$

and so that its leading m by m sub-matrix

A1 A_1 and its transpose are easily invertible.

Since there is considerable flexibility in choosing the "basis" A_1 from the rectangular matrix A by suitable column interchanges, assumption **A1** is often easily, and sometimes trivially, satisfied. Note that the problem of determining the "sparsest" A_1 is NP hard, [8, 9], while numerical considerations must be

[3] They actually propose the alternative

$$\begin{pmatrix} G + A^T D A & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} r + A^T D s \\ s \end{pmatrix}$$

although this is not significant.

given to ensure that A_1 is not badly conditioned if at all possible [23]. More generally, we do not necessarily assume that A_1 is sparse or has a sparse factorization, merely that there are effective ways to solve systems involving A_1 and A_1^T. For example, for many problems involving constraints arising from the discretization of partial differential equations, there are highly effective *iterative* methods for such systems [4].

Given **A1**, we shall be particularly concerned with the *reduced-space* basis matrix

$$N = \begin{pmatrix} R \\ I \end{pmatrix}, \quad \text{where} \quad R = -A_1^{-1} A_2. \tag{2.5}$$

Such basis matrices play vital roles in Simplex (pivoting)-type methods for linear programming [2,20], and more generally in active-set methods for nonlinear optimization [23,35,36].

Suppose that we partition G and H so that

$$G = \begin{pmatrix} G_{11} & G_{21}^T \\ G_{21} & G_{22} \end{pmatrix} \quad \text{and} \quad H = \begin{pmatrix} H_{11} & H_{21}^T \\ H_{21} & H_{22} \end{pmatrix}, \tag{2.6}$$

where G_{11} and H_{11} are (respectively) the leading m by m sub-matrices of G and H. Then (2.5) and (2.6) give

$$\begin{aligned} N^T G N &= G_{22} + R^T G_{21}^T + G_{21} R + R^T G_{11} R \\ \text{and} \quad N^T H N &= H_{22} + R^T H_{21}^T + H_{21} R + R^T H_{11} R. \end{aligned}$$

In order to improve the eigenvalue distribution resulting from our attempts to precondition K_H by K_G, we consider the consequences of picking G to reproduce certain portions of H.

First, consider the case where

$$G_{22} = H_{22}, \quad \text{but} \quad G_{11} = 0 \quad \text{and} \quad G_{21} = 0. \tag{2.7}$$

Theorem 3. *Suppose that G and H are as in (2.6) and that (2.7) holds. Suppose furthermore that H_{22} is positive definite, and let*

$$\begin{aligned} \rho = &\min \left[\operatorname{rank}(A_2), \operatorname{rank}(H_{21}) \right] \\ &+ \min \left[\operatorname{rank}(A_2), \operatorname{rank}(H_{21}) + \min[\operatorname{rank}(A_2), \operatorname{rank}(H_{11})] \right]. \end{aligned}$$

Then $K_G^{-1} K_H$ has at most

$$\operatorname{rank}(R^T H_{21}^T + H_{21} R + R^T H_{11} R) + 1 \leq \min(\rho, n-m) + 1 \leq \min(2m, n-m) + 1$$

distinct eigenvalues.

Proof. Elementary bounds involving the products and sums of matrices show that the difference

$$N^T H N - N^T G N = R^T H_{21}^T + H_{21} R + R^T H_{11} R$$

is a matrix of rank at most $\min(\rho, n-m)$. Since $N^T G N$ is, by assumption, positive definite, we may write $N^T G N = W W^T$ for some non-singular W. Thus

$$W^{-1} N^T H N W^{-T} = I + W^{-1}(R^T H_{21}^T + H_{21} R + R^T H_{11} R) W^{-T}$$

differs from the identity matrix by a matrix of rank at most $\min(\rho, n-m)$, and hence the generalized eigenproblem (2.2) has at most $\min(\rho, n-m)$ non-unit eigenvalues.

As we have seen from Theorem 2, the restriction that H_{22} be positive definite is not as severe as it might first seem, particularly if we can entertain the possibility of using the positive-definite $H_{22} + A_2^T D A_2$ instead.

The eigenvalue situation may be improved if we consider the case where

$$G_{22} = H_{22} \text{ and } G_{11} = H_{11} \text{ but } G_{21} = 0. \tag{2.8}$$

Theorem 4. *Suppose that G and H are as in (2.6) and that (2.8) holds. Suppose furthermore that $H_{22} + R^T H_{11}^T R$ is positive definite, and that*

$$\nu = 2 \min \left[\operatorname{rank}(A_2), \operatorname{rank}(H_{21}) \right].$$

Then $K_G^{-1} K_H$ has at most

$$\operatorname{rank}(R^T H_{11} R) + 1 \leq \nu + 1 \leq \min(2m, n-m) + 1$$

distinct eigenvalues.

Proof. The result follows as before since now $N^T H N - N^T G N = R^T H_{21}^T + H_{21} R$ is of rank at most ν.
The same is true when

$$G_{22} = H_{22} \text{ and } G_{21} = H_{21} \text{ but } G_{11} = 0. \tag{2.9}$$

Theorem 5. *Suppose that G and H are as in (2.6) and that (2.9) holds. Suppose furthermore that $H_{22} + R^T H_{21}^T + H_{21} R$ is positive definite, and that*

$$\mu = \min \left[\operatorname{rank}(A_2), \operatorname{rank}(H_{11}) \right].$$

Then $K_G^{-1} K_H$ has at most

$$\operatorname{rank}(R^T H_{11} R) + 1 \leq \mu + 1 \leq \min(m, n-m) + 1$$

distinct eigenvalues.

Proof. The result follows, once again, as before since now $N^T H N - N^T G N = R^T H_{11} R$ is of rank at most μ.

In Tables 1 and 2, we illustrate these results by considering the complete set of linear and quadratic programming examples from the Netlib [21] and CUTEr [29] test sets. All inequality constraints are converted to equations by adding slack variables, and a suitable "barrier" penalty term (in this case, 1.0) is added to the diagonal of H for each bounded or slack variable to simulate systems that might arise during an iteration of an interior-point method for such problems.

Given A, a suitable basis matrix A_1 is found by finding a sparse LU factorization of A^T using the HSL [32] packages `MA48` and `MA51` [17]. An attempt to correctly identify rank is controlled by tight threshold column pivoting, in which any pivot may not be smaller than a factor $\tau = 2$ of the largest entry in its (uneliminated) column [23, 24]. The rank is estimated as the number of pivots, $\rho(A)$, completed before the remaining uneliminated sub-matrix is judged to be numerically zero, and the indices of the $\rho(A)$ pivotal rows and columns of A define A_1—if $\rho(A) < m$, the remaining rows of A are judged to be dependent, and are discarded.[4] Although such a strategy may not be as robust as, say, a singular-value decomposition or a QR factorization with pivoting, both our and others' experience [23] indicate it to be remarkably reliable and successful in practice.

Having found A_1, the factors are discarded, and a fresh LU decomposition of A_1, with a looser threshold column pivoting factor $\tau = 100$, is computed in order to try to encourage sparse factors. All other estimates of rank in Tables 1 and 2 are obtained in the same way. The columns headed "iteration bounds" illustrate Theorems 1 ("any G"), 3 ("exact H_{22}") and 5 ("exact H_{22} & H_{21}"). Note that in the linear programming case, $H_{21} \equiv 0$, so that we have omitted the "exact H_{22}" statistics from Tables 1, since these would be identical to those reported as "exact H_{22} & H_{21}".

Table 1: NETLIB LP problems

			rank				iteration bound		
							any G	exact H_{22} & H_{21}	
name	n	m	A	A_2	H_{11}	H_{12}		$\mu + 1$	upper
25FV47	1876	821	820	725	820	0	1057	726	822
80BAU3B	12061	2262	2262	2231	2262	0	9800	2232	2263
ADLITTLE	138	56	56	53	56	0	83	54	57
AFIRO	51	27	27	21	27	0	25	22	25
AGG2	758	516	516	195	516	0	243	196	243
AGG3	758	516	516	195	516	0	243	196	243
AGG	615	488	488	123	488	0	128	124	128
BANDM	472	305	305	161	305	0	168	162	168
BCDOUT	7078	5414	5412	1102	2227	0	1667	1103	1667
BEACONFD	295	173	173	116	173	0	123	117	123
BLEND	114	74	74	37	74	0	41	38	41
BNL1	1586	643	642	458	642	0	945	459	644
BNL2	4486	2324	2324	1207	2324	0	2163	1208	2163

[4] Note that if this happens, the right-hand inequalities in Theorems 3–5 will depend on $n - \text{rank}(A)$ not $n - m$.

Table 1: NETLIB LP problems (continued)

			rank				iteration bound		
							any G	exact H_{22} & H_{21}	
name	n	m	A	A_2	H_{11}	H_{12}		$\mu+1$	upper
BOEING1	726	351	351	314	351	0	376	315	352
BOEING2	305	166	166	109	166	0	140	110	140
BORE3D	334	233	231	73	231	0	104	74	104
BRANDY	303	220	193	98	193	0	111	99	111
CAPRI	482	271	271	144	261	0	212	145	212
CYCLE	3371	1903	1875	1272	1868	0	1497	1273	1497
CZPROB	3562	929	929	732	929	0	2634	733	930
D2Q06C	5831	2171	2171	2059	2171	0	3661	2060	2172
D6CUBE	6184	415	404	403	404	0	5781	404	416
DEGEN2	757	444	442	295	442	0	316	296	316
DEGEN3	2604	1503	1501	1052	1501	0	1104	1053	1104
DFL001	12230	6071	6058	5313	6058	0	6173	5314	6072
E226	472	223	223	186	223	0	250	187	224
ETAMACRO	816	400	400	341	400	0	417	342	401
FFFFF800	1028	524	524	290	524	0	505	291	505
FINNIS	1064	497	497	456	497	0	568	457	498
FIT1D	1049	24	24	24	24	0	1026	25	25
FIT1P	1677	627	627	627	627	0	1051	628	628
FIT2D	10524	25	25	25	25	0	10500	26	26
FIT2P	13525	3000	3000	3000	3000	0	10526	3001	3001
FORPLAN	492	161	161	100	161	0	332	101	162
GANGES	1706	1309	1309	397	1309	0	398	398	398
GFRD-PNC	1160	616	616	423	616	0	545	424	545
GOFFIN	101	50	50	50	0	0	52	1	51
GREENBEA	5598	2392	2389	2171	2389	0	3210	2172	2393
GREENBEB	5598	2392	2389	2171	2386	0	3210	2172	2393
GROW15	645	300	300	300	300	0	346	301	301
GROW22	946	440	440	440	440	0	507	441	441
GROW7	301	140	140	140	140	0	162	141	141
SIERRA	2735	1227	1217	768	1217	0	1519	769	1228
ISRAEL	316	174	174	142	174	0	143	143	143
KB2	68	43	43	25	43	0	26	26	26
LINSPANH	97	33	32	32	32	0	66	33	34
LOTFI	366	153	153	110	153	0	214	111	154
MAKELA4	61	40	40	21	40	0	22	22	22
MAROS-R7	9408	3136	3136	3136	3136	0	6273	3137	3137
MAROS	1966	846	846	723	846	0	1121	724	847
MODEL	1557	38	38	11	38	0	1520	12	39
MODSZK1	1620	687	686	667	684	0	935	668	688
BCDOUT	7078	5414	5412	1107	5028	0	1667	1108	1667
NESM	3105	662	662	568	662	0	2444	569	663
OET1	1005	1002	1002	3	1000	0	4	4	4
OET3	1006	1002	1002	4	1000	0	5	5	5
PEROLD	1506	625	625	532	562	0	882	533	626
PILOT4	1123	410	410	367	333	0	714	334	411
PILOT87	6680	2030	2030	1914	2030	0	4651	1915	2031
PILOT-JA	2267	940	940	783	903	0	1328	784	941
PILOTNOV	2446	975	975	823	975	0	1472	824	976
PILOT	4860	1441	1441	1354	1441	0	3420	1355	1442
PILOT-WE	2928	722	722	645	662	0	2207	646	723
PT	503	501	501	2	499	0	3	3	3
QAP8	1632	912	853	697	853	0	780	698	780
QAP12	8856	3192	3089	2783	3089	0	5768	2784	3193
QAP15	22275	6330	6285	5632	6285	0	15991	5633	6331
QPBD_OUT	442	211	211	176	211	0	232	177	212
READING2	6003	4000	4000	2001	2001	0	2004	2002	2004
RECIPELP	204	91	91	78	91	0	114	79	92
SC105	163	105	105	58	105	0	59	59	59
SC205	317	205	205	112	205	0	113	113	113
SC50A	78	50	50	28	50	0	29	29	29
SC50B	78	50	50	28	50	0	29	29	29
SCAGR25	671	471	471	199	471	0	201	200	201

Table 1: NETLIB LP problems (continued)

			rank				iteration bound		
							any G	exact H_{22} & H_{21}	
name	n	m	A	A_2	H_{11}	H_{12}		$\mu+1$	upper
SCAGR7	185	129	129	56	129	0	57	57	57
SCFXM1	600	330	330	217	330	0	271	218	271
SCFXM2	1200	660	660	440	660	0	541	441	541
SCFXM3	1800	990	990	660	990	0	811	661	811
SCORPION	466	388	388	77	388	0	79	78	79
SCRS8	1275	490	490	341	490	0	786	342	491
SCSD1	760	77	77	77	77	0	684	78	78
SCSD6	1350	147	147	147	147	0	1204	148	148
SCSD8	2750	397	397	397	397	0	2354	398	398
SCTAP1	660	300	300	246	300	0	361	247	301
SCTAP2	2500	1090	1090	955	1090	0	1411	956	1091
SCTAP3	3340	1480	1480	1264	1480	0	1861	1265	1481
SEBA	1036	515	515	479	515	0	522	480	516
SHARE1B	253	117	117	72	117	0	137	73	118
SHARE2B	162	96	96	65	96	0	67	66	67
SHELL	1777	536	535	489	535	0	1243	490	537
SHIP04L	2166	402	360	343	360	0	1807	344	403
SHIP04S	1506	402	360	256	360	0	1147	257	403
SHIP08L	4363	778	712	679	712	0	3652	680	779
SHIP08S	2467	778	712	406	712	0	1756	407	779
SHIP12L	5533	1151	1042	828	1042	0	4492	829	1152
SHIP12S	2869	1151	1042	451	1042	0	1828	452	1152
SIERRA	2735	1227	1217	768	1217	0	1519	769	1228
SIPOW1M	2002	2000	2000	2	2000	0	3	3	3
SIPOW1	2002	2000	2000	2	1999	0	3	3	3
SIPOW2M	2002	2000	2000	2	2000	0	3	3	3
SIPOW2	2002	2000	2000	2	1999	0	3	3	3
SIPOW3	2004	2000	2000	4	1999	0	5	5	5
SIPOW4	2004	2000	2000	4	1999	0	5	5	5
SSEBLIN	218	72	72	72	72	0	147	73	73
STAIR	614	356	356	249	356	0	259	250	259
STANDATA	1274	359	359	283	359	0	916	284	360
STANDGUB	1383	361	360	281	360	0	1024	282	362
STANDMPS	1274	467	467	372	467	0	808	373	468
STOCFOR1	165	117	117	48	117	0	49	49	49
STOCFOR2	3045	2157	2157	888	2157	0	889	889	889
STOCFOR3	23541	16675	16675	6866	16675	0	6867	6867	6867
TFI2	104	101	101	3	100	0	4	4	4
TRUSS	8806	1000	1000	1000	1000	0	7807	1001	1001
TUFF	628	333	302	207	301	0	327	208	327
VTP-BASE	346	198	198	86	198	0	149	87	149
WOOD1P	2595	244	244	244	244	0	2352	245	245
WOODW	8418	1098	1098	1098	1098	0	7321	1099	1099

Table 2: CUTEr QP problems

			rank				iteration bound				
							any G	exact H_{22}		exact H_{22} & H_{21}	
name	n	m	A	A_2	H_{11}	H_{12}		$\rho+1$	upper	$\mu+1$	upper
AUG2DCQP	20200	10000	10000	10000	10000	0	10201	10001	10201	10001	10001
AUG2DQP	20200	10000	10000	10000	10000	0	10201	10001	10201	10001	10001
AUG3DCQP	27543	8000	8000	7998	8000	0	19544	7999	16001	7999	8001
AUG3DQP	27543	8000	8000	7998	8000	0	19544	7999	16001	7999	8001
BLOCKQP1	10011	5001	5001	5001	5001	5000	5011	5011	5011	5002	5002
BLOCKQP2	10011	5001	5001	5001	5001	5000	5011	5011	5011	5002	5002
BLOCKQP3	10011	5001	5001	5001	5001	5000	5011	5011	5011	5002	5002
BLOWEYA	4002	2002	2002	2000	2002	2000	2001	2001	2001	2001	2001
BLOWEYB	4002	2002	2002	2000	2002	2000	2001	2001	2001	2001	2001
BLOWEYC	4002	2002	2002	2000	2002	2000	2001	2001	2001	2001	2001
CONT-050	2597	2401	2401	192	2401	0	197	193	197	193	197

Table 2: CUTEr QP problems (continued)

			rank				iteration bound				
							any G	exact H_{22}		exact H_{22} & H_{21}	
name	n	m	A	A_2	H_{11}	H_{12}		$\rho+1$	upper	$\mu+1$	upper
CONT-101	10197	10098	10098	99	10098	0	100	100	100	100	100
CONT-201	40397	40198	40198	199	40198	0	200	200	200	200	200
CONT5-QP	40601	40200	40200	401	40200	0	402	402	402	402	402
CONT1-10	10197	9801	9801	392	9801	0	397	393	397	393	397
CONT1-20	40397	39601	39601	792	39601	0	797	793	797	793	797
CONT-300	90597	90298	90298	299	90298	0	300	300	300	300	300
CVXQP1	10000	5000	5000	2000	5000	2000	5001	4001	5001	2001	5001
CVXQP2	10000	2500	2500	2175	2500	1194	7501	3370	5001	2176	2501
CVXQP3	10000	7500	7500	1000	7500	2354	2501	2001	2501	1001	2501
DEGENQP	125050	125025	125024	26	125024	0	27	27	27	27	27
DUALC1	223	215	215	8	215	0	9	9	9	9	9
DUALC2	235	229	229	6	229	0	7	7	7	7	7
DUALC5	285	278	278	7	278	0	8	8	8	8	8
DUALC8	510	503	503	7	503	0	8	8	8	8	8
GOULDQP2	19999	9999	9999	9999	9999	0	10001	10000	10001	10000	10000
GOULDQP3	19999	9999	9999	9999	9999	9999	10001	10001	10001	10000	10000
KSIP	1021	1001	1001	20	1001	0	21	21	21	21	21
MOSARQP1	3200	700	700	700	700	3	2501	704	1401	701	701
NCVXQP1	10000	5000	5000	2000	5000	2000	5001	4001	5001	2001	5001
NCVXQP2	10000	5000	5000	2000	5000	2000	5001	4001	5001	2001	5001
NCVXQP3	10000	5000	5000	2000	5000	2000	5001	4001	5001	2001	5001
NCVXQP4	10000	2500	2500	2175	2500	1194	7501	3370	5001	2176	2501
NCVXQP5	10000	2500	2500	2175	2500	1194	7501	3370	5001	2176	2501
NCVXQP6	10000	2500	2500	2175	2500	1194	7501	3370	5001	2176	2501
NCVXQP7	10000	7500	7500	1000	7500	2354	2501	2001	2501	1001	2501
NCVXQP8	10000	7500	7500	1000	7500	2354	2501	2001	2501	1001	2501
NCVXQP9	10000	7500	7500	1000	7500	2354	2501	2001	2501	1001	2501
POWELL20	10000	5000	5000	4999	5000	0	5001	5000	5001	5000	5001
PRIMALC1	239	9	9	9	9	0	231	10	19	10	10
PRIMALC2	238	7	7	7	7	0	232	8	15	8	8
PRIMALC5	295	8	8	8	8	0	288	9	17	9	9
PRIMALC8	528	8	8	8	8	0	521	9	17	9	9
PRIMAL1	410	85	85	85	85	0	326	86	171	86	86
PRIMAL2	745	96	96	96	96	0	650	97	193	97	97
PRIMAL3	856	111	111	111	111	0	746	112	223	112	112
PRIMAL4	1564	75	75	75	75	0	1490	76	151	76	76
QPBAND	75000	25000	25000	25000	25000	0	50001	25001	50001	25001	25001
QPNBAND	75000	25000	25000	25000	25000	0	50001	25001	50001	25001	25001
QPCBOEI1	726	351	351	314	351	0	376	315	376	315	352
QPCBOEI2	305	166	166	109	166	0	140	110	140	110	140
QPCSTAIR	614	356	356	249	356	0	259	250	259	250	259
QPNBOEI1	726	351	351	314	351	0	376	315	376	315	352
QPNBOEI2	305	166	166	109	166	0	140	110	140	110	140
QPNSTAIR	614	356	356	249	356	0	259	250	259	250	259
SOSQP1	5000	2501	2501	2499	2501	2499	2500	2500	2500	2500	2500
STCQP1	8193	4095	1771	0	1771	317	6423	1	6423	1	4096
STCQP2	8193	4095	4095	0	4095	1191	4099	1	4099	1	4096
STNQP1	8193	4095	1771	0	1771	317	6423	1	6423	1	4096
STNQP2	8193	4095	4095	0	4095	1191	4099	1	4099	1	4096
UBH1	9009	6000	6000	3003	6	0	3010	7	3010	7	3010
YAO	4002	2000	2000	2000	2000	0	2003	2001	2003	2001	2001

We observe that in some cases there are useful gains to be made from trying to reproduce H_{22} and, less often, H_{21}. Moreover, the upper bounds on rank obtained in Theorems 3 and 5 can be significantly larger than even the estimates $\rho+1$ and $\mu+1$ of the number of distinct eigenvalues. However the trend is far from uniform, and in some cases there is little or no apparent advantage to be gained from reproducing portions of H. Nonetheless, since

significant improvements are possible, we now investigate efficient ways of computing decompositions which are capable of reproducing sub-blocks of H.

3 Implicit-factorization constraint preconditioners

It has long been common practice (at least in optimization circles) [3, 6, 10, 18, 22, 34, 39, 42] to use preconditioners of the form (1.2) by specifying G and factorizing K_G using a suitable symmetric, indefinite package such as `MA27` [16] or `MA57` [15]. While such techniques have often been successful, they have usually been rather *ad hoc*, with little attempt to improve upon the eigenvalue distributions beyond those suggested by the Theorem 1.

Recently, Dollar and Wathen [14] have suggested using a preconditioner of the form

$$K_G = PBP^T, \tag{3.1}$$

where solutions with each of the matrices P, B and P^T are easily obtained. In particular, rather than obtaining P and B from a given K_G, K_G *is derived implicitly from specially chosen* P *and* B. In this section, we examine a broad class of methods of this form.

3.1 Structural considerations

In general, we may write

$$P = \begin{pmatrix} P_1 & A^T \\ P_2 & 0 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} B_1 & B_2^T \\ B_2 & B_{33} \end{pmatrix} \tag{3.2}$$

where B_1 and B_{33} are symmetric and P_2 is of full rank; the zero block in P is selected so as to mimic that in K_G. Given this form, we have

$$K_G = \begin{pmatrix} P_1B_1P_1^T + A^TB_2P_1^T + P_1B_2^TA + A^TB_{33}A & P_1B_1P_2^T + A^TB_2P_2^T \\ P_2B_1P_1^T + P_2B_2^TA & P_2B_1P_2^T \end{pmatrix}$$

and since we wish (1.2) to hold, we require that

$$P_2B_1P_1^T + P_2B_2^TA = A \quad \text{and} \quad P_2B_1P_2^T = 0. \tag{3.3}$$

As A and P_2 are of full rank, we write

$$A = (A_1 \;\; A_2) \quad \text{and} \quad P_2 = (P_{31} \;\; P_{32})$$

for nonsingular m by m matrices A_1 and P_{31}, and shall likewise write

$$P_1 = \begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix}, \quad B_1 = \begin{pmatrix} B_{11} & B_{21}^T \\ B_{21} & B_{22} \end{pmatrix} \quad \text{and} \quad B_2 = (B_{31} \;\; B_{32}).$$

The second requirement in (3.3) is then that

$$P_{31}B_{11}P_{31}^T + P_{32}B_{21}P_{31}^T + P_{31}B_{21}^T P_{32}^T + P_{32}B_{22}P_{32}^T = 0.$$

Although there are a number of ways of guaranteeing this,[5] the simplest is to insist that

$$P_{32} = 0 \text{ and } B_{11} = 0.$$

The first requirement in (3.3) may be satisfied if

$$P_2 B_2^T = I \text{ and } P_2 B_1 P_1^T = 0, \tag{3.4}$$

although again there are other (more complicated) possibilities. It then follows that

$$B_{31} = P_{31}^{-T} \text{ and } P_{31}B_{21}^T(P_{12}^T \ P_{22}^T) = 0$$

and the second of these implies that

$$B_{21} = 0$$

since P_{31} is non singular and $(P_{12}^T \ P_{22}^T)$ must be of full rank.[6] Thus

$$P = \begin{pmatrix} P_{11} & P_{12} & A_1^T \\ P_{21} & P_{22} & A_2^T \\ B_{31}^{-T} & 0 & 0 \end{pmatrix} \text{ and } B = \begin{pmatrix} 0 & 0 & B_{31}^T \\ 0 & B_{22} & B_{32}^T \\ B_{31} & B_{32} & B_{33} \end{pmatrix}, \tag{3.5}$$

where B_{31} and B_{22} are non-singular. Furthermore, it follows trivially from Sylvester's law of inertia (see, for example, [11]) that

$$B_{22} \text{ must be positive definite} \tag{3.6}$$

if (2.1) is to hold.

3.2 Solution considerations

Solves involving P and its transpose

Suppose that B_{31} is chosen to be easily invertible—Dollar and Wathen [14] suggest picking $B_{31} = I$, but other simple choices are possible. Then, in order to solve systems involving the block (reverse) triangular matrix P and its transpose, it suffices to be able to do so for systems involving the sub-matrix

$$\begin{pmatrix} P_{12} & A_1^T \\ P_{22} & A_2^T \end{pmatrix}.$$

[5] In general $B_{11} = -P_{31}^{-1}\left(P_{32}B_{21}P_{31}^T + P_{31}B_{21}^T P_{32}^T + P_{32}B_{22}P_{32}^T\right)P_{31}^{-T}$ for any P_{32}.
[6] The latter follows since $P_{32} = 0$ and P is required to be non-singular.

Although **A1** allows a general (Schur-complement) pivot, in which such systems may be solved knowing factors of A_1 and $P_{22} + R^T P_{12}$, perhaps the easiest possibility is, again, to follow [14] and pick

$$P_{12} = 0. \tag{3.7}$$

This then presupposes that P_{22} is non-singular.

One further saving here in the solution of (2.3) via forward and backward substituting from (3.1) in the usual (preconditioning) case for which $s = 0$ is that the the block zero component of the right-hand-side may trivially be exploited in the initial forward substitution

$$\begin{pmatrix} P_{11} & 0 & A_1^T \\ P_{21} & P_{22} & A_2^T \\ B_{31}^{-T} & 0 & 0 \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \\ q \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \\ 0 \end{pmatrix}$$

for which $p_1 = 0$.

Solves involving B

It follows from (3.5) that solving systems of equations whose coefficient matrix is B relies on being able to solve systems with coefficient matrices B_{31}, B_{22} and B_{31}^T. The choice $B_{31} = I$ made by Dollar and Wathen [14] is again ideal from this perspective.

3.3 Considerations relating to preconditioning

So far, we simply require that P and B satisfy (3.5) in order to ensure K_G is of the form (1.2), but additionally that (3.6) holds for K_G to be a useful preconditioner. Note that without (3.6) we could choose the components of P and B to factorize K_G in the case where $H = G$, but if

$$\text{In} \begin{pmatrix} H & A^T \\ A & 0 \end{pmatrix} \neq (n, m, 0)$$

it will not be possible to find B_{22} satisfying (3.6) in this case.

Recovering G

The leading diagonal block G of K_G is

$$G = P_1 B_1 P_1^T + A^T B_2 P_1^T + P_1 B_2^T A + A^T B_{33} A. \tag{3.8}$$

In what remains, we shall thus assume that P and B_2 are given by (3.5), and that (3.7) holds, that is that

$$P = \begin{pmatrix} P_{11} & 0 & A_1^T \\ P_{21} & P_{22} & A_2^T \\ B_{31}^{-T} & 0 & 0 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 0 & 0 & B_{31}^T \\ 0 & B_{22} & B_{32}^T \\ B_{31} & B_{32} & B_{33} \end{pmatrix}. \tag{3.9}$$

It follows immediately from (2.6), (3.8) and (3.9) that

$$\begin{aligned} G_{11} &= P_{11}B_{31}^T A_1 + A_1^T B_{31} P_{11}^T + A_1^T B_{33} A_1 \\ G_{21} &= A_2^T B_{31} P_{11}^T + P_{21} B_{31}^T A_1 + P_{22} B_{32}^T A_1 + A_2^T B_{33} A_1 \quad \text{and} \\ G_{22} &= P_{22} B_{22} P_{22}^T + P_{21} B_{31}^T A_2 + P_{22} B_{32}^T A_2 + A_2^T B_{31} P_{21}^T + A_2^T B_{32} P_{22}^T + A_2^T B_{33} A_2. \end{aligned}$$

Notice that we have not as yet determined P_{11}, P_{21}, P_{22}, B_{22}, B_{31}, B_{32} and B_{33}, but that G involves significantly less information, and thus there is likely to be considerable freedom in our remaining choices even if we wish to recover a particular G.

It follows from (3.8) that

$$N^T G N = N^T P_1 B_1 P_1^T N$$

for any null-space basis matrix N, since $AN = 0$ It also follows from the required form (3.9) of P and B that

$$P_1 B_1 P_1^T = \begin{pmatrix} 0 & 0 \\ 0 & P_{22} B_{22} P_{22}^T \end{pmatrix}$$

and in the case of the reduced-space basis matrix (2.5) we have that

$$N^T G N = P_{22} B_{22} P_{22}^T.$$

3.4 Particular choices of P and B

Existing proposals

Schilders [40] sets $B_{31} = I$ and $B_{32} = 0$, and uses P_{11} and P_{22} as free parameters to determine P_{21}, B_{22} and B_{33} from G. Dollar and Wathen [14] consider the same choices for B_{31} and B_{32}, and use P_{11} and P_{22} and B_{33} as free parameters to determine P_{21}, B_{22} and G_{22} from G_{11} and G_{21}. So for example, if

$$P_{11} = 0, P_{21} = 0, \quad P_{22} = I, \quad B_{31} = I, \quad B_{22} = I, \quad B_{32} = 0 \quad \text{and} \quad B_{33} = 0$$

then

$$G_{11} = 0, \quad G_{21} = 0 \quad \text{and} \quad G_{22} = I.$$

Reproducing H_{22}

The simplest option is to set as many of free components of P and B as possible to zero; this corresponds to setting

$$P_{11} = 0, \;\; P_{21} = 0, \;\; B_{32} = 0 \;\text{ and }\; B_{33} = 0, \tag{3.10}$$

and results in

$$G_{11} = 0, \;\; G_{21} = 0 \;\text{ and }\; G_{22} = P_{22} B_{22} P_{22}^T.$$

Thus the requirement (3.6) forces G_{22} to be positive definite, and any positive-definite G_{22} may be accommodated by the choice (3.10). In particular, if H_{22} is positive-definite, Theorem 3 shows that picking $G_{22} = H_{22}$ leads to an improved eigenvalue bound over that for generic G. In this case P_{22} and B_{22} could accommodate (sparse) Cholesky or LDL^T factors of H_{22}.

Reproducing H_{21} and H_{22}

The choice

$$P_{11} = 0 \;\text{ and }\; B_{33} = 0 \tag{3.11}$$

gives

$$G_{11} = 0, \;\; G_{21} = P_{21} B_{31}^T A_1 + P_{22} B_{32}^T A_1 \;\text{ and}$$
$$G_{22} = P_{22} B_{22} P_{22}^T + P_{21} B_{31}^T A_2 + P_{22} B_{32}^T A_2 + A_2^T B_{31} P_{21}^T + A_2^T B_{32} P_{22}^T.$$

while choosing

$$P_{11} = 0, \;\; B_{32} = 0 \;\text{ and }\; B_{33} = 0 \tag{3.12}$$

gives

$$G_{11} = 0, \;\; G_{21} = P_{21} B_{31}^T A_1 \;\text{ and }\; G_{22} = P_{22} B_{22} P_{22}^T + P_{21} B_{31}^T A_2 + A_2^T B_{31} P_{21}^T.$$

Both of these possibilities allow us to choose $G_{22} = H_{22}$ and $G_{21} = H_{21}$, and Theorem 5 indicates that such choices lead to further improved eigenvalue bounds. Moreover, in both cases,

$$P_{22} B_{22} P_{22}^T = G_{22} + R^T G_{21}^T + G_{21} R$$

regardless of how we choose P_{21}, B_{31} and B_{32}.

Ensuring that G is positive definite

The role of the matrix B_{33} is interesting. For Theorem 2 and (3.8) suggest that by picking B_{33} sufficiently negative definite, the remaining terms

$$P_1 B_1 P_1^T + A^T B_2 P_1^T + P_1 B_2^T A$$

will be positive definite. However, since any significantly dense rows of A will result in dense blocks in $A^T B_{33} A$, it may well be wise to keep $B_{33} = 0$.

3.5 Factors in other orders

We have seen that specifying decompositions of the form (3.1) in which P and B have the block form (3.2) is an extremely flexible approach. A natural question is: are there other block forms which are equally useful? The most obvious alternative is to seek a decomposition

$$K_G = QEQ^T, \tag{3.13}$$

where

$$Q = \begin{pmatrix} Q_1 & Q_2 \\ A & 0 \end{pmatrix} \quad \text{and} \quad E = \begin{pmatrix} E_1 & E_2^T \\ E_2 & E_{33} \end{pmatrix} \tag{3.14}$$

where E_1 and E_{33} are symmetric and Q_2 is of full rank; here again the zero block in Q is selected so as to mimic that in K_G. In this case

$$K_G = \begin{pmatrix} Q_1E_1Q_1^T + Q_2E_2Q_1^T + Q_1E_2^TQ_2^T + Q_2E_{33}Q_2^T & Q_1E_1A^T + Q_2E_2A^T \\ AE_1Q_1^T + AE_2^TQ_2^T & AE_1A^T \end{pmatrix}. \tag{3.15}$$

But now we see a strong disadvantage of (3.13) compared with (3.1), namely that requiring that the 2,1 and 2,2 blocks of (3.15) reproduce A and 0 respectively place strong restrictions on E_1, E_2, Q_1 and Q_2. In particular, E_1A^T must lie in the null-space of A. Since this seems to limit the scope of (3.13)–(3.14) we do not pursue this further.

4 Numerical experiments

In this section we indicate that, in some cases, the implicit-factorization preconditioners proposed in Section 3 are very effective in practice.

We consider the set of quadratic programming examples from the CUTEr test set examined in Section 2. For each, we use the projected preconditioned conjugate-gradient method [27] to solve the resulting quadratic programming problem

$$\text{EQP:} \quad \underset{x \in \mathbb{R}^n}{\text{minimize}} \; q(x) = \tfrac{1}{2}x^THx + c^Tx \;\text{ subject to }\; Ax = b.$$

Firstly a feasible point $x = x_0$ is determined. Thereafter, iterates $x_0 + s$ generated by the conjugate-gradient method are constrained to satisfy $As = 0$ by means of the preconditioning system (2.3). Since, as frequently happens in practice, $q(x_0 + s)$ may be unbounded from below, a trust-region constraint $\|s\| \leq \Delta$ is also imposed, and the Generalized Lanczos Trust-Region (GLTR) method [28], as implemented in the GALAHAD library [30], is used to solve the resulting problem

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \; q(x_0 + s) \;\text{ subject to }\; As = 0 \;\text{ and }\; \|s\| \leq \Delta; \tag{4.1}$$

a large value of $\Delta = 10^{10}$ is used so as not to cut off the unconstrained solution for convex problems.

In Tables 1 and 2, we compare four preconditioning strategies for (approximately) solving the problem (4.1). We consider both low and high(er) accuracy solutions. For the former, we terminate as soon as the norm of the (preconditioned) gradient of $q(x_0 + s)$ has been reduced more than 10^{-2} from that of $q(x_0)$, while the latter requires a 10^{-8} reduction; these are intended to simulate the levels of accuracy required within a nonlinear programming solver in early (global) and later (asymptotic) phases of the solution process.

We consider two explicit factorizations, one using exact factors ($G = H$), and the other using a simple projection ($G = I$). The HSL package `MA57` [15] (version 2.2.1) is used to factorize K_G and subsequently solve (2.3); by way of comparison, we also include times for exact factorization with the earlier `MA27` [16], since this is still widely used. Two implicit factorizations of the form (3.1) with factors (3.9) are also considered. In the first, we use the method in Section 3.4 to get $G_{22} = I$. The second follows Section 3.4 and aims to reproduce $G_{22} = H_{22}$, and uses `MA57` to compute its factors. In particular, we exploit one of `MA57`'s options to make modest modifications [41] of the diagonals of H_{22} to ensure that G_{22} is positive definite if H_{22} fails to be—this proved only to be necessary for the `BLOWEY*` problems.

All of our experiments were performed using a single processor of a 3.05Mhz Dell Precision 650 Workstation with 4 Gbytes of RAM. Our codes were written in double precision fortran 90, compiled using the Intel ifort 8.1 compiler, and wherever possible made use of tuned ATLAS BLAS [43] for core computations. A single iteration of iterative refinement is applied, as necessary, when applying the preconditioner (2.3) to try to ensure small relative residuals.

For each option tested, we record the time taken to compute the (explicit or implicit) factors, the number of GLTR iterations performed (equivalently, the number of preconditioned systems solved), and the total time taken to solve the quadratic programming problem EQP (including the factorization). The initial feasible point x_0 is found by solving

$$\begin{pmatrix} G & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = \begin{pmatrix} 0 \\ b \end{pmatrix}$$

using the factors of K_G. Occasionally—in particular when $c = 0$ and $G = H$—such a point solves EQP, and the resulting iteration count is zero. In a few cases, the problems are so ill-conditioned that the trust-region constraint is activated, and more than one GLTR iteration is required to solve EQP even when $G = H$. Furthermore, rank deficiency of A occasionally resulted in unacceptably large residuals in (2.3) and subsequent failure of GLTR when $G = H$, even after iterative refinement.

In many cases, the use of an "exact" preconditioner $G = H$ is cost effective, particularly when the newer factorization package `MA57` is used to

Table 1: CUTEr QP problems—residual decrease of at least 10^{-2}

	Explicit factors									Implicit factors					
	$G = H$						$G = I$			$G_{22} = I$			$G_{22} = H_{22}$		
	MA27			MA57			MA57			MA57			MA57		
name	fact.	iter.	total	fact.	iter.	total	fact.	iter.	total	fact.	iter.	total	fact.	iter.	total
AUG2DCQP	0.08	1	0.13	0.47	1	0.54	0.46	1	0.53	0.04	125	1.54	0.25	125	2.01
AUG2DQP	0.08	1	0.13	0.47	1	0.54	0.46	2	0.53	0.04	120	1.49	0.25	125	2.03
AUG3DCQP	1.56	1	1.66	1.54	1	1.67	1.45	1	1.57	0.05	41	0.71	0.79	41	1.59
AUG3DQP	1.59	1	1.69	1.29	1	1.42	1.46	2	1.59	0.05	43	0.71	0.78	40	1.56
BLOCKQP1	0.06	0	0.08	0.21	0	0.23	0.23	1	0.26	0.33	2	0.35	0.39	2	0.41
BLOCKQP2	0.06	0	0.08	0.21	0	0.23	0.23	2	0.26	0.33	2	0.36	0.39	2	0.41
BLOCKQP3	0.06	0	0.08	0.21	0	0.23	0.23	1	0.25	0.33	2	0.35	0.38	2	0.41
BLOWEYA	26.50	1	26.60	0.04	1	0.05	0.05	35	0.21	0.03	50	0.13	0.04	50	0.15
BLOWEYB	26.29	1	26.39	0.04	1	0.05	0.05	13	0.11	0.03	32	0.09	0.04	32	0.11
BLOWEYC	26.27	1	26.36	0.04	1	0.05	0.05	36	0.21	0.03	50	0.12	0.04	50	0.15
CONT-050	0.17	1	0.19	0.12	1	0.14	0.12	1	0.14	0.09	3	0.10	0.09	3	0.11
CONT-101	3.03	1	3.18	0.73	2	0.85	0.70	2	0.82	0.86	2	0.91	0.86	2	0.91
CONT-201	35.96	4	38.38	5.78	5	6.99	5.63	6	7.04	10.14	2	10.41	10.10	2	10.37
CONT5-QP	33.89	1	34.59	3.37	1	3.83	3.35	2	3.80	20.01	39	22.36	19.94	37	22.20
CONT1-10	2.81	1	2.95	0.68	1	0.80	0.66	1	0.77	0.90	3	0.97	0.91	3	0.99
CONT1-20	30.94	1	31.65	6.85	1	7.46	6.67	2	7.28	10.83	3	11.22	10.86	3	11.26
CONT-300	140.10	9	146.23	19.33	5	22.26	18.33	5	21.25	40.82	2	41.46	41.00	2	41.64
CVXQP1	579.20	0	580.15	3.99	0	4.11	0.20	3	0.24	0.21	57	0.56	0.24	55	0.69
CVXQP2	139.11	0	139.48	1.70	0	1.78	0.10	3	0.12	0.01	14	0.07	0.10	14	0.23
CVXQP3	1353.52	0	1355.13	9.93	0	10.13	0.32	3	0.38	0.33	44	0.64	0.34	43	0.68
DEGENQP	3.85	1	4.14	14.36	1	14.72	0.01	2	0.01	2.43	3	2.87	2.45	3	2.89
DUALC1	0.01	5	0.01	0.00	2	0.01	0.00	1	0.00	0.00	8	0.00	0.00	8	0.00
DUALC2	0.01	9	0.01	0.00	1	0.01	0.01	2	0.01	0.00	6	0.00	0.00	6	0.01
DUALC5	0.01	8	0.02	0.01	1	0.01	0.01	2	0.01	0.00	6	0.01	0.00	6	0.01
DUALC8	0.11	5	0.13	0.01	2	0.01	0.20	0	0.23	0.01	7	0.01	0.01	7	0.01
GOULDQP2	0.05	0	0.07	0.23	0	0.27	0.20	2	0.25	0.03	0	0.05	0.08	0	0.10
GOULDQP3	0.07	1	0.11	0.32	1	0.40	0.05	5	0.06	0.03	6	0.11	0.08	6	0.17
KSIP	0.01	1	0.02	0.05	1	0.06	0.04	3	0.05	0.02	3	0.03	0.02	3	0.03
MOSARQP1	0.02	1	0.03	0.04	1	0.04	0.20	3	0.24	0.06	6	0.07	0.07	6	0.08
NCVXQP1	573.69	0	574.65	4.10	0	4.22	0.20	3	0.24	0.21	55	0.54	0.24	55	0.68
NCVXQP2	584.17	0	585.14	4.02	0	4.14	0.20	3	0.24	0.20	55	0.54	0.24	56	0.70
NCVXQP3	573.04	0	573.98	4.15	0	4.28	0.11	3	0.13	0.20	54	0.53	0.23	55	0.69
NCVXQP4	138.52	0	138.90	1.71	0	1.79	0.10	3	0.12	0.01	14	0.07	0.10	13	0.22
NCVXQP5	130.26	0	130.64	1.69	0	1.76	0.10	3	0.13	0.01	14	0.06	0.10	14	0.24
NCVXQP6	139.37	0	139.75	1.70	0	1.79	0.32	3	0.38	0.01	14	0.06	0.10	14	0.24
NCVXQP7	1363.85	0	1365.49	10.03	0	10.23	0.33	3	0.39	0.33	43	0.64	0.34	43	0.67
NCVXQP8	1386.80	0	1388.45	10.07	0	10.26	0.33	3	0.38	0.33	43	0.63	0.34	43	0.67
NCVXQP9	1357.68	0	1359.31	10.12	0	10.32	0.09	2	0.11	0.33	44	0.64	0.34	43	0.67
POWELL20	0.03	0	0.05	0.09	0	0.11	0.00	5	0.01	0.01	2	0.03	0.07	2	0.08
PRIMALC1	0.00	1	0.00	0.00	1	0.01	0.00	3	0.00	0.00	11	0.00	0.00	6	0.00
PRIMALC2	0.00	1	0.00	0.00	1	0.01	0.00	6	0.01	0.00	5	0.00	0.00	5	0.00
PRIMALC5	0.00	1	0.00	0.00	1	0.01	0.01	4	0.01	0.00	6	0.00	0.00	5	0.00
PRIMALC8	0.01	1	0.01	0.01	1	0.01	0.01	8	0.02	0.00	11	0.01	0.00	7	0.01
PRIMAL1	0.01	1	0.01	0.01	1	0.02	0.03	5	0.03	0.00	15	0.01	0.00	27	0.02
PRIMAL2	0.01	1	0.01	0.03	1	0.03	0.06	4	0.07	0.00	13	0.01	0.01	21	0.02
PRIMAL3	0.03	1	0.03	0.06	1	0.06	0.03	3	0.04	0.01	18	0.04	0.01	26	0.06
PRIMAL4	0.04	1	0.04	0.03	1	0.03	14.34	2	14.69	0.01	12	0.03	0.02	15	0.04
QPBAND	0.16	1	0.30	1.08	1	1.28	1.84	2	1.99	0.09	2	0.19	0.40	2	0.54
QPNBAND	0.17	1	0.30	1.07	1	1.27	1.83	3	2.03	0.09	3	0.24	0.41	2	0.55
QPCBOEI1	0.01	1	0.01	0.02	2	0.02	0.01	3	0.01	0.00	12	0.01	0.00	12	0.01
QPCBOEI2	0.00	1	0.01	0.00	1	0.01	0.00	3	0.01	0.00	12	0.00	0.00	12	0.00
QPCSTAIR	0.01	1	0.01	0.02	1	0.02	0.01	3	0.02	0.00	12	0.01	0.00	14	0.01
QPNBOEI1	0.01	1	0.01	0.02	2	0.02	0.01	3	0.01	0.01	12	0.01	0.00	12	0.01
QPNBOEI2	0.00	1	0.00	0.00	1	0.01	0.00	3	0.01	0.00	12	0.00	0.00	12	0.00
QPNSTAIR	0.01	1	0.01	0.02	1	0.02	0.01	3	0.02	0.00	12	0.01	0.00	12	0.01
SOSQP1	0.01	0	0.01	0.04	0	0.04	0.04	0	0.05	0.03	1	0.04	0.05	1	0.05
STCQP1	rank deficient A			rank deficient A			20.67	3	21.01	0.02	3	0.04	0.09	1	0.10
STCQP2	9.76	0	9.84	0.87	0	0.92	0.14	3	0.17	0.03	3	0.05	0.11	1	0.13
STNQP1	113.27	0	113.59	rank deficient A			20.75	3	21.09	0.02	3	0.04	0.09	1	0.11
STNQP2	9.64	0	9.72	0.87	0	0.92	0.14	3	0.17	0.03	3	0.05	0.11	1	0.13
UBH1	0.02	0	0.03	0.12	0	0.14	0.11	0	0.13	0.02	0	0.03	0.04	0	0.05
YAO	0.01	1	0.01	0.03	1	0.04	0.03	6	0.05	0.01	21	0.04	0.02	21	0.06

Table 2: CUTEr QP problems—residual decrease of at least 10^{-8}

	Explicit factors									Implicit factors					
	$G = H$						$G = I$			$G_{22} = I$			$G_{22} = H_{22}$		
	MA27			MA57			MA57			MA57			MA57		
name	fact.	iter.	total	fact.	iter.	total	fact.	iter.	total	fact.	iter.	total	fact.	iter.	total
AUG2DCQP	0.08	1	0.13	0.47	1	0.54	0.46	1	0.53	0.04	866	10.35	0.25	872	12.50
AUG2DQP	0.08	1	0.13	0.47	1	0.54	0.46	4	0.60	0.04	882	10.67	0.25	855	12.33
AUG3DCQP	1.56	1	1.66	1.54	1	1.67	1.45	1	1.57	0.05	378	6.04	0.79	377	8.18
AUG3DQP	1.59	1	1.69	1.29	1	1.42	1.46	5	1.75	0.05	381	5.90	0.78	380	8.27
BLOCKQP1	0.06	0	0.08	0.21	0	0.23	0.23	1	0.26	0.33	3	0.37	0.39	3	0.43
BLOCKQP2	0.06	0	0.08	0.21	0	0.23	0.23	2	0.26	0.33	3	0.37	0.39	3	0.43
BLOCKQP3	0.06	0	0.08	0.21	0	0.23	0.23	1	0.25	0.33	5	0.39	0.38	4	0.44
BLOWEYA	26.50	1	26.60	0.04	1	0.05	> 10000 iterations			> 10000 iterations			> 10000 iterations		
BLOWEYB	26.29	1	26.39	0.04	1	0.05	0.05	216	0.99	0.03	668	1.23	> 10000 iterations		
BLOWEYC	26.27	1	26.36	0.04	1	0.05	> 10000 iterations			> 10000 iterations			> 10000 iterations		
CONT-050	0.17	1	0.19	0.12	1	0.14	0.12	1	0.14	0.09	7	0.12	0.09	7	0.13
CONT-101	3.03	1	3.18	0.73	4	1.11	0.70	5	1.15	0.86	10	1.09	0.86	10	1.10
CONT-201	35.96	4	38.38	5.78	8	9.39	5.63	13	11.26	10.14	11	11.48	10.10	11	11.43
CONT5-QP	33.89	1	34.59	3.37	1	3.83	3.35	2	3.95	20.01	113	26.81	19.94	98	25.89
CONT1-10	2.81	1	2.95	0.68	1	0.80	0.66	1	0.77	0.90	10	1.13	0.91	10	1.16
CONT1-20	30.94	1	31.65	6.85	1	7.46	6.67	5	9.08	10.83	12	12.29	10.86	12	12.34
CONT-300	140.10	27	174.66	19.33	26	45.80	18.33	40	58.01	40.82	15	44.98	41.00	15	45.16
CVXQP1	579.20	0	580.15	3.99	0	4.11	0.20	5	0.27	0.21	211	1.49	0.24	207	1.94
CVXQP2	139.11	0	139.48	1.70	0	1.78	0.10	5	0.14	0.01	51	0.21	0.10	51	0.59
CVXQP3	1353.52	0	1355.13	9.93	0	10.13	0.32	5	0.42	0.33	183	1.52	0.34	178	1.71
DEGENQP	3.85	1	4.14	14.36	1	14.72	0.01	11	0.01	2.43	3	3.00	2.45	7	3.52
DUALC1	0.01	5	0.01	0.00	11	0.01	0.00	1	0.00	0.00	8	0.01	0.00	8	0.00
DUALC2	0.01	9	0.01	0.00	1	0.01	0.01	4	0.01	0.00	6	0.00	0.00	6	0.01
DUALC5	0.01	145	0.20	0.01	1	0.01	0.01	5	0.01	0.00	7	0.01	0.00	7	0.01
DUALC8	0.11	5	0.13	0.01	7	0.02	0.20	0	0.23	0.01	7	0.01	0.01	7	0.01
GOULDQP2	0.05	0	0.07	0.23	0	0.27	0.20	5	0.31	0.03	0	0.05	0.08	0	0.10
GOULDQP3	0.07	1	0.11	0.32	1	0.40	0.05	21	0.08	0.03	1614	18.95	0.08	1579	23.38
KSIP	0.01	1	0.02	0.05	1	0.06	0.04	5	0.05	0.02	18	0.05	0.02	10	0.04
MOSARQP1	0.02	1	0.03	0.04	1	0.04	0.20	5	0.27	0.06	36	0.10	0.07	35	0.13
NCVXQP1	573.69	0	574.65	4.10	0	4.22	0.20	5	0.27	0.21	215	1.51	0.24	204	1.89
NCVXQP2	584.17	0	585.14	4.02	0	4.14	0.20	6	0.28	0.20	212	1.50	0.24	212	2.00
NCVXQP3	573.04	0	573.98	4.15	0	4.28	0.11	5	0.14	0.20	210	1.46	0.23	204	1.92
NCVXQP4	138.52	0	138.90	1.71	0	1.79	0.10	5	0.14	0.01	51	0.20	0.10	51	0.60
NCVXQP5	130.26	0	130.64	1.69	0	1.76	0.10	6	0.15	0.01	51	0.20	0.10	50	0.59
NCVXQP6	139.37	0	139.75	1.70	0	1.79	0.32	5	0.42	0.01	51	0.21	0.10	51	0.61
NCVXQP7	1363.85	0	1365.49	10.03	0	10.23	0.33	5	0.43	0.33	189	1.69	0.34	176	1.67
NCVXQP8	1386.80	0	1388.45	10.07	0	10.26	0.33	5	0.42	0.33	191	1.69	0.34	176	1.70
NCVXQP9	1357.68	0	1359.31	10.12	0	10.32	0.09	20	0.23	0.33	193	1.69	0.34	179	1.71
POWELL20	0.03	0	0.05	0.09	0	0.11	0.00	11	0.01	0.01	40	0.21	0.07	40	0.31
PRIMALC1	0.00	1	0.00	0.00	1	0.01	0.00	4	0.00	0.00	25	0.01	0.00	12	0.01
PRIMALC2	0.00	1	0.00	0.00	1	0.01	0.00	10	0.01	0.00	9	0.00	0.00	9	0.00
PRIMALC5	0.00	1	0.00	0.00	1	0.01	0.01	7	0.01	0.00	15	0.01	0.00	10	0.01
PRIMALC8	0.01	1	0.01	0.01	1	0.01	0.01	14	0.02	0.00	20	0.01	0.00	10	0.01
PRIMAL1	0.01	1	0.01	0.01	1	0.02	0.03	8	0.03	0.00	153	0.08	0.00	158	0.09
PRIMAL2	0.01	1	0.01	0.03	1	0.03	0.06	6	0.07	0.00	86	0.06	0.01	92	0.08
PRIMAL3	0.03	1	0.03	0.06	1	0.06	0.03	5	0.04	0.01	74	0.14	0.01	80	0.15
PRIMAL4	0.04	1	0.04	0.03	1	0.03	14.34	2	14.80	0.01	41	0.07	0.02	44	0.09
QPBAND	0.16	1	0.30	1.08	1	1.28	1.84	5	2.19	0.09	7	0.46	0.40	5	0.78
QPNBAND	0.17	1	0.30	1.07	1	1.27	1.83	6	2.24	0.09	8	0.51	0.41	6	0.84
QPCBOEI1	0.01	1	0.01	0.02	5	0.02	0.01	5	0.02	0.00	47	0.03	0.00	47	0.03
QPCBOEI2	0.00	1	0.01	0.00	1	0.01	0.00	5	0.01	0.00	38	0.01	0.00	37	0.01
QPCSTAIR	0.01	1	0.01	0.02	1	0.02	0.01	8	0.02	0.00	40	0.02	0.00	52	0.03
QPNBOEI1	0.01	1	0.01	0.02	5	0.03	0.01	5	0.01	0.01	48	0.03	0.00	47	0.03
QPNBOEI2	0.00	1	0.00	0.00	1	0.01	0.00	5	0.01	0.00	37	0.01	0.00	37	0.01
QPNSTAIR	0.01	1	0.01	0.02	1	0.02	0.01	8	0.02	0.00	40	0.02	0.00	56	0.03
SOSQP1	0.01	0	0.01	0.04	0	0.04	0.04	0	0.05	0.03	1	0.04	0.05	1	0.05
STCQP1	rank deficient A			rank deficient A			20.67	6	21.35	0.02	6	0.05	0.09	1	0.10
STCQP2	9.76	0	9.84	0.87	0	0.92	0.14	7	0.20	0.03	7	0.07	0.11	1	0.13
STNQP1	113.27	0	113.59	rank deficient A			20.75	6	21.43	0.02	6	0.05	0.09	1	0.11
STNQP2	9.64	0	9.72	0.87	0	0.92	0.14	8	0.22	0.03	8	0.08	0.11	1	0.13
UBH1	0.02	0	0.03	0.12	0	0.14	0.11	0	0.13	0.02	0	0.03	0.04	0	0.05
YAO	0.01	1	0.01	0.03	1	0.04	0.03	26	0.11	0.01	107	0.18	0.02	106	0.23

compute the factors. For those problems for which the exact preconditioner is expensive—for example, the CVXQP* and NCVXQP* problems—the "inexact" preconditioners are often more effective, particularly when low accuracy solutions are required. The explicit preconditioner with $G = I$ is often a good compromise, although this may reflect the fact that H is often (almost) diagonal. The implicit factors are sometimes but not always cheaper to compute than the explicit ones. The cost of finding a good basis A_1 using MA48 is higher than we would have liked, and is usually the dominant cost of the overall implicit factorization. Nonetheless, for problems like the DUAL*, PRIMAL* and ST* examples, the implicit factors seem to offer a good alternative to the explicit ones. We must admit to being slightly disappointed that the more sophisticated implicit factors using $G_{22} = H_{22}$ seemed to show few advantages over the cheaper $G_{22} = I$, but again this might reflect the nature of H in our test set.

5 Comments and conclusions

We have developed a class of implicit-factorization constraint preconditioners for the iterative solution of symmetric linear systems arising from saddle-point problems. These preconditioners are flexible, and allow for improved eigenvalue distributions over traditional approaches. Numerical experiments indicate that these methods hold promise for solving large-scale problems, and suggest that such methods should be added to the arsenal of available preconditioners for saddle-point and related problems. A fortran 90 package which implements methods from our class of preconditioners will shortly be available as part of the GALAHAD library [30]. We are currently generalizing implicit-factorization preconditioners to cope with problems for which the 2,2 block in (1.1) may be nonzero [13].

One issue we have not really touched on—aside from the need for stable factors—is the effect of partitioning of the columns of A to produce a non-singular sub-matrix A_1. Consider the simple example

$$A = \begin{pmatrix} \times & 0 & \times & 0 \\ 0 & \times & \times & \times \end{pmatrix},$$

where each $\times$ is non-zero. If we chose A_1 as the sub-matrix corresponding to the first two columns of A, A_2 has rank two, while if A_1 were made up of columns one and three, A_2 then has rank one. This simple example indicates how the choice of A_1 may effect the iteration bounds obtained in Theorems 3–5, and significantly, leads us to ask just how much we can reduce the bounds indicated in these theorems by judicious choice of A_1. We plan to investigate this issue in future.

Acknowledgment

This work was supported in part by the O.U.C.L. Doctoral Training Account and EPSRC grant GR/S42170. Thanks are due to Mario Arioli, Iain Duff and John Reid for fruitful discussions on various aspects of this work.

References

1. M. Avriel. *Nonlinear Programming: Analysis and Methods.* Prentice-Hall, Englewood Cliffs, New Jersey, 1976.
2. R. H. Bartels and G. H. Golub. The simplex method of linear programming using lu decompositions. *Communications of the ACM*, 12:266–268, 1969.
3. L. Bergamaschi, J. Gondzio, and G. Zilli. Preconditioning indefinite systems in interior point methods for optimization. *Computational Optimization and Applications*, 28:149–171, 2004.
4. G. Biros and O. Ghattas. A Lagrange-Newton-Krylov-Schur method for pde-constrained optimization. *SIAG/Optimiztion Views-and-News*, 11(2):12–18, 2000.
5. J. H. Bramble and J. E. Pasciak. A preconditioning technique for indefinite systems resulting from mixed approximations of elliptic problems. *Mathematics of Computation*, 50:1–17, 1988.
6. R. H. Byrd, M. E. Hribar, and J. Nocedal. An interior point algorithm for large scale nonlinear programming. *SIAM Journal on Optimization*, 9(4):877–900, 2000.
7. Y. Chabrillac and J.-P. Crouzeix. Definiteness and semidefiniteness of quadratic forms revisited. *Linear Algebra and its Applications*, 63:283–292, 1984.
8. T. F. Coleman and A. Pothen. The null space problem I: complexity. *SIAM Journal on Algebraic and Discrete Methods*, 7(4):527–537, 1986.
9. T. F. Coleman and A. Pothen. The null space problem II: algorithms. *SIAM Journal on Algebraic and Discrete Methods*, 8(4):544–563, 1987.
10. A. R. Conn, N. I. M. Gould, D. Orban, and Ph. L. Toint. A primal-dual trust-region algorithm for non-convex nonlinear programming. *Mathematical Programming*, 87(2):215–249, 2000.
11. R. W. Cottle. Manifestations of the Schur complement. *Linear Algebra and its Applications*, 8:189–211, 1974.
12. G. Debreu. Definite and semidefinite quadratic forms. *Econometrica*, 20(2):295–300, 1952.
13. H. S. Dollar, N. I. M. Gould, W. H. A. Schilders, and A. J. Wathen. On iterative methods and implicit-factorization preconditioners for regularized saddle-point systems. Technical Report RAL-TR-2005-011, Rutherford Appleton Laboratory, Chilton, Oxfordshire, England, 2005.
14. H. S. Dollar and A. J. Wathen. Incomplete factorization constraint preconditioners for saddle point problems. Technical Report 04/01, Oxford University Computing Laboratory, Oxford, England, 2004.
15. I. S. Duff. MA57 - a code for the solution of sparse symmetric definite and indefinite systems. *ACM Transactions on Mathematical Software*, 30(2):118–144, 2004.

16. I. S. Duff and J. K. Reid. The multifrontal solution of indefinite sparse symmetric linear equations. *ACM Transactions on Mathematical Software*, 9(3):302–325, 1983.
17. I. S. Duff and J. K. Reid. The design of MA48: a code for the direct solution of sparse unsymmetric linear systems of equations. *ACM Transactions on Mathematical Software*, 22(2):187–226, 1996.
18. C. Durazzi and V. Ruggiero. Indefinitely constrained conjugate gradient method for large sparse equality and inequality constrained quadratic problems. *Numerical Linear Algebra with Applications*, 10(8):673–688, 2002.
19. H. C. Elman, D. J. Silvester, and A. J. Wathen. *Finite-Elements and Fast Iterative Solvers: with applications in Incompressible Fluid Dynamics*. Oxford University Press, Oxford, 2005, to appear.
20. J. J. H. Forrest and J. A. Tomlin. Updating triangular factors of the basis to maintain sparsity in the product form simplex method. *Mathematical Programming*, 2(3):263–278, 1972.
21. D. M. Gay. Electronic mail distribution of linear programming test problems. Mathematical Programming Society COAL Newsletter, December 1985. See `http://www.netlib.org/lp/data/`.
22. P. E. Gill, W. Murray, D. B. Ponceleón, and M. A. Saunders. Preconditioners for indefinite systems arising in optimization. *SIAM Journal on Matrix Analysis and Applications*, 13(1):292–311, 1992.
23. P. E. Gill, W. Murray, and M. A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Journal on Optimization*, 12(4):979–1006, 2002.
24. P. E. Gill and M. A. Saunders. private communication, 1999.
25. G. H. Golub and C. Greif. On solving block-structured indefinite linear systems. *SIAM Journal on Scientific Computing*, 24(6):2076–2092, 2003.
26. N. I. M. Gould. On practical conditions for the existence and uniqueness of solutions to the general equality quadratic-programming problem. *Mathematical Programming*, 32(1):90–99, 1985.
27. N. I. M. Gould, M. E. Hribar, and J. Nocedal. On the solution of equality constrained quadratic problems arising in optimization. *SIAM Journal on Scientific Computing*, 23(4):1375–1394, 2001.
28. N. I. M. Gould, S. Lucidi, M. Roma, and Ph. L. Toint. Solving the trust-region subproblem using the Lanczos method. *SIAM Journal on Optimization*, 9(2):504–525, 1999.
29. N. I. M. Gould, D. Orban, and Ph. L. Toint. CUTEr (and SifDec), a Constrained and Unconstrained Testing Environment, revisited. *ACM Transactions on Mathematical Software*, 29(4):373–394, 2003.
30. N. I. M. Gould, D. Orban, and Ph. L. Toint. GALAHAD—a library of thread-safe fortran 90 packages for large-scale nonlinear optimization. *ACM Transactions on Mathematical Software*, 29(4):353–372, 2003.
31. C. Greif, G. H. Golub, and J. M. Varah. Augmented Lagrangian techniques for solving saddle point linear systems. Technical report, Computer Science Department, University of British Columbia, Vancouver, Canada, 2004.
32. HSL. A collection of Fortran codes for large-scale scientific computation, 2004. See `http://hsl.rl.ac.uk/`.
33. C. Keller, N. I. M. Gould, and A. J. Wathen. Constraint preconditioning for indefinite linear systems. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1300–1317, 2000.

34. L. Lukšan and J. Vlček. Indefinitely preconditioned inexact Newton method for large sparse equality constrained nonlinear programming problems. *Numerical Linear Algebra with Applications*, 5(3):219–247, 1998.
35. B. A. Murtagh and M. A. Saunders. Large-scale linearly constrained optimization. *Mathematical Programming*, 14(1):41–72, 1978.
36. B. A. Murtagh and M. A. Saunders. A projected Lagrangian algorithm and its implementation for sparse non-linear constraints. *Mathematical Programming Studies*, 16:84–117, 1982.
37. J. Nocedal and S. J. Wright. *Large sparse numerical optimization.* Series in Operations Research. Springer Verlag, Heidelberg, Berlin, New York, 1999.
38. L. A. Pavarino. Preconditioned mixed spectral finite-element methods for elasticity and Stokes problems. *SIAM Journal on Scientific Computing*, 19(6):1941–1957, 1998.
39. I. Perugia and V. Simoncini. Block-diagonal and indefinite symmetric preconditioners for mixed finite element formulations. *Numerical Linear Algebra with Applications*, 7(7-8):585–616, 2000.
40. W. Schilders. A preconditioning technique for indefinite systems arising in electronic circuit simulation. Talk at the one-day meeting on preconditioning methods for indefinite linear systems, TU Eindhoven, December 9, 2002.
41. R. B. Schnabel and E. Eskow. A revised modified Cholesky factorization algorithm. *SIAM Journal on Optimization*, 9(4):1135–1148, 1999.
42. R. J. Vanderbei and D. F. Shanno. An interior point algorithm for nonconvex nonlinear programming. *Computational Optimization and Applications*, 13:231–252, 1999.
43. R. C. Whaley, A. Petitet, and J. Dongarra. Automated empirical optimization of software and the ATLAS project. *Parallel Computing*, 27(1–2):3–35, 2001.

Optimal algorithms for large sparse quadratic programming problems with uniformly bounded spectrum

Zdeněk Dostál

VŠB-Technical University Ostrava, Tř 17. listopadu, CZ-70833 Ostrava, Czech Republic (zdenek.dostal@vsb.cz)

Summary. Recently proposed algorithms for the solution of large quadratic programming problems are reviewed. An important feature of these algorithms is their capability to find an approximate solution of the convex equality and/or bound constrained quadratic programming problems with the uniformly bounded spectrum of the Hessian matrix at $O(1)$ iterations. The theoretical results are presented and illustrated by numerical experiments.

Key words: quadratic programming, box and equality constraints, augmented Lagrangians, adaptive precision control.

1 Introduction

An important ingredient in development of effective methods for the solution of very large problems is identification of algorithms that can solve some special classes of problems with optimal (i.e. asymptotically linear) complexity. For example, the interior point methods were applied successfully to the solution of very large problems of nonlinear optimization with many constraints and tens of thousands of decision variables observing that the Hessian matrix with a special pattern of the nonzero elements may be decomposed with nearly linear complexity ([18]).

Another important class of problems may be characterized by distribution of the spectrum of the Hessian matrix. For example, the major breakthrough in the development of effective algorithms for numerical solution of elliptic partial differential equations was the observation that application of the domain decomposition methods [25] or multigrid methods [19] to these problems may result in the class of unconstrained QP problems with the cost functions whose Hessian matrices are very sparse and have their spectrum in a given positive interval. Since there are well known algorithms, such as the conjugate gradient method, with the rate of convergence dependent on the condition

number of the Hessian matrix, it followed easily that the discretized elliptic problems may be solved with optimal (i. e. asymptotically linear) complexity.

Until recently, there were no theoretical results like this for inequality constrained problems, the only exception known to the author being the results by J. Schöberl based on the gradient projection [26,27]. However, experimental results were reported by several authors ([1,9,11,21–24]). The lack of the theoretical results is not very surprising if we realize that any result of this type requires identification of the active constraints for free and the rate of convergence independent of conditioning of the constraints. Let us point out that even though the Hessian matrix of any of the problems we are talking about is sparse, its sparsity pattern typically does not allow fast decomposition, so that the complexity of even single step of the interior point methods is far from optimal.

In this paper, we review our optimal algorithms [4–7,16] for the solution of convex bound and/or equality constrained quadratic programming problems. These algorithms can be implemented in such a way that, for the class of problems with the spectrum of the Hessian matrix in a given positive interval, they can find approximate solutions of each particular problem at the cost proportional to that of the matrix-vector multiplication. In combination with our variants of FETI [13, 14] or FETI-DP [15] methods, these algorithms turned out to be a powerful engine in development of scalable algorithms for numerical solution of elliptic variational inequalities. Numerical experiments with the solution of variational inequalities discretized by more than eight millions of nodal variables confirmed that the algorithms extend both the numerical and parallel scalability of the linear FETI method (e.g. [17]) to constrained problems.

We demonstrate the performance of our algorithms by solving for each type of the constraints the class of well conditioned problems of varying dimensions with the quadratic form q_t defined for $t \in \{2, 3, \dots\}$ by the symmetric Toeplitz matrix $A = A_t$ of the order $2 * t^2$ determined by the nonzero entries $a_{11} = 12, a_{12} = a_{1,t} = -1$ and by the vectors $b = b_t$ defined by the entries $b_i = -1, i = 1, \dots, 2 * t$. Using the Gershgorin theorem, it is easy to see that the eigenvalues λ_i of any A_t satisfy $8 \leq \lambda_i \leq 16$. The equality constraints were defined by the matrix $C = C_t$ with t rows comprising $2 * t^2$ entries which are zeros except $c_{i,t^2-i+1} = 1$ and $c_{i,t^2+i} = -1, i = 1, \dots, t$.

In the whole paper, $q(x) = \frac{1}{2}x^T Ax - b^T x$ will always denote a strictly convex quadratic function defined on $\mathbb{R}^n$ with the Hessian matrix $\nabla^2 q = A \in \mathbb{R}^{n \times n}$ symmetric positive definite and $x, b \in \mathbb{R}^n$. The eigenvalues of A will be denoted $\lambda_i(A)$,

$$\lambda_{\min}(A) = \lambda_1(A) \leq \cdots \leq \lambda_n(A) = \lambda_{\max}(A) = \|A\|.$$

The Euclidean norm and the A–energy norm of x will be denoted by $\|x\|$ and $\|x\|_A$, respectively. Thus $\|x\|^2 = x^T x$ and $\|x\|_A^2 = x^T Ax$. Analogous notation will be used for the induced matrix norms.

2 Equality constrained problems

We shall start with the problem of finding the minimizer of the quadratic function $q(x)$ subject to the linear equality constraints, that is

$$\text{minimize } q(x) \text{ subject to } x \in \Omega_E \tag{1}$$

with $\Omega_E = \{x \in \mathbb{R}^n : Cx = d\}, C \in \mathbb{R}^{m \times n}$, and $d \in \mathbb{R}^m$. We require neither that C is a full row rank matrix nor $m \leq n$, but we shall assume that d belongs to the range of C to guarantee that Ω_E is not empty. Our development is based on the augmented Lagrangian method [2] which reduces (1) to a sequence of the problems of the form

$$\text{minimize } L(x, \mu^k, \rho_k) \text{ subject to } x \in \mathbb{R}^p \tag{2}$$

where

$$L(x, \mu^k, \rho_k) = q(x) + (\mu^k)^T(Cx - d) + \frac{\rho_k}{2}\|Cx - d\|^2 \tag{3}$$

is known as the augmented Lagrangian function, $\mu^k = (\mu_1^k, \ldots, \mu_m^k)^T$ is the vector of the Lagrange multipliers for the equality constraints, and ρ_k is the penalty parameter. The precision of the approximate solution x^k of the auxiliary problems will be measured by the Euclidian norm of the feasibility error and of the gradient of the augmented Lagrangian. The latter is always denoted by g, so that

$$g(x, \mu, \rho) = \nabla_x L(x, \mu, \rho) \quad = \quad Ax - b + C^T\mu + \rho C^T(Cx - d). \tag{4}$$

Our algorithm with the adaptive precision control reads as follows.

Algorithm 1. Semi-monotonic augmented Lagrangians for equality constraints (SMALE)
Given $\eta > 0$, $\beta > 1$, $M > 0$, $\rho_0 > 0$, and $\mu^0 \in \mathbb{R}^m$, set $k = 0$.
Step 1. {Inner iteration with adaptive precision control.}
Find x^k such that

$$\|g(x^k, \mu^k, \rho_k)\| \leq \min\{M\|Cx^k - d\|, \eta\}. \tag{5}$$

Step 2. {Update μ.}

$$\mu^{k+1} = \mu^k + \rho_k(Cx^k - d). \tag{6}$$

Step 3. {Update ρ provided the increase of the Lagrangian is not sufficient.}
If $k > 0$ and

$$L(x^k, \mu^k, \rho_k) < L(x^{k-1}, \mu^{k-1}, \rho_{k-1}) + \frac{\rho_k}{2}\|Cx^k - d\|^2 \tag{7}$$

then

$$\rho_{k+1} = \beta\rho_k, \tag{8}$$

else

$$\rho_{k+1} = \rho_k. \tag{9}$$

Step 4. Set $k = k + 1$ and return to the Step 1.

In Step 1 we can use any convergent algorithm for minimizing the strictly convex quadratic function such as the conjugate gradient method [2]. Algorithm 1 differs from those considered by Hager [20] and Dostál, Friedlander and Santos [8] by the condition on the update of the penalization parameter in Step 3.

Algorithm 1 has been proved to be correctly defined and to enjoy a kind of optimal convergence of the feasibility error [4]. To present our optimality result related to the conjugate gradient implementation of Step 1, let $\mathcal{T}$ denote any set of indices and assume that for any $t \in \mathcal{T}$ there is defined a problem

$$\text{minimize } q_t(x) \text{ subject to } x \in \Omega_E^t \tag{10}$$

with $\Omega_E^t = \{x \in I\!R^{n_t} : C_t x = 0\}$, $q_t(x) = \frac{1}{2} x^T A_t x - b_t^T x$, $A_t \in I\!R^{n_t \times n_t}$ symmetric positive definite, $C_t \in I\!R^{m_t \times n_t}$, and $b_t, x \in I\!R^{n_t}$.

Theorem 1. *Let $\{x_t^k\}, \{\mu_t^k\}$ and $\{\rho_{t,k}\}$ be generated by Algorithm 1 for (10) with $\|b_t\| \geq \eta_t > 0$, $\beta > 1$, $M > 0$, $\rho_{t,0} = \rho_0 > 0$, $\mu_t^0 = 0$. Let $0 < a_{\min} < a_{\max}$ and $0 < c_{\min} < c_{\max}$ be given constants. Let Step 1 be implemented by the conjugate gradient method which generates the iterates $x_t^{k,0}, x_t^{k,1}, \ldots, x_t^{k,l} = x_t^k$ for the solution of (10) starting from $x_t^{k,0} = x_t^{k-1}$ with $x_t^{-1} = 0$, where $l = l_{k_t}$ is the first index satisfying*

$$\|g(x_t^{k,l}, \mu_t^k, \rho_k)\| \leq M \|C_t x_t^{k,l}\| \tag{11}$$

or

$$\|g(x_t^{k,l}, \mu_t^k, \rho_k)\| \leq \epsilon \|b_t\| \min\{1, M\}. \tag{12}$$

Let the class of problems (10) satisfies

$$a_{\min} \leq \lambda_{\min}(A_t) \leq \lambda_{\max}(A_t) \leq a_{\max}, \quad c_{\min} \leq \sigma_{\min}(C_t) \leq \|C_t\| \leq c_{\max}, \tag{13}$$

where $\sigma_{\min}(C_t)$ denotes the least nonzero singular value of C_t. Then the following statements hold:
(i) Algorithm 1 generates an approximate solution $x_t^{k_t}$ of any problem (10) which satisfies

$$\|x^k - \overline{x}\| \leq \epsilon \|b_t\| \tag{14}$$

at $O(1)$ matrix-vector multiplications by the Hessian of the augmented Lagrangian L_t for (10).
(ii) The images of the Lagrange multipliers $C^T \mu^k$ are bounded and converge to $C^T \overline{\mu}$, where $\overline{\mu}$ denotes any vector of Lagrange multipliers of the solution.

Proof: See [6].

We have implemented Algorithm 1 in Matlab and solved a class of problems of the varying dimension defined in the introduction. We solved the problem with $\eta_t = \|b_t\|, \beta = 10, \rho = 200, M = 1$ and $\mu_0 = 0$ using the stopping criterion $\|g_t(x, \mu, \rho)\| \leq 10^{-5}\|b_t\|$ and $\|C_t x\| \leq 10^{-5}\|b_t\|$. The results are in Table 1.

Table 1: Performance of SMALE

Equality constrains =bandwidth	Dimension n	cg iterations	Outer iterations
10	200	25	4
50	5000	22	4
100	20000	18	3
250	125000	18	3
500	500000	17	3

We conclude that we can observe optimality in practice for well conditioned problems. More numerical experiments and theoretical results may be found in [4, 6].

3 Bound constrained problems

We shall now be concerned with the problem

$$\text{minimize } q(x) \text{ subject to } x \in \Omega_B \tag{15}$$

with $\Omega_B = \{x : x \geq \ell\}$ and $\ell \in I\!R^n$. The unique solution $\overline{x}$ of (15) is fully determined by the Karush-Kuhn-Tucker optimality conditions [2] so that for $i = 1, \ldots, n$,

$$\overline{x}_i = \ell_i \text{ implies } \overline{g}_i \geq 0 \text{ and } \overline{x}_i > \ell_i \text{ implies } \overline{g}_i = 0 \tag{16}$$

where $g = g(x)$ denotes the gradient of q defined by

$$g = g(x) = Ax - b. \tag{17}$$

The conditions (16) can be described alternatively by the *free gradient* φ and the *chopped gradient* β that are defined by

$$\varphi_i(x) = g_i(x) \text{ for } x_i > \ell_i, \varphi_i(x) = 0 \text{ for } x_i = \ell_i$$

$$\beta_i(x) = 0 \text{ for } x_i > \ell_i, \beta_i(x) = g_i^-(x) \text{ for } x_i = \ell_i$$

where we have used the notation $g_i^- = \min\{g_i, 0\}$. Thus the conditions (16) are satisfied iff the *projected gradient* $g^P(x) = \varphi(x) + \beta(x)$ is equal to the zero.

The algorithm for the solution of (15) that we describe here exploits a given constant $\Gamma > 0$, a test to decide about leaving the face and three types of steps to generate a sequence of the iterates $\{x^k\}$ that approximate the solution of (15).

The *expansion step* may expand the current active set and is defined by

$$x^{k+1} = x^k - \overline{\alpha}\widetilde{\varphi}(x^k) \tag{18}$$

with the fixed steplength $\overline{\alpha} \in (0, \|A\|^{-1}]$ and the *reduced free gradient* $\widetilde{\varphi}(x)$ with the entries $\widetilde{\varphi}_i = \widetilde{\varphi}_i(x) = \min\{(x_i - \ell_i)/\overline{\alpha}, \varphi_i\}$.

If the inequality

$$\|\beta(x^k)\|^2 \le \Gamma^2 \widetilde{\varphi}(x^k)^T \varphi(x^k) \tag{19}$$

holds then we call the iterate x^k *strictly proportional*. The test (19) is used to decide which component of the projected gradient $g^P(x^k)$ will be reduced in the next step.

The *proportioning step* may remove indices from the active and is defined by

$$x^{k+1} = x^k - \alpha_{cg}\beta(x^k) \tag{20}$$

with the steplength α_{cg} that minimizes $q\left(x^k - \alpha\beta(x^k)\right)$. It is easy to check [2] that α_{cg} that minimizes $q(x - \alpha d)$ for a given d and x may be evaluated by the formula

$$\alpha_{cg} = \alpha_{cg}(d) = \frac{d^T g(x)}{d^T A d}. \tag{21}$$

The *conjugate gradient step* is defined by

$$x^{k+1} = x^k - \alpha_{cg} p^k \tag{22}$$

where p^k is the conjugate gradient direction [2] which is constructed recurrently. The recurrence starts (or restarts) from $p^s = \varphi(x^s)$ whenever x^s is generated by the expansion or proportioning step. If p^k is known, then p^{k+1} is given [2] by

$$p^{k+1} = \varphi(x^{k+1}) - \gamma p^k, \quad \gamma = \frac{\varphi(x^{k+1})^T A p^k}{(p^k)^T A p^k}. \tag{23}$$

Algorithm 2. Modified proportioning with reduced gradient projections (MPRGP)

Let $x^0 \in \Omega$, $\overline{\alpha} \in (0, \|A\|^{-1}]$, and $\Gamma > 0$ be given. For $k \ge 0$ and x^k known, choose x^{k+1} by the following rules:

Step 1. If $g^P(x^k) = 0$, set $x^{k+1} = x^k$.

Step 2. If x^k is strictly proportional and $g^P(x^k) \ne 0$, try to generate x^{k+1} by the conjugate gradient step. If $x^{k+1} \in \Omega$, then accept it, else use the expansion step.

Step 3. If x^k is not strictly proportional, define x^{k+1} by proportioning.

Algorithm 2 has been proved to enjoy the R-linear rate of convergence in terms of the spectral condition number [16]. To formulate the optimality results, let $\mathcal{T}$ denote any set of indices and assume that for any $t \in \mathcal{T}$ there is defined the problem

$$\text{minimize } q_t(x) \text{ subject to } x \in \Omega_B^t \tag{24}$$

with $\Omega_B^t = \{x \in I\!R^{n_t} : x \geq \ell\}$, $q_t(x) = \frac{1}{2}x^T A_t x - b_t^T x$, $A_t \in I\!R^{n_t \times n_t}$ symmetric positive definite, and $b_t, x, \ell_t \in I\!R^{n_t}$. Our optimality result then reads as follows.

Theorem 2. *Let the Hessian matrices $A_t = \nabla^2 q_t$ of (24) satisfy*

$$0 < a_{\min} \leq \lambda_{\min}(A_t) \leq \lambda_{\max}(A_t) \leq a_{\max},$$

let $\{x_t^k\}$ be generated by Algorithm 2 for (24) with a given $x_t^0 \in \Omega_B^t$, $\overline{\alpha} \in (0, a_{\max}^{-1}]$, and let $\Gamma > 0$. Let there be a constant a_b such that $\|x_t^0\| \leq a_b \|b_t\|$ for any $t \in \mathcal{T}$.
(i) If $\epsilon > 0$ is given, then the approximate solution $\overline{x}_t$ of (24) which satisfies

$$\|x_t^k - \overline{x}_t\| \leq \epsilon \|b_t\|$$

may be obtained at $O(1)$ matrix-vector multiplications by A_t.
(ii) If $\epsilon > 0$ is given, then the approximate solution x_t^k of (24) which satisfies

$$\|g_t^P(x_t^k)\| \leq \epsilon \|b_t\|$$

may be obtained at $O(1)$ matrix-vector multiplications by A_t.

Proof: See [7, 16].

Table 2: Performance of MPRGP

Bandwidth	Dimension n	Active constraints	Matrix-vector multiplications
10	200	48	11
50	5000	1244	16
100	20000	5049	18
250	125000	31940	20
500	500000	128370	22
710	1008200	259206	22

The results indicate that we can observe optimality for well conditioned problems. More numerical experiments and implementation details may be found in [16].

4 Bound and equality constrained problems

We shall be finally concerned with the problem of finding the minimizer of the strictly convex quadratic function $q(x)$ subject to the bound and linear equality constraints, that is

$$\text{minimize } q(x) \quad \text{subject to} \quad x \in \Omega_{BE} \tag{25}$$

with $\Omega_{BE} = \{x \in I\!R^n : x \geq \ell \text{ and } Cx = 0\}$ and $C \in I\!R^{m \times n}$. We do not require that C is a full row rank matrix, but we shall assume that Ω is not empty. Let us point out that confining ourselves to the homogeneous equality constraints does not mean any loss of generality, as we can use a simple transform to reduce any non-homogeneous equality constraints to our case. The algorithm that we describe here combines in a natural way the algorithms SMALE and MPRGP described above. It is related to the earlier work of Friedlander and Santos with the present author [10]. Let us recall that the basic scheme that we use was proposed by Conn, Gould and Toint [3] who adapted the augmented Lagrangian method to the solution of the problems with a general cost function subject to general equality constraints and simple bounds.

Algorithm 3. Semi-monotonic augmented Lagrangians for bound and equality constraints (SMALBE)
Given $\eta > 0$, $\beta > 1$, $M > 0$, $\rho_0 > 0$, and $\mu^0 \in I\!R^m$, set $k = 0$.
Step 1. {Inner iteration with adaptive precision control.}
Find x^k such that

$$\|g^P(x^k, \mu^k, \rho_k)\| \leq \min\{M\|Cx^k\|, \eta\}. \tag{26}$$

Step 2. {Update μ.}

$$\mu^{k+1} = \mu^k + \rho_k C x^k. \tag{27}$$

Step 3. {Update ρ provided the increase of the Lagrangian is not sufficient.}
If $k > 0$ and

$$L(x^k, \mu^k, \rho^k) < L(x^{k-1}, \mu^{k-1}, \rho_{k-1}) + \frac{\rho_k}{2}\|Cx^k\|^2 \tag{28}$$

then

$$\rho_{k+1} = \beta \rho_k, \tag{29}$$

else

$$\rho_{k+1} = \rho_k. \tag{30}$$

Step 4. Set $k = k + 1$ and return to *Step 1.*

Algorithm 3.1 has been shown to be well defined [10], that is, any convergent algorithm for the solution of the auxiliary problem required in Step 1 which guarantees convergence of the projected gradient to zero will generate either x^k that satisfies (26) in a finite number of steps or a sequence of

approximations that converges to the solution of (25). To present explicitly the optimality of Algorithm 3 with Step 1 implemented by Algorithm 2, let $\mathcal{T}$ denote any set of indices and let for any $t \in \mathcal{T}$ be defined the problem

$$\text{minimize } q_t(x) \text{ subject to } x \in \Omega^t_{BE} \tag{31}$$

with $\Omega^t_{BE} = \{x \in I\!R^{n_t} : C_t x = 0 \text{ and } x \geq \ell_t\}$, $q_t(x) = \frac{1}{2} x^T A_t x - b_t^T x$, $A_t \in I\!R^{n_t \times n_t}$ symmetric positive definite, $C_t \in I\!R^{m_t \times n_t}$, and $b_t, \ell_t \in I\!R^{n_t}$. Our optimality result reads as follows.

Theorem 3. *Let $\{x_t^k\}, \{\mu_t^k\}$ and $\{\rho_{t,k}\}$ be generated by Algorithm 3 for (31) with $\|b_t\| \geq \eta_t > 0$, $\beta > 1$, $M > 0$, $\rho_{t,0} = \rho_0 > 0$, $\mu_t^0 = 0$. Let Step 1 of Algorithm 3 be implemented by Algorithm 2 (MPRGP) which generates the iterates $x_t^{k,0}, x_t^{k,1}, \ldots, x_t^{k,l} = x_t^k$ for the solution of (31) starting from $x_t^{k,0} = x_t^{k-1}$ with $x_t^{-1} = 0$, where $l = l_{k_t}$ is the first index satisfying*

$$\|g^P(x_t^{k,l}, \mu_t^k, \rho_k)\| \leq M \|C_t x_t^{k,l_k}\| \tag{32}$$

or

$$\|g^P(x_t^{k,l}, \mu_t^k, \rho_k)\| \leq \epsilon \|b_t\| \min\{1, M^{-1}\}. \tag{33}$$

Let $0 < a_{\min} < a_{\max}$ and $0 < c_{\max}$ be given and let the class of problems (31) satisfies

$$a_{\min} \leq \lambda_{\min}(A_t) \leq \lambda_{\max}(A_t) \leq a_{\max} \quad \text{and} \quad \|C_t\| \leq c_{\max}. \tag{34}$$

Then Algorithm 3 generates an approximate solution $x_t^{k_t}$ of any problem (31) which satisfies

$$\|g^P(x_t^{k_t}, \mu_t^{k_t}, \rho_{t,k_t})\| \leq \epsilon \|b_t\| \quad \text{and} \quad \|C_t x_t^{k_t}\| \leq \epsilon \|b_t\| \tag{35}$$

at $O(1)$ matrix-vector multiplications by the Hessian of the augmented Lagrangian L_t.

Proof: See [7].

We have implemented Algorithm 3 in Matlab and solved the class of well conditioned problems of the varying dimensions specified in the introduction. We solved the problem with $\eta_t = \|b_t\|, \beta = 10, \rho = 200, M = 1$ and $\mu_0 = 0$ using the stopping criterion $\|g_t(x, \mu, \rho)\| \leq 10^{-5}\|b_t\|$, $\|C_t x\| \leq 10^{-5}\|b_t\|$. The results are in Table 3.

5 Conclusions

Theoretical results concerning optimality of the recently proposed algorithms for bound and/or equality constrained quadratic programming were presented

Table 3: Performance of SMALBE

Equality constrains =bandwidth	Dimension n	Active constraints	cg iterations	Outer iterations
10	200	57	38	9
50	5000	1239	41	9
100	20000	4997	39	8
250	125000	31193	44	8
500	500000	124887	44	8

and illustrated by numerical experiments. The unique feature of the presented algorithms is their capability to find the approximate solution of the class of problems with the uniformly bounded spectrum of the Hessian matrix at $O(1)$ matrix-vector multiplications. No assumptions concerning regularity of solution are used and the results are valid even for linearly dependent constraints. The algorithms were combined with FETI [5, 9, 11, 12] and FETI-DP [15] domain decomposition methods to develop scalable algorithms for numerical solution of elliptic variational inequalities and contact problems of elasticity and tested on problems discretized by up to more than 8 millions of nodal variables.

References

1. P. Avery, G. Rebel, M. Lesoinne, C. Farhat, *A umerically scalable dual-primal substructuring method for the solution of contact problems I: the frictionless case*, Comput. Methods Appl. Mech. Eng. 193, 23-26 (2004) 2403-2426.
2. D. P. Bertsekas, *Nonlinear Optimization*, Athena Scientific, Belmont (1999).
3. A. R. Conn, N. I. M. Gould, Ph. L. Toint, *LANCELOT: a Fortran package for large scale nonlinear optimization*, Springer Verlag, Berlin (1992).
4. Z. Dostál, Semi-monotonic inexact augmented Lagrangians for quadratic programming with equality constraints, to appear in *Optimization Methods & Software*.
5. Z. Dostál, Inexact semi-monotonic Augmented Lagrangians with optimal feasibility convergence for quadratic programming with simple bounds and equality constraints, *SIAM Journal for Numerical Analysis* 43,1 (2005) 96-115.
6. Z. Dostál, An optimal algorithm for a class of equality constrained quadratic programming problems with bounded spectrum, submitted.
7. Z. Dostál, An optimal algorithm for a class of bound and equality constrained quadratic programming problems with bounded spectrum, submitted.
8. Z. Dostál, A. Friedlander, S. A. Santos, Augmented Lagrangians with adaptive precision control for quadratic programming with equality constraints, *Computational Optimization and Applications,* 14 (1999) 37-53.
9. Z. Dostál, F. A. M. Gomes, S. A. Santos, Solution of contact problems by FETI domain decomposition with natural coarse space projection, *Computer Meth. in Appl. Mech. and Engineering* 190,13-14 (2000) 1611-1627.

10. Z. Dostál, A. Friedlander, S. A. Santos, Augmented Lagrangians with Adaptive Precision Control for Quadratic Programming with Simple Bounds and Equality constraints, *SIAM Journal on Optimization* 13,4 (2003) 1120-1140.
11. Z. Dostál, D. Horák, Scalability and FETI based algorithm for large discretized variational inequalities, *Math. and Comput. in Simulation* 61,3-6 (2003) 347-357.
12. Z. Dostál, D. Horák, Scalable FETI with Optimal Dual Penalty for a Variational Inequality, *Numerical Linear Algebra and Applications* 11, 5-6 (2004) 455 - 472.
13. Z. Dostál, D. Horák, On scalable algorithms for numerical solution of variational inequalities based on FETI and semi-monotonic augmented Lagrangians, *Domain Decomposition Methods in Science and Engineering*, eds. R. Kornhuber, R. H. W. Hoppe, J. Périaux, O. Pironneau, O. B. Widlund and J. Xu, Lecture Notes in Computational Science and Engineering, vol. 40, Berlin (2005) 487-494.
14. Z. Dostál, D. Horák, R. Kučera, V. Vondrák, J. Haslinger, J. Dobiáš, S. Pták, FETI based algorithms for contact problems: scalability, large displacements and 3D Coulomb friction, *Computer Methods in Applied Mechanics and Engineering* 194, 2-5 (2005) 395-409.
15. Z. Dostál, D. Horák, D. Stefanica, A scalable FETI-DP algorithm for coercive variational inequalities, *IMACS J. Appl. Num. Math.* 54, 3-4 (2005) 378-390.
16. Z. Dostál, J. Schöberl, Minimizing quadratic functions over non-negative cone with the rate of convergence and finite termination, *Computational Optimization and Applications* 30,1 (2005) 23-44.
17. C. Farhat, M. Lesoinne, K. Pierson, A scalable dual-primal domain decomposition method, *Numer. Linear Algebra Appl.* 7, No.7-8, 687-714 (2000).
18. J. Gondzio, R. Sarkissian, Parallel interior point solver for structured linear programs, *Mathematical Programming* 96,3 (2003) 561-584.
19. W. Hackbusch, *Multi-Grid Methods and Applications*, Springer-Verlag, Berlin 1985.
20. W. W. Hager, Analysis and implementation of a dual algorithm for constraint optimization, *JOTA*, 79 (1993) 37-71.
21. R. Kornhuber, *Adaptive monotone multigrid methods for nonlinear variational problems*, Teubner-Verlag, Stuttgart (1997).
22. R. Kornhuber, R. Krause, Adaptive multigrid methods for Signorini's problem in linear elasticity, *Computer Visualization in Science* 4,1 (2001) 9-20.
23. R. Krause, O. Sander, Fast solving of contact problems on complicated geometries, eds. R. Kornhuber, R. H. W. Hoppe, J. Périaux, O. Pironneau, O. B. Widlund and J. Xu, Lecture Notes in Computational Science and Engineering, vol. 40, Berlin (2005) 495-502.
24. R. Krause, B. I. Wohlmuth, A Dirichlet-Neumann type algorithm for contact problems with friction, *Computer Visualization in Science* 5, 3 (2002)139-148.
25. J. Mandel, R. Tezaur, Convergence of substructuring method with Lagrange multipliers, *Numerische Mathematik* 73 (1996) 473-487.
26. J. Schöberl, Solving the Signorini problem on the basis of domain decomposition techniques, *Computing* 60,4 (1998) 323-344.
27. J. Schöberl, Efficient contact solvers based on domain decomposition techniques, *Computers & Mathematics* 42 (1998) 1217-1228.

Numerical methods for separating two polyhedra

Yury G. Evtushenko, Alexander I. Golikov, and Saed Ketabchi

Dorodnicyn Computing Centre, Russian Academy of Sciences, Vavilov 40, Moscow, 119991, Russia ({evt,gol}@ccas.ru, saed_ketabchi@yahoo.com)

Summary. The problem of finding a family of parallel hyperplanes that separates two disjoint nonempty polyhedra is examined. The polyhedra are given by systems of linear inequalities or by systems of linear equalities with nonnegative variables. Constructive algorithms for solving these problems are presented. The proposed approach is based on the theorems of alternative.

Key words: polyhedra, separating hyperplane, theorems of alternative.

1 Introduction

The theorem on the existence of a separating hyperplane plays a key role in functional analysis, optimization theory, and operations research. In solving practical problems, one should not only know that there exists a separating hyperplane but also be able to constructively determine it. The method for finding a hyperplane that separates two polyhedra defined by a system of inequalities was developed by I.I. Eremin (see [1, Theorem 10.1]).

In this paper we propose the numerical methods for construction of a family of parallel hyperplanes that separate two disjoint nonempty polyhedra given by systems of linear inequalities or by systems of linear equalities with nonnegative variables. Our approach is based on new theorems of alternative described in [2,3].

In Section 2 we consider two polyhedra defined by systems of linear inequalities. If polyhedra are disjoint then the union of these systems (original system) is inconsistent. The alternative system is solved by minimizing the residuals of the inconsistent original system. The results of this minimization are used to find the normal solution (with a minimal Euclidean norm) to the alternative system. The replacement of the alternative system by the minimization of the residuals of the original system may be advantageous when the dimension of the new variables is less than that of starting ones. In this case,

such a reduction results in minimization problem in a space of lower dimension and allows one to obtain the normal solution to the alternative system. After finding the normal solution of the alternative system we construct the family of parallel hyperplanes which separates the polyhedra.

In Section 3, we construct families of separating hyperplanes for two polyhedra given by systems of linear equalities with nonnegative variables. In contrast to the previous case, every solution to the alternative system determines *two distinct* families of separating hyperplanes.

In Section 4, we examine the following problem: how can one distinguish a solution to the alternative system that generates a family of separating hyperplanes with a maximal thickness, which coincides with the minimal distance between the polyhedra?

In Section 5, we give a brief review of the generalized Newton method for calculating the normal solution to the alternative system. The normal solution is used for constructing a family of separating hyperplanes for polyhedra given by systems of linear inequalities. The generalized Newton method was implemented in Matlab and showed a good performance in solving large-scale test problems.

2 Separation of polyhedra defined by inequalities

Let $x \in \mathbb{R}^n$ and $b \in \mathbb{R}^m$, where $\|b\| \neq 0$, be given vectors and $A \in \mathbb{R}^{m \times n}$ be a given rectangular matrix. Define the two sets

$$X = \{x \in \mathbb{R}^n : Ax \geq b\}, \qquad U = \{u \in \mathbb{R}^m : A^\top u = 0_n,\ b^\top u = \rho,\ u \geq 0_m\},$$

where $\rho > 0$ is an arbitrary fixed positive number and 0_i is the zero vector of dimension i. The linear systems

$$Ax \geq b, \tag{1}$$

$$A^\top u = 0_n, \qquad b^\top u = \rho, \qquad u \geq 0_m \tag{2}$$

which determine the sets X and U, respectively, are alternative for any strictly positive value of ρ, which means that exactly one of them is consistent (this is the Gale theorem; e.g., see [2,3]). We take the scalar products of both sides of the first equality in (2) with the vector x and then subtract the second equality from the resulting relation. This yields

$$u^\top (Ax - b) = -\rho < 0. \tag{3}$$

This equality is a key tool for constructing a family of hyperplanes that separate two polyhedra given as intersections of half-spaces. We write A, b, and u in the form

$$A = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix}, \qquad b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}, \qquad u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix},$$

where A_1 and A_2 are matrices of sizes $m_1 \times n$ and $m_2 \times n$, respectively; $b_1, u_1 \in \mathbb{R}^{m_1}$; $b_2, u_2 \in \mathbb{R}^{m_2}$; and $m_1 + m_2 = m$. Define the two nonempty sets

$$X_1 = \{x \in \mathbb{R}^n : A_1 x \geq b_1\}, \qquad X_2 = \{x \in \mathbb{R}^n : A_2 x \geq b_2\},$$

which determine two polyhedra (or polyhedral sets; see [4]) such that $X = X_1 \cap X_2 = \emptyset$. Define the hyperplane $c^\top x - \gamma = 0$, where $c \in \mathbb{R}^n$, $\|c\| \neq 0$ is a normal vector and γ is a scalar. We say that this hyperplane separates X_1 and X_2 if $c^\top x - \gamma \geq 0$ for all $x \in X_1$ and $c^\top x - \gamma \leq 0$ for all $x \in X_2$. If we have the strict inequalities in both conditions, then we say that this hyperplane strictly separates X_1 and X_2.

Consider the problem of calculating the hyperplanes that separate X_1 and X_2. Taking into account the partition introduced above, we can rewrite systems (1) and (2) and relation (3) as follows:

$$A_1 x \geq b_1, \qquad A_2 x \geq b_2, \tag{4}$$

$$A_1^\top u_1 + A_2^\top u_2 = 0_n, \quad b_1^\top u_1 + b_2^\top u_2 = \rho, \quad u_1 \geq 0_{m_1}, \quad u_2 \geq 0_{m_2}, \tag{5}$$

$$u_1^\top (A_1 x - b_1) + u_2^\top (A_2 x - b_2) = -\rho < 0. \tag{6}$$

Define a linear function $\varphi(x, \alpha)$ of variable $x \in \mathbb{R}^n$ and a scalar parameter α ranging on the interval $[0, 1]$:

$$\varphi(x, \alpha) = u_1^\top (A_1 x - b_1) + \alpha\rho. \tag{7}$$

Relation (6) implies that $\varphi(x, \alpha)$ can be equivalently defined as

$$\varphi(x, \alpha) = u_2^\top (b_2 - A_2 x) + (\alpha - 1)\rho. \tag{8}$$

The equality $\varphi(x, \alpha) = 0$, where u_1 and u_2 satisfy (5) and α belongs to $[0, 1]$, determines the hyperplane that separates the sets X_1 and X_2. Indeed, if $x \in X_1$, then, according to (7), we have $\varphi(x, \alpha) \geq 0$, while if $x \in X_2$, then, according to (8), we have $\varphi(x, \alpha) \leq 0$. In view of system (5), the hyperplane $\varphi(x, \alpha) = 0$ determined by a function of form (7) or (8) can be written as

$$\varphi(x, \alpha) = c^\top x - \gamma = 0,$$

where

$$c = A_1^\top u_1 = -A_2^\top u_2, \qquad \gamma = b_1^\top u_1 - \alpha\rho = -b_2^\top u_2 - (\alpha - 1)\rho.$$

Here, u_1 and u_2 are arbitrary solutions to system (5). For fixed distinct vectors $u^\top = [u_1^\top, u_2^\top]$ that satisfy system (5), we examine the family of parallel hyperplanes given by the following equivalent definitions:

$$\Gamma(\alpha) = \{x \in \mathbb{R}^n : u_1^\top A_1 x - b_1^\top u_1 + \alpha\rho = 0\} = \{x \in \mathbb{R}^n : \varphi(x, \alpha) = 0\}, \tag{9}$$

$$\Gamma(\alpha) = \{x \in \mathbb{R}^n : -u_2^\top A_2 x + b_2^\top u_2 + (\alpha - 1)\rho = 0\}. \tag{10}$$

All the hyperplanes belonging to this family are parallel, because they have the common normal vector $c = A_1^\top u_1 = -A_2^\top u_2$.

The hyperplane $\Gamma(1)$ can be obtained from $\Gamma(0)$ with the help of the shift vector y:

$$\Gamma(1) = \Gamma(0) + y.$$

The norm of y (i.e., the distance between the hyperplanes $\Gamma(1)$ and $\Gamma(0)$) will be called the *thickness of the family of hyperplanes.*

According to [3,4] the projection $\bar{x}^*$ of a point $\bar{x}$ onto the hyperplane $\Gamma(\alpha)$ is determined by the formula

$$\bar{x}^* = \bar{x} + c(b_1^\top u_1 - c^\top \bar{x} - \alpha\rho)/\|c\|^2. \tag{11}$$

Denote by $\mathrm{pr}(0_n, \Gamma(\alpha))$ the projection of the origin onto the hyperplane $\Gamma(\alpha)$. Setting $\bar{x} = 0_n$ in (11), we obtain $\mathrm{pr}(0_n, \Gamma(\alpha)) = c(b_1^\top u_1 - \alpha\rho)/\|c\|^2$. From this, we find the shift vector y and the thickness $\|y\|$ of the family of hyperplanes $\Gamma(\alpha)$:

$$y = \mathrm{pr}(0_n, \Gamma(1)) - \mathrm{pr}(0_n, \Gamma(0)) = -c\rho/\|c\|^2, \tag{12}$$

$$\|y\| = \rho/\|c\|. \tag{13}$$

I.I.Eremin was the first who proposed to use any solution to system (5) for construction the separation hyperplan utilizing function $\phi(x, \alpha)$ with $\alpha = 1/2$ [1]. System (5) may have many solutions. In this section, we examine the properties of the family of separating hyperplanes, where u is the normal solution $\tilde{u}^*$ to system (5). The results of [2,3] allow us to relatively easily construct the normal solution, i.e., to solve the following quadratic programming problem:

$$\min_{u \in U} \frac{1}{2}\|u\|^2, \qquad U = \{u \in \mathbb{R}^m : A^\top u = 0_n,\ b^\top u = \rho,\ u \geq 0_m\}. \tag{14}$$

Henceforth, we use the Euclidean norm of vectors.

We introduce the following unconstrained minimization problem for the residual vector of system (1):

$$I_1 = \min_{x \in \mathbb{R}^n} \frac{1}{2}\|(b - Ax)_+\|^2. \tag{15}$$

Here, a_+ is the nonnegative part of a vector a; i.e., the ith component of a_+ is the same as the ith component of a if the latter is nonnegative; otherwise, the ith component of a_+ is zero. The unconstrained minimization problem (15) is dual to the following quadratic programming problem:

$$\begin{aligned} &I_2 = \max_{z \in Z} \left\{ b^\top z - \tfrac{1}{2}\|z\|^2 \right\}, \\ &Z = \{z^\top = [z_1^\top, z_2^\top] \in \mathbb{R}^m : A_1^\top z_1 + A_2^\top z_2 = 0_n,\ z_1 \geq 0_{m_1},\ z_2 \geq 0_{m_2}\}. \end{aligned} \tag{16}$$

Problems (15) and (16) are always solvable. Moreover, problem (16) has a unique solution, because its feasible set is nonempty and its objective function, which is quadratic and strictly concave, is bounded above by $\|b\|^2/2$. Theorem 1, given below, asserts the equivalence between the quadratic programming problems (14) and (16) in the sense that the solution to one problem determines the solution to the other. The solution $z^* \in \mathbb{R}^m$ to the quadratic programming problem (16) can be expressed in terms of the solution $x^* \in \mathbb{R}^n$ to the simpler problem (15) of the unconstrained minimization of a piecewise quadratic function. Usually, we have $n \ll m$ in the problem of separating polyhedra (4).

Theorem 1. *Let X_1 and X_2 be nonempty disjoint polyhedra. Every solution x^* to problem* (15) *determines a unique solution $z^{*\top} = [z_1^{*\top}, z_2^{*\top}]$ to problem* (16) *given by the formulas*

$$z_1^* = (b_1 - A_1 x^*)_+, \qquad z_2^* = (b_2 - A_2 x^*)_+. \tag{17}$$

The normal solution $\tilde{u}^*$ to system (5) *can be obtained from the solution z^* to problem* (16) *by the formula*

$$\tilde{u}^* = \rho z^* / \|z^*\|^2, \tag{18}$$

while the solution z^* to problem (16) *can be obtained from the solution $\tilde{u}^*$ to problem* (14) *by the formula*

$$z^* = \rho \tilde{u}^* / \|\tilde{u}^*\|^2. \tag{19}$$

It holds that $\|\tilde{u}^*\| \|z^*\| = \rho$. The optimal values of the objective functions in problems (15) *and* (16) *are the same: $I_1 = I_2 = \|z^*\|^2/2$.*

The assertions of Theorem 1 follow from the results of [2, 3]. The vector $z^{*\top} = [z_1^{*\top}, z_2^{*\top}]$ will be called the *vector of minimal residuals* of system (4). Consider the family of hyperplanes (9), (10) that uses the normal solution $\tilde{u}^*$ to system (5). This family is given by the two equivalent definitions

$$\Gamma(\alpha) = \{x \in \mathbb{R}^n : \ \tilde{u}_1^{*\top}(A_1 x - b_1) + \alpha\rho = 0\}, \tag{20}$$

$$\Gamma(\alpha) = \{x \in \mathbb{R}^n : \ -\tilde{u}_2^{*\top}(A_2 x - b_2) + (\alpha - 1)\rho = 0\}. \tag{21}$$

Note that if $\tilde{u}^*$ is replaced by z^* using (19), then families (20) and (21) can be written in yet another equivalent form

$$\Gamma(\alpha) = \{x \in \mathbb{R}^n : \ z_1^{*\top}(A_1 x - b_1) + \alpha\|z^*\|^2 = 0\},$$

$$\Gamma(\alpha) = \{x \in \mathbb{R}^n : \ z_2^{*\top}(b_2 - A_2 x) + (\alpha - 1)\|z^*\|^2 = 0\}.$$

Theorem 2 (on family (20), (21) of parallel separating hyperplanes). *Let X_1 and X_2 be nonempty disjoint polyhedra. Assume that x^* is a solution to problem (15), while the vectors z^* and $\tilde{u}^*$ are determined by (17) and (18). Then, the following is true:*

1) *there exists a solution to system (5); for every solution to this system, it holds that $\|u_1\| \neq 0$, $\|u_2\| \neq 0$, and $\|A_1^\top u_1\| = \|A_2^\top u_2\| \neq 0$;*
2) *when $0 \leq \alpha \leq 1$, the set $\Gamma(\alpha)$ determines a family of parallel hyperplanes that separate X_1 and X_2; if $0 < \alpha < 1$, then the hyperplanes $\Gamma(\alpha)$ strictly separate X_1 and X_2;*
3) *if α is equal to $\alpha_* = \|z_1^*\|^2/\|z^*\|^2$, then the point x^* belongs to the separating hyperplane corresponding to this value of α;*
4) *the shift vector $y = \Gamma(1) - \Gamma(0)$ and the thickness of the family $\Gamma(\alpha)$ are determined by the formulas*

$$y = \frac{-\rho A_1^\top \tilde{u}_1^*}{\|A_1^\top \tilde{u}_1^*\|^2}, \qquad \|y\| = \frac{\rho}{\|A_1^\top \tilde{u}_1^*\|};$$

5) *if $\alpha > 0$, then $X_1 \cap \Gamma(\alpha) = \emptyset$; if $\alpha < 1$, then $X_2 \cap \Gamma(\alpha) = \emptyset$;*
6) *if $X_1 \cap \Gamma(0) \neq \emptyset$, then $\Gamma(0)$ is a supporting hyperplane of the set X_1; if $X_2 \cap \Gamma(1) \neq \emptyset$, then $\Gamma(1)$ is a supporting hyperplane of the set X_2;*
7) *every solution x^* to problem (15) belongs to neither X_1 nor X_2.*

Proof.

1. System (5) is alternative to the inconsistent system (4); hence, there exists a solution to (5); moreover, $\|A_1^\top u_1\| = \|A_2^\top u_2\|$. We show that these norms cannot vanish. The relation $b_1^\top u_1 + b_2^\top u_2 = \rho > 0$ implies that at least one of the two summands on the left-hand side is strictly positive. Without loss of generality, we can assume that

$$b_1^\top u_1 = \rho_1 > 0. \tag{22}$$

By the condition of the theorem, $X_1 \neq \emptyset$. Hence, the system $A_1^\top u_1 = 0_n$, $b_1^\top u_1 = \rho_1$, $u_1 \geq 0_{m_1}$, which is alternative to the system $A_1 x \geq b_1$, is inconsistent. Thus, if (22) is fulfilled and $u_1 \geq 0_{m_1}$, then the vector $A_1^\top u_1$ cannot be zero. Therefore, $A_2^\top u_2$ is not a zero vector as well. It follows that the solutions u_1 and u_2 to system (5) are nonzero.

2. The necessary condition for a minimum in problem (15), combined with (17) and (18), leads to $A^\top z^* = 0_n$ and $A^\top \tilde{u}^* = 0_n$. Define the vector c as follows:

$$c = A_1^\top \tilde{u}_1^* = -A_2^\top \tilde{u}_2^*. \tag{23}$$

Since assertion 1 has already been proved, we have $\|c\| \neq 0$. Taking the normal solution $\tilde{u}^*$ as the vector u in formulas (7) and (8), we arrive at the relations

$$\varphi(x, \alpha) = \tilde{u}_1^{*\top}(A_1^\top x - b_1) + \alpha\rho = c^\top x - b_1^\top \tilde{u}_1^* + \alpha\rho, \tag{24}$$

$$\varphi(x, \alpha) = \tilde{u}_2^{*\top}(b_2 - A_2^\top x) + (\alpha - 1)\rho = c^\top x + b_2^\top \tilde{u}_2^* + (\alpha - 1)\rho. \tag{25}$$

If $x \in X_1$ and $\alpha \geq 0$, then $\varphi(x, \alpha) \geq 0$. If $x \in X_2$ and $\alpha \leq 1$, then $\varphi(x, \alpha) \leq 0$. Hence, $\Gamma(\alpha)$, where $0 \leq \alpha \leq 1$, is indeed a family of separating hyperplanes.

If $\alpha > 0$ and $x \in X_1$, then, by (24), we have $\varphi(x, \alpha) > 0$. Similarly, if $\alpha < 1$ and $x_2 \in X_2$, then (25) implies that $\varphi(x, \alpha) < 0$. Thus, in this case, $\Gamma(\alpha)$ defines a family of strictly separating hyperplanes.

3. We set $\bar{x}$ in (11) equal to the vector x^* and set α equal to α_*. Then, we find from (11) that

$$\bar{x}^* - x^* = c\left(b_1^\top \tilde{u}_1^* - \rho \frac{\|z_1^*\|^2}{\|z^*\|^2} - c^\top x^*\right).$$

Using this relation and taking into account (17) and (18), we arrive at the equality $\bar{x}^* - x^* = 0_n$; i.e., in this case, the vector x^* belongs to the separating hyperplane $\Gamma(\alpha_*)$. Similarly, substituting α_* into (21) and taking into account the relation $1 - \alpha_* = \|z_2^*\|/\|z^*\|^2$, we conclude that x^* belongs to the separating hyperplane $\Gamma(\alpha_*)$.

4. This assertion follows from formulas (12) and (13).

5. The conditions $x_1 \in X_1$ and $\tilde{u}_1^* \geq 0_{m_1}$ imply that $\tilde{u}_1^{*\top}(A_1 x_1 - b_1) \geq 0$. On the other hand, from the condition $x_1 \in \Gamma(\alpha)$, we have $\tilde{u}_1^{*\top}(A_1 x_1 - b) + \alpha\rho = 0$, which is impossible if $\alpha\rho > 0$. Hence, the intersection of X_1 and $\Gamma(\alpha)$ is empty for any $\alpha > 0$. The case $\alpha < 1$ is treated analogously.

6. The set X_1 has at least one common point x_1 with $\Gamma(0)$. Moreover, X_1 belongs to the half-space $c^\top x_1 - \tilde{u}_1^{*\top} b_1 \geq 0$, because this inequality can be rewritten as $\tilde{u}_1^{*\top}(A_1 x_1 - b_1) \geq 0$. It follows that $\Gamma(0)$ is a separating hyperplane of the set X_1 at its point x_1. The second assertion is proved analogously.

7. Assume the contrary; i.e., there exists a solution x^* to problem (15) such that $x^* \in X_1$. This means that $z_1^* = 0_{m_1}$. Then, by (18), the solution to system (5) is such that $\|\tilde{u}_1^*\| = 0$, which contradicts assertion 1. The theorem is proved. □

Theorem 2 suggests that the simplest method for constructing a family of separating hyperplanes is as follows. First, one solves in $\mathbb{R}^n$ the unconstrained minimization problem (15) for the residual of the inconsistent system (5) and calculates the normal solution $\tilde{u}^*$ to system (5). Then, one constructs $\Gamma(\alpha)$, using (20) or (21). The approach of Eremin is to find an arbitrary solution to the consistent system (5), where the number of unknowns is m. Since we usually have $n \ll m$, the approach suggested by Theorem 2 is preferable.

Note that the normal solution $\tilde{u}^*$ to system (5) can be found by a different method, namely, by solving the dual problem to the quadratic programming problem (14). The dual problem is the following unconstrained maximization problem for a piecewise quadratic function:

$$\max_{\beta \in \mathbb{R}^1} \max_{x \in \mathbb{R}^n} \left\{ \beta\rho - \frac{1}{2}\|(\beta b - Ax)_+\|^2 \right\}. \tag{26}$$

The number of variables in this problem is $n+1$.

If β', x' is a solution to problem (26), then the normal solution $\tilde{u}^*$ to system (5) is given by

$$\tilde{u}^* = (\beta' b - Ax')_+.$$

In the following theorem, we determine the distance between the supporting hyperplanes constructed with the help of the normal solution $\tilde{u}^*$ to system (5).

Theorem 3. *Let the conditions of Theorem 2 be fulfilled. Then, there exist $\hat{\alpha} \leq 0$ and $\tilde{\alpha} \geq 1$ such that the family of parallel hyperplanes (20), (21), where $\hat{\alpha} \leq \alpha \leq \tilde{\alpha}$, separates X_1 and X_2. The hyperplanes $\Gamma(\hat{\alpha})$ and $\Gamma(\tilde{\alpha})$ are supporting hyperplanes of the sets X_1 and X_2, respectively. The hyperplane $\Gamma(\tilde{\alpha})$ can be obtained from $\Gamma(\hat{\alpha})$ by the formula $\Gamma(\tilde{\alpha}) = \Gamma(\hat{\alpha}) + y$, where the shift vector y and its norm are given by $y = (\hat{\alpha} - \tilde{\alpha})c/\|c\|^2$ and $\|y\| = (\tilde{\alpha} - \hat{\alpha})/\|c\|$.*

Proof. The form of $\varphi(x, \alpha)$ implies that the inequalities

$$\varphi(x,\alpha) = \tilde{u}_1^{*\top}(A_1^\top x - b_1) + \alpha\rho = c^\top x - b_1^\top \tilde{u}_1^* + \alpha\rho \geq 0, \tag{27}$$

$$c^\top x \geq b_1^\top \tilde{u}_1^* - \alpha\rho \tag{28}$$

are fulfilled for all $x \in X_1$ and $\alpha \geq 0$. Similarly, the inequalities

$$\varphi(x,\alpha) = \tilde{u}_2^{*\top}(b_2 - A_2^\top x) + (\alpha - 1)\rho = c^\top x + b_2^\top \tilde{u}_2^* + (\alpha - 1)\rho \leq 0, \tag{29}$$

$$c^\top x \leq -b_2^\top \tilde{u}_2^* - (\alpha - 1)\rho \tag{30}$$

hold for all $x \in X_2$ and $\alpha \leq 1$. According to (28) and (30), there exist $\hat{x} \in X_1$ and $\tilde{x} \in X_2$ such that

$$c^\top \hat{x} = \min_{x \in X_1} c^\top x, \qquad c^\top \tilde{x} = \max_{x \in X_2} c^\top x. \tag{31}$$

Setting $x = \hat{x}$ in (27) yields

$$c^\top \hat{x} - b_1^\top \tilde{u}_1^* + \alpha\rho \geq 0. \tag{32}$$

If $\alpha = 0$, then we have

$$b_1^\top \tilde{u}_1^* - c^\top \hat{x} \leq 0. \tag{33}$$

Therefore, (32) is valid for any $\alpha \geq \hat{\alpha}$, where

$$\hat{\alpha} = (b_1^\top \tilde{u}_1^* - c^\top \hat{x})/\rho \leq 0. \tag{34}$$

Relation (29) implies that

$$\varphi(x,\alpha) = c^\top \tilde{x} + b_2^\top \tilde{u}_2^* + \rho(\alpha - 1) \leq 0$$

for $x \in X_2$ and $\alpha \leq 1$. If $\alpha = 1$, then we have

$$c^\top \tilde{x} + b_2^\top \tilde{u}_2^* \leq 0. \tag{35}$$

Hence, the inequality $\varphi(x, \alpha) \leq 0$ holds for all $x \in X_2$ and α such that $\alpha \leq \tilde{\alpha}$, where

$$\tilde{\alpha} = 1 - (c^\top \tilde{x} + b_2^\top \tilde{u}_2^*)/\rho \geq 1. \tag{36}$$

The hyperplane $\Gamma(\hat{\alpha}) = \{x \in \mathbb{R}^n : c^\top x = c^\top \hat{x}\}$ has the common point $\hat{x}$ with the set X_1. In view of (31), every point of X_1 belongs to the half-space $c^\top(\hat{x} - x) \leq 0$. It follows that $\Gamma(\hat{\alpha})$ is a supporting hyperplane of X_1. The vector c is a supporting vector of X_1 at the point $\hat{x}$. In particular, if $\hat{\alpha} = 0$, then $\Gamma(0)$ is a supporting hyperplane. In a similar way, we show that $\Gamma(\tilde{\alpha})$ is a supporting hyperplane of X_2 at the point $\tilde{x}$. The shift vector y is obtained by simple calculations similar to those in (12) and (13). The theorem is proved. □

In certain cases, the knowledge of the normal solution $\tilde{u}^*$ makes it possible to easily determine the optimal values of the objective functions in problems (31) and find out whether the hyperplanes $\Gamma(\alpha)$ corresponding to $\alpha = 0$ and $\alpha = 1$ are supporting hyperplanes for X_1 and X_2, respectively. Denote by $w_1 \in \mathbb{R}_+^{m_1}$ and $w_2 \in \mathbb{R}_+^{m_2}$ the Lagrange multipliers, and define the Lagrangian functions for problems (31):

$$L_1(x, w_1) = c^\top x + w_1^\top (b_1 - A_1 x), \quad L_2(x, w_2) = -c^\top x + w_2^\top (b_2 - A_2 x).$$

The pair $[x_1, w_1]$ is a Kuhn–Tucker point for the first problem in (31) if it holds that

$$c = A_1^\top w_1, \quad D(w_1)(b_1 - A_1 x_1) = 0_{m_1}, \quad w_1 \geq 0_{m_1}, \quad A_1 x_1 \geq b_1. \tag{37}$$

Analogously, the pair $[x_2, w_2]$ is a Kuhn–Tucker point for the second problem in (31) if

$$c = -A_2^\top w_2, \quad D(w_2)(b_2 - A_2 x_2) = 0_{m_2}, \quad w_2 \geq 0_{m_2}, \quad A_2 x_2 \geq b_2. \tag{38}$$

We take $\tilde{u}_1^*$ as the vector w_1 in (37). If there exists a vector x_1 that satisfies (37), then $[x_1, \tilde{u}_1^*]$ is a Kuhn–Tucker point. Moreover, (34) implies that $\hat{\alpha} = 0$; i.e., $\Gamma(0)$ is a supporting hyperplane of X_1 at the point x_1. Similarly, let us set $w_2 = \tilde{u}_2^*$ in the second problem in (31). If there exists a vector x_2 that satisfies (38), then, in view of (36), we conclude that $\tilde{\alpha} = 1$. Thus, $\Gamma(1)$ is a supporting hyperplane of X_2 at the point x_2. If $\hat{\alpha} < 0$ or $\tilde{\alpha} > 1$, then we seek the optimal Lagrange multipliers w_1^* or w_2^* for the corresponding problems in (31). The first conditions in (37) and (38) imply that these vectors satisfy the relation

$$A_1^\top w_1^* + A_2^\top w_2^* = 0.$$

Setting $\hat{x} = x_1$ and $\tilde{x} = x_2$ in (33) and (35), respectively, we obtain

$$c^\top x_1 \geq b_1 \tilde{u}_1^*, \quad -c^\top x_2 \geq b_2 \tilde{u}_2^*.$$

Adding these inequalities, we find that

$$b_1^\top w_1^* + b_2^\top w_2^* = c^\top (x_1 - x_2) \geq b_1^\top \tilde{u}_1^* + b_2^\top \tilde{u}_2^* = \rho \geq 0.$$

It follows that the vector $w^{*\top} = [w_1^{*\top}, w_2^{*\top}]$, together with $\tilde{u}^*$, satisfies system (5). The family of separating hyperplanes can be represented as

$$\Gamma(\alpha) = \{x \in \mathbb{R}^n : w_1^{*\top}(A_1 x - b_1) + \alpha\rho = 0\}, \tag{39}$$

$$\Gamma(\alpha) = \{x \in \mathbb{R}^n : -w_2^{*\top}(A_2 x - b_2) + (\alpha - 1)\rho = 0\}. \tag{40}$$

Thus, we have constructed two families of parallel separating hyperplanes of form (20), (21) and (39), (40), respectively. For the first family, we use the normal solution to system (5); for the second, the optimal Lagrange multipliers for the linear programming problems (31). Both vectors $\tilde{u}^*$ and w^* satisfy (5). The following theorem asserts that the normal vector c and the scalar γ which determine an arbitrary strictly separating hyperplane can be expressed in terms of a solution to system (5).

Theorem 4. *Let the hyperplane $c^\top x = \gamma$ strictly separate two nonempty disjoint polyhedra X_1 and X_2. Then, there exists a solution u_1, u_2 to the system*

$$A_1^\top u_1 + A_2^\top u_2 = 0, \quad b_1^\top u_1 + b_2^\top u_2 = \rho > 0, \qquad u_1 \geq 0, \quad u_2 \geq 0, \tag{41}$$

such that the vector c and the scalar γ are given by

$$c = A_1^\top u_1 = -A_2^\top u_2, \qquad \gamma = b_1^\top u_1 - \rho_1 = -b_2^\top u_2 + \rho_2,$$

where ρ_1 and ρ_2 are arbitrary positive constants such that $\rho_1 + \rho_2 = \rho$.

Proof. For definiteness, we assume that the given strictly separating hyperplane is such that all $x \in X_1$ satisfy the inequality $c^\top x > \gamma$, while all $x \in X_2$ satisfy the inequality $c^\top x < \gamma$. Then, the system

$$A_1 x \geq b_1, \qquad c^\top x \leq \gamma$$

is unsolvable, whereas the alternative system is consistent. Hence, there exist a vector $q \geq 0_{m_1}$ and a scalar $\eta \geq 0$, $\eta \in \mathbb{R}^1$ such that

$$A_1^\top q - c\eta = 0_n, \qquad b_1^\top q - \gamma\eta = \rho_1 > 0. \tag{42}$$

Here, ρ_1 is an arbitrary positive constant. The scalar η cannot vanish, since, otherwise, the consistent system (42) has the form

$$A_1^\top q = 0_n, \qquad b_1^\top q = \rho_1 > 0, \qquad q \geq 0_{m_1}.$$

Therefore, the alternative system $A_1 x \geq b_1$ is inconsistent, which contradicts the condition $X_1 \neq \emptyset$. Thus, (42) yields $c = A_1^\top q/\eta$ and $\gamma = b_1^\top q/\eta - \rho_1/\eta$. Using the notation $u_1 = q/\eta$ and $\rho_1/\eta = \rho_1$, we obtain

$$A_1^\top u_1 = c, \qquad b_1^\top u_1 - \gamma = \rho_1 > 0, \qquad u_1 \geq 0_{m_1}. \tag{43}$$

In a similar manner, we arrive at the relations

$$A_2^\top u_2 = -c, \qquad b_2^\top u_2 + \gamma = \rho_2 > 0. \tag{44}$$

Adding (43) and (44), we obtain the consistent system (41), which is alternative to (4). The theorem is proved. □

3 Separation of polyhedra defined by equations with nonnegative variables

Consider the case where two polyhedra are represented by equality systems given on the nonnegative orthant; i.e., we have two nonempty sets

$$X_1 = \{x \in \mathbb{R}^n : A_1 x = b_1, \ x \geq 0_n\}, \qquad X_2 = \{x \in \mathbb{R}^n : A_2 x = b_2, \ x \geq 0_n\},$$

such that $X = X_1 \cap X_2 = \emptyset$. According to Farkas' lemma, the inconsistency of the system

$$A_1 x = b_1, \qquad A_2 x = b_2, \qquad x \geq 0_n,$$

where the variables are nonnegative, implies that the system,

$$A_1^\top u_1 + A_2^\top u_2 \leq 0_n, \qquad b_1^\top u_1 + b_2^\top u_2 = \rho \tag{45}$$

is consistent. Here, ρ is a positive constant.

Theorem 5. *Let two nonempty disjont polyhedra X_1 and X_2 are defined by the equality systems given on nonnegative orthant. Then any solution $[u_1^\top, u_2^\top]$ to system (45) determines two families of parallel hyperplanes that separate X_1 and X_2*

$$\varphi_1(x, \alpha) = u_1^\top (A_1 x - b_1) + \alpha\rho = 0, \quad \varphi_2(x, \alpha) = -u_2^\top (A_2 x - b_2) - (1 - \alpha)\rho = 0,$$

where $\alpha \in [0 \ 1]$.

Proof. System (45) is solvable, and every its solution satisfies the inequalities $\|u_1\| \neq 0$, $\|u_2\| \neq 0$, $\|A_1^\top u_1\| \neq 0$, and $\|A_2^\top u_2\| \neq 0$. Indeed, assume the contrary; namely, let $A_1^\top u_1 = 0_n$. Since X_1 is nonempty, the alternative system $A_1^\top u_1 \leq 0$, $b_1^\top u_1 = \rho_1 \neq 0$, is inconsistent. By assumption, $A_1^\top u_1 = 0_n$; hence, $b_1^\top u_1 = 0$. In this case, (45) converts into the consistent system $A_2^\top u_2 \leq 0_n$, $b_2^\top u_2 = \rho$. However, this is the alternative system to $A_2 x = b_2$, $x \geq 0_n$.

By assumption, the latter system is consistent, because X_2 is nonempty. The contradiction obtained proves that the equality $A_1^\top u_1 = 0_n$ is impossible. If $u_1 = 0_{m_1}$, then $A_1^\top u_1 = 0_n$, which is impossible. Taking the scalar product of the inequality part in (45) with a nonnegative vector x and then subtracting the equality part, we obtain

$$u_1^\top (A_1 x - b_1) + u_2^\top (A_2 x - b_2) \le -\rho < 0. \tag{46}$$

Define two linear functions of variable $x \in \mathbb{R}^n$ and a parameter $\alpha \in [0,1]$ as follows:

$$\varphi_1(x,\alpha) = u_1^\top (A_1 x - b_1) + \alpha\rho, \qquad \varphi_2(x,\alpha) = -u_2^\top (A_2 x - b_2) - (1-\alpha)\rho.$$

Then, inequality (46) can be rewritten as

$$\varphi_1(x,\alpha) = u_1^\top (A_1 x - b_1) + \alpha\rho \le -u_2^\top (A_2 x - b_2) - (1-\alpha)\rho = \varphi_2(x,\alpha). \tag{47}$$

Fix vectors u_1 and u_2 constituting an arbitrary solution to system (45). Using $\varphi_1(x,\alpha)$ and $\varphi_2(x,\alpha)$, we obtain two families of hyperplanes that correspond to $\alpha \in [0,1]$ and separate X_1 and X_2. If $x \in X_1$, then $\varphi_1(x,\alpha) \ge 0$ for $\alpha \in [0,1]$. If $x \in X_2$, then $\varphi_2(x,\alpha) \le 0$ for $\alpha \in [0,1]$. Then, (47) implies that $\varphi_1(x,\alpha) \le 0$. It follows that the hyperplanes in the family $\varphi_1(x,\alpha) = 0$, $\alpha \in [0,1]$ separate X_1 and X_1. If $0 < \alpha < 1$, then inequality (47) shows that the hyperplane $\varphi_1(x,\alpha) = 0$ strictly separates these sets. Now, we show that the condition $\varphi_2(x,\alpha) = 0$ determines the family of hyperplanes that separate X_1 and X_1 for $\alpha \in [0,1]$ and strictly separate these sets if $0 < \alpha < 1$.

Indeed, if $x \in X_2$, then $\varphi_2(x,\alpha) \le 0$ for $\alpha \in [0,1]$ and $\varphi_2(x,\alpha) < 0$ if $0 \le \alpha < 1$. If $x \in X_1$, then (47) implies that $\varphi_2(x,\alpha) \ge 0$ for $\alpha \in [0,1]$ and $\varphi_2(x,\alpha) > 0$ if $0 < \alpha \le 1$. □

Thus, by solving the alternative consistent system (45), we obtain two families of separating hyperplanes determined by $\varphi_1(x,\alpha)$ and $\varphi_2(x,\alpha)$). It follows that the case under analysis differs from the case of polyhedra given by inequality systems, which was examined above.

Define a nonnegative linear combination of $\varphi_1(x,\alpha)$ and $\varphi_2(x,\alpha)$ by setting

$$\varphi_3(x,\alpha) = \lambda_1\varphi_1(x,\alpha) + \lambda_2\varphi_2(x,\alpha).$$

Here, $\lambda_1 \ge 0$ and $\lambda_2 \ge 0$.

Consider the following three families of separating hyperplanes and their unions:

$$\Gamma_i(\alpha) = \{x \in \mathbb{R}^n : \varphi_i(x,\alpha) = 0\}, \qquad \Gamma_i = \bigcup_{\alpha=0}^{1} \Gamma_i(\alpha), \qquad i = 1,2,3.$$

It is easy to show that the hyperplanes in $\Gamma_3(\alpha)$ separate X_1 and X_2 for any nonnegative scalars λ_1 and λ_2, of which at least one is nonzero. As in the preceding section, we denote by p^* the vector joining two nearest points in

X_1 and X_1; then, the distance between these sets is $\|p^*\|$. It is often possible to choose λ_1 and λ_2 such that the vector

$$p^* = \lambda_1 A_1^\top u_1 - \lambda_2 A_2^\top u_2,$$

where $u^\top = [u_1^\top, u_2^\top]$, satisfies condition (45). In this case, p^* is the normal vector of $\Gamma_3(x, \alpha)$, and the thickness of this family is equal to the minimal distance between X_1 and X_2. The following theorem is an analogue of Theorem 4.

Theorem 6. *Let the hyperplane $c^\top x - \gamma = 0$ strictly separate two nonempty disjoint polyhedra X_1 and X_1. Then, there exists a solution u_1, u_2 to the system*

$$A_1^\top u_1 + A_2^\top u_2 \le 0, \qquad b_1^\top u_1 + b_2^\top u_2 = \rho > 0,$$

such that

$$A_1^\top u_1 \le c, \qquad A_2^\top u_2 \le -c, \qquad \gamma = b_1^\top u_1 - \rho_1 = -b_2^\top u_2 + \rho_2,$$

where ρ_1 and ρ_2 are arbitrary positive constants such that $\rho_1 + \rho_2 = \rho_1$.

The proof is an almost word-for-word repetition of the proof of Theorem 4.

Theorem 6 asserts that polyhedra given by equality systems on the nonnegative orthant are different from polyhedra given by inequality systems in the sense that it is not always possible to find u_1 and u_2 that satisfy the consistent alternative system (45) and, at the same time, satisfy either the condition $c = A_1^\top u_1$, or the condition $c = -A_2^\top u_2$. In other words, there may not exist vectors u_1 and u_2 that satisfy (45) and have the property that the separating hyperplane $c^\top x - \gamma = 0$ belongs to either $\Gamma_1(\alpha)$ or $\Gamma_2(\alpha)$.

4 The thickest separating family of parallel hyperplanes

In this section we consider the polyhedra defined only by systems of linear inequalites. The problem of finding the minimal distance between two disjoint sets X_1 and X_2 can be written in the form

$$\min_{x_1 \in X_1} \min_{x_2 \in X_2} \frac{1}{2} \|x_1 - x_2\|^2. \tag{48}$$

We change the variable to $p = x_1 - x_2$ and rewrite problem (48) as

$$\min_{p \in \mathbb{R}^n} \min_{x_2 \in X_2} \frac{1}{2} \|p\|^2 \tag{49}$$

subject to

$$A_1 x_2 + A_1 p \ge b_1, \qquad A_2 x_2 \ge b_2. \tag{50}$$

The norm $\|p\|$ is the same as the distance between the convex sets X_1 and X_2. The vector p obtained by solving problem (49), (50) will be called the vector determining the distance between these sets. The vector y introduced above is not always the same as the vector p produced by solving problem (49).

The Lagrangian function for problem ((49) has the form

$$L(p, x_2, v) = \|p\|^2/2 + v_1^\top (b_1 - A_1 p - A_1 x_2) + v_2^\top (b_2 - A_2 x_2).$$

Using this function, we can write the dual problem:

$$\max_{v_1 \in \mathbb{R}_+^{m_1}} \max_{v_2 \in \mathbb{R}_+^{m_2}} \min_{x_2 \in \mathbb{R}^n} \min_{p \in \mathbb{R}^n} L(p, x_2, v). \tag{51}$$

The optimality conditions for the inner minimization problem in (51) are as follows:

$$L_p(p, x_2, v) = p - A_1^\top v_1 = 0_n, \tag{52}$$

$$L_{x_2}(p, x_2, v) = -A_1^\top v_1 - A_2^\top v_2 = 0_n. \tag{53}$$

Relations (52) and (53) imply that $p = A_1^\top v_1 = -A_2^\top v_2$. The substitution of this expression into the Lagrangian function results in the dual Lagrangian function

$$\begin{aligned}\tilde{L}(x_2, v) &= \|A_1^\top v_1\|^2/2 + v_1^\top (b_1 - A_1 A_1^\top v_1 - A_1 x_2) + v_2^\top (b_2 - A_2 x_2) = \\ &= b_1^\top v_1 + b_2^\top v_2 - \|A_1^\top v_1\|^2/2 - x_2^\top (A_1^\top v_1 + A_2^\top v_2).\end{aligned}$$

Taking into account (53), we obtain the dual problem to problem (49), (50):

$$\max_{v_1 \in \mathbb{R}_+^{m_1}} \max_{v_2 \in \mathbb{R}_+^{m_2}} \{b_1^\top v_1 + b_2^\top v_2 - \|A_1^\top v_1\|^2/2\} \tag{54}$$

subject to

$$A_1^\top v_1 + A_2^\top v_2 = 0_n, \qquad v_1 \geq 0_{m_1}, \quad v_2 \geq 0_{m_2}. \tag{55}$$

Denote by $[p^*, x_2^*]$ a solution to problem (49), (50) and by $[v_1^*, v_2^*]$ a solution to the dual problem (54), (55). By the duality theorem, we have

$$b_1^\top v_1^* + b_2^\top v_2^* - \|A_1^\top v_1^*\|^2/2 = \|p^*\|^2/2. \tag{56}$$

Substituting $p^* = A_1^\top v_1^*$ into (56), we obtain

$$b_1^\top v_1^* + b_2^\top v_2^* = \|A_1^\top v_1^*\|^2. \tag{57}$$

Thus, we have the following assertion.

Theorem 7. *Every solution* $v^{*\top} = [v_1^*, v_2^*]$ *to the dual problem* (54), (55) *determines the unique first component* p^* *in a solution* $[p^*, x_2^*]$ *to problem* (51), *which is given by the formulas*

$$p^* = A_1^\top v_1^* = -A_2^\top v_2^*. \tag{58}$$

Moreover, it holds that

$$b^\top v^* = \|A_1^\top v_1^*\|^2 = \|A_2^\top v_2^*\|^2 = \|p^*\|^2.$$

Theorem 7 implies that the vector v^* found from (54), (55) satisfies system (5) for $\rho = \|p^*\|^2$.

Note that a solution v^* to the dual problem (54), (55) determines only the first component p^* in a solution $[p^*, x_2^*]$ to the primal problem (49), (50). To determine x_2^*, one should substitute p^* into the constraints of the primal problem and solve the resulting system of inequalities

$$A_1 x_2 \geq b_1 - A_1 p^*, \qquad A_2 x_2 \geq b_2$$

with respect to the vector x_2. Thus, the situation here is different from that of a pair of mutually dual problems, which was examined above.

Theorem 8 (on a family of parallel separating hyperplanes). *Let X_1 and X_2 be nonempty disjoint polyhedra, and let v^*, p^*, x_2^* be a solution to problem (51). Then, the family of parallel hyperplanes that separate X_1 and X_2 can be represented in the form*

$$\Gamma(\alpha) = \{x \in \mathbb{R}^n : {p^*}^\top x - b_1^\top v_1^* + \alpha \|p^*\|^2 = 0\}, \tag{59}$$

$$\Gamma(\alpha) = \{x \in \mathbb{R}^n : {p^*}^\top x + b_2^\top v_2^* - (1-\alpha)\|p^*\|^2 = 0\}, \tag{60}$$

where $\alpha \in [0,1]$. Moreover, if $0 < \alpha < 1$, then these hyperplanes strictly separate X_1 and X_2. The hyperplanes $\Gamma(0)$ and $\Gamma(1)$ are supporting hyperplanes for the sets X_1 and X_2, respectively. The thickness of the family $\Gamma(\alpha)$ is equal to $\|p^*\|$; it is the same as the distance between the polyhedra X_1 and X_2.

Proof. Since v^* is a solution to problem (54), (55), we have

$$A_1^\top v_1^* + A_2^\top v_2^* = 0_n.$$

Multiplying this relation on the left by $x^\top$, where x is an arbitrary vector in $\mathbb{R}^n$, and subtracting the resulting equality from (57), we obtain

$$x^\top A_1^\top v_1^* + x^\top A_2^\top v_2^* - b_1^\top v_1^* - b_2^\top v_2^* = -\|A_1^\top v_1^*\|^2. \tag{61}$$

Using the coefficient $\alpha \in [0,1]$ and the equality $\|A_1^\top v_1^*\| = \|A_2^\top v_2^*\|$, we can rewrite (61) as

$${v_1^*}^\top (A_1 x - b_1) + \alpha \|A_1^\top v_1^*\|^2 = {v_2^*}^\top (b_2 - A_2 x) + (\alpha - 1)\|A_2^\top v_2^*\|^2.$$

Due to (58), we arrive at the equivalent representations (59) and (60) of the hyperplane $\Gamma(\alpha)$.

The triple $[v^*, p^*, x_2^*]$ is a Kuhn–Tucker point for problem (49). Therefore, the following complementary slackness conditions must hold:

$${v_1^*}^\top L_{v_1}(p^*, x_2^*, v^*) = 0, \quad {v_2^*}^\top L_{v_2}(p^*, x_2^*, v^*) = 0, \quad v_1^* \geq 0_{m_1}, \quad v_2^* \geq 0_{m_2},$$
$$L_{v_1}(p^*, x_2^*, v^*) \leq 0_n, \quad L_{v_2}(p^*, x_2^*, v^*) \leq 0_{m_2}.$$

From these conditions, we derive

$$v_1^{*\top}(b_1 - A_1x_1^*) = 0, \quad v_2^{*\top}(b_2 - A_2x_2^*) = 0, \quad A_1x_1^* \geq b_1, \quad A_2x_2^* \geq b_2. \quad (62)$$

The relations obtained imply that $x_1^* \in X_1$, $x_1^* \in \Gamma(0)$, $x_2^* \in X_2$, and $x_2^* \in \in \Gamma(1)$.

For an arbitrary point x in X_1, we have $A_1x \geq b_1$. Taking the scalar product of this inequality with the nonnegative vector v_1^*, we obtain $v_1^{*\top}(A_1x - -b_1) \geq 0$. Taking into account (58), we arrive at the relation

$$p^{*\top}x - b_1^\top v_1^* \geq 0. \quad (63)$$

From (62), we find that

$$b_1^\top v_1^* = p^{*\top}x_1^*. \quad (64)$$

From (63), we conclude that $p^{*\top}x \geq p^{*\top}x_1^*$ for any $x \in X_1$ and at least one point $x_1^* \in X_1$ belongs to the separating hyperplane $p^{*\top}x = p^{*\top}x_1^*$. Hence, X_1 belongs to one of the half-spaces determined by the hyperplane $\Gamma(0)$, and $\Gamma(0)$ is a supporting hyperplane for this set at its point x_1^*. In a similar way, we show that $\Gamma(1)$ is a supporting hyperplane for X_2 at the point x_2^*. It holds that

$$b_2^\top v_2^* = -p^{*\top}x_2^*. \quad (65)$$

All the points in X_2 satisfy the inequality $p^{*\top}x \leq p^{*\top}x_2^*$ and at least one point $x_2^* \in X_2$ belongs to the hyperplane $p^{*\top}x = p^{*\top}x_2^*$.

The distance between the supporting hyperplanes is the same as the distance between X_1 and X_2; both are equal to $\|p^*\|$, which follows from the formulation of problem (49), (50). The theorem is proved. □

Note that, if relations (64) and (65) are taken into account, then hyperplanes $\Gamma(\alpha)$ of form (59), (60) can be represented as

$$\Gamma(\alpha) = \{x \in \mathbb{R}^n : p^{*\top}x - (1-\alpha)p^{*\top}x_1^* - \alpha p^{*\top}x_2^* = 0\}, \quad (66)$$

i.e., each member of the family of separating hyperplanes can be represented as a convex combination of supporting hyperplanes for X_1 and X_2. To construct a family of hyperplanes of form (66), one needs to solve problem (49), (50) in a space of variables of dimension $2n$. To represent the same family in form (59), (60), it is required to solve the dual problem (54), (55) in a space of variables of dimension m. The vector p^* appearing in this representation is expressed in terms of v^* by formula (58).

Now, we examine the following issue: is it possible to distinguish a solution to system (5) that determines a family of hyperplanes whose thickness is equal to the distance between the sets X_1 and X_2? According to formulas (12) and (13), the shift vector y and the thickness $\|y\|$ of the family of separating hyperplanes $\Gamma(\alpha)$ are given by $y = -\rho A_1^\top u_1 / \|A_1^\top u_1\|^2$ and $\|y\| = \rho / \|A_1^\top u_1\|$, respectively; here, $u^\top = [u_1^\top, u_2^\top]$ is a solution to system (5). It is then natural

to pose the problem of finding a solution $u^{*\top} = [u_1^{*\top}, u_2^{*\top}] \in U$ to system (5) for which the thickness of the family of separating hyperplanes is maximal:

$$\frac{1}{2}\|A_1^\top u_1^*\|^2 = \min_{u \in U} \frac{1}{2}\|A_1^\top u_1\|^2, \tag{67}$$

$$U = \{u \in \mathbb{R}^m : A_1^\top u_1 + A_2^\top u_2 = 0_n,\ b_1^\top u_1 + b_2^\top u_2 = \rho,\ u_1 \geq 0_{m_1},\ u_2 \geq 0_{m_2}\}. \tag{68}$$

In this case, the shift vector $y = \Gamma(1) - \Gamma(0)$ yields the thickness of the family of hyperplanes, which is identical to the minimal distance between the polyhedra X_1 and X_2. This is explained by the fact that the solution u^* to problem (67), (68) allows us to find the minimal distance between X_1 and X_2. It turns out that problems (54), (55) and (67), (68) are equivalent in the sense that the solution to one of them can be found from the solution to the other.

Theorem 9. *Let X_1 and X_2 be nonempty disjoint polyhedra. Then, the solution v^* to problem* (54), (55) *and the solution u^* to problem* (67), (68) *satisfy the relations*

$$v^* = \frac{\rho u^*}{\|A_1^\top u_1^*\|^2}, \qquad u^* = \frac{\rho v^*}{b^\top v^*}. \tag{69}$$

The family of separating hyperplanes can be represented in each of the following forms:

$$\Gamma(\alpha) = \{x \in \mathbb{R}^n : u_1^{*\top} A_1 x - b_1^\top u_1^* + \alpha\rho = 0\}, \tag{70}$$

$$\Gamma(\alpha) = \{x \in \mathbb{R}^n : -u_2^{*\top} A_2 x + b_2^\top u_2^* + (\alpha - 1)\rho = 0\}, \tag{71}$$

where $0 \leq \alpha \leq 1$. The thickness of this family is identical to the minimal distance between X_1 and X_2.

Proof. The vector u^* satisfies system (5); hence, $\|A_1^\top u_1^*\| \neq 0$ (see the first assertion in Theorem 2). By Theorem 7, it holds that $b^\top v^* \neq 0$. Formulas (69) are obtained by comparing the Kuhn–Tucker conditions for problems (54), (55) and (67), (68). Using (58), (69) and (12), we have

$$p^* = A_1^\top v_1^* = \frac{\rho A_1^\top u_1^*}{\|A_1^\top u_1^*\|^2} = -y.$$

It follows that, for the family of separating hyperplanes (70), (71) the thickness is identical to the minimal distance between X_1 and X_2:

$$\|p^*\| = \frac{\rho}{\|A_1^\top u_1^*\|} = \|y\|.$$

The theorem is proved. □

The dual problem to problem (67), (68) is as follows:

$$\max_{q\in\mathbb{R}^n}\max_{x\in\mathbb{R}^n}\max_{\xi\in\mathbb{R}^1}\left\{\rho\xi-\frac{1}{2}\|q\|^2\right\}$$

subject to

$$A_1(q+x)-b_1\xi\geq 0_{m_1},\qquad A_2x-b_2\xi\geq 0_{m_2}.$$

The unknown vector q can be obtained from the solution u^* to the primal problem (67), (68) by using the formula $q^*=A_1^\top u_1^*=-A_2^\top u_2^*$. By the duality theorem, the solutions u^* and $[q^*,x^*,\xi^*]$ to the primal and dual problems satisfy the relation $\rho\xi^*=\|A_1^\top u_1^*\|^2$. Therefore, $\xi^*>0$ and $p^*=q^*/\xi^*$, and $x_2^*=x^*/\xi^*$.

Thus, we have constructed the three equivalent representations (59)–(60), (66), and (70)–(71), for the same family of separating hyperplanes whose thickness is equal to the minimal distance between the polyhedra. Each representation requires solving its own optimization problem. Geometrically we fined the thickest slab that separates two polyhedra.

5 The generalized Newton method

Since we usually have $n\ll m$ in the problem of separating polyhedra given by inequality systems (4), it is preferable to solve problem (15): minimize the function $F(x)=\|(b-Ax)_+\|^2/2$, which depends on n variables. The unconstrained minimization of $F(x)$ can be performed by any method, say, by the conjugate gradient method. However, Mangasarian showed that the generalized Newton method is especially efficient for the unconstrained optimization of a piecewise quadratic function (see [4,5]). We give a brief description of this method. The objective function $F(x)$ of problem (15) is convex, piecewise quadratic, and differentiable. Such a function does not have the conventional Hessian matrix. Indeed, the gradient

$$F_x(x)=-A^\top(b-Ax)_+$$

of $F(x)$ is not differentiable. However, for function $F(x)$, one can define the generalized Hessian matrix, which is an $n\times n$ symmetric positive semidefinite matrix of the form

$$\partial^2F(x)=A^\top D^\sharp(z)A.$$

Here, $D^\sharp(z)$ denotes the $n\times n$ diagonal matrix whose ith diagonal entry z^i is equal to one if $(b-Ax)^i>0$; z^i is equal to zero if $(b-Ax)^i\leq 0$, $i=1,\ldots,m$. Since the generalized Hessian matrix can be singular, the following modified Newton direction is used:

$$-(\partial^2F(x)+\delta I_n)^{-1}F_x(x),$$

where δ is a small positive number (in our calculations, we typically set $\delta = = 10^{-4}$ and I_n is the identity matrix of order n. In this case, the modified Newton method has the form

$$x_{s+1} = x_s - (\partial^2 F(x_s) + \delta I_n)^{-1} F_x(x_s). \tag{72}$$

We used the following stopping criterion for this method:

$$\|x_{s+1} - x_s\| \leq tol.$$

Mangasarian has studied the convergence of the generalized Newton method as applied to the unconstrained minimization of a convex piecewise quadratic function of this type with the step size chosen by the Armijo rule. The proof of the finite global convergence of the generalized Newton method can be found in [5–7]. The generalized Newton method as applied to the unconstrained minimization problem (15) was implemented in Matlab and showed a good performance in solving large-scale test problems. For instance, problem (15) with $n = 500$ and $m = 10^4$ whose matrix A was fully filled with nonzero entries was solved in less than one minute on a 2.24 GHz Pentium-IV computer.

Acknowledgments

This study was supported by the Russian Foundation for Basic Research (project no. 03-01-00465) and by the Program of the State Support of Leading Scientific Schools (project no. NSh-1737.2003.1).

References

1. Eremin I.I. (1998), *Theory of Linear Optimization*, Ural Branch of RAS, Ekaterinburg.
2. Evtushenko Yu.G., and Golikov A.I. (2003), New perspective on the theorem of alternative, *High Performance Algorithms and Software for Nonlinear Optimization*, G. Di Pillo and A. Murli, Editors, Kluwer Academic Publishers B.V., pp. 227–241.
3. Golikov A.I., and Evtushenko Yu.G. (2003), Theorems of the Alternative and Their Application in Numerical Methods, *Computational Mathematics and Mathematical Physics*, vol.40, pp. 338–358.
4. Vasil'ev F.P., and Ivanitskii A.Yu. (2003), *Linear Programming*, Faktorial, Moscow.
5. Mangasarian O.L. (2002), A Finite Newton Method for Classification, *Optimization Methods and Software*, vol. 17, pp. 913–930.
6. Mangasarian O.L. (2002), A Newton Method for Linear Programming, *Journal of Optimization Theory and Applications*, vol. 121, pp. 1–18.
7. Kanzow C., Qi H., and Qi L., (2003), On the Minimum Norm Solution of Linear Programs, *Journal of Optimization Theory and Applications*, vol. 116, pp. 333–345.

Exact penalty functions for generalized Nash problems

Francisco Facchinei[1] and Jong-Shi Pang[2]

[1] Università di Roma "La Sapienza", Dipartimento di Informatica e Sistemistica, Via Buonarroti 12, 00185 Roma, Italy (facchinei@dis.uniroma1.it)
[2] Rensselaer Polytechnic Institute, Department of Mathematical Sciences and Department of Decision Sciences and Engineering Systems, Troy, New York, 12180-3590 U.S.A. (pangj@rpi.edu)

Summary. We propose the use exact penalty functions for the solution of generalized Nash equilibrium problems (GNEPs). We show that by this approach it is possible to reduce the solution of a GNEP to that of a usual Nash problem. This paves the way to the development of numerical methods for the solution of GNEPs. We also introduce the notion of generalized stationary point of a GNEP and argue that convergence to generalized stationary points is an appropriate aim for solution algorithms.

Key words: (Generalized) Nash equilibrium problem, Penalty function.

1 Introduction

In this paper we consider the following Generalized Nash Equilibrium Problem (GNEP) with two players:

$$\begin{array}{lll} \text{minimize}_x\ \theta_{\mathrm{I}}(x,y) & & \text{minimize}_y\ \theta_{\mathrm{II}}(x,y) \\ \text{subject to } h^{\mathrm{I}}(x) \leq 0 & \text{and} & \text{subject to } h^{\mathrm{II}}(y) \leq 0 \\ \quad g^{\mathrm{I}}(x,y) \leq 0 & & \quad g^{\mathrm{II}}(x,y) \leq 0 \end{array} \quad (1)$$

where

- $x \in \mathbb{R}^{n_1}$, $y \in \mathbb{R}^{n_2}$;
- $\theta_{\mathrm{I}}(x,y)$ ($\theta_{\mathrm{II}}(x,y)$) is a continuously differentiable function from $\mathbb{R}^{n_1+n_2}$ to $\mathbb{R}$, such that, for every fixed y (x), $\theta_{\mathrm{I}}(\cdot,y)$ ($\theta_{\mathrm{II}}(x,\cdot)$) is convex;
- $h^{\mathrm{I}}(x)$ ($h^{\mathrm{II}}(y)$) is a continuously differentiable convex function from $\mathbb{R}^{n_1}$ ($\mathbb{R}^{n_2}$) to $\mathbb{R}^{m_1}$ ($\mathbb{R}^{m_2}$);
- $g^{\mathrm{I}}(x,y)$ ($g^{\mathrm{II}}(x,y)$) is a continuously differentiable function from $\mathbb{R}^{n_1+n_2}$ to $\mathbb{R}^{m_1}$ ($\mathbb{R}^{m_2}$) such that, for every fixed y (x), $g_{\mathrm{I}}(\cdot,y)$ ($g_{\mathrm{II}}(x,\cdot)$) is convex.

The extension of all the results of the paper to the case of a finite number of players is trivial and we do not discuss this more general case just for the sake of notational simplicity.

For every fixed y, denote by $\mathcal{S}(y)$ the (possibly empty) solution set of the first player and, similarly, for every fixed x, $\mathcal{S}(x)$ is the (possibly empty) solution set of the second player. We further denote by $\mathcal{F} \subseteq \mathbb{R}^{n_1} \times \mathbb{R}^{n_2}$ the feasible set of the GNEP, i.e. the set of points (x, y) that are feasible for the first and second player at the same time. Note that, by the assumptions made, once y is fixed, the first player's problem is convex in x and analogously we have that the second player's optimization problem is convex in y for every fixed x. A point $(\bar{x}, \bar{y})$ is a solution of the GNEP if $\bar{x} \in \mathcal{S}(\bar{y})$ and $\bar{y} \in \mathcal{S}(\bar{x})$. Solutions of the GNEP are also often called *equilibria.*

The presence of the "coupling" constraints $g^{\mathrm{I}}(x, y)$ and $g^{\mathrm{II}}(x, y)$ which make the feasible set of one player depend on the variables of the other player is what distinguish the GNEP from the standard Nash Equilibrium Problem NEP and actually makes the GNEP an extremely difficult problem. To date there have been very few attempts to define algorithms for the calculation of an equilibrium of a GNEP. One possibility is to write the optimality conditions of the two players using the minimum principle thus obtaining a Quasi Variational Inequality (QVI). This has been recognized by Bensoussan [1] as early as 1974 (see also [11] for the finite-dimensional case). However there is also a lack of proposals for the solution of QVIs, so this reduction is not very appealing from the algorithmic point of view. There are a few other scattered proposals in the literature, based either on fixed point approaches or the use of the Nikaido-Isoda function: [2, 3, 14, 16, 19] (see [15] for the definition of the Nikaido-Isoda function). These kind of algorithms, however, require quite stringent assumptions that cannot be expected to be satisfied in general. On the other hand there is certainly a wealth of interesting applications that call for the solution of GNEPs: see, as a way of example, [4, 10, 13, 17, 18]

Recently, Fukushima and Pang [10] proposed a promising *sequential* penalty approach whereby a solution is sought by solving a sequence of smooth NEPs problems for values of a penalty parameter increasing to infinity. The advantage of this idea is that the the penalized NEPs can be reformulated as Variational Inequalities (VI) to which, in principle, well understood solution methods can be applied, see [9]. In this work we propose a solution framework whereby, by using *exact* penalization techniques, the GNEP is reduced to the solution of a single NEP. The advantage is that we only deal with a single NEP with a finite value of the penalty parameter. The disadvantage is that the players in this NEP have nonsmooth objective functions.

Before describing our approach more in detail we think it is important to set the goal. The GNEP has a structure that exhibits many "convexities" and so one could think that a reasonable goal for a numerical method is to find a solution (or to determine that the problem has no solutions): this would parallel what happens for a convex optimization problem. However the GNEP is really a "non convex" problem. For example, under our assumptions, the

feasible set $\mathcal{F}$ is non convex and even finding a feasible point, let alone a solution, is a difficult task. Therefore we cannot expect that we can solve GNEP unless some (more) stringent assumptions are made.

In nonlinear programming a research line has emerged that attempts to analyze algorithms under minimal assumptions; an algorithm is considered successful if it can be shown to find a "generalized stationary point" under these minimal assumptions. Roughly speaking, a "generalized stationary point" of a nonlinear program is a either a Karush-Kuhn-Tucker (KKT) point or Fritz-John (FJ) point or an (unfeasible) stationary point of some measure of violation of the constraints. After this analysis has been carried out, it is investigated under which additional assumptions one can guarantee that, indeed, the algorithm converges to a KKT point, thus ruling out other undesirable possibilities. This point of view seems very sensible and enriches the usual approach in that when one applies an algorithm to the solution of a nonlinear optimization problem, one does not usually know in advance that the problem satisfies the regularity assumptions required by algorithm (i.e. linear independence of active constrains, positive linear independence of violated constraints and so on). It is then of interest to show that, in any case, the algorithm behaves in a reasonable way, locating a generalized stationary point, and to show that, if in addition some more stringent regularity conditions are satisfied, convergence actually occurs to a KKT point of the problem.

In this paper we parallel these kind of developments and show that the penalization technique we propose can only converge to a "Nash generalized stationary point". Then we give additional conditions to guarantee that convergence occurs to a solution. Our first task on the agenda is then to give a suitable definition of Nash generalized stationary point. Our definition is inspired by similar definitions in the case of nonlinear optimization problems and also takes into account the convexities that are present in the GNEP.

Definition 1. A point $(x, y) \in \mathbb{R}^{n_1+n_2}$ is a Nash generalized stationary point if

1. x is either a KKT or a FJ point of

$$\begin{aligned} &\text{minimize}_x\ \theta_{\mathrm{I}}(x, y) \\ &\text{subject to } h^{\mathrm{I}}(x) \quad \leq 0 \\ &\qquad\qquad g^{\mathrm{I}}(x, y) \leq 0 \end{aligned} \tag{2}$$

or a global minimizer of $\|h^{\mathrm{I}}(\cdot)_+, g^{\mathrm{I}}(\cdot, y)_+\|_2$ with $\|h^{\mathrm{I}}(x)_+, g^{\mathrm{I}}(x, y)_+\|_2 > 0$;

2. y is either a KKT or a FJ point of

$$\begin{aligned} &\text{minimize}_y\ \theta_{\mathrm{II}}(x, y) \\ &\text{subject to } h^{\mathrm{II}}(y) \quad \leq 0 \\ &\qquad\qquad g^{\mathrm{II}}(x, y) \leq 0 \end{aligned} \tag{3}$$

or a global minimizer of $\|h^{\mathrm{II}}(\cdot)_+, g^{\mathrm{II}}(x,\cdot)_+\|_2$ with $\|h^{\mathrm{I}}(y)_+, g^{\mathrm{I}}(x,y)_+\|_2 > 0$.

Two observations are in order. Every solution is clearly a Nash generalized stationary point, but the viceversa does not hold. If x is a KKT point (by which we mean that x, together with appropriate multipliers, satisfies the KKT conditions) then x solves, for the given fixed y, the optimization problem (2) and similarly for y. If x is FJ point, instead, this reflects a "lack of constraint qualification" and in this case we cannot say whether x is or is not a solution to (2). Finally, if x is a global minimizer of the function $\|h^{\mathrm{I}}(\cdot)_+, g^{\mathrm{I}}(\cdot,y)_+\|_2$ with $\|h^{\mathrm{I}}(x)_+, g^{\mathrm{I}}(x,y)_+\|_2 > 0$, this means that, for the given y, problem (2) is unfeasible. Note that the definition of generalized stationary point extends in a natural way similar definitions valid for nonlinear optimization problems. We remark that under the assumptions we will make to analyze the algorithm the existence of a solution is not guaranteed, so it would be unreasonable to expect that any algorithm can surely find one!

2 Exact penalty functions for the GNEP

Our aim is to transform the GNEP (1) problem into a(n unconstrained), nondifferentiable Nash problem by using a penalty approach. To this end we consider the following penalization of the GNEP:

$$\begin{array}{l} \min_x \ \theta_{\mathrm{I}}(x,y) + \rho_{\mathrm{I}} \, \|h^{\mathrm{I}}_+(x), g^{\mathrm{I}}_+(x,y)\|_2 \\ \\ \text{and} \\ \\ \min_y \ \theta_{\mathrm{II}}(x,y) + \rho_{\mathrm{II}} \, \|h^{\mathrm{II}}_+(y), g^{\mathrm{II}}_+(x,y)\|_2, \end{array} \tag{4}$$

where ρ_{I} and ρ_{II} are positive penalty parameters. In this paper all the norms are always Euclidean norms; therefore from now on we will always write $\|\cdot\|$ instead of $\|\cdot\|_2$. By setting

$$P_{\mathrm{I}}(x,y;\rho_{\mathrm{I}}) = \theta_{\mathrm{I}}(x,y) + \rho_{\mathrm{I}} \, \|h^{\mathrm{I}}_+(x), g^{\mathrm{I}}_+(x,y)\|$$

and

$$P_{\mathrm{II}}(x,y;\rho_{\mathrm{II}}) = \theta_{\mathrm{II}}(x,y) + \rho_{\mathrm{II}} \, \|h^{\mathrm{II}}_+(y), g^{\mathrm{II}}_+(x,y)\|,$$

problem (4) can be rewritten as

$$\min_x \ P_{\mathrm{I}}(x,y;\rho_{\mathrm{I}}) \qquad \text{and} \qquad \min_y \ P_{\mathrm{II}}(x,y;\rho_{\mathrm{II}}).$$

It is also possible to penalize only some of constraints, the most natural choice being to penalize only the coupling constraints $g^{\mathrm{I}}_+(x,y) \le 0$ and $g^{\mathrm{II}}_+(x,y) \le 0$. This would give rise to the following penalized Nash problem

$$\min_{x\in X} \ \theta_{\mathrm{I}}(x,y) + \rho_{\mathrm{I}} \, \|g^{\mathrm{I}}_+(x,y)\| \qquad \text{and} \qquad \min_{y\in Y} \ \theta_{\mathrm{II}}(x,y) + \rho_{\mathrm{II}} \, \|g^{\mathrm{II}}_+(x,y)\|,$$

where $X = \{x : h^{\mathrm{I}}(x) \leq 0\}$ and $Y = \{y : h^{\mathrm{II}}(y) \leq 0\}$. Different penalizations could further be considered where, maybe, also some "difficult" constraints in the definition of X or Y are penalized while the simple ones (e.g. box constraints) are not penalized. We do not consider all this variants in this paper, but with some obvious technical changes all the developments we consider for the penalization (4) can be extended to these different penalizations.

Note that for every fixed y, the first player subproblem in (4) is convex, although nondifferentiable, and the same holds for the second player. In this section we show that under appropriate conditions and for sufficiently large (but finite) values of the penalty parameters, the solutions sets of the GNEP (1) and that of the Penalized GNEP (PGNEP) (4) are strongly related. In the next section we will give a general scheme that allows us to iteratively update the penalty parameters in an appropriate way supposing that a minimization algorithm is available for the solution of the PGNEP (4) for fixed values of the penalty parameters.

The first result we discuss is basically known (see [12] for example), however we report it here with a proof for sake of completeness and also because we could not find a reference with the precise statement below that we need.

Proposition 1. Consider the minimization problem

$$\begin{aligned} &\text{minimize}_z \ f(z) \\ &\text{subject to } v(z) \leq 0, \end{aligned} \tag{5}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ and $v : \mathbb{R}^n \to \mathbb{R}^m$ are C^1 and (componentwise) convex. Let $\bar{z}$ be a solution of this problem and assume that set M of KKT multipliers is non-empty. Let λ be any multiplier in M. Then, for every $\rho > \|\lambda\|$, the solution sets of (5) and

$$\text{minimize}_z \ f(z) + \rho \|v_+(z)\|$$

coincide.

Proof. Let $\tilde{z}$ be a solution of (5) and suppose that $\rho > \|\lambda\|$. We show that $\tilde{z}$ is a minimum point of the penalty function $P(z;\rho) \equiv f(z) + \rho \|v_+(z)\|$ (which is also convex). We recall that $\tilde{z}$ is a minimum point of the Lagrangian of the constrained problem (5), that is

$$f(z) + \lambda^T g(z) \geq f(\tilde{z}) + \lambda^T g(\tilde{z}), \qquad \forall z \in \mathbb{R}^n$$

(λ is a fixed KKT multiplier; recall that the set M of multipliers of problem (5) does not depend on the solution considered). Therefore we get, for any $z \in \mathbb{R}^n$,

$$\begin{aligned} f(z) + \rho \|v_+(z)\| &> f(z) + \|\lambda\| \, \|v_+(z)\| \\ &\geq f(z) + \lambda^T v(z)_+ \\ &\geq f(z) + \lambda^T v(z) \\ &\geq f(\tilde{z}) + \lambda^T v(\tilde{z}) \\ &= f(\tilde{z}) \\ &= f(\tilde{z}) + \rho \|v_+(\tilde{z})\|, \end{aligned}$$

where we have used the facts that $\lambda \geq 0$, $\lambda^T v(\tilde{z}) = 0$ and $v_+(\tilde{z}) = 0$. Suppose now that $\tilde{z}$ is a (global) minimum point of the penalty function $P(z, \rho)$. Since the penalty function P and the objective function f coincide on the feasible set of problem (5), in order to show that $\tilde{z}$ is a solution of problem (5) it is sufficient to verify that $\tilde{z}$ is feasible to problem (5). If this were not the case

$$\begin{aligned} P(\tilde{z}; \rho) &> P(\tilde{z}; \|\lambda\|) \\ &\geq f(\tilde{z}) + \lambda^T v(\tilde{z})_+ \\ &\geq f(\tilde{z}) + \lambda^T v(\tilde{z}) \\ &\geq f(\bar{z}) + \lambda^T g(\bar{z}) \\ &= f(\bar{z}) \\ &= P(\bar{z}; \rho), \end{aligned}$$

(where the fourth inequality derives again from the fact that $\bar{z}$ is a minimum point of the Lagrangian of the constrained problem (5)). This would show that $\tilde{z}$ is not a global minimum of the penalty function thus giving a contradiction and therefore $\tilde{z}$ must be feasible. △

Consider now the first player's problem in (1). We can see it as a collection of convex optimization problems, one for each possible y. The same holds for the second player's problem, with the role of "parameter" taken here by x. The previous theorem suggests that if we want to penalize (1) and obtain some kind of equivalence to the penalized problem for finite values of the penalty parameters, we should require the boundedness of the multipliers of the player's problems for each value of the other player's variables. This lead us to the following assumption.

Assumption 2.1 (Generalized Sequential Boundedness Constraint Qualification (GSBCQ)) There exists a constant M such that for every solution $(\bar{x}, \bar{y})$ of the generalized Nash equilibrium problem there exists a corresponding couple (λ, μ) of KKT multipliers such that $\|(\lambda, \mu)\| \leq M$.

We refer to chapter 3 in [9] for more details on this constraint qualification (CQ) condition (in the case of optimization problems). Here we only stress that the GSBCQ appear to be a rather mild constraint qualification that unifies many standard CQ such as the Mangasarian-Fromovitz CQ and the constant rank one.

Under this assumption it is easy to prove the following result.

Theorem 1. Suppose that the GSBCQ is satisfied. Then there exists $\bar{\rho}_{\mathrm{I}} \geq 0$ and $\bar{\rho}_{\mathrm{II}} \geq 0$ such that for every $\rho_{\mathrm{I}} \geq \bar{\rho}_{\mathrm{I}}$ and $\rho_{\mathrm{II}} \geq \bar{\rho}_{\mathrm{II}}$ every solution of (1) is a solution of (4); viceversa every solution of (4) feasible to (1) is a solution of (1) (independent of the values of the penalty parameters).

Proof. Take ρ_{I} and ρ_{II} larger then the constant M in the definition of the GSBCQ. Suppose $(\bar{x}, \bar{y})$ is a solution of the GNEP. Then, by Proposition 1 $(\bar{x}, \bar{y})$ is also a solution of (4). Viceversa, assume that $(\bar{x}, \bar{y})$ is a solution of (4) feasible to (1). It is trivial to see then, that it is also a solution for (1). $\triangle$

This result is somewhat weaker than the corresponding one in the case of constrained optimization, where in the second part there is no necessity to postulate the feasibility of the solution of (4) to conclude that the point is a solution of (1). However we do not believe that it is possible and, actually, sensible to expect such a result in the case of a GNEP. In the case of penalty functions for constrained optimization, in fact, a basic assumption is always that the optimization problem is feasible. In the case of GNEP, instead, we deal with (looking at the first player, for example) an infinite number of optimization problems, one for each possible y, and some of this problems can be expected to have no solution or even no feasible points.

Theorem 1 is certainly of interest and basically shows that a penalty approach for GNEPs has sound bases. In the next section we give a general algorithmic scheme and show, on the basis of Theorem 1, that this penalty algorithmic scheme can locate generalized stationary points.

3 Updating the penalty parameters

In general the correct value of the penalty parameters for which the solutions of the generalized Nash problem (1) and those of the Nash problem (4) coincide is not known in advance. Therefore, a strategy must be envisaged that allows to update the values of penalty parameter so that eventually the correct values are reached. In this section we show how this is possible in a broad algorithmic framework.

The aim of the penalization method is to transform the original problem into one that is easier to solve. It is clear that, in principle, (4) is easier than (1), even if the non differentiabilty of the players' objective functions is somewhat problematic, at least in practice. There exist methods to deal with nondifferentiable (4) and the equivalent VI-type reformulation. Furthermore, ad-hoc methods (such as smoothing methods, for example) could be developed to deal with the very structured nondifferentiability of the objective functions of (4). In this paper we do not go into these technical details. Our aim is, instead, to give a broad framework that is as general as possible to show the viability of the approach. To this end, we simply assume that we have a "reasonable" algorithm for the solution of the Nash problem (4). To be more

precise we suppose that we have at our disposal an iterative algorithm $\mathcal{A}$ that, given a point (x^k, y^k), generates a new point $(x^{k+1}, y^{k+1}) = \mathcal{A}[(x^k, y^k)]$. We make the following absolutely natural and basic assumption on the algorithm $\mathcal{A}$.

Assumption 3.1 For every (x^0, y^0), the sequence $\{(x^k, y^k)\}$ obtained by iteratively applying the algorithm $\mathcal{A}$ is such that every of its limit points (if any) is a solution of (4).

It is clear that virtually every algorithm that can be said to "solve" (4) will satisfy this assumption. We can now consider the following algorithmic scheme. Below we denote by $\mathcal{F}(y)$ the feasible set of the first player for a given y and, similarly $\mathcal{F}(x)$ is the feasible region of the second player for a given x.

General penalty updating scheme

Algorithm. 2
Data: $(x^0, y^0) \in \mathbb{R}^{n_1+n_2}$, $\rho_{\mathrm{I}}, \rho_{\mathrm{II}} > 0$, $c_{\mathrm{I}}, c_{\mathrm{II}} \in (0,1)$. Set $k = 0$.
Step 1: If (x^k, y^k) is a solution of (1) STOP.
Step 2: If $x^k \notin \mathcal{F}(y^k)$ and

$$\|\nabla_x \theta_{\mathrm{I}}(x^k, y^k)\| > c_{\mathrm{I}} \left[\rho_{\mathrm{I}} \left\| \nabla_x \|h_{\mathrm{I}}^+(x^k), g_{\mathrm{I}}^+(x^k, y^k)\| \right\|\right], \tag{6}$$

then double ρ_{I} until (6) is not satisfied.
Step 3: If $y^k \notin \mathcal{F}(x^k)$ and

$$\|\nabla_y \theta_{\mathrm{II}}(x^k, y^k)\| > c_{\mathrm{II}} \left[\rho_{\mathrm{II}} \left\| \nabla_y \|h_{\mathrm{II}}^+(y^k), g_{\mathrm{II}}^+(x^k, y^k)\| \right\|\right], \tag{7}$$

then double ρ_{II} until (7) is not satisfied.
Step 4: Compute $(x^{k+1}, y^{k+1}) = \mathcal{A}[(x^k, y^k)]$; set $k \leftarrow k+1$ and go to step 1.

Note that if the perform the test (6) the point x^k is not feasible for the first player, so that $\|h_{\mathrm{I}}^+(x^k), g_{\mathrm{I}}^+(x^k, y^k)\| > 0$ and, since the norm is the Euclidean norm, the function $\|h_{\mathrm{I}}^+(\cdot), g_{\mathrm{I}}^+(\cdot, y^k)\|$ is continuously differentiable around x^k and the test (6) is well defined. Similar arguments can be made for the test at Step 3.

In what follows we assume that the stopping criterion at Step 1 is never satisfied and study the behavior of Algorithm 2.

Theorem 3. Let the sequence $\{(x^k, y^k)\}$ produced by the Algorithm 2 be bounded. If either ρ_{I} or ρ_{II} are updated an infinite number of times, then every limit point of the sequence $\{(x^k, y^k)\}$ is a generalized stationary point of the GNEP (1). If instead the penalty parameters ρ_{I} and ρ_{II} are updated only a finite number of times, then every limit point of the sequence $\{(x^k, y^k)\}$ is a solution of the GNEP (1).

Proof. Suppose the both penalty parameters are updated a finite number of times only. Therefore for k sufficiently large we are applying the algorithm $\mathcal{A}$

to problem (4) for fixed penalty parameters. We denote these fixed values by ρ_{I} and ρ_{II}. Hence, by the assumption made on $\mathcal{A}$ we know that every limit point of the sequence $\{(x^k, y^k)\}$ is a solution of (4). Let $(\bar{x}, \bar{y})$ be such a limit point. We want to show that $(\bar{x}, \bar{y})$ is also a solution of (1). By Theorem 1 we only have to show that $(\bar{x}, \bar{y})$ is feasible for (1). Suppose this is not true and assume, without loss of generality, that the constraints of the first player are not satisfied at $(\bar{x}, \bar{y})$. Furthermore, since $(\bar{x}, \bar{y})$ is a solution of (4) and $\|h_{\mathrm{I}}^+(\bar{x}), g_{\mathrm{I}}^+(\bar{x}, \bar{y})\| > 0$, we can write

$$\nabla_x \theta_{\mathrm{I}}(\bar{x}, \bar{y}) + \rho_{\mathrm{I}} \nabla_x \|h_{\mathrm{I}}^+(\bar{x}), g_{\mathrm{I}}^+(\bar{x}, \bar{y})\| = 0,$$

from which we deduce

$$\|\nabla_x \theta_{\mathrm{I}}(\bar{x}, \bar{y})\| = \rho_{\mathrm{I}} \left\| \nabla_x \|h_{\mathrm{I}}^+(\bar{x}), g_{\mathrm{I}}^+(\bar{x}, \bar{y})\| \right\|.$$

But this, together with $c_{\mathrm{I}} < 1$ and some simple continuity arguments, shows that the tests at Step 2 must be satisfied eventually and ρ_{I} updated. This contradiction shows that $(\bar{x}, \bar{y})$ is feasible for both players.

Consider now the case in which at least one penalty parameter is updated an infinite number of times. Without loss of generality assume that it is ρ_{I} that is updated an infinite number of times and that the infinite subsequence of iterations where the updating occurs is K. If $\{(\bar{x}, \bar{y})\}$ is the limit of a subsequence of $\{(x^k, y^k)\}$ with $k \notin K$ we can reason as in the previous case and conclude that $\{(\bar{x}, \bar{y})\}$ is a solution of (1). Let us analyze then the case in which $\{(\bar{x}, \bar{y})\}$ is the limit of a subsequence of $\{(x^k, y^k)\}$ with $k \in K$. We have that the the sequence (x^k, y^k) is bounded by assumption and so is, by continuity, $\{\nabla_x \theta_{\mathrm{I}}(x^k, y^k)\}_K$. Therefore, since the test (6) is satisfied for every $k \in K$ and the penalty parameter goes to infinity on the subsequence K, we can conclude that

$$\left\{ \left\| \nabla_x \|h_{\mathrm{I}}^+(x^k), g_{\mathrm{I}}^+(x^k, y^k)\| \right\| \right\}_K \to 0.$$

If $(\bar{x}, \bar{y})$ is infeasible we then have by continuity that $\nabla_x \|h_{\mathrm{I}}^+(\bar{x}), g_{\mathrm{I}}^+(\bar{x}, \bar{y})\| = 0$ and therefore, since $\|h_{\mathrm{I}}^+(x), g_{\mathrm{I}}^+(x, y)\|$ is convex in x (for a fixed y), this means that $\bar{x}$ is a global minimizer of the function $\|h^{\mathrm{I}}(\cdot)_+, g^{\mathrm{I}}(\cdot, y)_+\|$ with $\|h^{\mathrm{I}}(\bar{x})_+, g^{\mathrm{I}}(\bar{x}, \bar{y})_+\| > 0$.

If $(\bar{x}, \bar{y})$ is feasible, we have, taking into account that every x^k with $k \in K$ is infeasible for the first player, that

$$\nabla_x \|h_{\mathrm{I}}^+(x^k), g_{\mathrm{I}}^+(x^k, y^k)\| = \frac{\nabla_x h^{\mathrm{I}}(x^k)\, h_+^{\mathrm{I}}(x^k) + \nabla_x g^{\mathrm{I}}(x^k, y^k)\, g_+^{\mathrm{I}}(x^k, y^k)}{\sqrt{\|h_+^{\mathrm{I}}(x^k)\|^2 + \|g_+^{\mathrm{I}}(x^k, y^k)\|^2}}.$$

Passing to the limit for $k \to \infty$, $k \in K$, it is easy to check that $\bar{x}$ is a FJ point for the first player (when the second player chooses the strategy $\bar{y}$). It is now immediate to see, reasoning along similar lines, that also $\bar{y}$ must be either a solution or a FJ point for the second player or global minimizer of the

function $\|h^{\mathrm{II}}(\cdot)_+, g^{\mathrm{II}}(x,\cdot)_+\|$ with $\|h^{\mathrm{I}}(\bar{x})_+, g^{\mathrm{I}}(\bar{x},\bar{y})_+\| > 0$. Hence we conclude that in any case $(\bar{x},\bar{y})$ is a generalized stationary point of the GNEP (1). $\triangle$

It should be clear that there is no need to perform the tests at steps 2 and 3 for every k. It is enough that they are performed an infinite number of times. Also, if the updating test, say that at Step 2, is passed, it is sufficient to take the new ρ_{I} larger than

$$2\frac{\|\nabla_x \theta_{\mathrm{I}}(\bar{x},\bar{y})\|}{\left\|\nabla_x \|h_{\mathrm{I}}^+(\bar{x}), g_{\mathrm{I}}^+(\bar{x},\bar{y})\|\right\|} + 1.$$

Actually any updating rule for the penalty parameter ρ_{I} will be acceptable as far as it is guaranteed that if ρ_{I} is updated an infinite number of times then it grows to infinity. We leave the details to the reader and discuss instead the more interesting issue of whether it is possible to guarantee that every limit point is a solution and not just a generalized stationary point of the GNEP (1). Note that in Theorem 3 we did not make any regularity assumption on the problem; correspondingly, and quite naturally, we could prove convergence to generalized stationary points and not to solutions. However, Theorem 3 makes clear that convergence to generalized stationary points that are not solutions can only occur if a(t least one) penalty parameter goes to infinity. In turn, the proof of Theorem 3 shows that if this occurs, then we can find a sequence $\{x^k, y^k\}$ of infeasible points such that either $\{\nabla_x \|h_{\mathrm{I}}^+(x^k), g_{\mathrm{I}}^+(x^k,y^k)\|\}$ or $\{\nabla_y \|h_{\mathrm{II}}^+(y^k), g_{\mathrm{II}}^+(x^k,y^k)\|\}$ tend to zero. The following corollary then easily follows from the above considerations.

Corollary 1. Let the sequence $\{(x^k,y^k)\}$ produced by the algorithm 2 belong to a bounded set $\mathcal{B}$. Suppose that there exists a positive constant σ such that, for every infeasible point $(x,y) \in \mathcal{B}$,

$$\left\|\nabla_x \|h_{\mathrm{I}}^+(x^k), g_{\mathrm{I}}^+(x^k,y^k)\|\right\| \geq \sigma, \qquad \left\|\nabla_x \|h_{\mathrm{I}}^+(x^k), g_{\mathrm{I}}^+(x^k,y^k)\|\right\| \geq \sigma. \quad (8)$$

Then ρ_{I} or ρ_{II} are updated an finite number of times and every limit point of the sequence $\{(x^k,y^k)\}$ is a solution of the GNEP (1). $\triangle$

In the case of one player (i.e. in the case of optimization problems) the condition (8) has been used and analyzed in detail, see [5–8]. Basically this condition can be viewed as a sort of generalization of the Mangasarian-Fromovitz CQ. Its practical meaning is rather obvious: the functions $\|h_{\mathrm{I}}^+(x), g_{\mathrm{I}}^+(x,y)\|$ and $\|h_{\mathrm{II}}^+(y), g_{\mathrm{II}}^+(x,y)\|$ which represent the violation of the constraints of the first and second player respectively, must not have stationary points outside the feasible set. This condition seems very natural and says that the "feasibility" problem which is a "part" of the generalized Nash equilibrium problem is easy (in the sense that the only stationary points of the functions representing the violation of the constraints are the global minima). From this condition we could derive several sets of sufficient conditions for Corollary 1 to hold, along the lines developed in [5–8]. We leave this for future research.

4 Conclusions

In this paper we proposed the notion of generalized stationary point for the Generalized Nash Equilibrium Problem and argued that this is an appropriate and realistic target for any numerical solution method. Furthermore we introduced an exact penalization method for the solution of the GNEP. We gave a broad algorithmic scheme and showed that this scheme is able to generate sequences converging to a generalized stationary point under a mere boundedness assumption. Finally we also discussed an additional regularity condition that guarantees convergence to solutions (as opposed to generalized stationarity points). There are certainly still many issues that deserve more study, prominent among these an effective solution procedure for the nondifferentiable (unconstrained) problem arising from the application of the exact penalty approach. It certainly was not our intention to investigate all the issues connected to a penalization approach to the solution of a GNEP. However, we remark that, given the lack of results in the study of GNEPs, we believe that the approach proposed in this paper could not only be useful from the numerical point of view, but also lead to new sensitivity and stability results.

Acknowledgments

The work of the first author has been partially supported by MIUR-FIRB Research Project RBNE01WBBB "Large Scale Nonlinear Optimization".

References

1. A. Bensoussan. Points de Nash dans le cas de fonctionnelles quadratiques et jeux differentiels lineaires a N personnes. *SIAM Journal on Control* 12 (1974) 460–499.
2. T. Başar. Relaxation techniques and asynchronous algorithms for on-line computation of non-cooperative equilibria. *Journal of Economic Dynamics and Control* 11 (1987) 531–549.
3. S. Berridge and J.B. Krawczyc. Relaxation algorithms in finding Nash equilibria. Economic working papers archives(1997) URL: http://econwpa.wustl.edu/eprints/comp/papers/9707/9707002.abs
4. J. Contreras, M.K. Klusch and J.B. Krawczyc. Numerical solution to Nash-Cournot equilibria in coupled constraints electricity markets. *IEEE Transactions on Power Systems* 19 (2004) 195–206.
5. V.F. Demyanov, G. Di Pillo, and F. Facchinei. Exact penalization via Dini and Hadamard constrained derivatives. *Optimization Methods and Software* 9 (1998) 19–36.
6. G. Di Pillo and F. Facchinei. Exact penalty functions for nondifferentiable programming problems. In F.H. Clarke, V.F. Demyanov, and F. Giannessi, editors, Nonsmooth Optimization and Related Topics, 89–107, Plenum Press (New York 1989)

7. G. Di Pillo and F. Facchinei. Regularity conditions and exact penalty functions in Lipschitz programming problems. In F. Giannessi, editor, Nonsmooth Optimization Methods and Applications, 107–120, Gordon and Breach (New York 1992)
8. G. Di Pillo and F. Facchinei. Exact barrier functions methods for Lipschitz Programs. *Applied Mathematics and Optimization* 32 (1995) 1–31.
9. F. Facchinei and J.-S. Pang. *Finite-Dimensional Variational Inequalities and Complementarity Problems*, Springer Verlag (New York 2003).
10. M. Fukushima and J.-S. Pang. Quasi-variational inequalities, generalized Nash equilibria, and multi-leader-follower games. *Computational Management Science*, to appear.
11. P.T. Harker. Generalized Nash games and quasivariational inequalities. *European Journal of Operations research* 54 (1991) 81–94.
12. J.-B. Hiriart-Urruty, and C. Lemaréchal. *Convex Analysis and Minimization Algorithms*, Springer Verlag (New York 1993).
13. J.B. Krawczyc and S. Uryasev. Relaxation algorithms to find Nash equilibria with economic applications. *Environmental Modelling and Assessment* 5 (2000) 63–73.
14. S. Li and T. Başar. Distributed algorithms for the computation of noncooperative equilibria. *Automatica* 23 (1987) 523–533.
15. H. Nikaido and K. Isoda. Note on noncooperative convex games. *Pacific Journal of Mathematics* 5 (1955) 807–815.
16. G.P. Papavassilopoulos. Iterative techniques for the Nash solution in quadratic games with unknown parameters. *SIAM Journal on Control and Optimization* 24 (1986) 821–834.
17. S.M. Robinson. Shadow prices for measures of effectiveness. II. Linear Model. *Operations Research* 41 (1993) 518–535.
18. S.M. Robinson. Shadow prices for measures of effectiveness. II. General Model. *Operations Research* 41 (1993) 536–548.
19. S. Uryasev and R. Y. Rubinstein. On relaxation algorithms in computation of noncooperative equilibria. *IEEE Transactions on Automatic Control* 39 (1994) 1263–1267.

Parametric Sensitivity Analysis for Optimal Boundary Control of a 3D Reaction-Diffusion System

Roland Griesse[1] and Stefan Volkwein[2]

[1] Johann Radon Institute for Computational and Applied Mathematics (RICAM), Austrian Academy of Sciences, Altenbergerstraße 69, A–4040 Linz, Austria (roland.griesse@oeaw.ac.at)

[2] Institute for Mathematics and Scientific Computing, Heinrichstraße 36, A–8010 Graz, Austria (stefan.volkwein@uni-graz.at)

Summary. A boundary optimal control problem for an instationary nonlinear reaction-diffusion equation system in three spatial dimensions is presented. The control is subject to pointwise control constraints and a penalized integral constraint. Under a coercivity condition on the Hessian of the Lagrange function, an optimal solution is shown to be a directionally differentiable function of perturbation parameters such as the reaction and diffusion constants or desired and initial states. The solution's derivative, termed parametric sensitivity, is characterized as the solution of an auxiliary linear-quadratic optimal control problem. A numerical example illustrates the utility of parametric sensitivities which allow a quantitative and qualitative perturbation analysis of optimal solutions.

Key words: optimal control, reaction-diffusion equations, sensitivity analysis.

1 Introduction

Parametric sensitivity analysis for optimal control problems governed by partial differential equations (PDE) is concerned with the behavior of optimal solutions under perturbations of system data. The subject matter of the present paper is an optimal boundary control problem for a time-dependent coupled system of semilinear parabolic reaction-diffusion equations. The equations model a chemical or biological process where the species involved are subject to diffusion and reaction among each other. The goal in the optimal control problem is to drive the reaction-diffusion model from the given initial state as close as possible to a desired terminal state. However, the control has to be chosen within given upper and lower bounds which are motivated by physical or technological considerations.

In practical applications, it is unlikely that all parameters in the model are precisely known a priori. Therefore, we embed the optimal control problem into a family of problems, which depend on a parameter vector p. In our case, p can comprise physical parameters such as reaction and diffusion constants, but also desired terminal states, etc. In this paper we prove that under a coercivity condition on the Hessian of the Lagrange function, local solutions of the optimal control problem depend Lipschitz continuously and directionally differentiably on the parameter p. Moreover, we characterize the derivative as the solution of an additional linear-quadratic optimal control problem, known as the sensitivity problem. If these sensitivities are computed "offline", *i.e.*, along with the optimal solution of the nominal (unperturbed) problem belonging to the expected parameter value p_0, a first order Taylor approximation can give a real-time ("online") estimate of the perturbed solution.

Let us put the current paper into a wider perspective: Lipschitz dependence and differentiability properties of parameter-dependent optimal control problems for PDEs have been investigated in the recent papers [6,11–14,16,18]. In particular, sensitivity results have been derived in [6] for a two-dimensional reaction-diffusion model with distributed control. In contrast, we consider here the more difficult situation in three spatial dimensions and with boundary control and present both theoretical and numerical results. Other numerical results can be found in [3,7].

The main part of the paper is organized as follows: In Section 2, we introduce the reaction-diffusion system at hand and the corresponding optimal control problem. We also state its first order optimality conditions. Since this problem, without parameter dependence, has been thoroughly investigated in [9], we only briefly recall the main results. Section 3 is devoted to establishing the so-called *strong regularity property* for the optimality system. This necessitates the investigation of the *linearized* optimality system for which the solution is shown to be Lipschitz and differentiable with respect to perturbations. In Section 4, these properties for the linearized problem are shown to carry over to the original nonlinear optimality system, in virtue of a suitable implicit function theorem. Finally, we present some numerical results in Section 5 in order to further illustrate the concept of parametric sensitivities.

Necessarily all numerical results are based on a discretized version of our infinite-dimensional problem. Nevertheless we prefer to carry out the analysis in the continuous setting so that smoothness properties of the involved quantities become evident which could then be used for instance to determine rates of convergence under refinements of the discretization etc. In view of our problem involving a nonlinear time-dependent system of partial differential equations, its discretization yields a large scale nonlinear optimization problem, albeit with a special structure.

2 The Reaction-Diffusion Optimal Boundary Control Problem

Reaction-diffusion equations model chemical or biological processes where the species involved are subject to diffusion and reaction among each other. As an example, we consider the reaction $A+B \to C$ which obeys the law of mass action. To simplify the discussion, we assume that the backward reaction $C \to A+B$ is negligible and that the forward reaction proceeds with a constant (not temperature-dependent) rate. This leads to a coupled semilinear parabolic system for the respective concentrations (c_1, c_2, c_3) as follows:

$$\frac{\partial}{\partial t}c_1(t,x) = d_1 \Delta c_1(t,x) - k_1 c_1(t,x) c_2(t,x) \qquad \text{for all } (t,x) \in Q, \quad (1a)$$

$$\frac{\partial}{\partial t}c_2(t,x) = d_2 \Delta c_2(t,x) - k_2 c_1(t,x) c_2(t,x) \qquad \text{for all } (t,x) \in Q, \quad (1b)$$

$$\frac{\partial}{\partial t}c_3(t,x) = d_3 \Delta c_3(t,x) + k_3 c_1(t,x) c_2(t,x) \qquad \text{for all } (t,x) \in Q. \quad (1c)$$

The scalars d_i and k_i, $i = 1, \ldots, 3$, are the diffusion and reaction constants, respectively. Here and throughout, let $\Omega \subset \mathbb{R}^3$ denote the domain of reaction and let $Q = (0,T) \times \Omega$ be the time-space cylinder where $T > 0$ is the given final time. We suppose that the boundary $\Gamma = \partial\Omega$ is Lipschitz and can be decomposed into two disjoint parts $\Gamma = \Gamma_n \cup \Gamma_c$, where Γ_c denotes the control boundary. Moreover, we let $\Sigma_n = (0,T) \times \Gamma_n$ and $\Sigma_c = (0,T) \times \Gamma_c$. We impose the following Neumann boundary conditions:

$$d_1 \frac{\partial c_1}{\partial n}(t,x) = 0 \qquad \text{for all } (t,x) \in \Sigma, \quad (2a)$$

$$d_2 \frac{\partial c_2}{\partial n}(t,x) = u(t)\,\alpha(t,x) \qquad \text{for all } (t,x) \in \Sigma_c, \quad (2b)$$

$$d_2 \frac{\partial c_2}{\partial n}(t,x) = 0 \qquad \text{for all } (t,x) \in \Sigma_n, \quad (2c)$$

$$d_3 \frac{\partial c_3}{\partial n}(t,x) = 0 \qquad \text{for all } (t,x) \in \Sigma. \quad (2d)$$

Equation (2b) prescribes the boundary flux of the second substance B by means of a given shape function $\alpha(t,x) \geq 0$, modeling, *e.g.*, the location of a spray nozzle revolving with time around one of the surfaces of Ω, while $u(t)$ denotes the control intensity at time t which is to be determined. The remaining homogeneous Neumann boundary conditions simply correspond to a "no-outflow" condition of the substances through the boundary of the reaction vessel Ω.

In order to complete the description of the model, we impose initial conditions for all three substances involved, *i.e.*,

$$c_1(0,x) = c_{10}(x) \quad \text{for all } x \in \Omega, \quad (3a)$$

$$c_2(0,x) = c_{20}(x) \quad \text{for all } x \in \Omega, \quad (3b)$$

$$c_3(0,x) = c_{30}(x) \quad \text{for all } x \in \Omega. \quad (3c)$$

Our goal is to drive the reaction-diffusion model (1)–(3) from the given initial state near a desired terminal state. Hence, we introduce the cost functional

$$J_1(c_1, c_2, u) = \frac{1}{2} \int_\Omega \left(\beta_1 \left| c_1(T) - c_{1T} \right|^2 + \beta_2 \left| c_2(T) - c_{2T} \right|^2 \right) \mathrm{d}x + \frac{\gamma}{2} \int_0^T |u - u_d|^2 \, \mathrm{d}t.$$

Here and in the sequel, we will find it convenient to abbreviate the notation and write $c_1(T)$ instead of $c_1(T, \cdot)$ or omit the arguments altogether when no ambiguity arises.

In the cost functional, β_1, β_2 and γ are non-negative weights, c_{1T} and c_{2T} are the desired terminal states, and u_d is some desired (or expected) control. In order to shorten the notation, we have assumed that the objective J_1 does not depend on the product concentration c_3. This allows us to delete the product concentration c_3 from the equations altogether and consider only the system for (c_1, c_2). All results obtained can be extended to the three-component system in a straightforward way.

The control $u : [0, T] \to \mathbb{R}$ is subject to pointwise box constraints $u_a(t) \leq u(t) \leq u_b(t)$. It is reasonable to assume that $u_a(t) \geq 0$, which together with $\alpha(t, x) \geq 0$ implies that the second (controlled) substance B can not be withdrawn through the boundary. The presence of an upper limit u_b is motivated by technological reasons. In addition to the pointwise constraint, it may be desirable to limit the total amount of substance B added during the process, *i.e.*, to impose a constraint like

$$\int_0^T u(t) \, \mathrm{d}t \leq u_c.$$

In the current investigation, we do not enforce this inequality directly but instead we add a penalization term

$$J_2(u) = \frac{1}{\varepsilon} \max \left\{ 0, \int_0^T u(t) \, \mathrm{d}t - u_c \right\}^3$$

to the objective, which then assumes the final form

$$J(c_1, c_2, u) = J_1(c_1, c_2, u) + J_2(u). \tag{4}$$

Our optimal control problem can now be stated as problem (**P**)

$$\begin{array}{lll} \text{Minimize} \quad J(c_1, c_2, u) & \text{s.t.} & \text{(1a)–(1b), (2a)–(2c) and (3a)–(3b)} \\ & \text{and} & u_a(t) \leq u(t) \leq u_b(t) \quad \text{hold.} \end{array} \tag{**P**}$$

2.1 State Equation and Optimality System

The results in this section draw from the investigations carried out in [9] and are stated here for convenience and without proof. Our problem (**P**) can be posed in the setting

$$u \in U = L^2(0,T)$$
$$(c_1, c_2) \in Y = W(0,T) \times W(0,T).$$

That is, we consider the state equation (1a)–(1b), (2a)–(2c) and (3a)–(3b) in its weak form, see Remark 1 and Section 2.2 for details. Here and throughout, $L^2(0,T)$ denotes the usual Sobolev space [1] of square-integrable functions on the interval $(0,T)$ and the Hilbert space $W(0,T)$ is defined as

$$W(0,T) = \{\varphi \in L^2(0,T;H^1(\Omega)) : \frac{\partial}{\partial t}\varphi \in L^2(0,T;H^1(\Omega)')\}.$$

containing functions of different regularity in space and time. Here, $H^1(\Omega)$ is again the usual Sobolev space and $H^1(\Omega)'$ is its dual. At this point we note for later reference the compact embedding [17, Chapter 3, Theorem 2.1]

$$W(0,T) \hookrightarrow\hookrightarrow L^2(0,T;H^s(\Omega)) \quad \text{for any } 1/2 < s < 1 \tag{5}$$

involving the fractional-order space $H^s(\Omega)$. For convenience of notation, we define the admissible set

$$U_{\text{ad}} = \{u \in U \ : \ u_a(t) \le u(t) \le u_b(t)\}.$$

Let us summarize the fundamental results about the state equation and problem (**P**). We begin with the following assumption which is needed throughout the paper:

Assumption 1 *(a) Let $\Omega \subset \mathbb{R}^3$ be a bounded open domain with Lipschitz continuous boundary $\Gamma = \partial\Omega$, which is partitioned into the control part Γ_c and the remainder Γ_n. Let d_i and k_i, $i = 1,2$ be positive constants, and assume that $\alpha \in L^\infty(0,T;L^2(\Gamma_c))$ is non-negative. The initial conditions c_{i0}, $i = 1,2$ are supposed to be in $L^2(\Omega)$. $T > 0$ is the given final time of the process.*

(b) For the control problem, we assume desired terminal states $c_{iT} \in L^2(\Omega)$, $i = 1,2$, and desired control $u_d \in L^2(0,T)$ to be given. Moreover, let β_1, β_2 be non-negative and γ be positive. Finally, we assume that the penalization parameter ε is positive and that $u_c \in \mathbb{R}$ and u_a and u_b are in $L^\infty(0,T)$ such that $\int_0^T u_a(t)\,\mathrm{d}t \le u_c$.

Theorem 1. *Under Assumption 1(a), the state equation (1a)–(1b), (2a)–(2c) and (3a)–(3b) has a unique weak solution $(c_1, c_2) \in W(0,T) \times W(0,T)$ for any given $u \in L^2(0,T)$. The solution satisfies the a priori estimate*

$$\|c_1\|_{W(0,T)} + \|c_2\|_{W(0,T)} \le C\left(1 + \|c_{10}\|_{L^2(\Omega)} + \|c_{20}\|_{L^2(\Omega)} + \|u\|_{L^2(0,T)}\right)$$

with some constant $C > 0$.

In order to state the system of first order necessary optimality conditions, we introduce the active sets

$$A_-(u) = \{t \in [0,T] \ : \ u(t) = u_a(t)\}$$
$$A_+(u) = \{t \in [0,T] \ : \ u(t) = u_b(t)\}$$

for any given control $u \in U_{\mathrm{ad}}$.

Theorem 2. *Under Assumption 1, the optimal control problem (**P**) possesses at least one global solution in $Y \times U_{\mathrm{ad}}$. If $(c_1, c_2, u) \in Y \times U_{\mathrm{ad}}$ is a local solution, then there exists a unique adjoint variable $(\lambda_1, \lambda_2) \in Y$ satisfying*

$$-\frac{\partial}{\partial t}\lambda_1 - d_1 \Delta \lambda_1 = -k_1 c_2 \lambda_1 - k_2 c_2 \lambda_2 \qquad \text{in } Q, \tag{6a}$$
$$-\frac{\partial}{\partial t}\lambda_2 - d_2 \Delta \lambda_2 = -k_1 c_1 \lambda_1 - k_2 c_1 \lambda_2 \qquad \text{in } Q, \tag{6b}$$
$$d_1 \frac{\partial \lambda_1}{\partial n} = 0 \qquad \text{on } \Sigma, \tag{6c}$$
$$d_2 \frac{\partial \lambda_2}{\partial n} = 0 \qquad \text{on } \Sigma, \tag{6d}$$
$$\lambda_1(T) = -\beta_1(c_1(T) - c_{1T}) \qquad \text{in } \Omega, \tag{6e}$$
$$\lambda_2(T) = -\beta_2(c_2(T) - c_{2T}) \qquad \text{in } \Omega \tag{6f}$$

in the weak sense, and a unique Lagrange multiplier $\xi \in L^2(0,T)$ such that the optimality condition

$$\gamma(u(t) - u_d(t)) + \frac{3}{\varepsilon} \max\left\{0, \int_0^T u(t)\,\mathrm{d}t - u_c\right\}^2 - \int_{\Gamma_c} \alpha(t,x)\,\lambda_2(t,x)\,\mathrm{d}x + \xi(t) = 0 \tag{7}$$

holds for almost all $t \in [0,T]$, together with the complementarity condition

$$\xi|_{A_-(u)} \le 0, \quad \xi|_{A_+(u)} \ge 0. \tag{8}$$

Remark 1. The partial differential equations throughout this paper are always meant in their weak form. In case of the state and adjoint equations (1)–(3) and (6), respectively, the weak forms are precisely stated in Section 2.2 below, see the definition of F. However, we prefer to write the equations in their strong form to make them easier understandable.

Solutions to the optimality system (6)–(8), including the state equation, can be found numerically by employing, *e.g.*, semismooth Newton or primal-dual active set methods, see [8, 10, 19] and [2, 9], respectively.

In the sequel, we will often find it convenient to use the abbreviations $y = (c_1, c_2)$ for the vector of state variables, $x = (y, u)$ for state/control pairs, and $\lambda = (\lambda_1, \lambda_2)$ for the vector of adjoint states. In passing, we define the Lagrangian associated to our problem (**P**),

$$\mathcal{L}(x,\lambda) = J(x) + \int_0^T \left\{ \langle \frac{\partial}{\partial t} c_1, \lambda_1 \rangle + d_1 \int_\Omega \nabla c_1 \nabla \lambda_1 \,\mathrm{d}x + \int_\Omega k_1 c_1 c_2 \lambda_1 \,\mathrm{d}x \right\} \mathrm{d}t$$
$$+ \int_0^T \left\{ \langle \frac{\partial}{\partial t} c_2, \lambda_2 \rangle + d_2 \int_\Omega \nabla c_2 \nabla \lambda_2 \,\mathrm{d}x + \int_\Omega k_2 c_1 c_2 \lambda_2 \,\mathrm{d}x - d_2 \int_{\partial\Omega} \alpha\, u\, \lambda_2 \,\mathrm{d}x \right\} \mathrm{d}t$$
$$+ \int_\Omega (c_1(0) - c_{10})\, \lambda_1(0) \,\mathrm{d}x + \int_\Omega (c_2(0) - c_{20})\, \lambda_2(0) \,\mathrm{d}x \quad (9)$$

for any $x = (c_1, c_2, u) \in Y \times U$ and $\lambda = (\lambda_1, \lambda_2) \in Y$. The bracket $\langle u, v \rangle$ denotes the duality between $u \in H^1(\Omega)'$ and $v \in H^1(\Omega)$. The Lagrangian is twice continuously differentiable, and its Hessian with respect to the state and control variables is readily seen to be

$$\mathcal{L}_{xx}(x,\lambda)(\overline{x},\overline{x}) = \beta_1 \|\overline{c}_1(T)\|^2_{L^2(\Omega)} + \beta_2 \|\overline{c}_2(T)\|^2_{L^2(\Omega)} + \gamma \|\overline{u}\|^2_{L^2(0,T)}$$
$$+ \frac{6}{\varepsilon} \max\left\{0, \int_0^T u(t)\,\mathrm{d}t - u_c\right\} \left(\int_0^T \overline{u}(t)\,\mathrm{d}t\right)^2 + 2\int_Q (k_1\lambda_1 + k_2\lambda_2)\, \overline{c}_1 \overline{c}_2 \,\mathrm{d}x\,\mathrm{d}t. \quad (10)$$

The Hessian is a bounded bilinear form, *i.e.*, there exists a constant $C > 0$ such that

$$\mathcal{L}_{xx}(x,\lambda)(\overline{x}_1,\overline{x}_2) \le C\, \|\overline{x}_1\|_{Y\times U} \|\overline{x}_2\|_{Y\times U}$$

holds for all $(\overline{x}_1, \overline{x}_2) \in [Y \times U]^2$.

2.2 Parameter Dependence

As announced in the introduction, we consider problem $(\mathbf{P})$ in dependence on a vector of parameters p and emphasize this by writing $(\mathbf{P}(p))$. It is our goal to investigate the behavior of locally optimal solutions of $(\mathbf{P}(p))$, or solutions of the optimality system (6)–(8) for that matter, as p deviates from its given nominal value p^*. In practice, the parameter vector p can be thought of as problem data which may be subject to perturbation or uncertainty. The nominal value p^* is then simply the expected value of the data. Our main result (Theorem 3) states that under a coercivity condition on the Hessian (10) of the Lagrange function, the solution of the optimality system belonging to $(\mathbf{P}(p))$ depends directionally differentiably on p. The derivatives are called parametric sensitivities since they yield the sensitivities of their underlying quantities with respect to perturbations in the parameter. Our analysis can be used to predict the solution at p near the nominal value p^* using a Taylor expansion. This can be exploited to devise a solution algorithm for $(\mathbf{P}(p))$ with real-time capabilities, provided that the nominal solution to $(\mathbf{P}(p^*))$ along with the sensitivities are computed beforehand ("offline"). In addition, the sensitivities allow a qualitative perturbation analysis of optimal solutions.

In our current problem, we take

$$p = (d_1, d_2, k_1, k_2, \beta_1, \beta_2, \gamma, u_c, \varepsilon, c_{10}, c_{20}, c_{1T}, c_{2T}, u_d) \\ \in \mathbb{R}^9 \times L^2(\Omega)^4 \times L^2(0,T) =: \Pi \tag{11}$$

as the vector of perturbation parameters. Note that p belongs to an infinite-dimensional Hilbert space and that, besides containing physical parameters such as the reaction and diffusion constants k_i and d_i, it comprises non-physical data such as the penalization parameter ε.

In order to carry out our analysis, it is convenient to rewrite the optimality system (6)–(8) plus the state equation as a generalized equation, involving a set-valued operator. We notice that the complementarity condition (8) together with (7) is equivalent to the variational inequality

$$\int_0^T \xi(t)(\overline{u}(t) - u(t))\,\mathrm{d}t \le 0 \quad \forall \overline{u} \in U_{\mathrm{ad}}. \tag{12}$$

This can also be expressed as $\xi \in N(u)$ where

$$N(u) = \{v \in L^2(0,T) : \int_0^T v\,(\overline{u} - u)\,\mathrm{d}t \le 0 \text{ for all } \overline{u} \in U_{\mathrm{ad}}\}$$

if $u \in U_{\mathrm{ad}}$, and $N(u) = \emptyset$ if $u \notin U_{\mathrm{ad}}$. This set-valued operator is known as the dual cone of U_{ad} at u (after identification of $L^2(0,T)$ with its dual). To rewrite the remaining components of the optimality system into operator form, we introduce

$$F : W(0,T) \times L^2(0,T) \times W(0,T) \times Q \to Z$$

with the target space Z given by

$$Z = L^2(0,T; H^1(\Omega)')^2 \times L^2(\Omega)^2 \times L^2(0,T) \times L^2(0,T; H^1(\Omega)')^2 \times L^2(\Omega)^2.$$

The components of F are given next. Wherever it appears, ϕ denotes an arbitrary function in $L^2(0,T; H^1(\Omega))$. For reasons of brevity, we introduce $K = k_1\lambda_1 + k_2\lambda_2$.

$$F_1(y,u,\lambda,p)(\phi) = \int_0^T \left\{ \langle -\frac{\partial}{\partial t}\lambda_1, \phi\rangle + d_1 \int_\Omega \nabla\lambda_1 \cdot \nabla\phi\,\mathrm{d}x + \int_\Omega K c_2\,\phi\,\mathrm{d}x \right\} \mathrm{d}t$$

$$F_2(y,u,\lambda,p)(\phi) = \int_0^T \left\{ \langle -\frac{\partial}{\partial t}\lambda_2, \phi\rangle + d_2 \int_\Omega \nabla\lambda_2 \cdot \nabla\phi\,\mathrm{d}x + \int_\Omega K c_1\,\phi\,\mathrm{d}x \right\} \mathrm{d}t$$

$$F_3(y,u,\lambda,p) = \lambda_1(T) + \beta_1(c_1(T) - c_{1T})$$

$$F_4(y,u,\lambda,p) = \lambda_2(T) + \beta_2(c_2(T) - c_{2T})$$

$$F_5(y,u,\lambda,p) = \gamma(u - u_d) + \frac{3}{\varepsilon}\max\left\{0, \int_0^T u(t)\,\mathrm{d}t - u_c\right\}^2 - \int_{\Gamma_c} \alpha\,\lambda_2\,\mathrm{d}x$$

$$F_6(y,u,\lambda,p)(\phi) = \int_0^T \left\{ \langle \frac{\partial}{\partial t} c_1, \phi \rangle + d_1 \int_\Omega \nabla c_1 \cdot \nabla \phi \, \mathrm{d}x + \int_\Omega k_1 c_1 c_2 \, \phi \, \mathrm{d}x \right\} \mathrm{d}t$$

$$F_7(y,u,\lambda,p)(\phi) = \int_0^T \left\{ \langle \frac{\partial}{\partial t} c_2, \phi \rangle + d_2 \int_\Omega \nabla c_2 \cdot \nabla \phi \, \mathrm{d}x + \int_\Omega k_2 c_1 c_2 \, \phi \, \mathrm{d}x \right\} \mathrm{d}t - \int_\Sigma \alpha \, u \, \phi \, \mathrm{d}x \, \mathrm{d}t$$

$$F_8(y,u,\lambda,p) = c_1(0) - c_{10}$$

$$F_9(y,u,\lambda,p) = c_2(0) - c_{20}.$$

At this point it is not difficult to see that the optimality system (6)–(8), including the state equation (1a)–(1b), (2a)–(2c) and (3a)–(3b), is equivalent to the generalized equation

$$0 \in F(y,u,\lambda,p) + \mathcal{N}(u) \tag{13}$$

where we have set $\mathcal{N}(u) = (0,0,0,0,N(u),0,0,0,0)^\top \subset Z$. In the next section, we will investigate the following linearization around a given solution (y^*,u^*,λ^*) of (13) and for the given parameter p^*. This linearization depends on a new parameter $\delta \in Z$:

$$\delta \in F(y^*,u^*,\lambda^*,p^*) + F'(y^*,u^*,\lambda^*,p^*) \begin{pmatrix} y-y^* \\ u-u^* \\ \lambda-\lambda^* \end{pmatrix} + \mathcal{N}(u). \tag{14}$$

Herein F' denotes the Fréchet derivative of F with respect to (y,u,λ). Note that F is the gradient of the Lagrangian $\mathcal{L}$ and F' is its Hessian whose "upper-left block" was already mentioned in (10).

3 Properties of the Linearized Problem

In order to become more familiar with the linearized generalized equation (14), we write it in its strong form, assuming smooth perturbations $\delta = (\delta_1, \ldots, \delta_5)$. For better readability, the given parameter p^* is still denoted as in (11), without additional * in every component. We obtain from the linearizations of F_1 through F_4:

$$-\frac{\partial}{\partial t}\lambda_1 - d_1\Delta\lambda_1 + K\,c_2^* + K^*c_2 = K^*c_2^* + \delta_1 \quad \text{in } Q, \tag{15a}$$

$$-\frac{\partial}{\partial t}\lambda_2 - d_2\Delta\lambda_2 + K\,c_1^* + K^*c_1 = K^*c_1^* + \delta_2 \quad \text{in } Q, \tag{15b}$$

$$d_1\frac{\partial\lambda_1}{\partial n} = \delta_1|_\Sigma \quad \text{on } \Sigma, \tag{15c}$$

$$d_2\frac{\partial\lambda_2}{\partial n} = \delta_2|_\Sigma \quad \text{on } \Sigma, \tag{15d}$$

$$\lambda_1(T) = -\beta_1(c_1(T) - c_{1T}) + \delta_3 \quad \text{in } \Omega, \tag{15e}$$

$$\lambda_2(T) = -\beta_2(c_2(T) - c_{2T}) + \delta_4 \quad \text{in } \Omega, \tag{15f}$$

where we have abbreviated $K = k_1\lambda_1 + k_2\lambda_2$ and $K^* = k_1\lambda_1^* + k_2\lambda_2^*$. From the components F_6 through F_9 we obtain a linearized state equation:

$$\frac{\partial}{\partial t}c_1 - d_1\Delta c_1 + k_1c_1c_2^* + k_1c_1^*c_2 = k_1c_1^*c_2^* + \delta_6 \quad \text{in } Q, \tag{16a}$$

$$\frac{\partial}{\partial t}c_2 - d_2\Delta c_2 + k_2c_1c_2^* + k_2c_1^*c_2 = k_2c_1^*c_2^* + \delta_7 \quad \text{in } Q, \tag{16b}$$

$$d_1\frac{\partial c_1}{\partial n} = \delta_6|_\Sigma \quad \text{on } \Sigma, \tag{16c}$$

$$d_2\frac{\partial c_2}{\partial n} = \alpha\,u + \delta_7|_\Sigma \quad \text{on } \Sigma, \tag{16d}$$

$$c_1(0) = c_{10} + \delta_8 \quad \text{in } \Omega, \tag{16e}$$

$$c_2(0) = c_{20} + \delta_9 \quad \text{in } \Omega. \tag{16f}$$

Finally, the component F_5 becomes the variational inequality

$$\int_0^T \xi(t)(\overline{u}(t) - u(t))\,\mathrm{d}t \le 0 \quad \forall\overline{u} \in U_{\mathrm{ad}} \tag{17}$$

where in analogy to the original problem, $\xi \in L^2(0,T)$ is defined through

$$\gamma(u - u_d) + \frac{3}{\varepsilon}\max\Big\{0, \int_0^T u^*(t)\,\mathrm{d}t - u_c\Big\}^2 - \int_{\Gamma_c}\alpha\,\lambda_2\,\mathrm{d}x - \delta_5$$
$$+ \frac{6}{\varepsilon}\max\Big\{0, \int_0^T u^*(t)\,\mathrm{d}t - u_c\Big\}\int_0^T (u(t) - u^*(t))\,\mathrm{d}t + \xi(t) = 0. \tag{18}$$

In turn, the system (15)–(18) is easily recognized as the optimality system for an auxiliary linear quadratic optimization problem, which we term ($\mathbf{AQP}(\delta)$):

$$\begin{aligned}\text{Minimize } \frac{1}{2}\mathcal{L}_{xx}(x^*,\lambda^*)(x,x) - \beta_1 \int_\Omega c_{1T}\, c_1(T)\,\mathrm{d}x - \beta_2 \int_\Omega c_{2T}\, c_2(T)\,\mathrm{d}x \\ + \frac{3}{\varepsilon}\max\Big\{0, \int_0^T u^*(t)\,\mathrm{d}t - u_c\Big\}^2 \int_0^T u(t)\,\mathrm{d}t \\ - \frac{6}{\varepsilon}\max\Big\{0, \int_0^T u^*(t)\,\mathrm{d}t - u_c\Big\}\left(\int_0^T u^*(t)\,\mathrm{d}t\right)\left(\int_0^T u(t)\,\mathrm{d}t\right) \\ - \gamma\int_0^T u_d\, u\,\mathrm{d}t - \int_Q (k_1\lambda_1^* + k_2\lambda_2^*)(c_1^* c_2 + c_1 c_2^*)\,\mathrm{d}x\,\mathrm{d}t \\ - \langle\delta_1, c_1\rangle - \langle\delta_2, c_2\rangle - \int_\Omega \delta_3 c_1(T) - \int_\Omega \delta_4 c_2(T) - \int_0^T \delta_5 u\,\mathrm{d}t \quad (19)\end{aligned}$$

subject to the linearized state equation (16) above and $u \in U_{\text{ad}}$. The bracket $\langle\delta_1, c_1\rangle$ here denotes the duality between $L^2(0,T;H^1(\Omega))$ and its dual $L^2(0,T;H^1(\Omega)')$. In order for (**AQP**(δ)) to have a strictly convex objective and thus to have a unique solution, we require the following assumption:

Assumption 2 (Coercivity Condition)
We assume that there exists $\rho > 0$ such that

$$\mathcal{L}_{xx}(x^*,\lambda^*)(x,x) \geq \rho\,\|x\|^2_{Y\times U}$$

holds for all $x = (c_1, c_2, u) \in Y \times U$ which satisfy the linearized state equation (16) in weak form, with all right hand sides except the term $\alpha\, u$ replaced by zero.

Sufficient conditions for Assumption 2 to hold are given in [9, Theorem 3.15]. We now prove our first result for the auxiliary problem (**AQP**(δ)):

Proposition 1 (Lipschitz Stability for the Linearized Problem).
*Under Assumption 1, holding for the parameter p^*, and Assumption 2, (**AQP**(δ)) has a unique solution which depends Lipschitz continuously on the parameter $\delta \in Z$. That is, there exists $L > 0$ such that for all $\hat\delta, \check\delta \in Z$ with corresponding solutions $(\hat x, \hat\lambda)$ and $(\check x, \check\lambda)$,*

$$\begin{aligned}\|\hat c_1 - \check c_1\|_{W(0,T)} + \|\hat c_2 - \check c_2\|_{W(0,T)} + \|\hat u - \check u\|_{L^2(0,T)} \\ + \|\hat\lambda_1 - \check\lambda_1\|_{W(0,T)} + \|\hat\lambda_2 - \check\lambda_2\|_{W(0,T)} \leq L\,\|\hat\delta - \check\delta\|_Z\end{aligned}$$

hold.

Proof. The proof follows the technique of [18] and is therefore kept relatively short here. Throughout, we denote by capital letters the differences we wish to estimate, *i.e.*, $C_1 = \hat c_1 - \check c_1$, etc. To improve readability, we omit the differentials $\mathrm{d}x$ and $\mathrm{d}t$ in integrals whenever possible. We begin by testing the weak form of the adjoint equation (15) by C_1 and C_2, and testing the weak form of the state equation (16) by Λ_1 and Λ_2, using integration by parts with

respect to time and plugging in the initial and terminal conditions from (15) and (16). One obtains

$$\begin{aligned}
&\beta_1\|C_1(T)\|^2 + \beta_2\|C_2\|^2 + 2\int_Q K^* C_1\, C_2 + \int_\Sigma \alpha\, U\Lambda \\
&= -\langle C_1, \Delta_1\rangle - \langle C_2, \Delta_2\rangle + \int_\Omega C_1(T)\Delta_3 + \int_\Omega C_2(T)\Delta_4 - \langle \Lambda_1, \Delta_6\rangle - \langle \Lambda_2, \Delta_7\rangle \\
&\qquad - \int_\Omega \Lambda_1(0)\Delta_8 - \int_\Omega \Lambda_2(0)\Delta_9. \qquad (20)
\end{aligned}$$

From the variational inequality (17), using $\overline{u} = \hat{u}$ or $\overline{u} = \check{u}$ as test functions, we get

$$-\int_\Sigma \alpha\, U\Lambda_2 \le -\gamma\,\|U\|^2 + \int_0^T U\Delta_5 - \frac{6}{\varepsilon}\max\left\{0, \int_0^T u^*(t)\,\mathrm{d}t - u_c\right\}\left(\int_0^T U\,\mathrm{d}t\right)^2. \qquad (21)$$

Unless otherwise stated, all norms are the natural norms for the respective terms. Adding the inequality (21) to (20) above and collecting terms yields

$$\begin{aligned}
&\mathcal{L}_{xx}(x^*, \lambda^*)((C_1, C_2, U), (C_1, C_2, U)) \\
&\quad \le -\langle C_1, \Delta_1\rangle - \langle C_2, \Delta_2\rangle + \int_\Omega C_1(T)\Delta_3 + \int_\Omega C_2(T)\Delta_4 + \int_0^T U\Delta_5 \\
&\qquad - \langle \Lambda_1, \Delta_6\rangle - \langle \Lambda_2, \Delta_7\rangle - \int_\Omega \Lambda_1(0)\Delta_8 - \int_\Omega \Lambda_2(0)\Delta_9 \\
&\quad \le \kappa\,(1 + c^2)\big(\|C_1\|^2 + \|C_2\|^2 + \|\Lambda_1\|^2 + \|\Lambda_2\|^2\big) + \kappa\,\|U\|^2 + \frac{1}{4\kappa}\sum_{i=1}^{9}\|\Delta_i\|^2
\end{aligned} \qquad (22)$$

where the last inequality has been obtained using Hölder's inequality, the embedding $W(0,T) \hookrightarrow C([0,T]; L^2(\Omega))$ and Young's inequality in the form $ab \le \kappa a^2 + b^2/(4\kappa)$. The number $\kappa > 0$ denotes a sufficiently small constant which will be determined later at our convenience. Here and throughout, generic constants are denoted by c. They may take different values in different locations.

In order to make use of the Coercivity Assumption 2, we decompose $C_i = z_i + w_i$, $i = 1, 2$ and consider their respective equations, see (16). The z components account for the control influence while the w components arise from the perturbation differences $\Delta_1, \ldots, \Delta_4$. We have on Q, Σ and Ω, respectively,

$$\frac{\partial}{\partial t}z_1 - d_1\Delta z_1 + k_1 z_1 c_2^* + k_1 c_1^* z_2 = 0, \quad \frac{\partial}{\partial t}w_1 - d_1\Delta w_1 + k_1 w_1 c_2^* + k_1 c_1^* w_2 = \Delta_6$$
$$\frac{\partial}{\partial t}z_2 - d_2\Delta z_2 + k_2 z_1 c_2^* + k_2 c_1^* z_2 = 0, \quad \frac{\partial}{\partial t}w_2 - d_2\Delta w_2 + k_2 w_1 c_2^* + k_2 c_1^* w_2 = \Delta_7$$
$$d_1\frac{\partial z_1}{\partial n} = 0, \qquad d_1\frac{\partial w_1}{\partial n} = \Delta_6|_\Sigma$$
$$d_2\frac{\partial z_2}{\partial n} = \alpha U, \qquad d_2\frac{\partial w_2}{\partial n} = \Delta_7|_\Sigma$$
$$z_1(0) = 0, \qquad w_1(0) = \Delta_8$$
$$z_2(0) = 0, \qquad w_2(0) = \Delta_9.$$

Note that for (z_1, z_2, U), the Coercivity Assumption 2 applies and that standard a priori estimates yield $\|z_1\| + \|z_2\| \le c\|U\|$ and $\|w_1\| + \|w_2\| \le c(\|\Delta_6\| + \|\Delta_7\| + \|\Delta_8\| + \|\Delta_9\|)$. Using the generic estimates $\|z_i\|^2 \ge \|C_i\|^2 - 2\|C_i\|\|w_i\| + \|w_i\|^2$ and $\|z_i\| \le \|C_i\| + \|w_i\|$, the embedding $W(0,T) \hookrightarrow C([0,T]; L^2(\Omega))$ and the coercivity assumption, we obtain

$$\begin{aligned}
\mathcal{L}_{xx}(x^*,\lambda^*)((C_1,C_2,U),(C_1,C_2,U)) &= \mathcal{L}_{xx}(x^*,\lambda^*)((z_1,z_2,U),(z_1,z_2,U)) \\
&\quad + \beta_1\int_\Omega z_1(T)w_1(T) + \beta_2\int_\Omega z_2(T)w_2(T) + \frac{\beta_1}{2}\|w_1(T)\|^2 + \frac{\beta}{2}\|w_2(T)\|^2 \\
&\quad + \int_Q K^*(w_1 z_2 + z_1 w_2 + w_1 w_2) \\
&\ge \rho\left(\|C_1\|^2 + \|C_2\|^2 + \|U\|^2\right) - 2\rho\left(\|C_1\|\|w_1\| + \|C_2\|\|w_2\|\right) \\
&\quad - \beta_1 c\|w_1\|\left(\|C_1\| + \|w_1\|\right) - \beta_2 c\|w_2\|\left(\|C_2\| + \|w_2\|\right) \\
&\quad - c\|K^*\|_{L^2(Q)}\left(\|w_1\|\|C_2\| + \|C_1\|\|w_2\| + 3\|w_1\|\|w_2\|\right). \qquad (23)
\end{aligned}$$

For the last term, we have employed Hölder's inequality and the embedding $W(0,T) \hookrightarrow L^4(Q)$, see [4, p. 7]. Combining the inequalities (22) and (23) yields

$$\begin{aligned}
\rho\left(\|C_1\|^2 + \|C_2\|^2 + \|U\|^2\right) &\le 2\rho\left(\|C_1\|\|w_1\| + \|C_2\|\|w_2\|\right) \\
&\quad + \beta_1 c\|w_1\|\left(\|C_1\| + \|w_1\|\right) + \beta_2 c\|w_2\|\left(\|C_2\| + \|w_2\|\right) \\
&\quad + c\|K^*\|_{L^2(\Omega)}\left(\|w_1\|\|C_2\| + \|C_1\|\|w_2\| + 3\|w_1\|\|w_2\|\right) \\
&\quad + \frac{1}{8\kappa}\sum_{i=1}^{9}\|\Delta_i\|^2 + \frac{\kappa}{2}(1+c^2)\left(\|C_1\|^2 + \|C_2\|^2 + \|\Lambda_1\|^2 + \|\Lambda_2\|^2\right) + \frac{\kappa}{2}\|U\|^2 \qquad (24)
\end{aligned}$$

and the last two terms can be absorbed in the left hand side when choosing $\kappa > 0$ sufficiently small and observing that Λ_1 and Λ_2 depend continuously on

the data C_1 and C_2. By the a priori estimate stated above, w_i, $i = 1, 2$, can be estimated against the data $\Delta_7, \ldots, \Delta_9$. Using again Young's inequality on the terms $\|C_i\|\|w_j\|$ and absorbing the quantities of type $\kappa\|C_i\|^2$ into the left hand side, we obtain the Lipschitz dependence of (C_1, C_2, U) on $\Delta_1, \ldots, \Delta_9$. Invoking once more the continuous dependence of Λ_i on (C_1, C_2), Lipschitz stability is seen to hold also for the adjoint variable. □

If (x^*, λ^*) is a solution to the optimality system (6)–(8) and state equation, then the previous theorem implies that the generalized equation (13) is strongly regular at this solution, compare [15]. Before showing that the Coercivity Assumption 2 implies also directional differentiability of the solution of (**AQP**(δ)) in dependence on δ, we introduce the *strongly active subsets* for the solution (y^*, u^*, λ^*) with multiplier ξ^* given by (7),

$$A_-^0(u^*) = \{t \in [0,T] \ : \ \xi^*(t) < 0\}$$
$$A_+^0(u^*) = \{t \in [0,T] \ : \ \xi^*(t) > 0\}$$

Note that necessarily $u^* = u_a$ on $A_-^0(u^*)$ and $u^* = u_b$ on $A_+^0(u^*)$ hold in view of the variational inequality (12). Based on the notion of strongly active sets, we define $\widehat{U}_{\mathrm{ad}}$, the set of admissible control variations:

$$u \in \widehat{U}_{\mathrm{ad}} \Leftrightarrow u \in L^2(0,T) \text{ and } \begin{cases} u = 0 \text{ on } A_-^0(u^*) \cup A_+^0(u^*) \\ u \geq 0 \text{ on } A_-(u^*) \\ u \leq 0 \text{ on } A_+(u^*). \end{cases}$$

This definition reflects the fact that if the solution u^* associated to the parameter value p^* is equal to the lower bound u_a at some point $t \in [0,T]$, we can approach it only from above (and vice versa for the upper bound). In addition, if the control constraint is strongly active at some point, *i.e.*, if it has a nonzero multiplier ξ^* there, the variation is zero.

Proposition 2 (Differentiability for the Linearized Problem). *Under Assumptions 1 and 2, the unique solution to (***AQP**(δ)*) depends directionally differentiably on the parameter $\delta \in Z$. The directional derivative in the direction of $\hat{\delta} \in Z$ is given by the solution of the auxiliary linear quadratic problem (***DQP**($\hat{\delta}$)*),*

$$\textit{Minimize } \frac{1}{2}\mathcal{L}_{xx}(x^*, \lambda^*)(x,x) - \langle \hat{\delta}_1, c_1 \rangle - \langle \hat{\delta}_2, c_2 \rangle - \int_\Omega \hat{\delta}_3 c_1(T) - \int_\Omega \hat{\delta}_4 c_2(T) - \int_0^T \hat{\delta}_5 u \, \mathrm{d}t$$

subject to $u \in \widehat{U}_{\mathrm{ad}}$ and the linearized state equation

$$\frac{\partial}{\partial t}c_1 - d_1\Delta c_1 + k_1 c_1 c_2^* + k_1 c_1^* c_2 = \hat{\delta}_6 \qquad \text{in } Q \tag{25a}$$

$$\frac{\partial}{\partial t}c_2 - d_2\Delta c_2 + k_2 c_1 c_2^* + k_2 c_1^* c_2 = \hat{\delta}_7 \qquad \text{in } Q \tag{25b}$$

$$d_1\frac{\partial c_1}{\partial n} = \hat{\delta}_6|_\Sigma \qquad \text{on } \Sigma \tag{25c}$$

$$d_2\frac{\partial c_2}{\partial n} = \alpha\, u + \hat{\delta}_7|_\Sigma \qquad \text{on } \Sigma \tag{25d}$$

$$c_1(0) = \hat{\delta}_8 \qquad \text{in } \Omega \tag{25e}$$

$$c_2(0) = \hat{\delta}_9 \qquad \text{in } \Omega. \tag{25f}$$

Proof. Let $\hat{\delta} \in Z$ be any given direction of perturbation and let $\{\tau_n\}$ be a sequence of real numbers such that $\tau_n \searrow 0$. We set $\delta_n = \tau_n \hat{\delta}$ and denote the solution of (**AQP**(δ_n)) by $(c_1^n, c_2^n, u^n, \lambda_1^n, \lambda_2^n)$. Note that $(c_1^*, c_2^*, u^*, \lambda_1^*, \lambda_2^*)$ is the solution of (**AQP**(0)). Then, by virtue of Proposition 1, we have

$$\left\|\frac{c_1^n - c_1^*}{\tau_n}\right\| + \left\|\frac{c_2^n - c_2^*}{\tau_n}\right\| + \left\|\frac{u_n - u^*}{\tau_n}\right\| + \left\|\frac{\lambda_1^n - \lambda_1^*}{\tau_n}\right\| + \left\|\frac{\lambda_2^n - \lambda_2^*}{\tau_n}\right\| \leq L\,\|\hat{\delta}\| \tag{26}$$

in the norms of $W(0,T)$, $L^2(0,T)$, and Z, respectively, and with some Lipschitz constant $L > 0$. We can thus extract weakly convergent subsequences (still denoted by index n) and use the compact embedding of $W(0,T)$ into $L^2(Q)$ to obtain

$$\frac{u^n - u^*}{\tau_n} \rightharpoonup \tilde{u} \quad \text{in } L^2(0,T) \tag{27}$$

$$\frac{c_1^n - c_1^*}{\tau_n} \rightharpoonup \hat{c}_1 \quad \text{in } W(0,T) \quad \text{and} \quad \to \hat{c}_1 \quad \text{in } L^2(Q) \tag{28}$$

and similarly for the remaining components. Taking yet another subsequence, all components except the control are seen also to converge pointwise almost everywhere in Q. From here, we only sketch the remainder of the proof since it closely parallels the ones given in [6, 12]. In addition to the arguments given there, our analysis relies on the strong convergence (and thus pointwise convergence almost everywhere on $[0,T]$ of a subsequence) of

$$\int_{\Gamma_c} \alpha\,\frac{\lambda_2^n - \lambda_2^*}{\tau^n} \to \int_{\Gamma_c} \alpha\,\hat{\lambda}_2 \quad \text{in } L^2(0,T) \tag{29}$$

which follows from the compact embedding of $W(0,T)$ into $L^2(0,T;H^s(\Omega))$ for $1/2 < s < 1$ (see (5)) and the continuity of the trace operator $H^s(\Omega) \to L^2(\Gamma_c)$. One expresses u^n as the pointwise projection of $u^n + \xi^n/\gamma$ onto the admissible set U_{ad} with ξ^n given by (18) evaluated at (u^n, λ_2^n). Using (27) and (29), one shows that $(u^n - u^*)/\tau^n$ possesses a pointwise convergent subsequence (still denoted by index n). Distinguishing cases, one finds the pointwise

limit $\hat{u}$ of $(u^n - u^*)/\tau^n$ to be the pointwise projection of $\lim_{n\to\infty}(u^n + \xi^n/\gamma)$ onto the new admissible set $\widehat{U}_{\mathrm{ad}}$. Using a suitable upper bound in Lebesgue's Dominated Convergence Theorem, one shows that $\hat{u}$ is also the limit in the sense of $L^2(0,T)$ and thus $\hat{u} = \tilde{u}$ must hold. It remains to show that the limit $(\hat{c}_1, \hat{c}_2, \hat{u}, \hat{\lambda}_1, \hat{\lambda}_2)$ satisfy the first order optimality system for $(\mathbf{DQP}(\hat{\delta}))$ (which is routine) and that the limits actually hold in their strong senses in $W(0,T)$ (which follows from standard a priori estimates). Since we could have started with a subsequence of τ^n in the first place and since the limit $(\hat{c}_1, \hat{c}_2, \hat{u}, \hat{\lambda}_1, \hat{\lambda}_2)$ must always be the same in view of the Coercivity Assumption 2, the convergence extends to the whole sequence. □

4 Properties of the Nonlinear Problem

In the current section, we shall prove that the solutions to the original nonlinear generalized equation (13) depend on p in the same way as the solutions to the linearized generalized equation (14) depend on δ. To this end, we invoke an implicit function theorem for generalized equations. Throughout this section, let again p^* be a given nominal (or unperturbed or expected) value of the parameter vector

$$\begin{aligned} p &= (d_1, d_2, k_1, k_2, \beta_1, \beta_2, \gamma, u_c, \varepsilon, c_{10}, c_{20}, c_{1T}, c_{2T}, u_d) \\ &\in \mathbb{R}^9 \times L^2(\Omega)^4 \times L^2(0,T) =: \Pi \end{aligned}$$

satisfying Assumption 1. Moreover, let $(x^*, \lambda^*) = (c_1^*, c_2^*, u^*, \lambda_1^*, \lambda_2^*)$ be a solution of the first order necessary conditions (6)–(8) plus the state equation, or, in other words, of the generalized equation (13).

Theorem 3 (Lipschitz Continuity and Directional Differentiability). *Under Assumptions 1 and 2, there exists a neighborhood $\mathcal{B}(p^*) \subset \Pi$ of p^* and a neighborhood $\mathcal{B}(y^*, u^*, \lambda^*) \subset Y \times U \times Y$ and a Lipschitz continuous function*

$$\mathcal{B}(p^*) \ni p \mapsto (y_p, u_p, \lambda_p) \in \mathcal{B}(y^*, u^*, \lambda^*)$$

such that (y_p, u_p, λ_p) solves the optimality system (6)–(8) plus the state equation for parameter p and such that it is the only critical point in $\mathcal{B}(y^*, u^*, \lambda^*)$. Moreover, the map $p \mapsto (y_p, u_p, \lambda_p)$ is directionally differentiable, and its derivative in the direction $\hat{p} \in \Pi$ is given by the unique solution of $(\mathbf{DQP}(\hat{\delta}))$, in the direction of $\hat{\delta} = -F_p(y^*, u^*, \lambda^*, p^*)\,\hat{p}$.

Proof. The proof is based on the implicit function theorem for generalized equations from [5, 15]. It relies on the strong regularity property, which was shown in Proposition 1. It remains to verify that F is Lipschitz in p near p^*, uniformly in a neighborhood of (y^*, u^*, λ^*), and that F is differentiable with respect to p, which is straightforward. The formula for its derivative is given in the remark below. □

Remark 2. In order to compute the parametric sensitivities of the nominal solution $(c_1^*, c_2^*, u^*, \lambda_1^*, \lambda_2^*)$ for $(\mathbf{P}(p^*))$ in a perturbation direction $\hat{p}$, we need to solve the linear-quadratic problem $(\mathbf{DQP}(\hat{\delta}))$ with

$$\begin{aligned}\hat{\delta} &= -F_p(y^*, u^*, \lambda^*, p^*)\,\hat{p} \\ &= -\Big(\hat{d}_1 \int_Q \nabla\lambda_1^* \nabla\cdot + (\hat{k}_1\lambda_1^* + \hat{k}_2\lambda_2^*)c_2^*\cdot, \quad \hat{d}_2 \int_Q \nabla\lambda_2^* \nabla\cdot + (\hat{k}_1\lambda_1^* + \hat{k}_2\lambda_2^*)c_1^*\cdot, \\ &\qquad \hat{\beta}_1(c_1^*(T) - c_{1T}^*) - \beta_1^*\hat{c}_{1T}, \qquad \hat{\beta}_2(c_2^*(T) - c_{2T}^*) - \beta_2^*\hat{c}_{2T}, \\ &\qquad \hat{\gamma}(u^* - u_d^*) - \gamma^*\hat{u}_d - \frac{3\hat{\varepsilon}}{(\varepsilon^*)^2} I^2 - \frac{6}{\varepsilon^*} I\hat{u}_c, \\ &\qquad \hat{d}_1 \int_Q \nabla c_1^* \cdot \nabla\cdot + \int_Q \hat{k}_1 c_1^* c_2^*\cdot, \qquad \hat{d}_2 \int_Q \nabla c_2^* \cdot \nabla\cdot + \int_Q \hat{k}_2 c_1^* c_2^*\cdot, \\ &\qquad -\hat{c}_{10}, \qquad -\hat{c}_{20} \Big)^\top,\end{aligned}$$

where I means $\max\left\{0, \int_0^T u^*(t)\,\mathrm{d}t - u_c^*\right\}$. We close this section by remarking that the parametric sensitivities allow to compute a second-order expansion of the value of the objective, see [6, 12] for details. In addition, the Coercivity Assumption 2 implies that second order sufficient conditions hold at the nominal and also at the perturbed solutions, so that points satisfying the first order necessary conditions are indeed strict local optimizers.

5 Numerical Results

In this section, we present some numerical results and show evidence that the parametric sensitivities yield valuable information which is useful in making qualitative and quantitative estimates of the solution under perturbations. In our example, the three-dimensional geometry of the problem is given by the annular cylinder between the planes $z = 0$ and $z = 0.5$ with inner radius 0.4 and outer radius 1.0 whose rotational axis is the z-axis (Figure 1). The control boundary Γ_c is the upper annulus, and we use the control shape function

$$\alpha(t, x) = \exp\left(-5\left[(x_1 - 0.7\cos(2\pi t))^2 + (x_2 - 0.7\sin(2\pi t))^2\right]\right).$$

which corresponds to a nozzle circling for $t \in [0, 1]$ once around in counterclockwise direction at a radius of 0.7. For fixed t, α is a function which decays exponentially with the square of the distance from the current location of the nozzle. The problem was discretized using the finite element method on a mesh consisting of 1797 points and 7519 tetrahedra. The 'triangulation' of the domain Ω by tetrahedra is also shown in Figure 1. In the time direction, the interval $[0, T]$ was uniformly divided into 100 parts. By controlling the second substance B, we wish to steer the concentration of the first substance A to zero at terminal time $T = 1$, *i.e.*, we choose

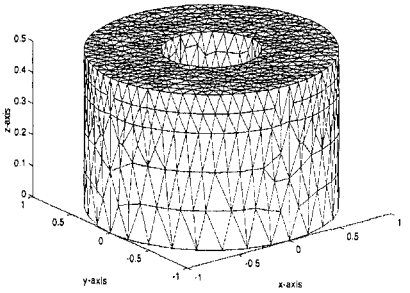


Fig. 1: Domain $\Omega \subset \mathbb{R}^3$ and its triangulation with tetrahedra

$$\beta_1^* = 1 \qquad \beta_2^* = 0 \qquad c_{1T}^* \equiv 0$$

The control cost parameter is $\gamma^* = 10^{-2}$ and the control bounds are chosen as

$$u_a \equiv 1 \qquad u_b \equiv 5.$$

The chemical reaction is governed by equations (1)–(3) with parameters

$$d_1^* = 0.15 \qquad d_2^* = 0.20 \qquad k_1^* = 1.0 \qquad k_2^* = 1.0.$$

As initial concentrations, we use

$$c_{10}^* \equiv 1.0 \qquad c_{20}^* \equiv 0.0.$$

The discrete optimal solution without the contribution from the penalized integral constraint J_2 (corresponding to $\varepsilon = \infty$) yields

$$\int_0^T u^*(t)\,\mathrm{d}t = 4.2401, \qquad J_1(c_1^*, c_2^*, u^*) = 0.2413.$$

In order for this constraint to become relevant, we choose $u_c^* = 3.5$ and enforce it using the penalization parameter $\varepsilon^* = 1$. Details on the numerical implementation are given in [8, 9]. For the discretization described above, we obtain a problem size of approximately 726 000 variables, including the adjoint states, which takes a couple of minutes to solve on a standard desktop PC.

In Figures 3–4 (left columns) and Figure 2 (left), we show the individual components of the optimal solution. We note that the optimal control lies on the upper bound in the first part of the time interval, then in the interior of the admissible interval $[1, 5]$ and finally on the lower bound. From Figure 3 (left) we infer that as time advances, substance A decays and approaches the desired value of zero to the extent permitted by the control cost parameter γ and the control bounds. Figure 4 (left) nicely shows the influence of the revolving

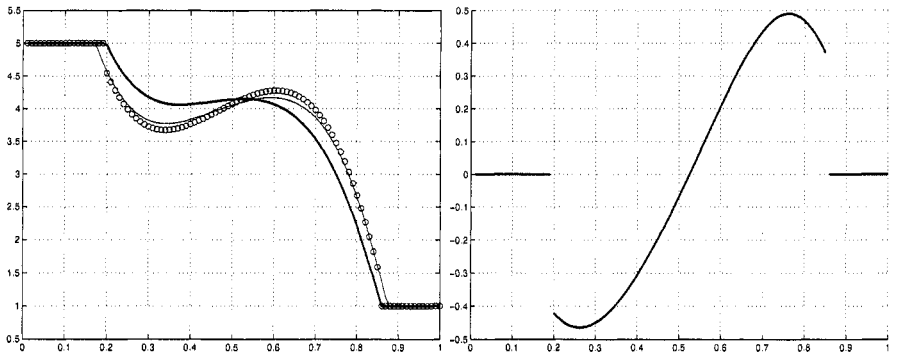


Fig. 2: Left: Optimal control u^* (thick solid), true perturbed control u_p (thin solid) and predicted control (circles). Right: Parametric sensitivity du_{p^*}/dp in the direction of $p - p^*$.

control nozzle on the upper surface of the annular cylinder, adding amounts of substance B over time which then diffuse towards the interior of the reaction vessel and react with substance A.

In order to illustrate the sensitivity calculus, we perturb the reaction constants k_1^* and k_2^* by 50%, taking

$$k_1 = 1.5 \qquad k_2 = 1.5$$

as their new values. With the reaction now proceeding faster, one presumes that the desired goal of consuming substance A within the given time interval will be achieved to a higher degree, which will in fact be confirmed below from sensitivity information. Figure 2 (left) shows, next to the nominal control, the solution obtained by a first order Taylor approximation using the sensitivity of the control variable, *i.e.*,

$$u_p \approx u_{p^*} + \frac{d}{dp} u_{p^*}(p - p^*).$$

To allow a comparison, the true perturbed solution is also depicted, which of course required the repeated solution of the nonlinear optimal control problem $(\mathbf{P}(p))$. It is remarkable how well the perturbed solution can be predicted in face of a 50% perturbation using the sensitivity information, without recomputing the solution to the nonlinear problem. We observe that the perturbed control is lower than the nominal one in the first part of the time interval, later to become higher. This behavior can not easily be predicted without any sensitivity information at hand. Besides, a qualitative analysis of the state sensitivities reveals more interesting information. We have argued above that with the reaction proceeding faster, the control goal can more easily be reached. This can be inferred from Figure 3 (right column), showing that the

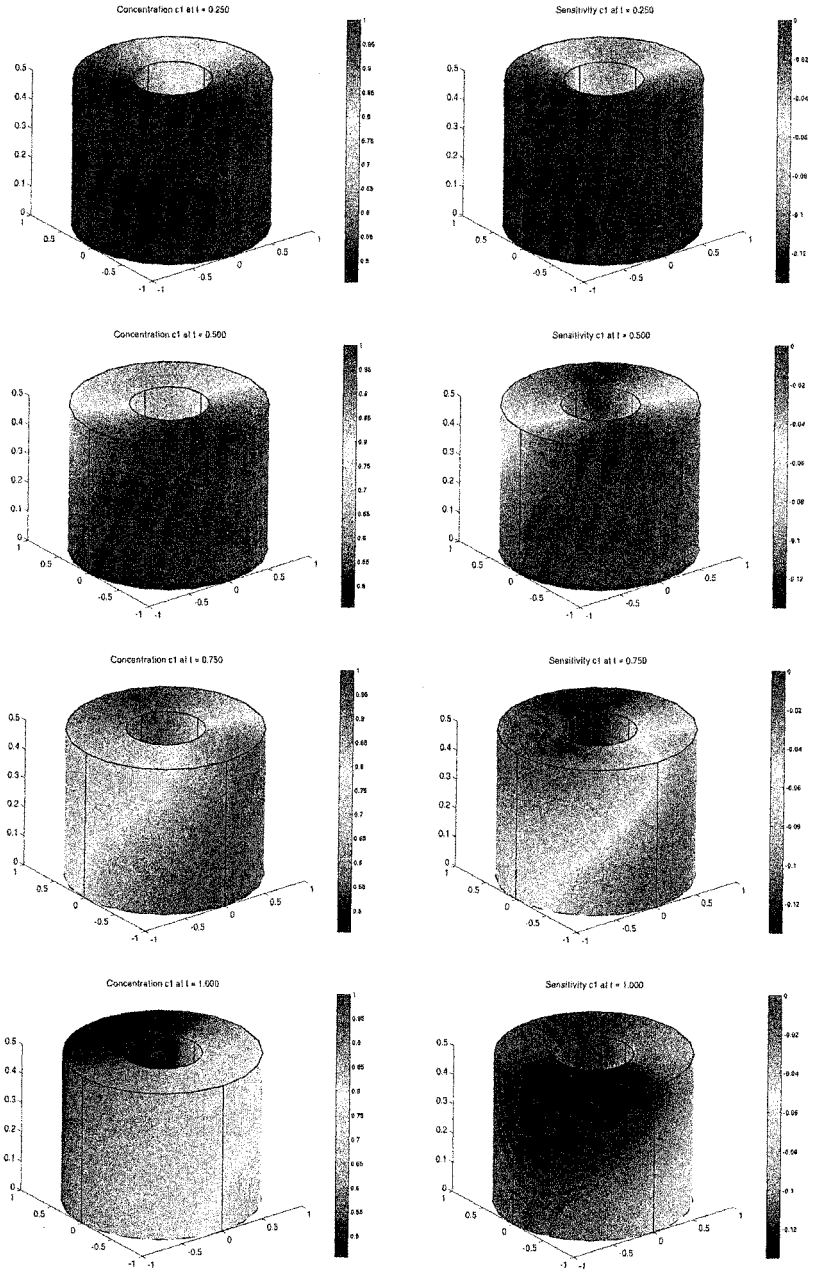


Fig. 3: Concentrations of substance A (left) and its sensitivity (right) at times $t = 0.25$, $t = 0.50$, $t = 0.75$, and $t = 1.00$.

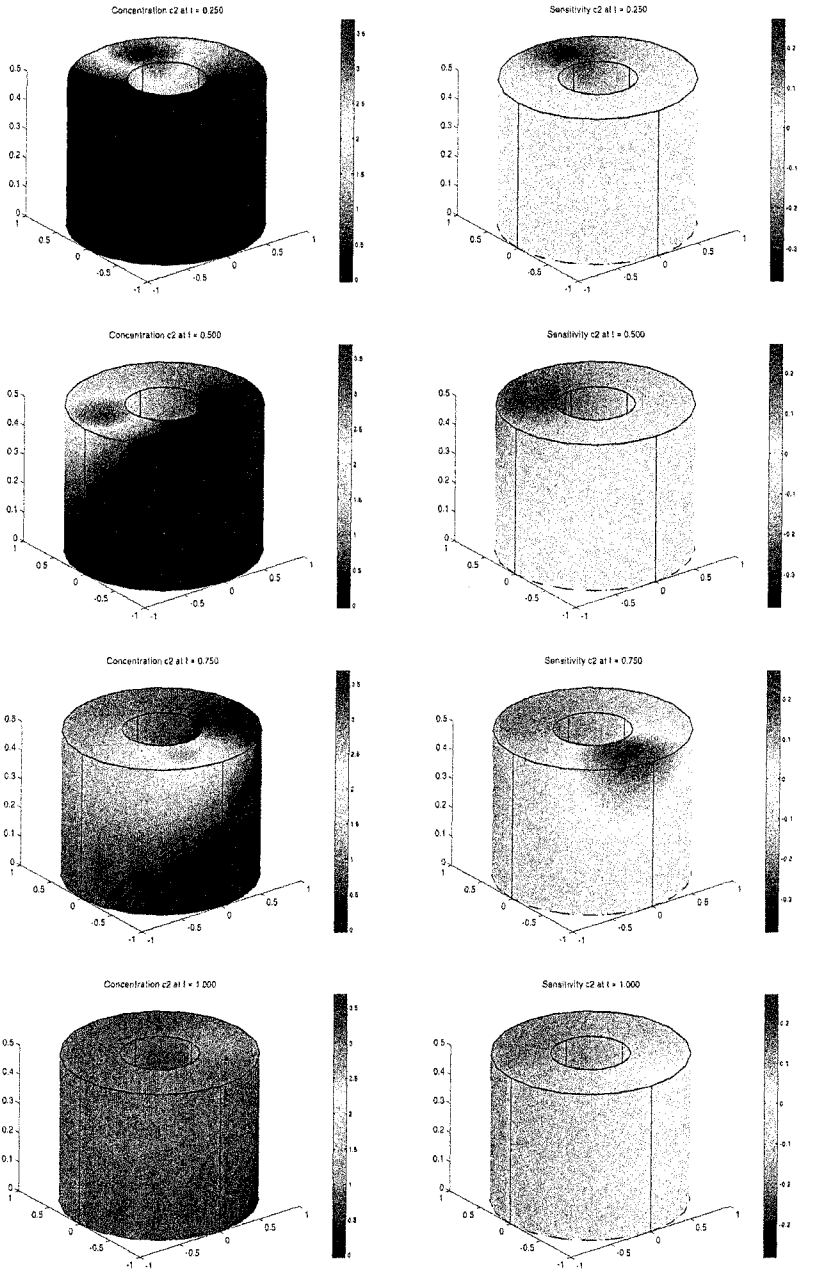


Fig. 4: Concentrations of substance B (left) and its sensitivity (right) at times $t = 0.25$, $t = 0.50$, $t = 0.75$, and $t = 1.00$.

sensitivity derivatives of the first substance are negative throughout, *i.e.*, the perturbed solution comes closer in a pointwise sense to the desired zero terminal state (to first order). The sensitivities for the second state component (see Figure 4, right column) nicely reflect the expected behavior inferred from the control sensitivities, see Figure 2 (right). As the perturbed control is initially lower than the unperturbed one after leaving the upper bound, the sensitivity of the second substance is below zero there. Later, it becomes positive, as does the sensitivity for the control variable.

References

1. R. Adams. *Sobolev Spaces*. Academic Press, New York, 1975.
2. M. Bergounioux, M. Haddou, M. Hintermüller, and K. Kunisch. A comparison of a Moreau-Yosida-based active set strategy and interior point methods for constrained optimal control problems. *SIAM Journal on Optimization*, 11(2):495–521, 2000.
3. C. Büskens and R. Griesse. Computational parametric sensitivity analysis of perturbed PDE optimal control problems with state and control constraints. *Journal of Optimization Theory and Applications*, to appear. http://www.ricam.oeaw.ac.at/people/page/griesse/publications.html.
4. E. DiBenedetto. *Degenerate Parabolic Equations*. Springer, Berlin, 1993.
5. A. Dontchev. Implicit function theorems for generalized equations. *Math. Program.*, 70:91–106, 1995.
6. R. Griesse. Parametric sensitivity analysis in optimal control of a reaction-diffusion system—part I: Solution differentiability. *Numerical Functional Analysis and Optimization*, 25(1–2):93–117, 2004.
7. R. Griesse. Parametric sensitivity analysis in optimal control of a reaction-diffusion system—part II: Practical methods and examples. *Optimization Methods and Software*, 19(2):217–242, 2004.
8. R. Griesse and S. Volkwein. A semi-smooth Newton method for optimal boundary control of a nonlinear reaction-diffusion system. In *Proceedings of the Sixteenth International Symposium on Mathematical Theory of Networks and Systems (MTNS), Leuven, Belgium*, 2004.
9. R. Griesse and S. Volkwein. A primal-dual active set strategy for optimal boundary control of a nonlinear reaction-diffusion system. *Accepted for publication in: SIAM Journal on Control and Optimization*, 2005. Preprint available from http://www.ricam.oeaw.ac.at/people/page/griesse/publications/RD3D.pdf.
10. M. Hintermüller, K. Ito, and K. Kunisch. The primal-dual active set strategy as a semismooth Newton method. *SIAM Journal on Optimization*, 13(3):865–888, 2002.
11. K. Malanowski. Sensitivity analysis for parametric optimal control of semilinear parabolic equations. *Journal of Convex Analysis*, 9(2):543–561, 2002.
12. K. Malanowski. Solution differentiability of parametric optimal control for elliptic equations. In E. W. Sachs and R. Tichatschke, editors, *System Modeling and Optimization XX*, Proceedings of the 20th IFIP TC 7 Conference, pages 271–285. Kluwer Academic Publishers, 2003.

13. K. Malanowski and F. Tröltzsch. Lipschitz stability of solutions to parametric optimal control for parabolic equations. *Journal of Analysis and its Applications*, 18(2):469–489, 1999.
14. K. Malanowski and F. Tröltzsch. Lipschitz stability of solutions to parametric optimal control for elliptic equations. *Control and Cybernetics*, 29:237–256, 2000.
15. S. Robinson. Strongly regular generalized equations. *Math. Oper. Res.*, 5(1):43–62, 1980.
16. T. Roubíček and F. Tröltzsch. Lipschitz stability of optimal controls for the steady-state Navier-Stokes equations. *Control and Cybernetics*, 32(3):683–705, 2003.
17. R. Temam. *Navier-Stokes Equations, Theory and Numerical Analysis*. North-Holland, Amsterdam, 1984.
18. F. Tröltzsch. Lipschitz stability of solutions of linear-quadratic parabolic control problems with respect to perturbations. *Dynamics of Continuous, Discrete and Impulsive Systems Series A Mathematical Analysis*, 7(2):289–306, 2000.
19. M. Ulbrich. Semismooth Newton methods for operator equations in function spaces. *SIAM Journal on Control and Optimization*, 13(3):805–842, 1999.

Projected Hessians for Preconditioning in One-Step One-Shot Design Optimization

Andreas Griewank[1]

Institut für Mathematik Fakultät Math.-Nat. II
Humboldt-Universität, Rudower Chaussee 25, Adlershof
Post: Unter den Linden 6, 10099 Berlin
(griewank@math.hu-berlin.de)

Summary. *One-shot* optimization aims at attaining feasibility and optimality simultaneously, especially on problems where even the linearized constraint equations cannot be resolved economically. Here we consider a scenario where forming and factoring the active Jacobian is out of the question, as is for example the case when the constraints represent some discretization of the Navier Stokes equation. Assuming that the 'user' provides us with a linearly converging solver that gradually restores feasibility after each change in the design variables, we derive a corresponding adjoint iteration and attach an optimization (sub)step.

The key question addressed is how the approximate reduced gradient generated by the adjoint iteration should be preconditioned in order to achieve overall convergence at a reasonable speed. An eigenvalue analysis yields necessary conditions on the preconditioning matrix, which are typically not satisfied by the familiar reduced Hessian. Some other projection of the Lagrangian Hessian appears more promising and is found to work very satisfactorily on a nonlinear test problem.

The analyzed approach is *one-step* in that the normal, dual and design variables are always updated simultaneously on the basis of one function evaluation and its adjoint. Multi-step variants are promising but remain to be investigated.

Key words: Jacobian-free optimization, adjoint equation, algorithmic differentiation, R-linear Convergence, fixed point iteration.

1 Introduction

Design optimization problems are distinguished from general NLPs by the fact that the set of variables is split a priori into a state vector $y \in Y$ and a set of design or control variables $u \in U$. For simplicity we will assume that not only Y but also U and thus their Cartesian product $X = Y \times U$ are Hilbert

[1] Supported by the DFG Research Center MATHEON "Mathematics for Key Technologies" in Berlin

spaces. With $c : X \to Y$ the state residual and $f : X \to \mathbb{R}$ the objective we obtain the equality constrained optimization problem

$$\mathbf{Min} f(y, u) \quad \text{s.t.} \quad c(y, u) = 0 \in Y.$$

While the solution space Y has often a natural scalar product, the design space U may be rather heterogeneous, representing for example material properties and/or shapes in various parameterizations [14]. We will therefore strive to avoid any dependences on the inner product in U.

For notational convenience we will assume that Y and U have finite dimensions $n = \dim(Y)$ and $m = dim(U)$ with $n \gg m$, typically. Then elements $y \in Y$ and $u \in U$ may be identified with their coordinate vectors in $\mathbb{R}^n$ and $\mathbb{R}^m$ with respect to suitable Hilbert bases. This convention allows us to write duals as transposed vectors and inner products as the usual scalar products in Euclidean space. Also we will be able to compute determinants and characteristic polynomials to determine eigenvalues rather than performing spectral theory in Hilbert spaces.

A key assumption throughout the paper is that at all arguments $x = (y, u)$ of interest

$$\det(c_y(y, u)) \neq 0 \quad \text{where} \quad c_y \equiv \partial c(y, u)/\partial y. \tag{1}$$

Here and elsewhere subscripting by real vectors indicates partial differentiation, whereas integer subscripts are iteration counters. Throughout we will denote a locally unique solution of $c(y, u) = 0$ for fixed u as $y_* = y_*(u)$ and also annotate other quantities evaluated at such a feasible point by subscript $*$.

Related Concepts

The scenario sketched above is similar to the one described by Biros and Ghattas in [2] under the heading 'PDE Constrained Optimization'. They also consider a situation, where there are 'only' equality constraints, which makes the problem look simple from a superficial point of view. However, the key assumption is that the state constraint system is so complex that it is impossible to directly evaluate and manipulate the active constraint Jacobian in any way. In particular it is considered very troublesome to compute and represent the generally dense matrix of tangential directions because that requires matrix factorizations or a large number of iterative equation solves. Instead one wishes to get by with just a few approximate solves for the state equation and its adjoint per outer iteration.

In another excellent paper [13] Heinkenschloss and Vicente develop a framework, which is even notationally very close to the one used here. They rightly emphasize the importance of inner products and the corresponding norms for the definition of gradients and second derivatives as well as iterative equation solvers of the Krylov and quasi-Newton type. These aspects may sometimes be ignored with impunity on smaller problems but become

absolutely crucial on large scale NLPs that arise from the discretization of design or control problems involving differential equations. Ideally, the overall optimization algorithm should be set up such that it behaves mesh-invariant in that it generates approximately the same iterates and function values for all sufficiently fine meshes [5].

Complexity Growth

Especially when partial differential equations are involved, the process of simulation, i.e. the computation of a feasible state y given values of the design or control parameters u may take a substantial amount of computational resources. It is then very important that the transition to optimization causes neither the operations count nor the number of data transfers and total memory required to grow by more than a moderate factor. Otherwise one has to perform the optimization calculation on cruder models, or rely on shear intuition altogether. Ideally, the factor limiting the growth in complexity should be a problem independent constant. However, even theoretically, one has to allow for a logarithmic dependence on the problem size in order to limit the storage requirement [7] for gradient calculations on evolutionary, or instationary, problems. Even though for actual runtimes the characteristics of the computing platform and many other issues are important, we will keep in mind that the complexity of each outer optimization step as well as their total number required to achieve an acceptable solution accuracy does not exceed the corresponding costs for the underlying simulation by more than a moderate factor.

2 (R)SQP variants on the structured KKT System

Whenever the regularity assumption (1) is satisfied in the vicinity of the feasible set $c^{-1}(0) = \{x \in X : c(x) = 0\}$, the subspace $Y \equiv Y \times \{0\} \subset X$ can be used as a subspace of normal directions. The reason is that it must then be transversal to the nullspace spanned by the row vectors of the matrix

$$[Z^T, I_m] \quad \text{where} \quad Z = Z(y,u) \equiv -c_y(y,u)^{-1} c_u(y,u).$$

Here I_m denotes the identity matrix of order m. We may therefore use the adjective *normal* to qualify all terms and actions that are concerned with solving the state space equations without changing the design vector u. In contrast we will use the adjective *adjoint* for all terms and actions related to the adjoint equation associated with $c(y,u) = 0$, namely

$$0 = \bar{c}(y, \bar{y}, u) \equiv \bar{y}\, c_y(y,u) + \mu\, f_y(y,u) \quad \text{with} \quad \bar{y}^T \in \mathbb{R}^n. \tag{2}$$

Here $\mu > 0$ is a constant weighting of the objective function, which will be handy later on as relative scale of normal and adjoint errors. It is well known

that the solution vector $\bar{y}$ may be interpreted as the gradient of the weighted objective function value μf with respect to perturbations of the right hand side of the state equation $c(y,u) = 0$. We will view all barred quantities and all partial derivatives as row vectors, since they really belong to the dual spaces of Y or U. At feasible points $(y(u),u) \in c^{-1}(0)$ the total derivative, or reduced gradient, of f with respect to u is given by the adjoint vector

$$\bar{u}(y,\bar{y},u) \equiv \bar{y}\, c_u(y,u) + \mu\, f_u(y,u) \quad \text{with} \quad \bar{u}^T \in \mathbb{R}^m.$$

We note that by their definition the adjoint vectors $\bar{y}$ and $\bar{u}$ like all Fréchet derivatives are heavily dependent on the inner products in Y and U. We will assume that for Y the inner product has been given in a 'natural' way and strive to eliminate the influence of the inner product in U, which we view as more or less arbitrary. In other words we are looking for invariance with respect to linear transformations on the design space U. Whenever $\bar{c}(y,\bar{y},u) = 0$ we will speak of *adjoint feasibility*. It should be noted that due to the given variable partitioning this is not quite the same as the concept of *dual feasible* familiar from linear optimization, in particular.

Adding the optimality condition $\bar{u} = 0$ to the state equation $c(y,0) = 0$ and its adjoint (2) we get the coupled system

$$\begin{bmatrix} c(y,u) \\ \bar{c}(y,\bar{y},u) \\ \bar{u}(y,\bar{y},u) \end{bmatrix} = 0. \tag{3}$$

This is nothing but the KKT system for an equality constrained optimization problem with the a priori partitioned variable vector $x = (y,u)$. The notational association of the Lagrange Multiplier vector $\bar{y}$ with the state variables y is meaningful because (2) has the exact solution $\bar{y} = \mu\, f_y(y,u)\; c_y(y,u)^{-1}$. In other words $\bar{c}(y,\bar{y},u) = 0$ defines the Lagrange multiplier estimates such that the partial gradient of the Lagrangian

$$L(y,\bar{y},u) \equiv \mu\, f(y,u) + \bar{y}\, c(y,u) \tag{4}$$

with respect to the state vector y is always zero. Computationally this makes especially sense when the state space Jacobian $c_y(y,u)$ is not only nonsingular but also has sufficient structure to facilitate the economical computation of Newton steps $\Delta y = -c_y(y,u)^{-1} c(y,u)$. Whether this is done by factorization or using iterative solvers, the exact or approximate computation of $\bar{y}$ by the same scheme applied to the transposed system will then only require a similar amount of computing time and very little extra coding. This applies in particular when $c(y,u)$ is selfadjoint with respect to y, i.e. the partial Jacobian c_y is symmetric. However, in view of nonelliptic problems we will later in this paper be mainly concerned with the case, where solving even the linearized state or adjoint equation for fixed u with any degree of accuracy is already very expensive.

Inexact Subspace Iteration

The structure of the KKT system (3) with the state equation Jacobian c_y being nonsingular suggests a block solution approach consisting of three substeps in each major iteration or cycle. First compute a normal step

$$y \;=\; y + \Delta y \quad \text{with} \quad \Delta y = -c_y(y,u)^{-1} c(y,u) \tag{5}$$

towards primal feasibility. Then, at the new y point evaluate

$$\bar{y} \;=\; \mu\, f_y(y,u)\, c_y(y,u)^{-1} \quad \text{and} \quad \bar{u} = \bar{y}\, c_u(y,u) + \mu\, f_u(y,u). \tag{6}$$

Finally, given a symmetric positive matrix $H \in \mathbb{R}^{n\times n}$ perform an optimization step

$$u \;=\; u + \Delta u \quad \text{with} \quad \Delta u \equiv\; -H^{-1}\bar{u}^T. \tag{7}$$

In other words we perform a cycle of three steps on the primal state, adjoint state, and design vector, to enhance feasibility, adjoint feasibility, and optimality, respectively.

On nonlinear problems one needs conceptually infinitely many such cycles, and it does not matter for the asymptotic behavior with which of the three steps one begins. Hence we may characterize this approach schematically as

$$\cdots \rightarrow \textbf{normal} \rightarrow \textbf{adjoint} \rightarrow \textbf{optim} \rightarrow \cdots$$

An (R)SQP interpretation

When the Lagrange multipliers are really recomputed from scratch as suggested by (6) one may merge the second step with the third and obtains a scheme similar to the Reduced SQP Algorithm 1 detailed in [12] by Hazra and Schulz. The only difference is that they avoid the evaluation of the constraints at two different arguments in each cycle by replacing the right hand side for the computation of Δy with the approximation $c(y,\tilde{u}) + c_u(y,\tilde{u})(u-\tilde{u}) \approx c(y,u)$. Here $\tilde{u} = u - \Delta u$ denotes the old design at which c and its derivatives were already evaluated in the previous cycle. Either variant yields locally 2-step, or more precisely in our context 2-cycle Q-superlinear convergence, if the preconditioner H equals at least asymptotically the reduced Hessian

$$H(1) \;=\; L_{uu} + Z^T L_{yu} + L_{uy} Z + Z^T L_{yy} Z, \tag{8}$$

where the subscripts u and y denote partial differentiation with respect to the design and state variables, respectively. Throughout the paper we will use this notation and refer to $H(1)$ as the *reduced* rather than the projected Hessian, since there will be other projections of interest.

R-superlinear convergence has also been established for more classical RSQP, i.e. reduced SQP, methods where the last step is properly tangential. One should notice that this is not the case for our optimization step (7),

which leaves the state variables uncorrected and relies on the subsequent normal step (5) to straighten them out. Thus we obtain in some way a method of alternating direction, which like other nonlinear block Gauss-Jordan variants is generally prone to yield only linear convergence. Proper RSQP methods do not need to apply the Schulz correction mentioned above and can evaluate all function and derivative evaluations at a single argument per cycle. There is, however, a variant due to Coleman and Conn [3] that expends a second residual evaluation after the normal step in order to achieve 1-cyle Q-superlinear convergence. In contrast we require a second residual evaluation after the optimization step or need to approximate its value by the Schulz correction. This still does not give us 1-cycle Q-superlinear convergence unless we perform two normal corrections in a row thus applying the scheme

$$\cdots \rightarrow \textbf{normal} \rightarrow \textbf{adjoint} \rightarrow \textbf{optim} \rightarrow \textbf{normal} \rightarrow \cdots$$

For other RSQP variants and a very detailed convergence analysis see the seminal paper by Biegler, Nocedal and Schmid [1].

3 Pseudo-Newton Solvers and Piggy-backing

The discussion above tacitly assumed that it is possible to approximate normal and adjoint Newtons steps with some degree of accuracy at a reasonable cost. Especially on discretizations of nonelliptic PDEs this assumption is rather optimistic and one may have to make do with a possibly rather slowly convergent fixed point iteration of the form

$$y_{k+1} \;=\; G(y_k, u) \quad \text{with} \quad G : X = Y \times U \rightarrow Y. \tag{9}$$

Naturally, we assume that the *iteration function* G is stationary only at feasible points, i.e.

$$y = G(y, u) \quad \Longleftrightarrow \quad c(y, u) = 0.$$

This holds for example when $c(y, u) = P(y, u)[G(y, u) - y]$ with $P(x) = I_m$ or some other family of nonsingular matrices, e.g. $P(y, u) = c_y(y, u)$ when G is defined by Newton's methods.

Linear Convergence Rates

In aerodynamics it is not unusual that thousands of iterations are needed to obtain a high accuracy solution of the state equation. We will assume throughout that G is contractive such that for some constant ρ in the induced matrix norm $\|\cdot\|$

$$\|G_y(y, u)\| \;\leq\; \rho < 1 \implies \|G(y, u) - G(z, u)\| \leq \rho \, \|y - z\|. \tag{10}$$

The implication follows from the mean value theorem on any convex subdomain of Y. Another consequence is that no eigenvalue of $G_y(y,u)$ can be in modulus greater than ρ at any point (y,u).

Now it follows from Banach's fixed point theorem that the normal iteration (9) converges for any given u to the unique fixed point $y_* = y_*(u)$ solving $c(y(u),u) = 0$. Moreover, the rate of convergence is linear with the Q-factor

$$\rho_0 \equiv Q\{y_k - y_*\}_{k\in\mathbb{N}} \equiv \limsup_k \|y_{k+1} - y_*\| / \|y_k - y_*\| \leq \rho.$$

The norm independent R-factor may be smaller [15] and is given by

$$\rho_* \equiv R\{y_k - y_*\}_{k\in\mathbb{N}} \equiv \limsup_k \sqrt[k]{\|y_k - y_*\|} \leq \rho_0 \leq \rho.$$

When even ρ_* is quite close to 1.0 it makes little sense to refer to (9) as a Newton-like iteration. In his thesis [4] Courty has considered such slowly convergent solvers for Navier Stokes variants and called them more appropriately *pseudo-Newton* iterations.

Our key assumption is that the 'user' provides (only) such a pseudo-Newton solver, which may have been developed over a long period of time by various people. Presumably, they were experts at exploiting the particular structure of the state equation, so one cannot easily accelerated or otherwise improve the given fixed point solver. Now the optimizer faces the task of converting this given simulation tool into an optimization tool without performing major surgery on the code. We have coined the term *piggy-backing* for this hopefully convenient transition.

The Adjoint Iteration

Assuming that the 'user' also provides a code for the objective function $f(y,u)$ one can apply automatic, or algorithmic differentiation in the reverse mode to evaluate the associated adjoints

$$\bar{y}_{k+1} = \bar{G}(y_k, \bar{y}_k, u) \equiv \bar{y}_k\, G_y(y_k,u) + \mu\, f_y(y_k,u) \tag{11}$$

and

$$\bar{u}_{k+1} \equiv \bar{y}_k\, G_u(y_k,u) + \mu f_u(y_k,u). \tag{12}$$

It is in the nature of the reverse mode that both $\bar{y}_{k+1}$ and $\bar{u}_{k+1}$ are obtained simultaneously and with a basic operations count not exceeding 4 times that for evaluating G and f. While this complexity result from [7] takes into account memory moves as operations, it does rely on the ability to temporarily store the intermediate results of all arithmetic operations and special function evaluations. Alternatively, one may employ checkpointing techniques, which allow various trade-offs between the increase in the temporal and the spatial complexity for the adjoint (11) relative to the normal (9) iteration. In

particular one can achieve an only logarithmic growth in both complexity components. For more details see again [7] and [8].

In the current context it is only important to note that performing the adjoint step (11,12) is not much more expensive than computing the normal step (9) and that, except in linear cases, no saving is made if the former is executed without the latter. More specifically, the computation of the adjoint quantities $\bar{y}_{k+1}$ and $\bar{u}_{k+1}$ requires the execution of a forward and a backward sweep through the evaluation procedure for $[G, f]$, of which the first sweep already yields $\mathbf{y}_{k+1}$. As a cautionary note we add that the mechanical application of an AD tool in reverse mode to l successive steps (9) will lead to a memory requirement that grows proportional to l. This memory growth proportional to the number of iterations is only acceptable for true evolutions, e.g. discretized dynamical systems. It can and should be avoided whenever the iterates converge to a stationary solution, in other words if we have a pseudo-time evolution.

Notice that in (11) the Lagrange multipliers $\bar{y}_k$ have a true iteration history, rather than being recalculated from scratch at each primal iterate. The latter is usually done in NLP solvers that have access to the active constraint Jacobian. Naturally, the resulting approximate reduced gradient $\bar{u}_{k+1}$ will be as good or bad as the current multiplier estimate $\bar{y}_k$. For $c(y,u) = P(y,u)\,[G(y,u) - y]$ we have at feasible points (y,u) the identity $c_y(y,u) = P(y,u)\,[G_y(y,u) - I_n]$. Then it is easy to see that $\bar{y}_* = \bar{y}_*(u)$ is exactly a fixed point of (11) at $(y,u) \in c^{-1}(0)$ if $\bar{y}_* P(y_*,u)^{-1}$ is a solution of the original adjoint equation (2).

It is well understood that if one first solves the state equation $c(y,u) = 0$ to full accuracy and only then executes the adjoint iteration (11) the convergence rates of the two iterative processes will be very similar to each other. The reason is that the transposed Jacobian

$$\partial \bar{G}(y,\bar{y},u)/\partial \bar{y} \;=\; G_y^T \;=\; [\partial G(y,u)/\partial y]^T$$

has of course exactly the same spectral properties as G_y. Since, for fixed (y,u), the adjoint $\bar{G}$ is linear with respect to $\bar{y}$ we may even expect a more regular convergence pattern for the adjoints than for the underlying normal iteration. It was first observed in [6] that if G describes Newton's method so that $G_y(y,u)$ vanishes at any feasible (y,u), then the adjoints will converge in a single step, which amounts simply to an application of the implicit function theorem. For slowly converging solvers on the other hand, it can often be observed that executing the adjoint iteration simultaneously with the underlying state space iteration does not slow the former down significantly. This simultaneous iteration certainly promises a significant reduction in wall-clock time, especially on a multiprocessor machine.

The Coupled Iteration

For the subsequent analysis we write the combination of (9) and (11) in a Hamiltonian like fashion as

$$\left[y_{k+1}^T, \bar{y}_{k+1}\right] \;=\; \nabla N(y_k, \bar{y}_k, u) \;=\; \left[N_{\bar{y}}(y_k, \bar{y}_k, u), N_y(y_k, \bar{y}_k, u)\right], \tag{13}$$

where N is defined as

$$N(y, \bar{y}, u) \;\equiv\; \bar{y}\, G(y, u) + \mu\, f(y, u). \tag{14}$$

When we set $c(y, u) = G(y, u) - y$ then the function N is related to the Lagrange function L as defined in (4) by the shift

$$\bar{y}\, y = N(y, \bar{y}, u) - L(y, \bar{y}, u).$$

Consequently, we have then

$$N_y = L_y + \bar{y} \quad \text{and} \quad N_{\bar{y}} = L_{\bar{y}} + y \quad \text{but} \quad N_u = L_u$$

and, for the subsequent analysis more importantly, all second derivatives are identical:

$$N_{yy} = L_{yy},\; N_{yu} = L_{yu}, N_{uu} = L_{uu}.$$

In the more general situation $c(y, u) = P(y, u)[G(y, u) - y]$ the second derivatives are related in such a way that second order necessary and sufficiency conditions applied to N and L are still equivalent.

In any case we note that N is linear in the weighting $(\bar{y}, \mu)$ and thus homogeneous in μ, provided $\bar{y}$ solves the adjoint equation $N_y(y, \bar{y}, u) = \bar{y}$ or is initialized and consistently updated proportionally to μ. Hence we can make N and its derivatives arbitrarily small compared to G by reducing the weighting μ on the objective appropriately.

A side benefit of iterating on $\bar{y}_k$ simultaneously with y_k is that the Lagrangian value

$$\mu\, f(y_k, u) + \bar{y}_{k+1}\,(y_{k+1} - y_k) \;=\; \mu\, f(y_*(u), u) + \mathcal{O}(\|\bar{y}_{k+1} - \bar{y}_*\| \|y_{k+1} - y_*\|) \tag{15}$$

converges much faster than $\mu\, f(y_k, u)$ towards the limit $\mu\, f(y_*(u), u)$. This result was derived in [9] using a very general duality argument. It merely reflects that the Lagrange multipliers indicate the sensitivity of the optimal function value with respect to perturbations of the state equations and that the current iterate exactly satisfies a slight perturbation of these 'constraints'.

Adjoint Convergence Analysis

The Jacobian of the coupled iteration (13) takes the block triangular form

$$J_k \;\equiv\; \frac{\partial(y_{k+1}, \bar{y}_{k+1})}{\partial(y_k, \bar{y}_k)} \;=\; \begin{bmatrix} G_y(y_k, u) & 0 \\ N_{yy}(y_k, \bar{y}_k, u) & G_y^T(y_k, u) \end{bmatrix}.$$

By elementary arguments one derives that in the l_2 matrix norm

$$\|J_k\| \;\leq\; \max\left(\|G_y\|, \|G_y^T\|\right) + \|N_{yy}\| \;=\; \rho + \mathcal{O}(\mu).$$

In other words for sufficiently small μ the combined iteration (13) is, like G, also a contraction on the product space $Y \times \bar{Y}$. Using the homogeneity of all adjoint quantities with respect to μ, one can then deduce that for any fixed μ also

$$\bar{\rho}_* \;\equiv\; R\{\bar{y}_k - \bar{y}_*\}_{k \in \mathbb{N}} \;\equiv\; \limsup_k \sqrt[k]{\|\bar{y}_k - \bar{y}_*\|} \;\leq\; \rho.$$

This R-linear convergence result was generalized in [9] to contractive iterations of the form $y_{k+1} = G_k(y_k, u) = y_k - P_k\, c(y_k, u)$ coupled with the corresponding adjoints. Here the matrices P_k may vary discontinuously from step to step as is for example the case for quasi-Newton methods.

On the other hand, the following observation was made in [11] for twice continuously differentiable $G(y, u)$ and $f(y, u)$. Since J_k is block triangular and has in its diagonal the Jacobian G_y and its transposed G_y^T, every eigenvalue of G_y is a double eigenvalue of J_k, which has thus generically Jordan blocks of size 2. As a consequence, for fixed u the adjoints $\bar{y}_k$ and $\bar{u}_k$ converge a little slower than the normal iterates y_k in that in general

$$\|\bar{u}_k - \bar{u}_*\| \;\sim\; \|\bar{y}_k - \bar{y}_*\| \;\sim\; k\; \|y_k - y_*\|. \tag{16}$$

In other words, despite the common R-factor, the Lagrange multipliers $\bar{y}_k$ and the corresponding reduced gradients $\bar{u}_k$ lag somewhat behind the normal iterates y_k. In hindsight that is not all that surprising since the adjoints are heading for a moving target, namely the solution of an equation parameterized by y_k. A practical consequence seems that any optimization algorithms that is based on the approximate reduced gradients $\bar{u}_k$ will likely exhibit tangential convergence to constrained optima. In other words feasibility will be reached faster than adjoint feasibility and thus optimality, a pattern that is familiar from NLP solvers based on exact Jacobians but (secant) approximated Hessians.

Reduced Hessian as Second Order Adjoints

Differentiating (13) and (12) once more with respect to $x = (y, u)$ in some direction $(\dot{y}_k, \dot{u})$ one obtains an iteration of the form

$$\begin{bmatrix} \dot{y}_{k+1} \\ \dot{\bar{y}}_{k+1} \\ \dot{\bar{u}}_{k+1} \end{bmatrix} \;=\; \begin{bmatrix} G_y(y_k, u)\dot{y}_k + G_u(y_k, \bar{y}_k, u)\dot{u} \\ \dot{\bar{y}}_k G_y(y_k, u) + (N_{yy}(y_k, \bar{y}_k, u)\dot{y}_k + N_{yu}(y_k, \bar{y}_k, u)\dot{u})^T \\ \dot{\bar{y}}_k G_u(y_k, u) + (N_{uy}(y_k, \bar{y}_k, u)\dot{y}_k + N_{uu}(y_k, \bar{y}_k, u)\dot{u})^T \end{bmatrix}. \tag{17}$$

Here we have separated on the right hand side the leading terms that are dependent on $\dot{y}_k \in \mathbb{R}^n$ and $\dot{\bar{y}}_k^T \in \mathbb{R}^n$, respectively. Since they are defined again in terms of the contractive state space Jacobian G_y one gets once more linear convergence as follows.

Proposition 1 *With* $Z_* \equiv (I - G_y)^{-1} G_u \in \mathbb{R}^{m \times n}$ *the derivatives* $\dot{y}_k \in \mathbb{R}^n, \dot{\bar{y}}_k^T \in \mathbb{R}^n$ *and* $\dot{\bar{u}}_k^T \in \mathbb{R}^m$ *converge with R-Factors* $\leq \rho$ *such that in the limit*

$$\lim_k \begin{bmatrix} \dot{y}_k \\ \dot{\bar{y}}_k \\ \dot{\bar{u}}_k \end{bmatrix} = \begin{bmatrix} Z_* \dot{u} \\ \dot{u}^T [Z_*^T, I] N_{xy} (I - G_y)^{-1} \\ \dot{u}^T [Z_*^T, I] N_{xx} [Z_*^T, I]^T \end{bmatrix}.$$

Proof: The fact of linear convergence with R-factor $\leq \rho$ was proven in [9]. The limits can be derived in the current framework as follows. Simply replacing in (17) all subscripts k by $*$ and collecting the unknown vector onto the left side we obtain linear equations for $\dot{y}_*, \dot{\bar{y}}_*$ and $\dot{\bar{u}}_*$, all in terms of the matrix $I - G_y$ or its transpose. The first equation has the right hand side G_u, which immediately yields the solution $(I - G_y)^{-1} G_u \dot{u} = Z_* \dot{u}$. Substituting this result into the second equation we obtain the transposed right hand side

$$N_{yx} \dot{x}_* = N_{yy} \dot{y}_* + N_{yu} \dot{u}_* = (N_{yy} Z_* + N_{yu}) \dot{u}_* = N_{yx} \begin{bmatrix} Z_* \\ I \end{bmatrix} \dot{u}.$$

Transposition and post-multiplication by $(I - G_y)^{-1}$ yields the assertion. Substitution of this result into the third yields firstly the term $\bar{y}_* G_u = \dot{u}^T [Z_*^T, I] N_{xy} (I - G_y)^{-1} G_u = \dot{u}^T [Z_*^T, I] N_{xy} Z_*$ to which we have to add $\dot{x}_*^T N_{xu} = \dot{u}_*^T (Z_*^T, I) N_{xu}$. The sum can than be written as the asserted partitioned matrix product.

The proposition states that for each selected direction $\dot{u}$ in the domain space, the first derivative vectors $\dot{y}_k$ converge to the state space direction $\dot{y}_* = Z_* \dot{u}$, which makes $\dot{x} = (\dot{y}_*, \dot{u})$ a feasible direction in the full space $X = Y \times U$. The corresponding adjoint vectors $\dot{\bar{y}}_k$ converge to the product of the Lagrangian Hessian and the vector $\dot{x} = (\dot{y}_*, \dot{u})$, except for the nonsingular transformation $I - G_y$. The corresponding second order adjoints $\dot{\bar{u}}_k$ converge to the transposed direction $\dot{u}^T$ multiplied by the reduced Hessian. As in Section 1 we will denote the latter by $H_*(1) = [Z_*^T, I] N_{xx} [Z_*^T, I]^T \in \mathbb{R}^{m \times m}$ for reasons that will become clear later.

While the iteration (17) looks reasonably complicated it can also be realized in a completely automatic fashion, which does not increase the complexity by more than a constant factor. If one replaces $\dot{u}$ in the above results by the identy I_m on the design space the whole reduced Hessian $H_*(1)$ is obtained as $\dot{\bar{u}}_*$ by what is sometimes called the *vector mode* of automatic differentiation. Naturally the cost will then also grow by a factor of m, namely the dimension of the design space. In our test calculations we have so far made that effort, though we prefer a different projected Hessian denoted by $H_*(-1)$ that can be obtained using a slight variation of (17).

The analysis in [11] shows that the errors in $\dot{\bar{y}}_k$ and thus also in $\dot{\bar{u}}_k$ lag asymptotically behind those in y_k by a factor proportional to k^2 so that

$$\|\dot{\bar{u}}_k - \dot{\bar{u}}_*\| \sim \|\dot{\bar{y}}_k - \dot{\bar{y}}_*\| \sim k^2 \|y_k - y_*\|. \tag{18}$$

In view of (16) this effect is not surprising and probably not so serious, because the accuracy of reduced Hessians is not nearly as critical as that of reduced gradients for rapid and robust convergence.

Inexact Block Gauss-Seidel

Now suppose we decide to always execute the coupled step (13) $r > 0$ times, before executing one optimization step of the form (7). Schematically, we may write this approach as

$$\cdots \to [\mathbf{normal}, \mathbf{adjoint}]^r \to \mathbf{optim} \to \cdots \tag{19}$$

Here [**normal**, **adjoint**] represents (13) with $u_k = u$ remaining unchanged and **optim** represents the update

$$u_{k+1} \;=\; u_k - H_k^{-1} \left[\bar{y}_k \, G_u(y_k, u_k) + \mu \, f_u(y_k, u_k)\right]^T . \tag{20}$$

(Thus k is employed to count individual steps rather than whole cycles.)

When we select r very large the first two equation blocks in the KKT system (3) are solved quite accurately, so that we have a block Gauss-Seidel type method. However, it is clearly doubtful whether it makes sense to expend a large computational effort for gaining feasibility and adjoint feasibility at a design argument u that may still be far from the optimal. This observation leads naturally to the concept of 'one-shot' optimization as it is called in the aerodynamics literature [16]. The basic idea is to adjust the design u as frequently as possible in order to gain optimality more or less simultaneously with feasibility.

From an 'algebraic' point of view it seems attractive to adjust u at every step. Here we tacitly assume that changing u from a current value u_k to a different design u_{k+1} causes no extra cost for the subsequent evaluation of the iteration function G at (y_{k+1}, u_{k+1}). However, in applications there may be a significant setup cost for example due to the need to regrid after changes in a wing design. In that situation our hypothesis that $\bar{u}_{k+1}$ costs little extra on top of the evaluation of $\bar{y}_{k+1}$ may also be a little naive. To model these effects one would have to consider some partial preevaluation of $G(y, u)$ and then distinguish the cost of evaluating G and its derivatives with or without a change in u. For simplicity we shy away from doing so, but will keep this effect in mind. On a theoretical level one may refine G as a subcyles of l normal steps, such that its combined cost clearly dominates the setup cost incurred by changing the design. In any case we may realistically assume that $m = \dim(u)$ is so small that the linear algebra associated with the optimization step especially storing and manipulating the preconditioning matrix H is comparatively cheap.

Finally, as we have observed in the previous section for an RSQP-like method it may make sense to perform more normal steps than coupled ones so that one arrives at a the following most general scheme

$$\cdots \rightarrow [\mathbf{normal}]^s \rightarrow [\mathbf{normal}, \mathbf{adjoint}]^r \rightarrow \mathbf{optim} \rightarrow \cdots \tag{21}$$

Here **normal** represents (9) with $u_k = u_k$ remaining unchanged.

Now the key question is naturally how to first select the repetition numbers s and r and then how to best define and compute the matrix H_k in (20). Since there is no separate line-search it should absorb the step multiplier and achieve the invariance with respect to linear transformations on the design space. Naturally, when $\rho^{(s+r)}$ is still close to 1.0 we must choose H_k large and thus H_k^{-1} small such that the design variables u_k are adjusted rather cautiously. Otherwise the iteration may diverge immediately as was we have observed on our test problem. On the other hand we wish to choose H_k small enough such that the overall optimization proceeds at a reasonable pace that is not much smaller than normal iteration for pure simulation at a fixed design u. While there are some experimental studies and corresponding recommendations for particular application areas we are not aware of a systematic study on the best choice of the repetition numbers s, r and the possible choices of H_k. There is a natural tendency for optimization people to assume that the reduced Hessian $H_k = H_*$ as defined in (8) is the optimal choice. However, we believe this only to be true when ρ^r is quite small so that a large degree of normal and adjoint feasibility is restored before each optimization step. It certainly is not valid at the other extreme namely for the one-step method

$$\cdots \rightarrow [\mathbf{normal}, \mathbf{adjoint}, \mathbf{optim}] \rightarrow \cdots \tag{22}$$

In contrast to the scheme above with $s = 0$ and $r = 1$ we assume here that the reduced gradient (12) is still evaluated with the old values $(y_k, \bar{y}_k)$ rather than the new versions $(y_{k+1}, \bar{y}_{k+1})$ just obtained in the same cycle. As we have discussed above, from an algebraic view point this value $\bar{y}_k \, G_u(y_k, u_k) + f_u(y_k, u_k)$ is a cheap by-product of evaluating $[G, f]$ and $\bar{G}$ at $(y_k, \bar{y}_k, u_k)$, but obtaining the newer reduced gradient $\bar{y}_{k+1} \, G_u(y_{k+1}, u_k) + f_u(y_k, u_k)$ would require a completely new evaluation of the same quantities at $(y_{k+1}, \bar{y}_{k+1}, u_k)$. While this simplification may cause the loss of of superlinear convergence for an RSQP like method, we believe that it makes very little difference for the convergence speed in a pseudo-Newton setting.

4 Necessary Convergence Condition on H_*

Throughout this section we assume that we are within the neighborhood of a KKT point $(y_*, \bar{y}_*, u_*)$, i.e. a solution to (3) in which the contractivity assumption (10) holds and G as well as f are twice Lipschitz continuously differentiable. The contractivity implies linear independence of the constraints $G(y, u) - y = 0$ and thus uniqueness of the multipliers $\bar{y}_*$ for given (y_*, u_*). Using the function $N(y, \bar{y}, u)$ defined above we may write one step of the one-step scheme (22) as

$$\begin{bmatrix} y_{k+1} \\ \bar{y}_{k+1} \\ u_{k+1} \end{bmatrix} = \begin{bmatrix} G(y_k, u_k) \\ N_y(y_k, \bar{y}_k, u_k)^T \\ u_k - H_k^{-1}[N_u(y_k, \bar{y}_k, u_k)]^T \end{bmatrix}. \tag{23}$$

Omitting the common argument $(y_*, \bar{y}_*, u_*)$ and assuming that the H_k converge to a limit H_* we obtain at the fixed point the block 3×3 Jacobian

$$\begin{aligned} \hat{J}_* &= \left. \frac{\partial(y_{k+1}, \bar{y}_{k+1}, u_{k+1})}{\partial(y_k, \bar{y}_k, u_k)} \right|_{(y_*, \bar{y}_*, u_*)} \\ &= \begin{bmatrix} G_y & 0 & G_u \\ N_{yy} & G_y^T & N_{yu} \\ -H_*^{-1}N_{uy} & -H_*^{-1}G_u^T & I - H_*^{-1}N_{uu} \end{bmatrix}. \end{aligned} \tag{24}$$

The eigenvalues of the system Jacobian can be characterized as those of its Schur complement.

Proposition 2 *For any nonsingular $H_* = H_*^T$ the eigenvalues of the matrix $\hat{J}_*$ are either eigenvalues of G_y or solve the nonlinear eigenvalue problem* $\det(M(\lambda)) = 0$ *with*

$$M(\lambda) \equiv (\lambda - 1)H_* + [-G_u^T(G_y^T - \lambda I)^{-1}, I]N_{xx} \begin{bmatrix} -(G_y - \lambda I)^{-1}G_u \\ I \end{bmatrix}, \tag{25}$$

where $N_{xx} = \nabla^2_{(y,u)}N \in \mathbb{R}^{(n+m)\times(n+m)}$.

Proof: The eigenvalues λ must solve the characteristic equation

$$0 = \det \begin{bmatrix} G_y - \lambda I & 0 & G_u \\ N_{yy} & G_y^T - \lambda I & N_{yu} \\ -H_*^{-1}N_{uy} & -H_*^{-1}G_u^T & (1-\lambda)I - H_*^{-1}N_{uu} \end{bmatrix}. \tag{26}$$

Without changing the roots we can premultiply the last block row by the the negative of the nonsingular matrix $-H_*$, which now occurs in place of I. Unless a λ is eigenvalue of G_y the leading 2×2 block has the inverse

$$\begin{bmatrix} (G_y - \lambda I)^{-1} & 0 \\ -(G_y - \lambda I)^{-T}N_{yy}\,(G_y - \lambda I)^{-1} & (G_y - \lambda I)^{-T} \end{bmatrix}. \tag{27}$$

Using this block to eliminate the last block row and block column we obtain after some rearrangements as Schur complement exactly the negative of matrix $M(\lambda)$.

Now we look for conditions that exclude the existence of eigenvalues with modulus greater than or equal to 1. To this end we denote the right term in $M(\lambda)$ by

$$H(\lambda) \equiv [Z(\lambda)^T, I]\, N_{xx}\, [Z(\lambda)^T, I]^T,$$

where

$$Z(\lambda) \;=\; [\lambda I - G_y]^{-1} G_u \in \mathbb{R}^{n\times m}.$$

Since we may write the state constraints as $G(y,u) - y = 0$ the columns of the particular matrix $Z(1)$ represent feasible directions restricted to the state space. Therefore the positive definiteness property of $H(1)$ being positive definite, denoted by $H(1) \succ 0$, corresponds to the usual second order sufficiency condition at constrained minima. More generally the rows of $[Z(\lambda)^T, I]$ span the tangent space of the manifold $\{G(y,u) = \lambda y\}$ and $H(\lambda)$ represents the projection of N_{xx} onto that m-dimensional subspace.

The nicest situation occurs when G_y has n distinct real eigenvalues and our optimization problem is strongly convex in that the Lagrangian Hessian N_{xx} is positive definite on the whole space $X = \mathbb{R}^{n+m}$. Then the rational function $\det M(\lambda)$ has n double-poles in the interval $(-1,1)$ and for positive definite H_* we may expect two eigenvalues of $\hat{J}_*$ close to each one of them. Of those only the ones beyond the largest and below the smallest eigenvalue of G_y may destroy contractivity. However, there is no guarantee that there may not be complex eigenvalues of modulus greater than 1.0 even in this convenient scenario. Therefore the necessary conditions for convergence derived in the following proposition are probably quite far from sufficient.

Proposition 3 *Provided $H(1) \succeq 0$, it is a necessary condition for $\hat{J}_*$ to have no real eigenvalues $\lambda \geq 1$ that $H_* \succ 0$ and $\det(H(1)) \neq 0$. Moreover, then for $\hat{J}_*$ also to have no real eigenvalues $\lambda \leq -1$ it is necessary that $H_* \succeq \frac{1}{2} H(-1)$.*

Proof: Since $M(1) = H(1)$ the singularity of $H(1)$ would immediately imply that $\lambda = 1$ is an eigenvalue of $\hat{J}_*$, which establishes $\det(H(1)) \neq 0$. As clearly $Z(\lambda) = \mathcal{O}(|\lambda|^{-1})$ we have $H(\lambda) = N_{uu} + \mathcal{O}(|\lambda|^{-1})$ and the leading term $(\lambda - 1)H_*$ of $M(\lambda)$ as defined in (25) dominates $M(\lambda)$ for large $|\lambda|$. Now if H_* had a negative eigenvalue the matrix $M(\lambda)$, which reduces to $H(1) = M(1)$ at $\lambda = 1$, would have to become singular for some $\lambda \geq 1$. Because H_* is by definition symmetric and nonsingular we must therefore have $H_* \succ 0$. Now suppose the second condition was violated so that $M(-1) = -2H_* + H(-1)$ has a positive eigenvalue. Then there would have to be a real eigenvalue $\lambda < -1$ since $(\lambda - 1)H_*$ and thus $M(\lambda)$ must be negative definite for large negative λ.

According to Proposition 2 it is necessary for the iteration (23) to be contractive that

$$0 \;\prec\; 2\,H_* \succ H(-1) \;=\; [G_u^T (I + G_y)^{-T}, I] N_{xx} [(I + G_y)^{-1} G_u, I]^T,$$

provided the second order sufficiency conditions are satisfied at the KKT point in question. If the underlying optimization problem is strongly convex then $H(-1)$ is like all $H(\lambda)$ positive definite and it looks like a reasonable candidate for the preconditioner H_*. On the other hand there is no reason, why the standard projected Hessian $H(1)$ should be greater than $H(-1)/2$ and if it fails to do so then using it in (23) must prevent even local convergence.

On our test problem the standard choice $H_* = H_*(1)$ leads to immediate blow up of the iteration, while $H_* = H_*(-1)$ yields full step convergence without any line-search. While the problem is not everywhere convex we found that eventually all components of the Lagrange multiplier vector $\bar{y}_k$ were positive so that naturally both $H_* = H_*(1)$ and $H_* = H_*(-1)$ allowed a Cholesky factorization.

To make the surprising result of our eigenvalue analysis a little more intuitive we offer the following, admittedly very vague explanation. The matrix $Z(1) = (I - G_y)^{-1} G_u$ spans the tangent space of $\{G(y,u) = y\}$, which is rich in the eigenvectors of G_y for which the corresponding eigenvalues are close to $+1.0$. We may call them mononotic modes in contrast to the alternating modes associated with eigenvalues close to -1.0 The latter are richly represented in $Z(-1) = (I + G_y)^{-1} G_u$ whose columns span the tangent space of $\{G(y,u) = -y\}$. By preconditioning the approximate reduced gradient with $H(-1) = Z(-1)^T N_{xx} Z(-1)$ we seem to be monitoring more the effects of these alternating modes. Their strength is perhaps more detrimental to a regular convergence behavior than that of the monotonic modes.

To actually compute approximations $Z_k \approx Z(-1) \in \mathbb{R}^{n \times m}$ and $H_k \approx H_*(-1) \in \mathbb{R}^{m \times m}$ one may use the following modification of the vector iteration (17).

$$\begin{bmatrix} -Z_{k+1} \\ -\bar{Z}_{k+1} \\ H_{k+1} \end{bmatrix} = \begin{bmatrix} G_y(y_k,u) Z_k + G_u(y_k, \bar{y}_k, u) \\ \bar{Z}_k G_y(y_k,u) + (N_{yy}(y_k,\bar{y}_k,u) Z_k + N_{yu}(y_k,\bar{y}_k,u))^T \\ \bar{Z}_k G_u(y_k,u) + (N_{uy}(y_k,\bar{y}_k,u) Z_k + N_{uu}(y_k,\bar{y}_k,u))^T \end{bmatrix}. \tag{28}$$

Here the matrix $\bar{Z}_k \in \mathbb{R}^{m \times n}$ represents the adjoint of Z_k. The two minus signs in front of Z_{k+1} and $\bar{Z}_{k+1}$ on the left are quite deliberate and result from setting $\lambda = -1$. Of course we can move them to the right, especially in order to verify that we have again linear contractions in the state space Jacobian. Leaving them on the left only illustrates that we can use standard differentiation procedures to evaluate the right hand sides and then only have to flip signs to get the correct values for Z_{k+1} and $\bar{Z}_{k+1}$.

Finally, let us consider what happens when the design variable u is replaced by $C\,u$ with $C \in \mathbb{R}^{m \times m}$ and $\det(C) \neq 0$. Then all differentiations with respect to the new design variable will 'eject' the extra factor C according to the chainrule. In particular the reduced gradient $\bar{u}$ and the matrices $Z(\lambda)$ will be multiplied from the right by C, and $H(\lambda)$ will be multiplied from the right by C and from the left by C^T. The same will be true for any preconditioner H_* that we define as a linear combination of one or several of the $H(\lambda)$. Then it can be checked without too much trouble that the corrections to the design variables according to (20) are C^{-1} times those for the original setting and thus unchanged except for the coordinate transformation. Hence we have indeed achieved invariance with respect to the inner product norm and the design space U.

5 Numerical Verification

The following results were obtained on the unit square for a variant of the Bratu equation frequently used in combustion modeling.

$$\Delta_x y(x) + e^{y(x)} = 0 \quad \text{for} \quad x = (\zeta, \eta) \in [0,1]^2 \quad \text{s.t.}$$

$$y(0,\eta) = y(1,\eta), \quad y(\zeta,0) = \sin(2\pi\zeta), \quad y(\zeta,1) = u(\zeta) \quad \text{for} \quad \zeta,\eta \in [0,1]$$

Hence the problem is periodic with respect to the horizontal coordinate ζ and has Dirichlet boundary conditions on the lower and upper edge of the unit square. The function u is viewed as a boundary control that can be varied to minimize the objective

$$f(y,u) = \int_0^1 \left[\frac{\partial y(\eta,\zeta)}{\partial \eta}\Big|_{\eta=1} - 4 - \cos(2\pi\zeta) \right]^2 d\zeta + \sigma \int_0^1 \left[u(\zeta)^2 + u'(\zeta)^2 \right] d\zeta.$$

In the following calculations we used $\sigma = 0.001$ and the control u is set initially to the constant $u(\zeta) = 2.2$. This value is not all that far from the fold point where solutions cease to exist due to overheating of the system.

We use a central difference discretization with the meshwidth 1/12.0 so that the resulting algebraic system involves 144 equations in as many variables. Since the nonlinearities occur only on the diagonal one can easily implement Jacobi's method to obtain the basic function $G(y,u)$. For this simple example we also coded by hand the corresponding adjoint iteration function $\bar{G}$ defined in (11) and even the somewhat more complicated right hand side of (17). The results were later confirmed using the automatic differentiation tool ADOL-C [10].

As can be seen in Fig.1 the convergence of the Jacobi method is rather slow with the common R-factor being about $(1-1/300)$. The lowest curve represents the natural logarithms of the Euclidean norm ratios $\|y_{k+1}-y_k\|/\|y_1-y_0\|$, which provide some indication of the norm ratios $\|y_k - y_*\|/\|y_0 - y_*\|$. In view of the very slow convergence this relation need certainly not be very close. Nevertheless the theory is basically confirmed with the first derivatives $\|\dot{y}_{k+1} - \dot{y}_k\|/\|\dot{y}_1 - \dot{y}_0\|$ and $\|\bar{y}_{k+1} - \bar{y}_k\|/\|\bar{y}_1 - \bar{y}_0\|$ lagging somewhat behind, and the second derivatives $\|\dot{\bar{y}}_{k+1} - \dot{\bar{y}}_k\|/\|\dot{\bar{y}}_1 - \dot{\bar{y}}_0\|$ coming in last.

On closer inspection it was found in [11] that the ratio between these derivative quantities and the original iterates confirms after an initial transition phase the asymptotic relations (16) and (18). While the adjoints $\bar{y}_k$ were defined as vectors, the direct differentiation was performed simultaneously with respect to all components of the discretized u so that the quantity $\dot{u}$ occurring in (17) was in fact the identity matrix of order 12. Consequently, $\dot{y}_k = Z_k$ and $\dot{\bar{y}}_k = \bar{Z}_k$ had also 12 times as many components as the underlying y_k and $\bar{y}_k$, which had both $n = 144$ components.

Without and with the sign switch the Hessians H_k converged for fixed u quite rapidly to the respective limits $H(1)$ and $H(-1)$ displayed in Fig.2

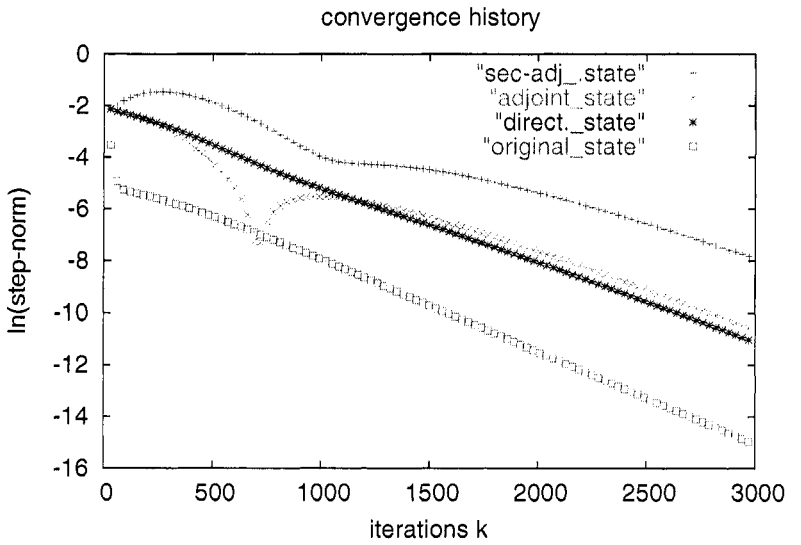


Fig. 1: Convergence pattern for fixed design $u = 2.2$

and Fig. 3. Both are Toeplitz matrices and look rather similar to discretized Laplacians in 1D. It was shown in [12] by Hazra and Schulz that the reduced Hessian must have this property exactly on the original test problem without the nonlinear term, which causes only a relatively compact perturbation. By comparison the diagonal elements in $H(-1)$ are about 50% larger than those of $H(1)$, which results in slightly shorter and thus more cautious optimization steps. Moreover, there are also structural differences as $H(1)$ is apparently an M-matrix, whereas the subdiagonals of $H(-1)$ have alternating signs and don't decline quite as rapidly as those of $H(1)$.

In any case, as soon as the optimization correction (7) is added using either approximations $H_k \approx H(1)$ or $H_k \approx H(-1)$ the convergence behavior is very different. The first, seemingly natural choice results in an immediate blow up whereas the second achieves convergence of the full steps without any line-search. The resulting values of the objective function $f(y_k, u_k)$, its Lagrange correction according to (15) and the Euclidean norm of the reduced gradient $\bar{u}_k$ are displayed in Fig. 4. As one can see the optimum is reached with about 6 Figure accuracy after some 1600 iterations, which is roughly twice as many as it took for the simulation with constant design u to reach that level of accuracy. The reason for the subsequent oscillation in the reduced gradient remains as yet a little mysterious. In order to make the method converge with steps of the form $-\alpha H_*(1)^{-1}\bar{u}_k$ we had to use a step multiplier α smaller than 0.04. On the other hand we could only use steps $-\alpha H_*(-1)^{-1}\bar{u}_k$ with α up

$$\begin{bmatrix}
34.90 & -27.62 & -1.11 & -0.38 & -0.15 & -0.07 & -0.06 & -0.07 & -0.15 & -0.38 & -1.11 & -3.62 \\
-27.62 & 58.90 & -27.62 & -1.11 & -0.38 & -0.15 & -0.07 & -0.06 & -0.07 & -0.15 & -0.38 & -1.11 \\
-1.11 & -27.62 & 58.90 & -27.62 & -1.11 & -0.38 & -0.15 & -0.07 & -0.06 & -0.07 & -0.15 & -0.38 \\
-0.38 & -1.11 & -27.62 & 58.90 & -27.62 & -1.11 & -0.38 & -0.15 & -0.07 & -0.06 & -0.07 & -0.15 \\
-0.15 & -0.38 & -1.11 & -27.62 & 58.90 & -27.62 & -1.11 & -0.38 & -0.15 & -0.07 & -0.06 & -0.07 \\
-0.07 & -0.15 & -0.38 & -1.11 & -27.62 & 58.90 & -27.62 & -1.11 & -0.38 & -0.15 & -0.07 & -0.06 \\
-0.06 & -0.07 & -0.15 & -0.38 & -1.11 & -27.62 & 58.90 & -27.62 & -1.11 & -0.38 & -0.15 & -0.07 \\
-0.07 & -0.06 & -0.07 & -0.15 & -0.38 & -1.11 & -27.62 & 58.90 & -27.62 & -1.11 & -0.38 & -0.15 \\
-0.15 & -0.07 & -0.06 & -0.07 & -0.15 & -0.38 & -1.11 & -27.62 & 58.90 & -27.62 & -1.11 & -0.38 \\
-0.38 & -0.15 & -0.07 & -0.06 & -0.07 & -0.15 & -0.38 & -1.11 & -27.62 & 58.90 & -27.62 & -1.11 \\
-1.11 & -0.38 & -0.15 & -0.07 & -0.06 & -0.07 & -0.15 & -0.38 & -1.11 & -27.62 & 58.90 & -27.62 \\
-3.62 & -1.11 & -0.38 & -0.15 & -0.07 & -0.06 & -0.07 & -0.15 & -0.38 & -1.11 & -27.62 & 34.90
\end{bmatrix}$$

Fig. 2: The reduced Hessian $H_*(1)$

$$\begin{bmatrix}
69.81 & -33.54 & 4.82 & -2.81 & 1.88 & -1.46 & 1.34 & -1.46 & 1.88 & -2.81 & 4.82 & -9.54 \\
-33.54 & 93.81 & -33.54 & 4.82 & -2.81 & 1.88 & -1.46 & 1.34 & -1.46 & 1.88 & -2.81 & 4.82 \\
4.82 & -33.54 & 93.81 & -33.54 & 4.82 & -2.81 & 1.88 & -1.46 & 1.34 & -1.46 & 1.88 & -2.81 \\
-2.81 & 4.82 & -33.54 & 93.81 & -33.54 & 4.82 & -2.81 & 1.88 & -1.46 & 1.34 & -1.46 & 1.88 \\
1.88 & -2.81 & 4.82 & -33.54 & 93.81 & -33.54 & 4.82 & -2.81 & 1.88 & -1.46 & 1.34 & -1.46 \\
-1.46 & 1.88 & -2.81 & 4.82 & -33.54 & 93.81 & -33.54 & 4.82 & -2.81 & 1.88 & -1.46 & 1.34 \\
1.34 & -1.46 & 1.88 & -2.81 & 4.82 & -33.54 & 93.81 & -33.54 & 4.82 & -2.81 & 1.88 & -1.46 \\
-1.46 & 1.34 & -1.46 & 1.88 & -2.81 & 4.82 & -33.54 & 93.81 & -33.54 & 4.82 & -2.81 & 1.88 \\
1.88 & -1.46 & 1.34 & -1.46 & 1.88 & -2.81 & 4.82 & -33.54 & 93.81 & -33.54 & 4.82 & -2.81 \\
-2.81 & 1.88 & -1.46 & 1.34 & -1.46 & 1.88 & -2.81 & 4.82 & -33.54 & 93.81 & -33.54 & 4.82 \\
4.82 & -2.81 & 1.88 & -1.46 & 1.34 & -1.46 & 1.88 & -2.81 & 4.82 & -33.54 & 93.81 & -33.54 \\
-9.54 & 4.82 & -2.81 & 1.88 & -1.46 & 1.34 & -1.46 & 1.88 & -2.81 & 4.82 & -33.54 & 69.81
\end{bmatrix}$$

Fig. 3: The projected Hessian $H_*(-1)$

to about 1.3 before convergence was lost. This indicates that $-H_*(-1)^{-1}\bar{u}_k$ is fairly optimal.

6 Summary and Outlook

We have considered the problem of minimizing an objective function subject to a very large dimensional state equation with a separate design space. After reviewing some reduced SQP variants we considered one-shot methods based on a rather slowly converging state equation solver. This approach promises a convenient transition from simulation to optimization at a limited increase in computational cost. Based on the convergence analysis in [9] for fixed design, we addressed the task of selecting a suitable preconditioner H_* for the approximate reduced gradient to allow an adjustment of the design variable

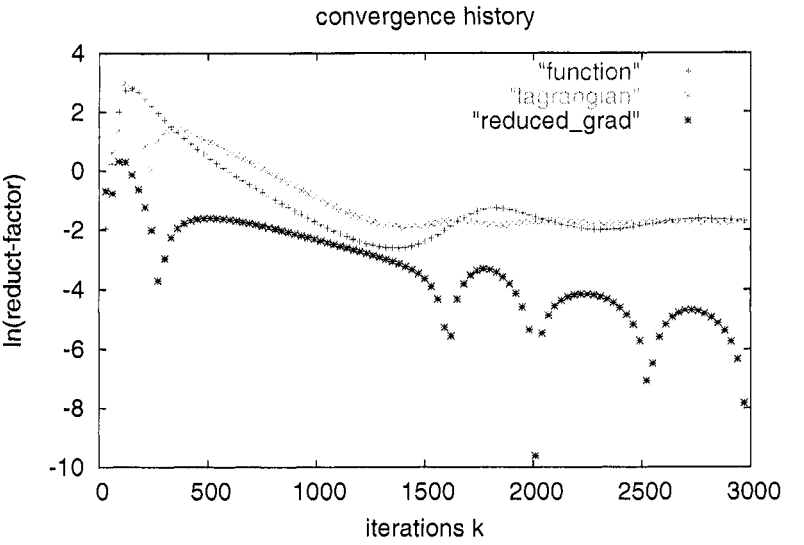


Fig. 4: One-step one-shot convergence with variable design

at each iteration. This was referred to as one-step one-shot design. Our eigenvalue analysis yielded certain necessary convergence conditions, which are not necessarily satisfied by the conventional reduced Hessian $H_*(1)$. A more suitable choice seems to be another projection $H_*(-1)$, though there is certainly no guarantee even of local convergence either.

In any case our current approach of evaluating either one of the projected Hessians as second order vector adjoints is certainly too expensive. To avoid a cost that grows proportional to the number of design variables we will pursue a low rank updating approach. Another possible improvement concerns the spacing of optimization steps. There is some indication that normal and adjoint iterations should be repeated at least twice before each correction on the design variables. That would turn the alternating modes of the state space Jacobian into monotone modes and thus possibly favor the reduced Hessian as preconditioner. However, this simple remedy need not work if there are complex eigenvalues as one should certainly expect in general. In general one would like to obtain a relation between the contractivity factor ρ of the given simulation code and the best convergence rate on can achieve for a multi-step optimization scheme of the most general form (21). For that one can not simply ignore complex eigenvalues as we have done so far. Of course, there are many unresolved nonlocal issue, chiefly how to force convergence and how to deal with inequalities.

References

1. Biegler, L.T., Nocedal, J., Schmid, C.: A reduced Hessian method for large-scale constrained optimization. SIAM J. on Optimization, **5**, 314-347 (1995).
2. Biros, G., Ghattas, O.: A Lagrange-Newton-Krylov-Schur Method for PDE-Constrained Optimization. SIAG/OPT Views-and-News, **11**(2), 1-6 (2000).
3. Coleman, T.F., Conn, A.R.: On the local convergence of a quasi-Newton method for the nonlinear programming problem. SIAM J. Num. Anal., **21**, 755-769 (1984).
4. Courty, F.: Optimization différentiable en mécanique des fluides numérique, Ph.D. Thesis, Université Paris-Sud (2003).
5. Deuflhard, P. , Potra, F.: Asymptotic mesh independence of Newton–Galerkin methods via a refined Mysovskii theorem. SIAM Journal on Numerical Analysis, **29**(5), 1395–1412 (1992).
6. Gilbert. J.Ch.: Automatic Differentiation and Iterative Processes. Optim. Methods Softw., **1**, 13–21 (1992).
7. Griewank, A: Evaluating Derivatives, Principles and Techniques of Algorithmic Differentiation. SIAM, Philadelphia, **19** (2000).
8. Griewank, A.: A mathematical view of automatic differentiation. Acta Numerica, **12**, 312–398 (2003).
9. Griewank, A., Faure, C.: Reduced functions, gradients and Hessian from fixed point iteration for state equations. Numerical Algorithms, **30**(2), 113–139 (2002).
10. Griewank, A., Juedes, D., Utke, J.: ADOL-C, a package for the automatic differentiation of algorithms written in C/C++ . TOMS, **22**(2), 131–167 (1996).
11. Griewank, A., Ponomarenko, A.: Time-lag in derivative convergence for fixed point iterations. Proceedings of CARI'04, 7th African Conference on Research in Computer Science, 295–304 (2004).
12. Hazra, S.B., Schulz, V.: Simultaneous pseudo-timestepping for PDE-model based optimization problems. Bit Numerical Math., **44**(3), 457-472 (2004).
13. Heinkenschloss, M., Vicente, L.N.: An interface between optimization and application for the numerical solution of optimal control problems. ACM Transactions on Math. Software, **25**(2), 157-190 (1999).
14. Mohammadi, B., Pironneau, O.: Applied Shape Optimization for Fluids. Numerical Mathematics and Scientific Computation, Cladenen Press, Oxford (2001).
15. Ortega, J.M., and Rheinboldt, W.C.: Iterative Solution of Nonlinear Equations in Several Variables. Academic Press, New York (1979).
16. Ta'asan, S., Kuruvila, G., Salas, M.D.: Aerodynamic design and optimization in one shot. In: 30th AIAA Aerospace Sciences Meeting and Exhibit, American Institute of Aeronautics and Astronautics, Reno, Paper 91-0025 (1992).

Conditions and parametric representations of approximate minimal elements of a set through scalarization*

César Gutiérrez[1], Bienvenido Jiménez[2], and Vicente Novo[3]

[1] Departamento de Matemática Aplicada, E.T.S.I. Informática, Universidad de Valladolid, Edificio de Tecnologías de la Información y las Telecomunicaciones, Campus Miguel Delibes, s/n, 47011 Valladolid, Spain (cesargv@mat.uva.es)
[2] Departamento de Economía e Historia Económica, Facultad de Economía y Empresa, Universidad de Salamanca, Campus Miguel de Unamuno, s/n, 37007 Salamanca, Spain (bjimen1@encina.pntic.mec.es)
[3] Departamento de Matemática Aplicada, E.T.S.I. Industriales, Universidad Nacional de Educación a Distancia, c/ Juan del Rosal, 12, Ciudad Universitaria, 28040 Madrid, Spain (vnovo@ind.uned.es)

Summary. This work deals with approximate solutions in vector optimization problems. These solutions frequently appear when an iterative algorithm is used to solve a vector optimization problem. We consider a concept of approximate efficiency introduced by Kutateladze and widely used in the literature to study this kind of solutions. Working in the objective space, necessary and sufficient conditions for Kutateladze's approximate elements of the image set are given through scalarization in such a way that these points are approximate solutions for a scalar optimization problem. To obtain sufficient conditions we use monotone functions. A new concept is then introduced to describe the idea of parametric representation of the approximate efficient set. Finally, through scalarization, characterizations and parametric representations for the set of approximate solutions in convex and nonconvex vector optimization problems are proved.

Key words: ε-efficiency, approximate solution, parametric representation, monotone function.

1 Introduction

In this paper, we deal with the following vector optimization problem:

$$\min\{f(x) : x \in S\}, \tag{VP}$$

* This research was partially supported by Ministerio de Ciencia y Tecnología (Spain), project BFM2003-02194.

where $f : X \to \mathbb{R}^p$, X is a normed space, $S \subset X$, $S \neq \emptyset$ and $\mathbb{R}^p$ is ordered via Pareto order (we denote the nonnegative orthant of $\mathbb{R}^p$ by $\mathbb{R}^p_+$):

$$y_1, y_2 \in \mathbb{R}^p,\ y_1 \leq y_2 \iff y_1 - y_2 \in -\mathbb{R}^p_+.$$

Usually, to solve (VP), i.e., to find the set of efficient points

$$\mathrm{MIN}(f, S) := \{x \in S : (f(x) - \mathbb{R}^p_+ \backslash \{0\}) \cap f(S) = \emptyset\},$$

or the set of weak efficient points (we denote the interior of a set A by $\mathrm{int}(A)$)

$$\mathrm{WMIN}(f, S) := \{x \in S : (f(x) - \mathrm{int}(\mathbb{R}^p_+)) \cap f(S) = \emptyset\},$$

a scalarization procedure and an iterative algorithm are used. Scalarization methods transform (VP) into a scalar optimization problem

$$\min\{(\varphi \circ f)(x) : x \in S\}, \tag{SP}$$

with $\varphi : \mathbb{R}^p \to \mathbb{R}$, in such a way that solutions of (SP) are efficient points of (VP). However, when an iterative algorithm is used to solve (SP), the following approximate solutions are just obtained:

$$\mathrm{ASMIN}(\varphi \circ f, S, \varepsilon) = \{x_0 \in S : \varphi(f(x_0)) - \varepsilon \leq \varphi(f(x)), \forall x \in S\},$$

where $\varepsilon \geq 0$ gives the precision achieved.

To analyze these questions, we introduce a notion of parametric representation for the set of approximate solutions of (VP). With this notion, we extend the usual concept of parametric representation (see [Wie86, Section 2] for the definition) from the set of efficient points to the set of approximate efficient points.

The following concept of approximate efficiency due to Kutateladze [Kut79] is considered to define the notion of approximate solution of a vector optimization problem.

Definition 1

(*a*) *Let $\varepsilon \geq 0$ and $q \in \mathbb{R}^p_+ \backslash \{0\}$. It is said that a point $x \in S$ is a Pareto εq-efficient solution of (VP) if*

$$(f(x) - \varepsilon q - \mathbb{R}^p_+ \backslash \{0\}) \cap f(S) = \emptyset.$$

(*b*) *Given $\varepsilon \geq 0$ and $q \in \mathbb{R}^p_+ \backslash \{0\}$, it is said that a point $x \in S$ is a weak Pareto εq-efficient solution of (VP) if*

$$(f(x) - \varepsilon q - \mathrm{int}(\mathbb{R}^p_+)) \cap f(S) = \emptyset.$$

We denote the set of Pareto εq-efficient solutions of (VP) by $\mathrm{AMIN}(f, S, \varepsilon)$ and the set of weak Pareto εq-efficient solutions of (VP) by $\mathrm{WAMIN}(f, S, \varepsilon)$. It is

clear that $\mathrm{AMIN}(f,S,\varepsilon) \subset \mathrm{WAMIN}(f,S,\varepsilon)$, $\forall \varepsilon \geq 0$. Moreover, if $q \in \mathrm{int}(\mathbb{R}^p_+)$ then

$$\mathrm{WAMIN}(f,S,\varepsilon) \subset \mathrm{AMIN}(f,S,\nu), \quad \forall \nu > \varepsilon \geq 0. \tag{1}$$

The work is structured as follows: in Section 2, necessary and sufficient conditions for approximate elements of a set, in the sense of Kutateladze, are obtained through scalarization. After that, in Section 3, several characterizations for Kutateladze's approximate solutions of (VP) are deduced. Moreover, a new concept of parametric representation is introduced and, using the previous necessary and sufficient conditions, some parametric representations are obtained for the set of approximate efficient solutions of (VP). Finally, in Section 4, some conclusions that summarize this work are presented.

2 Necessary and sufficient conditions

Let us redefine $\mathrm{MIN}(f,S)$, $\mathrm{WMIN}(f,S)$, $\mathrm{AMIN}(f,S,\varepsilon)$ and $\mathrm{WAMIN}(f,S,\varepsilon)$ to consider efficient, weak efficient, εq-efficient and weak εq-efficient elements of a set $K \subset \mathbb{R}^p$.

$$\mathrm{E}(K) := \{y \in K : (y - \mathbb{R}^p_+ \backslash \{0\}) \cap K = \emptyset\},$$

$$\mathrm{WE}(K) := \{y \in K : (y - \mathrm{int}(\mathbb{R}^p_+)) \cap K = \emptyset\}.$$

Definition 2

(*a*) *Consider $\varepsilon \geq 0$ and $q \in \mathbb{R}^p_+ \backslash \{0\}$. A point $y \in K$ is called a Pareto εq-efficient element of K if*

$$(y - \varepsilon q - \mathbb{R}^p_+ \backslash \{0\}) \cap K = \emptyset.$$

(*b*) *Given $\varepsilon \geq 0$ and $q \in \mathbb{R}^p_+ \backslash \{0\}$, it is said that a point $y \in K$ is a weak Pareto εq-efficient element of K if*

$$(y - \varepsilon q - \mathrm{int}(\mathbb{R}^p_+)) \cap K = \emptyset.$$

We denote the set of Pareto εq-efficient points of K by $\mathrm{AE}(K,\varepsilon)$ and the set of weak Pareto εq-efficient points of K by $\mathrm{WAE}(K,\varepsilon)$. It is clear that $f^{-1}(\mathrm{AE}(f(S),\varepsilon)) \cap S = \mathrm{AMIN}(f,S,\varepsilon)$ and $f^{-1}(\mathrm{WAE}(f(S),\varepsilon)) \cap S = \mathrm{WAMIN}(f,S,\varepsilon)$. Moreover, $\mathrm{AE}(K,\varepsilon) \subset \mathrm{WAE}(K,\varepsilon)$, $\mathrm{AE}(K,0) = \mathrm{E}(K)$ and $\mathrm{WAE}(K,0) = \mathrm{WE}(K)$. In [HP94, Section 3.1] and [GJN05b] important properties of the set $\mathrm{AE}(K,\varepsilon)$, considered as an approximation to $\mathrm{E}(K)$, are analyzed. One of them is related with the limit behavior of the set-valued map $F : (0,\infty) \rightrightarrows X$ defined by $F(\varepsilon) = \mathrm{AE}(K,\varepsilon)$ when $\varepsilon \to 0$:

$$\lim_{\varepsilon \to 0} \mathrm{AE}(K,\varepsilon) = \bigcap_{\varepsilon > 0} \mathrm{AE}(K,\varepsilon) = \mathrm{WE}(K). \tag{2}$$

For more details we refer the reader to [HP94] and [AF90, Sections 1.3 and 1.4].

In this section, our objective is to describe the sets $\mathrm{AE}(K,\varepsilon)$ and $\mathrm{WAE}(K,\varepsilon)$ through approximate solutions of several scalar optimization problems, i.e., to relate the points of $\mathrm{AE}(K,\varepsilon)$ and $\mathrm{WAE}(K,\varepsilon)$ with elements of the set

$$\mathrm{ASE}(K,\varphi,\varepsilon)=\{y_0\in K:\varphi(y_0)-\varepsilon\leq\varphi(y),\forall\, y\in K\}.$$

Notice that $\mathrm{ASMIN}(\varphi\circ f,S,\varepsilon)=f^{-1}(\mathrm{ASE}(f(S),\varphi,\varepsilon))\cap S$.

As it is usual in the study on efficient points of a set via scalarization, sufficient conditions to Pareto εq-efficiency are obtained through points in $\mathrm{ASE}(K,\varphi,\varepsilon)$, when the function φ is monotone (see for instance [Tam94, Theorem 2 and Corollary 1(2)]).

Definition 3 *Let us consider* $\varphi:\mathbb{R}^p\to\mathbb{R}$.

(*a*) φ *is monotone if for every* $y\in\mathbb{R}^p$,

$$\varphi(z)\leq\varphi(y),\ \forall\, z\in y-\mathbb{R}^p_+.$$

(*b*) φ *is strongly monotone if for every* $y\in\mathbb{R}^p$,

$$\varphi(z)<\varphi(y),\ \forall\, z\in y-\mathbb{R}^p_+\backslash\{0\}$$

(*c*) φ *is strictly monotone if for every* $y\in\mathbb{R}^p$,

$$\varphi(z)<\varphi(y),\ \forall\, z\in y-\mathrm{int}(\mathbb{R}^p_+).$$

It is clear that (*b*) implies (*a*), (*b*) implies (*c*), and if φ is continuous then (*c*) implies (*a*). Next, we give several examples of these notions. We denote the components of a point $y\in\mathbb{R}^p$ by $(y_1,y_2,\ldots,y_p)$ and the usual scalar product in $\mathbb{R}^p$ by $\langle\cdot,\cdot\rangle$.

Example 1

(*a*) A linear function $\varphi(y)=\langle\lambda,y\rangle$ is continuous strictly monotone (and consequently monotone) if $\lambda\in\mathbb{R}^p_+\backslash\{0\}$ and it is strongly monotone if $\lambda\in\mathrm{int}(\mathbb{R}^p_+)$.

(*b*) The max type function $\varphi(y)=\max\limits_{1\leq i\leq p}\{v_i(y_i-z_i)\}$, where $z\in\mathbb{R}^p$, is monotone if $v\in\mathbb{R}^p_+$ and it is continuous strictly monotone (and consequently monotone) if $v\in\mathrm{int}(\mathbb{R}^p_+)$.

Next, we show under mild conditions that if φ is a monotone (or strongly monotone) function, then some points in $\mathrm{ASE}(K,\varphi,\delta)$ are Pareto εq-efficient elements of K and the error ε depends on the precision δ. Similarly, we see that if φ is a strictly monotone function, then some elements of $\mathrm{ASE}(K,\varphi,\delta)$ are weak Pareto εq-efficient points of K.

For a function $\varphi:\mathbb{R}^p\to\mathbb{R}$ and errors $\varepsilon,\ \delta\geq 0$ we consider the following sets:

$$\mathrm{S}_1(\varphi,\varepsilon,\delta)=\{y\in\mathbb{R}^p:\varphi(y)-\varphi(y-\varepsilon q)>\delta\},$$
$$\mathrm{S}_2(\varphi,\varepsilon,\delta)=\{y\in\mathbb{R}^p:\varphi(y)-\varphi(y-\varepsilon q)\geq\delta\}.$$

Theorem 1 *Let us consider* $\varepsilon, \delta \geq 0$ *and* $q \in \mathbb{R}^p_+ \backslash \{0\}$.

(*a*) *If* φ *is monotone then*

$$\mathrm{S}_1(\varphi, \varepsilon, \delta) \cap \mathrm{ASE}(K, \varphi, \delta) \subset \mathrm{AE}(K, \varepsilon).$$

(*b*) *If* φ *is strongly monotone then*

$$\mathrm{S}_2(\varphi, \varepsilon, \delta) \cap \mathrm{ASE}(K, \varphi, \delta) \subset \mathrm{AE}(K, \varepsilon).$$

(*c*) *If* φ *is strictly monotone then*

$$\mathrm{S}_2(\varphi, \varepsilon, \delta) \cap \mathrm{ASE}(K, \varphi, \delta) \subset \mathrm{WAE}(K, \varepsilon).$$

Proof. Part (*a*). Suppose that the result is false. Then, there exists a point $y_0 \in \mathrm{S}_1(\varphi, \varepsilon, \delta) \cap \mathrm{ASE}(K, \varphi, \delta)$ such that $y_0 \notin \mathrm{AE}(K, \varepsilon)$. Consequently, there exists $y \in K$ such that $y \in y_0 - \varepsilon q - \mathbb{R}^p_+ \backslash \{0\}$. As φ is monotone we have that

$$\varphi(y) \leq \varphi(y_0 - \varepsilon q). \tag{3}$$

Since $y_0 \in \mathrm{ASE}(K, \varphi, \delta)$ and $y \in K$, we deduce that

$$\varphi(y_0) - \delta \leq \varphi(y) \tag{4}$$

and, from (3), it follows that

$$\varphi(y_0) - \varphi(y_0 - \varepsilon q) \leq \delta,$$

contrary to $y_0 \in \mathrm{S}_1(\varphi, \varepsilon, \delta)$.

The proofs of parts (*b*) and (*c*) are similar. In (*b*), since φ is strongly monotone, we have that $\varphi(y) < \varphi(y_0 - \varepsilon q)$ in equation (3). Therefore, (4) shows that

$$\varphi(y_0) - \varphi(y_0 - \varepsilon q) < \delta,$$

contrary to $y_0 \in \mathrm{S}_2(\varphi, \varepsilon, \delta)$. In part (*c*), if there exists a point $y_0 \in \mathrm{S}_2(\varphi, \varepsilon, \delta) \cap \mathrm{ASE}(K, \varphi, \delta)$ such that $y_0 \notin \mathrm{WAE}(K, \varepsilon)$ then there exists $y \in K$ such that $y \in y_0 - \varepsilon q - \mathrm{int}(\mathbb{R}^p_+)$. From here, the proof follows as in part (b) since φ is strictly monotone. □

Theorem 1(*b*)-(*c*) extends for example the results in [Wie86, first part of Theorem 9] and Corollary 1 of this paper considering as set K the image set $f(S)$ of (VP). These results are obtained by taking $\varepsilon = 0$ in Theorem 1(*b*)-(*c*). Moreover, several usual sufficient conditions for εq-efficient and weak εq-efficient solutions obtained through approximate solutions of scalarizations based on linear or max functions are generalized (see for instance Loridan [Lor84, Proposition 3.2(ii)], White [Whi86, Lemma 3.2], Deng [Den97, Theorem 2.1], Li and Wang [LW98, Theorems 4 and 5], Liu [Liu99, Theorem 1], Dutta and Vetrivel [DV01, Theorem 2.1] and Gutiérrez, Jiménez and Novo [GJN05a, Lemma 3.1]).

Necessary conditions for weak Pareto εq-efficient elements are now proved through scalarizations given by linear and max type functions. With max type functions, we obtain necessary conditions for weak Pareto εq-efficient points of a general set, and with linear functions we obtain necessary conditions for weak Pareto εq-efficient elements of $\mathbb{R}^p_+$-convex sets (a set K is $\mathbb{R}^p_+$-convex when $K+\mathbb{R}^p_+$ is a convex set).

Let us consider $q \in \operatorname{int}(\mathbb{R}^p_+)$. For every $y \in K$ we denote

$$\varphi_y(z) = \max_{1\le i\le p}\{(z_i - y_i)/q_i\}.$$

Theorem 2

(*a*) $\operatorname{WAE}(K,\varepsilon) \subset \bigcup_{y\in K} \operatorname{ASE}(K,\varphi_y,\varepsilon)$.

(*b*) *If K is a $\mathbb{R}^p_+$-convex set and $q \in \mathbb{R}^p_+\backslash\{0\}$, then*

$$\operatorname{WAE}(K,\varepsilon) \subset \bigcup_{\lambda\in\mathbb{R}^p_+,\|\lambda\|=1} \operatorname{ASE}(K,\langle\lambda,\cdot\rangle,\varepsilon\langle\lambda,q\rangle).$$

Proof. Part (*a*). We prove that

$$y \in \operatorname{WAE}(K,\varepsilon) \Rightarrow y \in \operatorname{ASE}(K,\varphi_y,\varepsilon).$$

Indeed, $\varphi_y(y) = \max_{1\le i\le p}\{(y_i - y_i)/q_i\} = 0$ and $\varphi_y(z) \ge -\varepsilon$, $\forall\, z \in K$, since if there exists a point $z \in K$ such that

$$\varphi_y(z) = \max_{1\le i\le p}\{(z_i - y_i)/q_i\} < -\varepsilon$$

then $z_i < y_i - \varepsilon q_i$, $\forall\, i = 1,2,\ldots,p$ and it follows that $z \in y - \varepsilon q - \operatorname{int}(\mathbb{R}^p_+)$, which is a contradiction since $y \in \operatorname{WAE}(K,\varepsilon)$.

Part (*b*). Let us consider $y_0 \in \operatorname{WAE}(K,\varepsilon)$. As $\operatorname{int}(\mathbb{R}^p_+) + \mathbb{R}^p_+ = \operatorname{int}(\mathbb{R}^p_+)$ we have that

$$(y_0 - \varepsilon q - \operatorname{int}(\mathbb{R}^p_+)) \cap (K + \mathbb{R}^p_+) = \emptyset,$$

where $K+\mathbb{R}^p_+$ and $y_0 - \varepsilon q - \operatorname{int}(\mathbb{R}^p_+)$ are convex sets. Then, by the separation theorem (see for instance [BSS93, Corollary 1, pg. 50]), we deduce that there exists $\lambda \in \mathbb{R}^p\backslash\{0\}$ such that

$$\langle\lambda, y_0 - \varepsilon q - d_1\rangle \le \langle\lambda, y + d_2\rangle, \quad \forall\, d_1, d_2 \in \mathbb{R}^p_+, \forall\, y \in K. \tag{5}$$

We can assume that $\|\lambda\| = 1$ (considering $\lambda/\|\lambda\|$ if it is necessary), since $\lambda \neq 0$. As $\mathbb{R}^p_+$ is a cone and $\langle\lambda,\cdot\rangle$ is a linear function, taking $y = y_0$ and $d_2 = 0$ in (5) we obtain

$$-\varepsilon\langle\lambda,q\rangle \le \alpha\langle\lambda,d_1\rangle, \quad \forall\, d_1 \in \mathbb{R}^p_+, \forall\, \alpha > 0.$$

Therefore, $\langle\lambda,d_1\rangle \ge 0, \forall\, d_1 \in \mathbb{R}^p_+$ and then $\lambda \in \mathbb{R}^p_+$. Finally, taking $d_1 = d_2 = 0$ in (5) we have that

$$\langle \lambda, y_0 \rangle - \varepsilon \langle \lambda, q \rangle = \langle \lambda, y_0 - \varepsilon q \rangle \leq \langle \lambda, y \rangle, \quad \forall\, y \in K$$

and so $y_0 \in \mathrm{ASE}(K, \langle \lambda, \cdot \rangle, \varepsilon \langle \lambda, q \rangle)$. □

From Theorem 2(a), an approximate nonconvex separation between the set K and some points $y_0 \in \mathrm{int}(K)$ can be obtained. When $\varepsilon = 0$ this result becomes the usual nonconvex separation theorem between a set K and points $y_0 \notin K + \mathrm{int}(\mathbb{R}^p_+)$ (see for instance [Wie86, Theorem 10 applied to function (32) in this reference]). Notice that Theorem 2(a) is direct, in the following sense: we know a priori which test function gives a necessary condition to check if a previously fixed point is εq-efficient.

In Theorem 2(b) we have achieved a nondirect approximate separation between a $\mathbb{R}^p_+$-convex set K and some points $y_0 \in \mathrm{int}(K)$ by means of linear functions and using the classical convex separation theorem (see for instance [BSS93, Corollary 1, pg. 50]). This classical result can be obtained from Theorem 2(b) taking $\varepsilon = 0$.

3 Characterizations and parametric representations

With the above necessary and sufficient conditions proved in the image space we now describe the sets of εq-efficient and weak εq-efficient solutions of (VP) using approximate solutions of several scalarizations. This description is done through a new notion of parametric representation that we introduce as follows.

Let $\{\varphi_\alpha\}_{\alpha \in \mathcal{P}}$ be a family of scalar functions $\varphi_\alpha : \mathbb{R}^p \to \mathbb{R}$, where $\mathcal{P}$ is a parametric index set, and let $G : \mathcal{P} \rightrightarrows X$ be a set-valued map. Consider the following two properties:

(P1) There exists $c_s > 0$ such that

$$\bigcup_{\alpha \in \mathcal{P}} \mathrm{ASMIN}(\varphi_\alpha \circ f, S \cap G(\alpha), \delta) \subset \mathrm{AMIN}(f, S, c_s \delta), \quad \forall\, \delta > 0. \tag{6}$$

(P2) There exists $c_n > 0$ such that

$$\mathrm{AMIN}(f, S, \varepsilon) \subset \bigcup_{\alpha \in \mathcal{P}} \mathrm{ASMIN}(\varphi_\alpha \circ f, S \cap G(\alpha), c_n \varepsilon), \quad \forall\, \varepsilon > 0. \tag{7}$$

Definition 4 *We say that* $\{\varphi_\alpha\}_{\alpha \in \mathcal{P}}$ *and* G *give a parametric representation of* $\mathrm{AMIN}(f, S, \varepsilon)$ *if properties* (P1) *and* (P2) *hold.*

The notion of parametric representation has been used in the literature to describe the efficiency set of vector optimization problems as (VP) (see for instance [Wie86, Section 2]). With Definition 4 we extend this concept to approximate solutions. Notice that the conditions considered in [Wie86, Section 2] to define the concept of parametric representation of $\mathrm{MIN}(f, S)$ are obtained from Definition 4 taking $\delta = 0$ and $\varepsilon = 0$ in (6) and (7), respectively.

In the following definition we introduce a similar concept for the set of weak Pareto εq-efficient solutions of (VP).

Definition 5 *We say that* $\{\varphi_\alpha\}_{\alpha\in\mathcal{P}}$ *and* G *give a parametric representation of* WAMIN(f,S,ε) *if properties* (P1) *and* (P2) *hold replacing* AMIN$(f,S,c_s\delta)$ *by* WAMIN$(f,S,c_s\delta)$ *in* (P1) *and* AMIN(f,S,ε) *by* WAMIN(f,S,ε) *in* (P2).

Remark 1

(*a*) $G(\alpha)$ represents a possible additional constraint motivated by the scalarization φ_α.

(*b*) Conditions (P1) and (P2) ensure that all εq-efficient solutions, and only this kind of solutions, can be obtained by solving the scalar optimization problems given by the objective functions $\varphi_\alpha \circ f$, $\alpha \in \mathcal{P}$. Moreover, (P1) ensures that an improved εq-efficient solution is obtained when the scalar objective decreases, since $c_s\delta \to 0$ if $\delta \to 0$, and this improvement depends on the properties of the set AMIN(f,S,ε) when $\varepsilon \to 0$ (see for instance property (2)).

(*c*) It is clear that (P1) and (P2), respectively, imply sufficient and necessary conditions. However, the concept of parametric representation is stronger than the notion of characterization because the former ensures that sufficient conditions do not depend on a fixed point, that is to say, all approximate solutions in ASMIN$(\varphi_\alpha \circ f, S \cap G(\alpha), \delta)$ are $c_s\delta q$-efficient solutions of (VP).

By Remark 1(*b*)-(*c*) we conclude that Definitions 4 and 5 give an useful framework to analyze approximate solutions of scalarizations obtained via iterative algorithms.

Next, we deduce several parametric representations.

Theorem 3 *Let us consider* $q \in \operatorname{int}(\mathbb{R}^p_+)$.

(*a*) *The family* $\{\varphi_{f(x)}\}_{x\in S}$, *where*

$$\varphi_{f(x)}(y) = \max\{(y_i - f_i(x))/q_i\},$$

gives a parametric representation of WAMIN(f,S,ε) *and* AMIN(f,S,ε) *with* $c_s = c_n = 1$ *and* $c_s > c_n = 1$, *respectively.*

(*b*) *If* f *is a convex function and* S *is a convex set then the family* $\{\langle\lambda,\cdot\rangle\}_{\lambda\in\mathcal{P}}$, *where*

$$\mathcal{P} = \{\lambda \in \mathbb{R}^p_+ : \|\lambda\| = 1\},$$

gives a parametric representation of WAMIN(f,S,ε) *and* AMIN(f,S,ε) *with* $c_s = k_s := 1/\min\{\langle\lambda,q\rangle : \lambda \in \mathcal{P}\}$, $c_n = k_n := \max\{\langle\lambda,q\rangle : \lambda \in \mathcal{P}\}$ *and* $c_s > k_s, c_n = k_n$, *respectively.*

Proof. Part (*a*). For every $\varepsilon > 0$, by Theorem 2(*a*), it follows that

$$\operatorname{AE}(f(S),\varepsilon) \subset \operatorname{WAE}(f(S),\varepsilon) \subset \bigcup_{x\in S} \operatorname{ASE}(f(S),\varphi_{f(x)},\varepsilon). \tag{8}$$

From Example 1(*b*) we see that $\varphi_{f(x)}$ is a strictly monotone function for every $x \in S$ and consequently, by Theorem 1(*c*), we deduce that

$$S_2(\varphi_{f(x)}, \varepsilon, \delta) \cap \mathrm{ASE}(f(S), \varphi_{f(x)}, \delta) \subset \mathrm{WAE}(f(S), \varepsilon), \quad \forall x \in S. \tag{9}$$

As

$$\varphi_{f(x)}(y) - \varphi_{f(x)}(y - \varepsilon q) = \varepsilon, \quad \forall y \in \mathbb{R}^p,$$

then $S_2(\varphi_{f(x)}, \varepsilon, \varepsilon) = \mathbb{R}^p$ and by (8)-(9) we have that

$$\mathrm{AE}(f(S), \varepsilon) \subset \bigcup_{x \in S} \mathrm{ASE}(f(S), \varphi_{f(x)}, \varepsilon) = \mathrm{WAE}(f(S), \varepsilon).$$

Thus,

$$f^{-1}(\mathrm{AE}(f(S), \varepsilon)) \cap S \subset \bigcup_{x \in S} f^{-1}(\mathrm{ASE}(f(S), \varphi_{f(x)}, \varepsilon)) \cap S = f^{-1}(\mathrm{WAE}(f(S), \varepsilon)) \cap S$$

and

$$\mathrm{AMIN}(f, S, \varepsilon) \subset \bigcup_{x \in S} \mathrm{ASMIN}(\varphi_{f(x)} \circ f, S, \varepsilon)$$
$$= \mathrm{WAMIN}(f, S, \varepsilon) \subset \mathrm{AMIN}(f, S, c\varepsilon), \quad \forall c > 1, \forall \varepsilon > 0, \tag{10}$$

where (10) is a consequence of (1). From here we see that the family $\{\varphi_{f(x)}\}_{x \in S}$ and the set-valued map $G(f(x)) = X$ give a parametric representation of $\mathrm{WAMIN}(f, S, \varepsilon)$ with $c_s = c_n = 1$ and they give a parametric representation of $\mathrm{AMIN}(f, S, \varepsilon)$ for every $c_s > 1$ and $c_n = 1$.

Part (b). As f is a convex function and S is a convex set then $f(S)$ is a $\mathbb{R}^p_+$-convex set. Moreover, for every $\lambda \in \mathcal{P}$, $\langle \lambda, \cdot \rangle$ is a strictly monotone function such that

$$\langle \lambda, y \rangle - \langle \lambda, y - \varepsilon q \rangle = \varepsilon \langle \lambda, q \rangle, \quad \forall y \in \mathbb{R}^p,$$

and so $S_2(\langle \lambda, \cdot \rangle, \varepsilon, \varepsilon \langle \lambda, q \rangle) = \mathbb{R}^p$. Then, for every $\varepsilon > 0$, by Theorems $1(c)$ and $2(b)$ we have that

$$\mathrm{AE}(f(S), \varepsilon) \subset \bigcup_{\lambda \in \mathcal{P}} \mathrm{ASE}(f(S), \langle \lambda, \cdot \rangle, \varepsilon \langle \lambda, q \rangle) = \mathrm{WAE}(f(S), \varepsilon).$$

Therefore,

$$f^{-1}(\mathrm{AE}(f(S), \varepsilon)) \cap S \subset \bigcup_{\lambda \in \mathcal{P}} f^{-1}(\mathrm{ASE}(f(S), \langle \lambda, \cdot \rangle, \varepsilon \langle \lambda, q \rangle)) \cap S = f^{-1}(\mathrm{WAE}(f(S), \varepsilon)) \cap S$$

and

$$\mathrm{AMIN}(f, S, \varepsilon) \subset \bigcup_{\lambda \in \mathcal{P}} \mathrm{ASMIN}(\langle \lambda, \cdot \rangle \circ f, S, \varepsilon \langle \lambda, q \rangle) \tag{11}$$
$$= \mathrm{WAMIN}(f, S, \varepsilon) \subset \mathrm{AMIN}(f, S, c\varepsilon), \quad \forall c > 1, \forall \varepsilon > 0, \tag{12}$$

where (12) is a consequence of (1). Let us consider $k_n := \max\{\langle \lambda, q\rangle : \lambda \in \mathcal{P}\}$ and $k_s := 1/\min\{\langle \lambda, q\rangle : \lambda \in \mathcal{P}\}$. As $q \in \operatorname{int}(\mathbb{R}^p_+)$ and $\mathcal{P}$ is a compact set, by Weierstrass' Theorem it follows that $k_n, k_s \in (0, \infty)$. Then, by (11)-(12), we conclude that the family $\{\langle \lambda, \cdot\rangle\}_{\lambda \in \mathcal{P}}$ and the set-valued map $G(\lambda) = X$ give a parametric representation of $\mathrm{WAMIN}(f, S, \varepsilon)$ taking $c_n = k_n$, $c_s = k_s$ and they give a parametric representation of $\mathrm{AMIN}(f, S, \varepsilon)$ for every $c_s > k_s$ and $c_n = k_n$. □

The family $\{\varphi_{f(x)}\}_{x \in S}$ has been considered by Wierzbicki in [Wie86] to obtain a parametric representation of the efficiency set of (VP). Theorem $3(a)$ extends [Wie86, scalarization (32) and Theorem 10] from the set of weak efficient solutions to the set of weak Pareto εq-efficient solutions of (VP). Theorem $3(b)$ also extends Theorem 1 in [Wie86], that describes a parametric representation of the set of weak efficient solutions in convex vector optimization problems, to weak approximate solutions of (VP) in the sense of Kutateladze.

Example 2 Let (VP) be the vector optimization problem given by the following data: $X = \mathbb{R}^2$, $p = 2$, $f(x, y) = (x, y)$, $S = \mathbb{R}^2_+$ and $q = (1, 1)$. It follows that

$$\begin{aligned}\mathrm{WAMIN}(f, S, \varepsilon) = &\{(x, y) \in \mathbb{R}^2 : 0 \le x \le \varepsilon,\ y \ge 0\} \\ &\cup \{(x, y) \in \mathbb{R}^2 : x \ge 0,\ 0 \le y \le \varepsilon\}.\end{aligned}$$

In order to obtain a parametric representation of $\mathrm{WAMIN}(f, S, \varepsilon)$ from Theorem $3(a)$, let $(a, b) \in \mathbb{R}^2_+$ be a feasible point and consider the scalar optimization problem

$$\min\{(\varphi_{a,b} \circ f)(x, y) : x, y \ge 0\},$$

where $\varphi_{a,b}(x, y) = \max\{x - a, y - b\}$. It is easy to check that

$$\mathrm{ASMIN}(\varphi_{a,b} \circ f, S, \varepsilon)$$

$$= \begin{cases} \{(x, y) \in \mathbb{R}^2 : 0 \le x \le \varepsilon,\ 0 \le y \le \varepsilon + (b - a)\} \text{ if } b \ge a \\ \{(x, y) \in \mathbb{R}^2 : 0 \le x \le \varepsilon + (a - b),\ 0 \le y \le \varepsilon\} \text{ if } a \ge b \end{cases}$$

and so, it is clear that the set

$$\bigcup_{(a,b) \in \mathbb{R}^2_+} \mathrm{ASMIN}(\varphi_{a,b} \circ f, S, \varepsilon)$$

gives a "parametric representation" of $\mathrm{WAMIN}(f, S, \varepsilon)$.

From Theorem 3 we obtain the following characterizations for Pareto and weak Pareto εq-efficient solutions of (VP). Notice that $(a.1)(\Rightarrow)$ and $(a.2)$ are direct necessary conditions and, however, $(b.1)(\Rightarrow)$ and $(b.2)$ are not direct necessary conditions.

Corollary 1

(*a*) *Let us consider* $q \in \text{int}(\mathbb{R}^p_+)$.
  (*a*.1) $x \in \text{WAMIN}(f, S, \varepsilon) \iff x \in \text{ASMIN}(\varphi_{f(x)} \circ f, S, \varepsilon)$.
  (*a*.2) $x \in \text{AMIN}(f, S, \varepsilon) \Rightarrow x \in \text{ASMIN}(\varphi_{f(x)} \circ f, S, \varepsilon)$.
  (*a*.3) $x \in \text{ASMIN}(\varphi_{f(x)} \circ f, S, \varepsilon) \Rightarrow x \in \text{AMIN}(f, S, \nu)$, $\forall \nu > \varepsilon$.

(*b*) *Let us consider that* f *is a convex function,* S *is a convex set and* $q \in \mathbb{R}^p_+ \backslash \{0\}$. *Then*
  (*b*.1) $x \in \text{WAMIN}(f, S, \varepsilon) \iff x \in \bigcup_{\lambda \in \mathbb{R}^p_+, \|\lambda\| = 1} \text{ASMIN}(\langle \lambda, \cdot \rangle \circ f, S, \varepsilon \langle \lambda, q \rangle)$.
  (*b*.2) $x \in \text{AMIN}(f, S, \varepsilon) \Rightarrow$ *there exists* $\lambda \in \mathbb{R}^p_+$ *with* $\|\lambda\| = 1$ *such that* $x \in \text{ASMIN}(\langle \lambda, \cdot \rangle \circ f, S, \varepsilon \langle \lambda, q \rangle)$.
  (*b*.3) *If* $q \in \text{int}(\mathbb{R}^p_+)$, $\lambda \in \mathbb{R}^p_+ \backslash \{0\}$ *and* $x \in \text{ASMIN}(\langle \lambda, \cdot \rangle \circ f, S, \varepsilon) \Rightarrow x \in \text{AMIN}(f, S, \nu)$, $\forall \nu > \varepsilon / \langle \lambda, q \rangle$.

In [LW98, Theorem 1], Li and Wang have obtained a characterization for weak εq-efficient solutions of (VP) via approximate solutions of several scalarizations. In Corollary 1(*a*.1) we have proved an equivalent condition, which is simpler since it uses a single scalarization.

4 Conclusions

In this work, approximate efficient solutions of vector optimization problems have been analyzed using a concept of approximate efficiency introduced by Kutateladze in [Kut79]. We have obtained relations between these approximate solutions and approximate solutions of several scalarizations. This relations are important because the methods used to solve a vector optimization problem are usually based on scalarization processes.

Next, a notion of parametric representation for the set of approximate efficient solutions defined by Kutateladze has been introduced and two specific parametric representations have been given via linear and max type functions. Our theorems extend several previous results since we consider nonconvex problems.

Acknowledgements

The authors are grateful to the anonymous referees for their helpful comments and suggestions.

References

[AF90] Aubin, J.P., Frankowska, H.: Set-Valued Analysis. Birkhäuser, Boston (1990).

[BSS93] Bazaraa, M.S., Sheraly, H.D., Shetty, C.M.: Nonlinear Programming. Theory and Algorithms. John Wiley & Sons, New York (1993).

[Den97] Deng, S.: On approximate solutions in convex vector optimization. SIAM J. Control Optim., **35**, 2128–2136 (1997).

[DV01] Dutta, J., Vetrivel, V.: On approximate minima in vector optimization. Numer. Funct. Anal. Optim., **22**, 845–859 (2001).

[GJN05a] Gutiérrez, C., Jiménez, B., Novo, V.: Multiplier rules and saddle-point theorems for Helbig's approximate solutions in convex Pareto problems. J. Global Optim., **32**, (2005).

[GJN05b] Gutiérrez, C., Jiménez, B., Novo, V.: A property of efficient and ε-efficient solutions in vector optimization. Appl. Math. Lett., **18**, 409–414 (2005).

[HP94] Helbig, S., Pateva, D.: On several concepts for ε-efficiency. OR Spektrum, **16**, 179–186 (1994).

[Kut79] Kutateladze, S.S.: Convex ε-programming. Soviet Math. Dokl., **20**, 391–393 (1979).

[Liu99] Liu, J.C.: ε-properly efficient solutions to nondifferentiable multiobjective programming problems. Appl. Math. Lett., **12**, 109–113 (1999).

[Lor84] Loridan, P.: ε-solutions in vector minimization problems. J. Optim. Theory Appl., **43**, 265–276 (1984).

[LW98] Li, Z., Wang, S.: ε-approximate solutions in multiobjective optimization. Optimization, **44**, 161–174 (1998).

[Tam94] Tammer, Chr.: Stability results for approximately efficient solutions. OR Spektrum, **16**, 47–52 (1994).

[Whi86] White, D.J.: Epsilon efficiency. J. Optim. Theory Appl., **49**, 319–337 (1986).

[Wie86] Wierzbicki, A.P.: On the completeness and constructiveness of parametric characterizations to vector optimization problems. OR Spektrum, **8**, 73–87 (1986).

Efficient methods for large-scale unconstrained optimization

Ladislav Lukšan[1] and Jan Vlček[2]

[1] Institute of Computer Science, Academy of Sciences of the Czech Republic, Pod Vodárenskou věží 2, 182 07 Praha 8 and Technical University of Liberec, Hálkova 6, 461 17 Liberec (luksan@cs.cas.cz)
[2] Institute of Computer Science, Academy of Sciences of the Czech Republic, Pod Vodárenskou věží 2, 182 07 Praha 8 (vlcek@cs.cas.cz)

Summary. This contribution contains a description of efficient methods for large-scale unconstrained optimization. Many of them have been developed recently by the authors. It concerns limited memory methods for general smooth optimization, variable-metric bundle methods for partially separable nonsmooth optimization, hybrid methods for sparse least squares and methods for solving large-scale trust-region subproblems.

Key words: Unconstrained optimization, large-scale optimization, nonsmooth optimization, limited-memory methods, bundle-type methods, variable metric methods, nonlinear least squares, nonlinear minimax optimization, trust-region subproblems, computational experiments.

1 Introduction

Modern numerical methods for unconstrained optimization have been studied and developed since the sixties of the last century. Nevertheless, many new problems and approaches have appeared only recently. It especially concerns general large-scale problems, which challenged the development of limited-memory variable metric methods [Noc80], and structured large-scale problems, which stimulated the development of variable metric methods for partially separable problems [GT82] and hybrid methods for sparse least-square problems [Luk96b]. Additional approaches arose in connection with nonsmooth unconstrained optimization. In this case, various bundle-type methods [Kiw85], [Lem89], [MN92] were developed including variable-metric bundle methods [LV99], [VL01], which substantially reduce the size of bundles and, therefore, the number of constraints in the quadratic programming subproblems. Variable-metric bundle methods were recently generalized to solve

large-scale nonsmooth problems using a limited-memory variable metric approach [HMM04], [HMM03] or a partially-separable variable metric framework [LV04]. Furthermore, new methods [GLRT99], [LMV04] for solving large-scale trust-region subproblems were proposed, which can be used in connection with the Newton method for general sparse unconstrained optimization or with the Gauss-Newton method for sparse nonlinear least squares.

In this contribution, we deal with the local minimization of the objective function $F : \mathcal{R}^n \to \mathcal{R}$. In Section 2, the function F is assumed to be twice continuously differentiable and limited-memory methods are reviewed, including the most recent methods proposed in [VL02] and [VL05]. Section 3 is devoted to the nonsmooth optimization. After introducing basic principles of the bundle methods and describing variable-metric bundle methods, we focus our attention on methods for large-scale nonsmooth problems. Section 4 contains a description of hybrid methods for nonlinear least squares and Section 5 is devoted to efficient methods for solving large-scale trust-region subproblems. All the methods presented were carefully tested and compared using extensive computational experiments.

2 Limited-memory variable metric methods

Limited-memory variable metric methods can be efficiently used for large-scale unconstrained optimization in case the Hessian matrix is not known or is not sparse. These methods are usually realized in the line-search framework so that they generate a sequence of points $x_k \in \mathcal{R}^n$, $k \in \mathcal{N}$, by the simple process

$$x_{k+1} = x_k + t_k d_k, \tag{1}$$

where $d_k = -H_k g_k$ is a direction vector, H_k is a positive definite approximation of the inverse Hessian matrix and $t_k > 0$ is a scalar step-size chosen in such a way that

$$F_{k+1} - F_k \leq \varepsilon_1 t_k d_k^T g_k, \quad d_k^T g_{k+1} \geq \varepsilon_2 d_k^T g_k \tag{2}$$

(the weak Wolfe conditions), where $F_k = F(x_k)$, $g_k = \nabla F(x_k)$ and $0 < \varepsilon_1 < 1/2$, $\varepsilon_1 < \varepsilon_2 < 1$. Matrices H_k, $k \in \mathcal{N}$, are computed either by using a limited (small) number of variable metric updates applied to the unit matrix or by updating low dimension matrices. First, we shortly describe two known limited-memory variable metric methods. Then we focus our attention on new shifted limited-memory variable metric methods.

2.1 Limited memory BFGS method

The most known and commonly used limited-memory BFGS (L-BFGS) method [Noc80] works with matrices $H_k = H_k^k$, where $H_{k-m}^k = \gamma_k I$ (usually $\gamma_k = b_{k-1}/a_{k-1}$) and

$$H_{j+1}^k = V_j^T H_j^k V_j + \frac{\rho_j}{b_j} s_j s_j^T, \quad V_j = I - \frac{1}{b_j} y_j s_j^T \tag{3}$$

for $k - m \le j \le k - 1$. Here $s_j = x_{j+1} - x_j$, $y_j = g_{j+1} - g_j$, $a_j = y_j^T H_j y_j$, $b_j = y_j^T s_j$ and ρ_j are correction parameters. Matrix $H_k = H_k^k$ need not be constructed explicitly since we need only vector $d_k = -H_k^k g_k$, which can be computed using two recurrences (the Strang formula). First, vectors

$$u_j = -\left(\prod_{i=j}^{k-1} V_i\right) g_k,$$

$k - 1 \ge j \ge k - m$, are computed using the backward recurrence

$$\sigma_j = s_j^T u_{j+1}/b_j,$$
$$u_j = u_{j+1} - \sigma_j y_j,$$

where $u_k = -g_k$. Then vectors

$$v_{j+1} = \frac{b_{k-1}}{a_{k-1}} \left(\prod_{i=k-m}^{j} V_i\right)^T u_{k-m} + \sum_{l=k-m}^{j} \frac{\rho_l}{b_l} \left(\prod_{i=l+1}^{j} V_i\right)^T s_l s_l^T u_{l+1},$$

$k - m \le j \le k - 1$, are computed using the forward recurrence

$$v_{j+1} = v_j + (\rho_j \sigma_j - y_j^T v_j/b_j) s_j,$$

where $v_{k-m} = (b_{k-1}/a_{k-1}) u_{k-m}$. Finally we set $d_k = v_k$. Note that $2m$ vectors s_j, y_j, $k - m \le j \le k - 1$ are used and stored.

Matrix $H_k = H_k^k$, obtained by updates (3), can be expressed in the compact form using low order matrices [BNS94]. In this case

$$H_k = \gamma_k I - [S_k, \gamma_k Y_k] M_k [S_k, \gamma_k Y_k]^T, \tag{4}$$

where $S_k = [s_{k-m}, \ldots, s_{k-1}]$, $Y_k = [y_{k-m}, \ldots, y_{k-1}]$, and

$$M_k = \begin{bmatrix} (R_k^{-1})^T (C_k + \gamma_k Y_k^T Y_k) R_k^{-1} & -(R_k^{-1})^T \\ -R_k^{-1} & 0 \end{bmatrix}, \tag{5}$$

where C_k is a diagonal matrix containing diagonal elements of $S_k^T Y_k$ and R_k is an upper triangular matrix having the same upper triangular part as $S_k^T Y_k$. Again $2m$ vectors s_j, y_j, $k - m \le j \le k - 1$ are used and stored.

The above idea can be used for some other variable metric updates. The symmetric rank-one (SR1) update can be expressed in the form

$$H_k = \gamma_k I + (S_k - \gamma_k Y_k)(R_k + R_k^T - C_k - \gamma_k Y_k^T Y_k)^{-1} (S_k - \gamma_k Y_k)^T. \tag{6}$$

It is necessary to note that update (3) with Strang recurrences is more stable than expressions (4)–(5). On the other hand, compact-form formulas are very important, since they can be easily inverted (using duality) and applied directly to $B_k = H_k^{-1}$, which is necessary in trust-region approach or in constrained optimization.

2.2 Methods based on reduced Hessian matrices

Another limited-memory variable metric method, proposed in [GL03], is based on updating reduced Hessian matrices. Let B_k, $k \in \mathcal{N}$, be approximations of Hessian matrices obtained by the BFGS method (with $B_1 = I$). If $\mathcal{G}_k$ and $\mathcal{D}_k$ are linear subspaces spanned respectively by the columns of matrices $G_k = [g_1, \ldots, g_k]$ and $D_k = [d_1, \ldots, d_k]$, then $\mathcal{D}_k = \mathcal{G}_k$. Moreover, $B_k v \in \mathcal{G}_k$ for $v \in \mathcal{G}_k$ and $B_k w = w$ for $w \in \mathcal{G}_k^{\perp}$. Let Z_k be a matrix whose columns form an orthonormal basis in $\mathcal{G}_k$ and let $Q_k = [Z_k, W_k]$ be a square orthogonal matrix. The above consideration implies that

$$Q_k^T B_k Q_k = \begin{bmatrix} Z_k^T B_k Z_k & 0 \\ 0 & I \end{bmatrix}, \quad Q_k^T g_k = \begin{bmatrix} Z_k^T g_k \\ 0 \end{bmatrix}$$

and the direction vector can be obtained from the reduced system

$$d_k = Z_k \tilde{d}_k, \quad Z_k^T B_k Z_k \tilde{d}_k = -\tilde{g}_k, \quad \tilde{g}_k = Z_k^T g_k. \tag{7}$$

Thus complete information concerning the variable metric update is contained in the reduced Hessian approximation $Z_k^T B_k Z_k$. We usually use the Choleski decomposition $R_k^T R_k = Z_k^T B_k Z_k$ and update the upper triangular matrix R_k. More details can be found in [GL01].

Consider now a limited-dimension subspace $\mathcal{D}_k$ spanned by the columns of matrix $D_k = [d_{k-m+1}, \ldots, d_k]$ (note that $\mathcal{D}_k$ and D_k were redefined). This subspace is changed on every iteration. Let Z_k be a matrix whose columns form an orthonormal basis in $\mathcal{D}_k$. In efficient implementations of limited-memory methods based on reduced Hessians, matrices Z_k and $Z_k^T B_k Z_k$ are not used explicitly. An upper triangular matrix T_k such that $D_k = Z_k T_k$ and the Choleski decomposition $R_k^T R_k = Z_k^T B_k Z_k$ are used instead. At the first iteration, we set

$$D_1 = [g_1], \quad T_1 = [\|g_1\|], \quad R_1 = [1], \quad \tilde{g}_1 = [\|g_1\|].$$

On every iteration, we first solve two equations $R_k^T R_k \tilde{d}_k = -\tilde{g}_k$, $T_k v_k = \tilde{d}_k$ and set $d_k = D_k v_k$. Then the line-search is performed to obtain a new point $x_{k+1} = x_k + t_k d_k$ and matrices D_k, T_k are changed according to the subspace $\mathcal{D}_k$. Therefore, we replace the last column of D_k by d_k and the last column of T_k by $\tilde{d}_k$. Now a representation of the subspace $\mathcal{D}_{k+1}$ has to be formed. First, we project the new gradient $g_{k+1} = g(x_{k+1})$ into the subspace $\mathcal{D}_k$ by solving the equation $T_k^T r_{k+1} = D_k^T g_{k+1}$. Then we determine the quantity $\rho_{k+1} = \|g_{k+1}\| - \|r_{k+1}\|$, set $D_{k+1} = [D_k, g_{k+1}]$ and

$$T_{k+1} = \begin{bmatrix} T_k & r_{k+1} \\ 0 & \rho_{k+1} \end{bmatrix}, \quad \tilde{g}_{k+1} = \begin{bmatrix} r_{k+1} \\ \rho_{k+1} \end{bmatrix}.$$

Thus we obtain a temporary representation of the reduced Hessian approximation in the form $Z_{k+1}^T B_k Z_{k+1} = R_{k+1}^T R_{k+1}$, where

$$R_{k+1} = \begin{bmatrix} R_k & 0 \\ 0 & \sqrt{1/\gamma_{k+1}} \end{bmatrix}$$

(γ_{k+1} is the scaling parameter). This factorization has to be updated to satisfy the quasi-Newton condition $R_{k+1}^T R_{k+1} \tilde{s}_k = \tilde{y}_k$, where

$$\tilde{s}_k = t_k \begin{bmatrix} \tilde{d}_k \\ 0 \end{bmatrix}, \quad \tilde{y}_k = \tilde{g}_{k+1} - \begin{bmatrix} \tilde{g}_k \\ 0 \end{bmatrix}.$$

Numerically stable methods described in [GMS75] can be used for this purpose. If the subspace $\mathcal{D}_{k+1}$ has dimension $m+1$, then it has to be reduced before the new iteration is started. Denote the matrices after such reduction by $\bar{D}_{k+1}$, $\bar{T}_{k+1}$, $\bar{R}_{k+1}$. Then $\bar{D}_{k+1}$ is obtained from D_{k+1} by deleting its first column and matrices $\bar{T}_{k+1}$, $\bar{R}_{k+1}$ can be constructed by using elementary Givens rotations (see [GL03] for more details).

2.3 Shifted variable metric methods

Consider line-search methods of the form (1)–(2). Limited-memory variable metric methods based on reduced Hessians use low-rank matrices $H_k = Z_k(Z_k^T B_k Z_k)^{-1} Z_k^T = U_k U_k^T$, where U_k has m columns at most. Thus H_k is singular and the case when d_k is almost perpendicular to g_k can occur. For this reason, it is advantageous to set $H_k = \zeta_k I + U_k U_k^T$, where $\zeta_k > 0$ is a parameter, which is carefully selected in every iteration. In this subsection, we assume that the rank of $A_k = U_k U_k^T$ is $\min(k, n)$ (i.e., $m = n$).

Shifted variable metric methods use matrices $H_k = \zeta_k I + A_k$, $k \in \mathcal{N}$, where $\zeta_k > 0$ and A_k is positive semidefinite. Starting from the zero matrix, these methods generate a sequence of positive semidefinite matrices A_k, $k \in \mathcal{N}$, satisfying the (modified) quasi-Newton condition $A_{k+1} y_k = \varrho_k \tilde{s}_k$, where $s_k = x_{k+1} - x_k$, $y_k = g_{k+1} - g_k$ and $\tilde{s}_k = s_k - \zeta_{k+1} y_k$. Here ϱ_k is a correction parameter and $\zeta_{k+1} > 0$ is a shift parameter. Update

$$A_{k+1} = A_k + \varrho_k \frac{\tilde{s}_k \tilde{s}_k^T}{\tilde{b}_k} - \frac{A_k y_k y_k^T A_k}{\bar{a}_k} + \frac{\eta_k}{\bar{a}_k} \left(\frac{\bar{a}_k}{\tilde{b}_k} \tilde{s}_k - A_k y_k \right) \left(\frac{\bar{a}_k}{\tilde{b}_k} \tilde{s}_k - A_k y_k \right)^T \tag{8}$$

is used, where $\bar{a}_k = y_k^T A_k y_k$ and $\tilde{b}_k = y_k^T \tilde{s}_k$. The shifted BFGS method corresponds to $\eta_k = 1$. The following theorem is proved in [VL02].

Theorem 1. Let A_k be positive semidefinite and $\eta_k \geq 0$. If $0 < \zeta_{k+1} < y_k^T s_k / y_k^T y_k$, then A_{k+1} is positive semidefinite.

A crucial part of shifted variable metric methods is the determination of the shift parameter. Theorem 1 implies condition

$$\zeta_{k+1} = \mu_k \, b_k / \hat{a}_k, \quad 0 < \mu_k < 1,$$

where $b_k = y_k^T s_k$ and $\hat{a}_k = y_k^T y_k$. If μ_k is too small, then matrix H_k is usually unsuitable (A_k is singular in the first n iterations). If μ_k is too large, the stability is usually lost (numerical explosion). Two basic choices were tested. The simplest choice uses constant $\mu_k = \mu$, $0 < \mu < 1/2$, in every iteration. If $\mu \to 1/2$, then the shifted BFGS method becomes unstable. Efficient values lie in the interval $0.20 \leq \mu \leq 0.25$, e.g., $\mu = 0.22$. A more sophisticated choice, derived by using a theoretical investigation of stability and global convergence (see [VL02]), is given by the formula

$$\mu_k = \sqrt{1 - \bar{a}_k/a_k} \Big/ \left(1 + \sqrt{1 - b_k^2/(\hat{a}_k |s_k|^2)}\right) \tag{9}$$

(the numerator assures the global convergence and the denominator assures the stability).

For proving the global convergence, we need the following assumptions.

Assumption 1. The objective function $f : R^n \to R$ is uniformly convex and has bounded second-order derivatives, i.e., there are constants $0 < \underline{G} \leq \overline{G} < \infty$ such that

$$\underline{G} \leq \underline{\lambda}(G(x)) \leq \overline{\lambda}(G(x)) \leq \overline{G}$$

for all $x \in \mathcal{R}^n$, where $\underline{\lambda}(G(x))$ and $\overline{\lambda}(G(x))$ are the lowest and the greatest eigenvalues of the Hessian matrix $G(x)$.

Assumption 2. Parameters ϱ_k and μ_k of the shifted variable metric method are uniformly positive and bounded, in the sense that

$$0 < \underline{\varrho} \leq \varrho_k \leq \overline{\varrho} < \infty,$$

$$0 < \underline{\mu} \leq \mu_k \leq \overline{\mu} < 1,$$

for every $k \geq 1$.

The following theorem is proved in [VL05].

Theorem 2. Consider a shifted variable metric method satisfying Assumption 2 with the line-search fulfilling the weak Wolfe conditions. Let the objective function satisfy Assumption 1. Then, if $0 \leq \eta_k \leq 1$ and $\mu_k^2 \leq 1 - \bar{a}_k/a_k$, one has

$$\liminf_{k \to \infty} \|g_k\| = 0.$$

Remark 1. Condition $\mu_k^2 \leq 1 - \hat{a}_k/a_k$ has been used for the choice of the numerator in (9). The denominator in (9) minimizes the condition number of H_{k+1} in the first iteration.

Shifted variable metric methods were tested by using a set of 92 relatively difficult (ill-conditioned) test problems with 50 and 200 variables implemented in subroutine `TEST28` (see `www.cs.cas.cz/~luksan/test.html`). The results are presented in Table 1, where n is the number of variables, "method" is

the method used (SBFGS - the shifted BFGS method, SDFP - the shifted DFP method, BFGS - the standard BFGS method, DFP - the standard DFP method), NIT is the total number of iterations, NEV is the total number of function and gradient evaluations, NF is the number of failures for a given set (i.e., the number of problems which were not successfully solved) and "time" is the total computational time in seconds.

n	method	NIT	NEV	NF	time
50	SBFGS	11256	12178	-	1.03
	SDFP	46010	48237	8	3.78
	BFGS	14958	16474	1	1.26
	DFP	79486	84215	35	6.66
200	SBFGS	30429	36080	1	25.11
	SDFP	92799	100461	15	74.88
	BFGS	36099	39991	2	27.21
	DFP	146851	158979	32	113.75

Table 1

The results presented in this table imply the following conclusions for the variable metric (VM) methods:

- The shifted VM methods are more efficient than standard implementations of the classic VM methods. However, the classic VM methods can be improved by a suitable scaling, which is problematic in the case of shifted VM methods.
- The shifted VM methods were not developed for solving problems that can be successfully solved by the classic VM methods. They have been discovered in the framework of shifted limited memory VM methods. Having the same form, they are ideal as starting methods (that give a suitable initial matrix U_1) for algorithms described in the next section.

2.4 Shifted limited-memory variable metric methods

Shifted limited-memory variable metric methods use recurrences (1)–(2) with matrix $H_k = \zeta_k I + A_k = \zeta_k I + U_k U_k^T$, where $n \times m$ matrix U_k is updated by formula $U_{k+1} = V_k U_k$ with a low rank matrix V_k chosen in such a way that the (modified) quasi-Newton condition $A_{k+1} y_k = U_{k+1} U_{k+1}^T y_k = \rho_k \tilde{s}_k$ is satisfied. This condition can be replaced by equations

$$U_{k+1}^T y_k = z_k, \quad U_{k+1} z_k = \varrho_k \tilde{s}_k, \quad z_k^T z_k = \varrho_k \tilde{b}_k. \tag{10}$$

The following theorem is proved in [VL05].

Theorem 3. Let T_k be a symmetric positive definite matrix and $z_k \in \mathcal{R}^m$. Then the unique solution U_{k+1} to the problem

$$\text{minimize} \quad \|T_k^{-1/2}(U_{k+1} - U_k)\|_F^2 \quad \text{subject to (10)}$$

is

$$U_{k+1} = U_k - \frac{T_k y_k}{y_k^T T_k y_k} y_k^T U_k + (\varrho_k \tilde{s}_k - U_k z_k + \frac{y_k^T U_k z_k}{y_k^T T_k y_k} T_k y_k) \frac{z_k^T}{z_k^T z_k} \tag{11}$$

($T_k y_k$ and z_k are vector parameters defining a class of shifted limited-memory variable metric methods).

Remark 2. Formula (11) can be writen in the form

$$U_{k+1} = \frac{\tilde{s}_k z_k^T}{\tilde{b}_k} + \left(I - \frac{T_k y_k y_k^T}{y_k^T T_k y_k}\right) U_k \left(I - \frac{z_k z_k^T}{z_k^T z_k}\right),$$

which implies

$$U_{k+1} U_{k+1}^T = \rho_k \frac{\tilde{s}_k \tilde{s}_k^T}{\tilde{b}_k} + \left(I - \frac{T_k y_k y_k^T}{y_k^T T_k y_k}\right) U_k \left(I - \frac{z_k z_k^T}{z_k^T z_k}\right) U_k^T \left(I - \frac{y_k y_k^T T_k}{y_k^T T_k y_k}\right).$$

Usually $T_k y_k = \tilde{s}_k$. This choice gives the (full) shifted BFGS method if term $z_k z_k^T / z_k^T z_k$ is omitted.

Using suitable values of the vector parameters we obtain particular methods. Assuming that $T_k y_k$ and $\rho_k \tilde{s}_k - U_k z_k$ are linearly dependent and setting

$$z_k = \vartheta_k U_k^T B_k s_k, \quad \vartheta_k = \pm\sqrt{\varrho_k \tilde{b}_k / \bar{c}_k} \tag{12}$$

we obtain rank 1 variationally derived method (VAR1), where

$$U_{k+1} = U_k - \frac{\varrho_k \tilde{s}_k - \vartheta_k A_k B_k s_k}{\varrho_k \tilde{b}_k - \vartheta_k \bar{b}_k} (y_k - \vartheta_k B_k s_k)^T U_k, \tag{13}$$

which gives the best results for the choice $\operatorname{sgn}(\vartheta_k \bar{b}_k) = -1$.

Using z_k given by (12) and setting $T_k y_k = \tilde{s}_k$, we obtain rank 2 variationally derived method (VAR2), where

$$U_{k+1} = U_k - \frac{\tilde{s}_k}{\tilde{b}_k} y_k^T U_k + \left(\varrho_k \frac{\tilde{s}_k}{\vartheta_k} - A_k B_k s_k + \frac{\bar{b}_k}{\tilde{b}_k} \tilde{s}_k\right) \frac{s_k^T B_k U_k}{\bar{c}_k}. \tag{14}$$

The efficiency of both these methods significantly depends on the value of the correction parameter ϱ_k. Very good results were obtained with choices $\varrho_k = \nu_k$, $\varrho_k = \varepsilon_k$, $\varrho_k = \sqrt{\nu_k \varepsilon_k}$ and $\varrho_k = \zeta_k/(\zeta_k + \zeta_{k+1})$, where $\nu_k = \mu_k/(1 - \mu_k)$, μ_k is a relative shift parameter and $\varepsilon_k = \sqrt{1 - \bar{a}_k/a_k}$ is the damping factor of μ_k.

Using the above formulas, the global convergence of VAR1 and VAR2 is a consequence of Theorem 2 (see [VL05]).

Theorem 4. Consider a shifted variable metric method VAR1 or VAR2 satisfying Assumption 2 and inequality $\mu_k^2 \leq \zeta_k \hat{a}_k / a_k$ together with the line search (1)-(2). Let the objective function satisfy Assumption 1. Then if

$$\vartheta_k = -\operatorname{sgn} \bar{b}_k \min\left(C, \sqrt{\varrho_k \tilde{b}_k / \bar{c}_k}\right) \quad \text{or} \quad \vartheta_k = \pm\sqrt{\varrho_k \tilde{b}_k / \bar{c}_k}$$

(for VAR1 or VAR2) hold in all iterations ($C > 0$ can be chosen arbitrarily), one has

$$\liminf_{k \to \infty} \|g_k\| = 0.$$

Shifted limited-memory variable metric methods were tested by using a set of 22 test problems with 1000 and 5000 variables implemented in subroutine `TEST14` (see `www.cs.cas.cz/~luksan/test.html`). The results are presented in Table 2, where n is the number of variables, "method is the method used (VAR1 - the rank 1 variationally derived method, VAR2 - the rank 2 variationally derived method, LBFGSS - the limited-memory BFGS method with Strang recurrences, LBFGSC - the limited-memory BFGS method with compact matrices, LBFGSR - the limited-memory BFGS method with reduced Hessians, CG - the nonlinear conjugate gradient method), NIT is the total number of iterations, NEV is the total number of function and gradient evaluations, NF is the number of failures for a given set (i.e., the number of problems which were not successfully solved) and "time" is the total computational time in seconds. Always 10 vectors (or pairs) were stored for $n = 1000$ and 5 vectors (or pairs) were stored for $n = 5000$.

n	method	NIT	NEV	NF	time
1000	VAR1	19260	19660	-	10.63
	VAR2	18430	18693	-	10.39
	LBFGSS	20341	21389	-	11.55
	LBFGSC	21022	22101	-	12.06
	LBFGSR	21892	33442	-	18.91
	CG	20087	40122	-	13.70
5000	VAR1	97057	98888	-	292.56
	VAR2	87725	89528	-	270.09
	LBFGSS	117670	121053	1	322.38
	LBFGSC	112448	115573	1	326.95
	LBFGSR	118139	189451	1	509.75
	CG	71388	178417	1	340.13

Table 2

The results presented in this table and our other extensive experiments imply the following conclusions:

- Methods VAR1 and VAR2 are very efficient, competitive with LBFGS methods, for our set of test problems. However, LBFGS methods can be better than VAR1 and VAR2 for very ill-conditioned problems.
- Method CG is very efficient for large-scale problems, but it needs a larger number of function evaluations and frequently terminates before a required precision is achieved.
- Shifted limited-memory VM methods are still under development. Our limited computational experience indicates that they could be improved by using a more suitable choice of parameters.

3 Methods for large-scale nonsmooth optimization

We assume that objective function $F : R^n \to R$ is locally Lipschitz and we are able to compute a (Clarke) subgradient $g \in \partial F(x)$ at any point $x \in \mathcal{R}^n$. Since a locally Lipschitz function is differentiable almost everywhere by the Rademacher theorem, then usually $g = \nabla F(x)$. A special feature of nonsmooth problems is the fact that the gradient $\nabla F(x)$ changes discontinuously and is not small in the neighborhood of a local extremum. Thus the standard optimization methods cannot be used efficiently.

3.1 Principles of bundle methods

The quantities $F(x^k)$, $g(x^k) \in \partial F(x^k)$ at a single point x^k do not suffice for describing the local properties of the nonsmooth objective function. A bundle of quantities $F^j = F(y^j)$, $g^j \in \partial F(y^j)$ obtained at trial points y^j, $j \in \mathcal{J}_k \subset \{1, \ldots, k\}$, gives much better information. These values serve for the construction of the piecewise linear function

$$F_L^k(x) = \max_{j \in \mathcal{J}_k} \{F^j + (x - y^j)^T g^j\} = \max_{j \in \mathcal{J}_k} \{F(x^k) + (x - x^k)^T g^j - \alpha_j^k\},$$

where $\alpha_j^k = F(x^k) - F_j^k$, $j \in \mathcal{J}_k$, are linearization errors and $F_j^k = F^j + (x^k - x^j)^T g^j$, $j \in \mathcal{J}_k$. In the convex case, this piecewise linear function is majorized by the objective function and, moreover, $\alpha_j^k \geq 0$ for $j \in \mathcal{J}_k$. To guarantee nonnegativity of these numbers in the nonconvex case, the subgradient locality measures

$$\alpha_j^k = \max \left\{ |F(x^k) - F_j^k|, \gamma (s_j^k)^\nu \right\},$$

where $\gamma > 0$, $\nu \geq 1$ and

$$s_j^k = \|x^j - y^j\| + \sum_{i=j}^{k-1} \|x^{i+1} - x^i\|$$

for $j \in \mathcal{J}_k$, are used instead of linearization errors. Since we can only work with limited-size bundles where $|\mathcal{J}_k| \leq m$ ($|\mathcal{J}_k|$ is the cardinality of set $\mathcal{J}_k$),

the set $\mathcal{J}_k$ is usually determined in such a way that $\mathcal{J}_k = \{1, \ldots, k\}$ for $k \leq m$, and $\mathcal{J}_{k+1} = \mathcal{J}_k \cup \{k+1\} \backslash \{k+1-m\}$ for $k \geq m$. In this case, one possibility guaranteeing the global convergence of the bundle method is the use of transformed aggregate values F_a^k, g_a^k, s_a^k and

$$\alpha_a^k = \max\left\{|F(x^k) - F_a^k|, \gamma(s_a^k)^\nu\right\},$$

which accumulate information from the previous iterations. These values represent a linear function which is added to the set of linear functions contained in the bundle. New aggregate values $\tilde{F}_a^k$, $\tilde{g}_a^k$, $\tilde{s}_a^k$ are obtained by solving the quadratic programming subproblem (see (18)) and are transformed to the next iteration by (24).

Direction vector $d^k \in \mathcal{R}^n$ is usually obtained as a minimum of the piecewise quadratic function

$$F_Q^k(x) = \frac{1}{2}(x - x^k)^T G^k (x - x^k) + \max\{F_L^k(x), F(x^k) + (x - x^k)^T g_a^k - \alpha_a^k\},$$

where $(1/2)(x - x^k)^T G^k (x - x^k)$ is the regularizing term with symmetric positive definite matrix G^k. This term restricts the size of the direction vector (in a similar way as in the trust region methods). This minimization problem is equivalent to the quadratic programming problem: Minimize function

$$\frac{1}{2} d^T G^k d + v \tag{15}$$

subject to

$$-\alpha_j^k + d^T g^j \leq v, \quad j \in \mathcal{J}_k, \quad -\alpha_a^k + d^T g_a^k \leq v \tag{16}$$

(v is an extra variable). The solution of the primal QP subproblem can be expressed in the form

$$d^k = -(G^k)^{-1} \tilde{g}_a^k, \quad v^k = -(d^k)^T G^k d^k - \tilde{\alpha}_a^k, \tag{17}$$

where

$$\begin{gathered} \tilde{g}_a^k = \sum_{j \in \mathcal{J}_k} \lambda_j^k g^j + \lambda_a^k g_a^k, \\ (\tilde{\alpha}_a^k, \tilde{F}_a^k, \tilde{s}_a^k) = \sum_{j \in \mathcal{J}_k} \lambda_j^k (\alpha_j^k, F_j^k, s_j^k) + \lambda_a^k (\alpha_a^k, F_a^k, s_a^k) \end{gathered} \tag{18}$$

and where λ_j^k, $j \in \mathcal{J}_k$, λ_a^k, are corresponding Lagrange multipliers. These Lagrange multipliers are also solutions of the dual QP problem: Minimize function

$$\frac{1}{2} \left(\sum_{j \in \mathcal{J}_k} \lambda_j g^j + \lambda_a g_a^k \right)^T (G^k)^{-1} \left(\sum_{j \in \mathcal{J}_k} \lambda_j g^j + \lambda_a g_a^k \right) + \sum_{j \in \mathcal{J}_k} \lambda_j \alpha_j^k + \lambda_a \alpha_a^k \tag{19}$$

subject to

$$\lambda_j \geq 0, \quad j \in \mathcal{J}_k, \quad \lambda_a \geq 0, \quad \sum_{j \in \mathcal{J}_k} \lambda_j + \lambda_a = 1. \tag{20}$$

The minimum value of the dual function is

$$w^k = \frac{1}{2}(\tilde{g}_a^k)^T (G^k)^{-1} \tilde{g}_a^k + \tilde{\alpha}_a^k = -v^k - \frac{1}{2}(\tilde{g}_a^k)^T (G^k)^{-1} \tilde{g}_a^k. \tag{21}$$

Using direction vector d^k, we can compute a new approximation of the minimizer of the objective function. It is usually not possible to just set $x^{k+1} = x^k + d^k$. To guarantee the global convergence of the bundle method, we use a line search procedure which generates two points

$$\begin{aligned} x^{k+1} &= x^k + t_L^k d^k, \\ y^{k+1} &= x^k + t_R^k d^k, \end{aligned}$$

where $0 \leq t_L^k \leq t_R^k \leq 1$ are stepsizes, in such a way that exactly one of the two possibilities, the descent step or the zero step, occurs. The descent step implies the conditions

$$t_R^k = t_L^k > 0, \qquad F(x^k + t_L^k d^k) \leq F(x^k) - \varepsilon_L t_L^k w^k, \tag{22}$$

while the zero step implies the conditions

$$t_R^k > t_L^k = 0, \qquad (d^k)^T g(x^k + t_R^k d^k) \geq \alpha^{k+1} - \varepsilon_R w^k \tag{23}$$

with

$$\alpha^{k+1} = \max\left\{|F(x^k) - F(x^k + t_R^k d^k) + t_R^k (d^k)^T g(x^k + t_R^k d^k)|, \gamma |t_R^k d^k|^\nu\right\}.$$

Here $0 < \varepsilon_L < 1/2$ and $\varepsilon_L < \varepsilon_R < 1$.

In case the descent step is performed, it is necessary to transform all bundle quantities to the new point x_{k+1}. This is realized by using the formulas

$$\begin{aligned} F_j^{k+1} &= F_j^k + (x^{k+1} - x^k)^T g^j, \qquad j \in J_k, \\ F_a^{k+1} &= \tilde{F}_a^k + (x^{k+1} - x^k)^T \tilde{g}_a^k, \\ F_{k+1}^{k+1} &= F^{k+1} + (x^{k+1} - y^{k+1}) g^{k+1}, \\ g_a^{k+1} &= \tilde{g}_a^k, \\ s_j^{k+1} &= s_j^k + \|x^{k+1} - x^k\|, \qquad j \in J_k, \\ s_a^{k+1} &= \tilde{s}_a^k + \|x^{k+1} - x^k\|, \\ s_{k+1}^{k+1} &= \|x^{k+1} - y^{k+1}\|. \end{aligned} \tag{24}$$

It remains to specify the way for determining matrices G^k. To ensure the global convergence of a bundle method, we assume for simplicity that matrices G^k are uniformly positive definite and uniformly bounded (their eigenvalues

are positive and lay in the compact interval that does not contain zero). Moreover, if the k-th step is a zero step, then we assume that $G^{k+1} - G^k$ is positive semidefinite. These assumptions are relatively strong, but they can be weakened for individual bundle methods. In the most frequently used proximal bundle method, where matrix G^k is a diagonal of the form $G^k = \sigma^k I$, the above assumptions are satisfied if weights σ^k are positive and lay in the compact interval that does not contain zero and $\sigma^{k+1} \geq \sigma^k$ holds in the zero step. Note that the proximal bundle method requires relatively large bundles ($m \sim n$) to be computationally efficient so that the solution of the quadratic programming subproblem (15)–(16) is time consuming.

It can be proved under mild assumptions (see e.g. [Kiw85]) that the number of consecutive zero steps is finite and that every cluster point of the sequence $\{x^k\}$ is a stationary point of the objective function. This follows from the fact that the norms of aggregate subgradients tend to zero implying $0 \in \partial F(x_k)$, if the number of consecutive zero steps is infinite. An infinite sequence of the descent steps can be investigated by the standard way.

3.2 Variable metric methods for nonsmooth problems

Standard bundle methods require relatively large bundles to be computationally efficient. Therefore, we need to solve quadratic programming subproblems with a relatively large number of constraints. At the same time, standard variable metric methods successfully solve many nonsmooth problems. For this reason, it is advantageous to develop special variable metric methods, which combine good properties of both mentioned approaches. Following [VL01], we apply variable metric updates with current subgradients to matrix $H^k = (G^k)^{-1}$ (used in (19)), which allows us to decrease the bundle dimension significantly. At the same time, we use aggregate subgradients after zero steps and a line search described in the previous subsection to guarantee the global convergence.

Variable metric methods described in this subsection use, for the direction determination, the current subgradient after a descent step and the aggregate subgradient after a zero step. The aggregation procedure uses only three subgradients $g^m \in \partial F(x^k)$, $g^{k+1} \in \partial F(y^{k+1})$, $\tilde{g}^k$ and three subgradient locality measures $\alpha_m = 0$, $\alpha_{k+1} \geq 0$, $\tilde{\alpha}_k \geq 0$ (m is the index of the last descent step and the tilde denotes aggregate quantities). The quadratic programming subproblem (19)-(20) reduces to the minimization of the function

$$\varphi(\lambda_1, \lambda_2, \lambda_3) = \frac{1}{2} \left\| (H^k)^{1/2} (\lambda_1 g^m + \lambda_2 g^{k+1} + \lambda_3 \tilde{g}^k) \right\|^2 + \lambda_2 \alpha^{k+1} + \lambda_3 \tilde{\alpha}^k, \quad (25)$$

where $\lambda_i \geq 0$, $i \in \{1, 2, 3\}$ and $\lambda_1 + \lambda_2 + \lambda_3 = 1$. The optimal values $\lambda_i^k \geq 0$, $i \in \{1, 2, 3\}$ can be computed in a simple way. The new aggregate subgradient and the new aggregate subgradient locality measure are computed from the formulas

$$\tilde{g}^{k+1} = \lambda_1^k g^m + \lambda_2^k g^{k+1} + \lambda_3^k \tilde{g}^k, \quad \tilde{\alpha}^{k+1} = \lambda_2^k \alpha^{k+1} + \lambda_3^k \tilde{\alpha}^k. \tag{26}$$

In the first iteration or after a descent step, we set $\tilde{g}^k = g^k$, $\tilde{\alpha}^k = 0$ and $m = k$. The direction vector is determined by formula $d^k = -H^k \tilde{g}^k$. At the same time, we set $w^k = (1/2)(\tilde{g}^k)^T H^k \tilde{g}^k + \tilde{\alpha}^k$. If w^k is sufficiently small, then an approximate solution is found.

Positive semidefiniteness of $H^k - H^{k+1}$ (which is equivalent to positive semidefiniteness of $G^{k+1} - G^k$) after a zero step is usually guaranteed by the symmetric rank-one (SR1) update. Therefore, we use the BFGS update after a descent step and the SR1 update after a zero step. The BFGS update

$$H^{k+1} = H^k + \left(t_L^k + \frac{(u^k)^T H^k u^k}{(u^k)^T d^k}\right) \frac{d^k (d^k)^T}{(u^k)^T d^k} - \frac{H^k u^k (d^k)^T + d^k (u^k)^T H^k}{(u^k)^T d^k},$$

where $u^k = g^{k+1} - g^m$, is used only if $(u^k)^T d^k > 0$. Otherwise we set $H^{k+1} = H^k$. The SR1 update

$$H^{k+1} = H^k - v^k (v^k)^T / (u^k)^T v^k,$$

where $v^k = H^k u^k - t_R^k d^k$, is used only if $(v^k)^T \tilde{g}^k < 0$ (which implies $(u^k)^T v^k > 0$). Otherwise we set $H^{k+1} = H^k$.

Detailed descriptions of variable metric methods for nonsmooth functions can be found in [LV99] and [VL01]. The following result is proved in [VL01].

Theorem 5. Assume that function $F : R^n \to R$ is locally Lipschitz and the level set $\{x \in \mathcal{R}^n : F(x) \leq F(x_1)\}$ is bounded. Then every cluster point of sequence $\{x_k\}$ generated by the nonsmooth variable metric method is stationary for F.

Two methods for nonsmooth optimization (PBM - the proximal bundle method, NVM - the nonsmooth variable metric method) were tested by using a set of 25 test problems with 2–50 variables implemented in subroutine `TEST19` (see `www.cs.cas.cz/~luksan/test.html`). The results are presented in Table 3, where P is the number of the problem, NIT is the total number of iterations, NEV is the total number of function and subgradient evaluations and F is the reached function value. The last row contains the summary values and the total computational time (in seconds).

Table 3 demonstrates the high efficiency of the nonsmooth variable metric method. It is competitive with the proximal bundle method measured by the number of iterations, even if it uses bundles of dimension at most 2. Moreover, it is more efficient than the proximal bundle method measured by the computational time, since it does not use the time consuming quadratic programming subproblem (with $m \sim n$ constraints).

	PBM			NVM		
P	NIT	NEV	F	NIT	NEV	F
1	42	45	0.38117064-06	34	34	0.27598807-10
2	18	20	0.46154993-08	15	16	0.94894120-10
3	31	33	1.9522245	17	17	1.9522247
4	14	16	2.0000000	17	17	2.0000000
5	17	19	-3.0000000	20	20	-2.9999996
6	13	15	7.2000014	19	19	7.2000000
7	11	12	-1.4142135	10	10	-1.4142133
8	66	68	-.99999940	55	59	-.99999247
9	13	15	-1.0000000	37	37	-.99999979
10	43	46	-7.9999999	14	14	-7.9999998
11	43	45	-43.999999	38	38	-43.999999
12	27	29	22.600162	40	40	22.600162
13	60	62	-32.348678	52	53	-32.348678
14	154	155	-2.9196975	32	32	-2.9197003
15	92	93	.55981566	81	83	.55981553
16	74	75	-.84140828	89	89	-.84140570
17	160	162	9.7857723	241	241	9.7858732
18	128	143	16.703861	88	89	16.703838
19	150	151	0.16712381-06	123	123	0.14683215-05
20	39	40	0.12440972-12	23	23	.00000000
21	245	251	-638530.48	357	359	-638564.91
22	52	53	0.11665945-11	358	360	0.41534959-05
23	19	20	0.51313988-08	65	66	0.32729678-05
24	27	28	0.23412735-07	67	67	0.94570857-06
25	428	450	32.349182	313	315	32.349159
Σ	1966	2046	TIME = 1.48	2205	2221	TIME = 0.93

Table 3

3.3 Variable metric methods for large-scale nonsmooth problems

Proximal bundle methods are not suitable for solving large-scale nonsmooth problems, since they lead to large-scale quadratic programming subproblems, where constraint Jacobian matrices are usually dense. Nonsmooth variable metric methods described in the previous subsection are also unsuitable, since they use dense variable metric updates. Fortunately, these updates can be replaced by updates based on a limited-memory approach or by updates which utilize sparsity. All other algorithmic details can remain unchanged.

A limited-memory approach is investigated in [HMM04]. The resulting method utilizes matrix (4)–(5) after a descent step and matrix (6) after a zero step. Nevertheless, the updating strategy is not simple, since the condition requiring positive semidefiniteness of $H^k - H^{k+1}$ after a zero step considerably complicates a logical structure of the algorithm. Algorithmic details

of this method together with encouraging computational results are given in [HMM04]. Global convergence of this method is proved in [HMM03].

An efficient method based on partitioned variable metric updates is proposed in [LV04]. This method has been developed for minimizing partially separable functions of the form

$$F(x) = \sum_{i=1}^{m} f_i(x)$$

where $f_i : \mathcal{R}^n \to \mathcal{R}$, $1 \leq i \leq m$ (m is usually large), are nonsmooth functions depending on a small number of variables (n_i, say). A typical example is

$$F(x) = \sum_{i=1}^{m} |f_i(x)|.$$

If $n_i \ll n$ for $1 \leq i \leq m$, subgradients g_i, generalized Hessian matrices G_i and their approximations B_i are sparse. Let $\mathcal{R}_i^n \subset \mathcal{R}^n$ be the subspace defined by n_i variables appearing in f_i and $Z_i \in R^{n \times n_i}$ be the matrix whose columns form the canonical orthonormal basis in $\mathcal{R}_i^n$ (i.e., they are columns of the unit matrix). To simplify the notation, we introduce packed subgradients $\hat{g}_i = Z_i^T g_i \in \mathcal{R}^{n_i}$, packed generalized Hessian matrices $\hat{G}_i = Z_i^T G_i Z_i \in \mathcal{R}^{n_i \times n_i}$ and their approximations $\hat{B}_i \in \mathcal{R}^{n_i \times n_i}$. Defining vectors $\hat{x}_i = Z_i^T x_i \in \mathcal{R}^{n_i}$ as parts of vector $x \in \mathcal{R}^n$, we can write packed quasi-Newton conditions in the form $\hat{B}_i^{k+1} \hat{s}_i^k = \hat{y}_i^k$, where $\hat{s}_i^k = \hat{x}_i^{k+1} - \hat{x}_i^k$ and $\hat{y}_i^k = \hat{g}_i^{k+1} - \hat{g}_i^k$. Packed quasi-Newton conditions imply packed quasi-Newton updates, which are used instead of dense variable metric updates.

Matrices $B_i^k = Z_i \hat{B}_i^k Z_i^T$ and subgradients $g_i^k = Z_i g_i^k$ (determined from packed matrices $\hat{B}_i^k$ and packed subgradients $\hat{g}_i^k$) define matrix B^k and subgradient g^k as sums

$$B^k = \sum_{i=1}^{m} B_i^k, \quad g^k = \sum_{i=1}^{m} g_i^k.$$

Denoting by $\tilde{g}^k = \sum_{i=1}^{m} \tilde{g}_i^k$ the corresponding aggregate subgradient (see (28)), direction vector d^k is determined by solving the equation

$$B^k d^k = -\tilde{g}^k. \tag{27}$$

Furthermore, we define $w^k = -(1/2)(s^k)^T \tilde{g}^k + \tilde{\alpha}^k$. If w^k is sufficiently small, then an approximate solution is found. Since matrix B^k is large and sparse, we use a sparse Choleski (or Gill-Murray [GM74]) decomposition $B^k = L^k D^k (L^k)^T$. This decomposition is also used in the quadratic programming subproblem (25) instead of H^k. Thus corresponding matrix multiplications are replaced by solutions of systems with triangular matrices (back elimination). Solving (25) we obtain Lagrange multipliers λ_1, λ_2, λ_3. The aggregate subgradients are obtained by the formula

$$\tilde{g}_i^{k+1} = \lambda_1^k g_i^k + \lambda_2^k g_i^{k+1} + \lambda_3^k \tilde{g}_i^k, \quad 1 \le i \le m. \tag{28}$$

Packed matrices $\hat{B}_i^k$, $1 \le i \le m$, are updated by packed variable metric updates. We use the packed BFGS update

$$\hat{B}_i^{k+1} = \hat{B}_i^k + \frac{\hat{y}_i^k (\hat{y}_i^k)^T}{(\hat{s}_i^k)^T \hat{y}_i^k} - \frac{\hat{B}_i^k \hat{s}_i^k (\hat{B}_i^k \hat{s}_i^k)^T}{(\hat{s}_i^k)^T \hat{B}_i^k \hat{s}_i^k}, \quad (\hat{s}_i^k)^T \hat{y}_i^k > 0,$$
$$\hat{B}_i^{k+1} = \hat{B}_i^k, \quad (\hat{s}_i^k)^T \hat{y}_i^k \le 0,$$

after a descent step and packed symmetric rank-one (SR1) update

$$\hat{B}_i^{k+1} = \hat{B}_i^k + \frac{\hat{v}_i^k (\hat{v}_i^k)^T}{(\hat{s}_i^k)^T \hat{v}_i^k}, \quad (\hat{s}_i^k)^T \hat{v}_i^k > 0,$$
$$\hat{B}_i^{k+1} = \hat{B}_i^k, \quad (\hat{s}_i^k)^T \hat{v}_i^k \le 0,$$

with $\hat{v}_i^k = \hat{y}_i^k - \hat{B}_i^k \hat{s}_i^k$, after a zero step.

Methods for large-scale nonsmooth optimization were tested by using a set of 22 test problems with 50, 200, 500 and 1000 variables implemented in subroutine `TEST15` (see `www.cs.cas.cz/~luksan/test.html`). The results are presented in Table 4, where n is the number of variables, "method" is the method used (PBM - the proximal bundle method, NVM - the nonsmooth variable metric method, PNVM - the partitioned nonsmooth variable metric method), NIT is the total number of iterations, NEV is the total number of function and subgradient evaluations, NF is the number of failures for a given set (i.e., the number of problems which were not successfully solved) and "time" is the total computational time in seconds.

n	method	NIT	NEV	NF	time
50	PBM	99665	103814	-	38.36
	NVM	34390	34475	-	3.48
	PNVM	50629	50676	-	4.22
200	PBM	214320	241845	8	1085.39
	NVM	97439	97703	2	67.53
	PNVM	42310	42418	-	15.83
500	NVM	173175	173744	4	880.15
	PNVM	45105	46086	-	51.77
1000	PNVM	52729	53604	-	135.00

Table 4

The results presented in this table imply the following conclusions:

- Nonsmooth variable metric method NVM is more efficient than proximal bundle method for small-size partially separable sums of absolute values.

- Partitioned nonsmooth variable metric method PNVM is very robust, much more efficient than other methods used for solving our sets of medium-size and large-scale test problems.

3.4 Variable metric methods for partially separable minimax problems

Consider functions of the form

$$F(x) = \max_{1 \le i \le m} f_i(x)$$

where $f_i : \mathcal{R}^n \to \mathcal{R}$, $1 \le i \le m$ (m is usually large), are nonsmooth functions depending on a small number of variables (n_i, say). Let $F(x) = f_i(x)$ for some $1 \le i \le m$. Then any subgradient of $f_i(x)$ is a subgradient of $F(x)$. Thus we can easily find a sparse subgradient $g(x) = g_i(x)$ (containing only n_i nonzero elements) at an arbitrary point $x \in \mathcal{R}^n$, which means that the constraint Jacobian matrix of the quadratic programming subproblem: minimize

$$\frac{1}{2} d^T G^k d + v$$

subject to

$$-\alpha_j^k + d^T g^j \le v, \quad j \in \mathcal{J}_k, \quad -\alpha_a^k + d^T g_a^k \le v$$

is sparse (note that aggregate subgradient g_a^k need not be sparse, which implies that the constraint Jacobian matrix can have one dense row). If $G^k = \sigma^k I$, we obtain a sparse quadratic programming subproblem. Thus having an efficient sparse QP solver, we can use the proximal bundle method.

Let

$$F(x) = \max_{1 \le i \le m} |f_i(x)|,$$

where $f_i(x)$, $1 \le i \le m$, are smooth functions depending on a small number of variables. Then minimization of F is equivalent to the sparse nonlinear programming problem with $n + 1$ variables $x \in \mathcal{R}^n$, $z \in \mathcal{R}$: Minimize z subject to

$$-z \le f_i(x) \le z, \quad 1 \le i \le m.$$

This problem can be solved by an arbitrary nonlinear programming method utilizing sparsity (SQP, interior point, nonsmooth equation). A special form of this problem allows us to use some simplifications in comparison with general problems. Choosing a suitable initial value of z we obtain a feasible starting point. Moreover, function $F(x)$ is an ideal merit function for the above problem.

4 Hybrid methods for large-scale nonlinear least squares

Consider functions of the form

$$F(x) = \frac{1}{2}\sum_{i=1}^{m} f_i^2(x) = \frac{1}{2} f^T(x) f(x)$$

(sum of squares), where $f_i : \mathcal{R}^n \to \mathcal{R}$, $1 \leq i \leq m$ (m is usually large), are smooth functions depending on a small number of variables (n_i, say). In this case, the Jacobian matrix $J(x) = [J_{ij}(x)] = [\partial f_i(x)/\partial x_j]$ is sparse. Using the Jacobian matrix, we can express gradient $g(x)$ and Hessian matrix $G(x)$ in the form $g(x) = J^T(x)f(x)$ and

$$G(x) = \sum_{i=1}^{m} \left(g_i(x) g_i^T(x) + f_i(x) G_i(x)\right) = J^T(x)J(x) + C(x)$$

($g_i(x)$ and $G_i(x)$ are gradients and Hessian matrices of $f_i(x)$, respectively).

The most known Gauss-Newton method uses matrix $B(x) = J^T(x)J(x)$ instead of the Hessian matrix $G(x) = J^T(x)J(x) + C(x)$ (i.e., it omits the second order information contained in $C(x)$). We assume that matrix $J^T(x)J(x)$ is sparse (then also $C(x)$ is sparse). Matrix $J^T(x)J(x)$ is frequently ill-conditioned (even singular), thus the Gauss-Newton method requires a trust-region realization. If the minimum value $F(x^*)$ is large (large residual problem), then the Gauss-Newton method can be inefficient. Therefore, modifications based on variable metric updates have been developed. The following theorem is proved in [AF85].

Theorem 5. If $F_k \to 0$ Q-superlinearly, then $(F_k - F_{k+1})/F_k \to 1$. If $F_k \to F^* > 0$, then $(F_k - F_{k+1})/F_k \to 0$.

Theorem 5 implies the following philosophy of hybrid Gauss-Newton methods with second order corrections. Direction vector d is obtained by a trust-region strategy using the quadratic model $(1/2)d^T Bd + f^T Jd$ and the constraint $\|d\| \leq \Delta$. Then $x_+ = x + d$, $F_+ = F(x_+)$ and $J_+ = J(x_+)$. If $F - F_+ > \underline{\vartheta} F$, then $B_+ = J_+^T J_+$ (Gauss-Newton method). If $F - F_+ \leq \underline{\vartheta} F$, then $B_+ = J_+^T J_+ + C_+$, where C_+ is an approximation of the second order term. Usually $\underline{\vartheta} \approx 10^{-4}$.

For medium-size problems with dense matrices, matrix C is usually obtained by variable metric updates [AF85] [DGW81], which are unsuitable in the large-scale case. Fortunately, simple corrections utilizing sparsity considerably increase efficiency of the Gauss-Newton method. We shortly describe two hybrid methods proposed in [Luk96b].

- Gauss-Newton method with the Newton corrections. In the first iteration we use matrix $B = J^T J$. In the subsequent iterations, we set

$$B_+ = J_+^T J_+, F - F_+ > \underline{\vartheta} F,$$
$$B_+ = J_+^T J_+ + \sum_{i=1}^{m} f_i^+ G_i^+, F - F_+ \leq \underline{\vartheta} F,$$

where $f_i^+ = f_i(x_+)$, $G_i^+ \approx G_i(x_+)$, $1 \leq k \leq m$, (G_i^+ is a difference approximation of the Hessian matrix $G_i(x_+)$).

- Gauss-Newton method with the Marwil corrections. In the first iteration we use matrix $B = J^T J$. In the subsequent iterations, we set

$$B_+ = J_+^T J_+, F - F_+ > \underline{\vartheta} F,$$
$$B_+ = \mathcal{P}_S \mathcal{P}_{QG}(J_+^T J_+), F - F_+ \leq \underline{\vartheta} F,$$

where

$$\mathcal{P}_S W = (W + W^T)/2$$

for a given square matrix W and

$$\mathcal{P}_{QG} M = \mathcal{P}_G(M + us^T).$$

for a given symmetric positive semidefinite matrix M. Here $u \in \mathcal{R}^n$ solves linear system $Du = y - Ms$ with diagonal matrix D such that

$$D_{ii} = \sum_{M_{ij} \neq 0} s_j^2$$

and

$$(\mathcal{P}_G W)_{ij} = W_{ij}, \quad (J^T J)_{ij} \neq 0,$$
$$(\mathcal{P}_G W)_{ij} = 0, \quad (J^T J)_{ij} = 0$$

($\mathcal{P}_G$ is the so-called gangster operator).

Methods for large-scale nonlinear least squares were tested by using a set of 52 test problems with 1000 variables implemented in subroutines `TEST15` and `TEST18` (see `www.cs.cas.cz/~luksan/test.html`). The results are presented in Table 5, where "step" is the strategy for step-length selection (MS - the optimum trust-region step of Moré and Sorensen [MS83], DL - the dog-leg strategy of Powell [Pow70], LS - the standard line-search procedure), "method" is the method used (GN - the Gauss-Newton method, GNN - the Gauss-Newton method with the Newton corrections, GNM - the Gauss-Newton method with the Marwil corrections, DN - the discrete Newton method, where the second order derivatives are approximated by differences, PVM - the partitioned variable metric method), NIT is the total number of iterations, NEV is the total number of function evaluations, NF is the number of failures for a given set (i.e., the number of problems which were not successfully solved) and "time" is the total computational time in seconds.

step	method	NIT	NEV	NF	time
MS	GN	8542	8929	1	72.00
	GNN	5499	5801	-	51.94
	GNM	6434	6801	-	62.88
	DN	7804	52398	1	202.07
DL	GN	9244	9602	-	38.84
	GNN	7767	8216	-	35.68
	GNM	6851	7029	-	25.87
	DN	10326	91181	-	171.98
LS	PVM	12093	16285	1	99.17

Table 5

The results presented in this table imply the following conclusions:

- Modifications of the Gauss-Newton method implemented with the trust-region strategy are very robust for our set of test problems, much better than discrete versions of the Newton method and more efficient than partitioned variable metric methods.
- The Newton corrections or the Marwil variable metric updates improve the efficiency of the Gauss-Newton method especially if direct methods for solving trust-region subproblems are used. Hybrid methods GNN and GNM are shown to be the most efficient methods for solving our set of test problems.

5 Methods for solving large-scale trust-region subproblems

Trust-region methods can be used when the Hessian matrix (or its approximation) is known. These methods are very convenient when this matrix is indefinite, ill-conditioned or singular. This situation often arises in connection with the Newton method for general objective function (indefiniteness) or with the Gauss-Newton method for nonlinear least-squares (near-singularity).

The crucial part of each trust region method is the direction determination. We restrict our attention to problems with large dimensions. To simplify the notation, we omit index k and use symbol $\succeq$ for ordering by positive semidefiniteness. Let

$$Q(d) = \frac{1}{2} d^T B d + g^T d.$$

We seek a direction vector $d \in \mathcal{R}^n$ in such a way that

$$\|d\| \leq \Delta, \tag{29}$$

$$\|d\| < \Delta \Rightarrow \|Bd + g\| \leq \omega \|g\| \tag{30}$$

with $0 \leq \omega < 1$ and

$$Q(d) \geq \sigma \|g\| \min\left(\Delta, \frac{\|g\|}{\|B\|}\right) \tag{31}$$

with $0 < \sigma \leq 1/2$. It can be shown [Pow84] that conditions (29)–(31) guarantee that the trust-region method is globally convergent if matrices B are uniformly bounded (or the sum of the reciprocal values of its norms is equal to infinity). There are various commonly known methods for computing direction vectors satisfying conditions (29)-(31) which we now shortly mention.

The most sophisticated method is based on the computation of the optimal locally constrained step. In this case, vector $d \in \mathcal{R}^n$ is obtained by solving subproblem

$$\text{minimize} \quad Q(d) = \frac{1}{2} d^T B d + g^T d \quad \text{subject to} \quad \|d\| \leq \Delta. \tag{32}$$

Necessary and sufficient conditions for this solution are

$$\|d\| \leq \Delta, \quad (B+\lambda I)d+g = 0, \quad B+\lambda I \succeq 0, \quad \lambda \geq 0, \quad \lambda(\Delta - \|d\|) = 0. \tag{33}$$

The Moré-Sorensen method [MS83] is based on solving nonlinear equation $1/\|d(\lambda)\| = 1/\Delta$ with $(B + \lambda I)d(\lambda) + g = 0$ by the Newton method using the sparse Choleski decomposition of $B + \lambda I$. This method is very robust but requires 2-3 Choleski decompositions per iteration.

Simpler methods are based on minimization of $Q(d)$ on the two-dimensional subspace containing Cauchy step $d_C = -(g^T g / g^T B g) g$ and Newton step $d_N = -B^{-1} g$. The most popular is the dog-leg method [Pow70], [DM75], where $d = d_N$ if $d_N \leq \Delta$ and $d = (\Delta / \|d_C\|) d_C$ if $\|d_C\| \geq \Delta$. In the remaining case, d is a convex combination of d_C and d_N such that $\|d\| = \Delta$. This method requires only one Choleski decomposition per iteration.

If B is not sufficiently sparse, then the sparse Choleski decomposition of B is expensive. In this case, iterative methods based on conjugate gradients are more suitable. Steihaug [Ste83] and Toint [Toi81] proposed a method based on the fact that $Q(d_{k+1}) < Q(d_k)$ and $\|d_{k+1}\| > \|d_k\|$ hold in the subsequent CG iterations if CG coefficients are positive. We either obtain an unconstrained solution with a sufficient precision or stop on the trust-region boundary if a negative curvature is indicated or the trust-region is left. This method is very efficient in practice especially when suitable preconditioning is used. Note that $\|d_{k+1}\|_C > \|d_k\|_C$ (where $\|d_k\|_C^2 = d_k^T C d_k$) holds instead of $\|d_{k+1}\| > \|d_k\|$ if preconditioner C (symmetric and positive definite) is used. Thus the solution on the trust-region boundary obtained by the preconditioned CG method can be further from the optimal locally constrained step than the solution obtained without preconditioning. This insufficiency is usually compensated by the rapid convergence of the preconditioned CG method.

The CG steps can be combined with Newton step d_N in the multiple dog-leg method [Ste83], [Luk96a]. Let $k \ll n$ (usually $k = 5$) and d_k be a vector

obtained after k CG steps of the Steihaug-Toint method. If $\|d_k\| < \Delta$, we use d_k instead of $d_C = d_1$ in the dog-leg method.

The solution on the trust-region boundary obtained by the Steihaug-Toint method can be rather far from the optimal solution. This insufficiency can be overcame by using the Lanczos process [GLRT99]. Initially, the conjugate gradient algorithm is used as in the Steihaug-Toint method. At the same time, the Lanczos tridiagonal matrix is constructed from the CG coefficients. If a negative curvature is indicated or the trust-region is left, we turn to the Lanczos process. In this case, $d = Z\tilde{d}$, where $\tilde{d}$ is obtained by minimizing quadratic function

$$\frac{1}{2}\tilde{d}^T T\tilde{d} + \|g\|e_1^T\tilde{d}$$

subject to $\|\tilde{d}\| \leq \Delta$. Here $T = Z^T BZ$ (with $Z^T Z = I$) is the Lanczos tridiagonal matrix and e_1 is the first column of the unit matrix. This method cannot be successfully preconditioned, since preconditioning changes the original trust-region subproblem to $\|d\|_C \leq \Delta$, where C changes in each major iteration and can be ill-conditioned.

To overcome the insufficiency of the previous method, the Lanczos process can be combined with the Steihaug-Toint method. The shifted Steihaug-Toint method proposed in [LMV04] consists of three steps:

- Let $m \ll n$ (usually $m = 5$). Determine tridiagonal matrix T of order m by m steps of the (unpreconditioned) Lanczos method applied to matrix B with the initial vector g.
- Solve subproblem

$$\text{minimize} \quad \frac{1}{2}\tilde{d}^T T\tilde{d} + \|g\|e_1^T\tilde{d} \quad \text{subject to} \quad \|\tilde{d}\| \leq \Delta \tag{34}$$

  using the method of Moré and Sorensen to obtain Lagrange multiplier $\tilde{\lambda}$.
- Apply the (preconditioned) Steihaug-Toint method to subproblem

$$\text{minimize} \quad \frac{1}{2}d^T(B + \tilde{\lambda}I)d + g^T d \quad \text{subject to} \quad \|d\| \leq \Delta \tag{35}$$

  to obtain direction vector $d = d(\tilde{\lambda})$.

The following theorem is proved in [LMV04].

Theorem 6. Let $\tilde{\lambda}$ be the Lagrange multiplier of the small-size subproblem (34) and λ be the Lagrange multiplier obtained by the Moré-Sorensen method applied to the original problem. Then $0 \leq \tilde{\lambda} \leq \lambda$.

As a consequence of Theorem 6, one has that $\lambda = 0$ implies $\tilde{\lambda} = 0$ so that $\|d\| < \Delta$ implies $\tilde{\lambda} = 0$. Thus the shifted Steihaug-Toint method reduces to the standard one in this case. At the same time, if B is positive definite and $\tilde{\lambda} > 0$, then one has $\Delta \leq \|(B + \tilde{\lambda}I)^{-1}g\| < \|B^{-1}g\|$. Thus the unconstrained minimizer of the shifted quadratic function (35) is closer to the trust-region

boundary than the unconstrained minimizer of the original quadratic function (32) and we can expect that $d(\tilde{\lambda})$ is closer to the optimal locally constrained step than d. Finally, if $\tilde{\lambda} > 0$, then matrix $B + \tilde{\lambda} I$ is better conditioned than B and we can expect that the shifted Steihaug-Toint method will converge more rapidly than the original one.

Methods for solving large-scale trust-region subproblems were tested by using a set of 22 sparse test problems with 1000 and 5000 variables implemented in subroutine `TEST14` (see `www.cs.cas.cz/~luksan/test.html`). The results are presented in Table 6, where n is the number of variables, "method" is the method used (MS - the optimum trust-region step of Moré and Sorensen [MS83], DL - the dog-leg strategy of Powell [Pow70], MDL - the multiple dog-leg strategy [Luk96a] with $m = 5$, ST - the basic Steihaug-Toint method, GLRT - the method of Gould, Lucidi, Roma and Toint [GLRT99] based on the Lanczos process, PST - the preconditioned Steihaug-Toint method (with the incomplete Choleski preconditioner), PSST - the preconditioned shifted Steihaug-Toint method [LMV04] with $m = 5$), NIT is the total number of iterations, NEV is the total number of function evaluations, NCG is the total number of CG iterations and "time" is the total computational time in seconds.

n	method	NIT	NEV	NCG	time
1000	MS	1918	1955	-	4.65
	DL	2515	2716	-	4.42
	MDL	2292	2456	12203	4.61
	ST	3329	3784	53573	8.20
	GLRT	3107	3444	55632	8.53
	PST	2631	2823	910	5.14
	PSST	1999	2046	1161	4.25
5000	MS	8391	8566	-	122.44
	DL	9657	10133	-	115.77
	MDL	8938	9276	47236	122.84
	ST	16894	19163	358111	364.42
	GLRT	14679	16383	366695	401.45
	PST	10600	11271	3767	145.42
	PSST	8347	8454	4329	108.87

Table 6

The results presented in this table imply the following conclusions:

- Direct methods MS and DL based on the sparse Choleski decomposition are very efficient for our set of test problems. Iterative methods require a suitable preconditioning.

- The Moré-Sorensen strategy MS gives the best approximation of the optimum locally constrained step and decreases the number of the major iterations.
- New strategy PSST can be efficiently preconditioned. It gives a relatively good approximation of the optimum locally constrained step. Method PSST is the most efficient method for solving our set of test problems.

Acknowledgment

This work was supported by the Grant Agency of the Czech Academy of Sciences, project code IAA1030405

References

[AF85] Al-Baali, M. Fletcher, R.: Variational Methods for Nonlinear Least Squares. Journal of Optimization Theory and Applications, **36**, 405-421 (1985).

[BNS94] Byrd, R.H., Nocedal, J., Schnabel, R.B.: Representation of quasi-Newton matrices and their use in limited memory methods. Mathematical Programming, **63**, 129-156 (1994).

[DGW81] Dennis, J.E., Gay, D., Welsch, R.E.: An adaptive nonlinear least squares algorithm. ACM Transactions on Mathematical Software, **7**, 348-368 (1981).

[DM75] Dennis, J.E., Mei, H.H.W: An unconstrained optimization algorithm which uses function and gradient values. Report No. TR 75-246, Department of Computer Science, Cornell University, Ithaca 1975.

[GL01] Gill, P.E., Leonard, M.W: Reduced-Hessian quasi-Newton methods for unconstrained optimization. SIAM Journal on Optimization, **12**, 209-237 (2001).

[GL03] Gill, P.E., Leonard, M.W: Limited-memory reduced-Hessian methods for large-scale unconstrained optimization. SIAM Journal on Optimization, **14**, 380-401 (2003).

[GM74] Gill, P.E., Murray, W: Newton type methods for unconstrained and linearly constrained optimization. Mathematical Programming, **7**, 311-350 (1974).

[GMS75] Gill, P.E., Murray, W, Saunders, M.A.: Methods for computing and modifying LDV factors of a matrix. Mathematics of Computation, **29**, 1051-1077 (1975).

[GLRT99] Gould, N.I.M, Lucidi, S., Roma, M., Toint, P.L.: Solving the trust-region subproblem using the Lanczos method. SIAM Journal on Optimization, **9**, 504-525 (1999).

[GT82] Griewank, A., Toint, P.L.: Partitioned variable metric updates for large-scale structured optimization problems. Numerische Mathematik, **39**, 119-137 (1982).

[HMM03] Haarala, M., Mietinen, K., Mäkelä, M.M.: Limited memory bundle method for large-scale nonsmooth optimization: Convergence analysis. Report No. B 12/2003, Department of Mathematical Information Technology, University of Jyväskylä, Jyväskylä 2003.

[HMM04] Haarala, M., Mietinen, K., Mäkelä, M.M.: New limited memory bundle method for large-scale nonsmooth optimization. Optimizaton Methods and Software, **19**, 673-692 (2004).

[Kiw85] Kiwiel, K.C.: Methods of Descent for Nondifferentiable Optimization, Lecture Notes in Mathematics 1133, Springer-Verlag, Berlin 1985.

[Lem89] Lemarechal, C.: Nondifferentiable Optimization. In: Optimization (Nemhauser, G.L., Rinnooy-Kan, A.H.G, Todd, M.J. eds.), Elsevier Science Publishers, North-Holland, Amsterdam 1989.

[Luk96a] Lukšan, L.: Combined trust region methods for nonlinear least squares. Kybernetika, **32**, 121-138 (1996).

[Luk96b] L.Lukšan: Hybrid methods for large sparse nonlinear least squares. Journal of Optimization Theory and Applications, **89**, (1996) 575-595.

[LMV04] Lukšan, L., Matonoha, C., Vlček, J.: A shifted Steihaug-Toint method for computing a trust-region step. Report V-914, Institute of Computer Science, Academy of Sciences, Prague 2004.

[LS00] Lukšan, L., Spedicato, E.: Variable metric methods for unconstrained optimization and nonlinear least squares. Journal of Computational and Applied Mathematics, **124**, 61-93 (2000).

[LV99] Lukšan, L., Vlček, J.: Globally convergent variable metric method for convex nonsmooth unconstrained minimization. Journal of Optimization Theory and Applications, **102**, 593-613 (1999).

[LV04] Lukšan, L., Vlček, J.: Variable metric method for minimization of partially separable nonsmooth functions. Report V-916, Institute of Computer Science, Academy of Sciences, Prague 2004.

[MN92] Mäkelä, M.M., Neittaanmäki, P: Nonsmooth Optimization. World Scientific Publishing Co., London 1992.

[MS83] Moré, J.J., Sorensen, D.C.: Computing a trust region step. SIAM Journal on Scientific and Statistical Computations, **4**, 553-572 (1983).

[Noc80] Nocedal, J.: Updating quasi-Newton matrices with limited storage. Mathematics of Computation **35** (1980) 773-782.

[Pow70] Powell, M.J.D.: A new algorithm for unconstrained optimization. In: Nonlinear Programming (Rosen, J.B., Mangasarian, O.L., Ritter, K., eds.) Academic Press, London 1970.

[Pow84] Powell, M.J.D.: On the global convergence of trust region algorithms for unconstrained optimization. Mathematical Programming, **29**, 297-303 (1984).

[Ste83] Steihaug, T.: The conjugate gradient method and trust regions in large-scale optimization. SIAM Journal on Numerical Analysis, **20**, 626-637 (1983).

[Toi81] Toint, P.L.: Towards an efficient sparsity exploiting Newton method for minimization. In: Sparse Matrices and Their Uses (Duff, I.S., ed.), Academic Press, London 1981.

[VL01] Vlček, J., Lukšan, L.: Globally convergent variable metric method for nonconvex nondifferentiable unconstrained minimization. Journal of Optimization Theory and Applications, **111**, 407-430 (2001).

[VL02] Vlček, J., Lukšan, L.: New variable metric methods for unconstrained minimization covering the large-scale case. Report V-876, Institute of Computer Science, Academy of Sciences, Prague 2002.

[VL05] Vlček, J., Lukšan, L.: Shifted limited-memory variable metric methods for large-scale unconstrained minimization. Journal of Computational and Applied Mathematics (in press).

A variational approach for minimum cost flow problems

Giandomenico Mastroeni[1]

Department of Mathematics, Largo B. Pontecorvo 5, 56127 - PISA, Italy
(mastroen@dm.unipi.it)

Summary. We consider a variational model for traffic network problems, which generalizes the classic minimum cost flow problem. This model has the peculiarity of being formulated by means of a suitable variational inequality. The Lagrangean approach to the study of this variational inequality allows us to consider dual variables associated with the constraints of the feasible set, and to generalize the classic Bellman optimality conditions in order to obtain a stopping criterion for a gap function algorithm.

Key words: network flows, variational inequalities, equilibrium problems, gap functions.

1 Introduction

In this paper we aim to deepen the analysis of a variational model for minimum cost network flow problems, introduced in [MP04]. The model requires the solution of a variational inequality defined on the usual feasible set of the linear minimum cost flow problem, where the constraints are given by the flow conservation equations at the nodes and the capacities on the arcs. One of the main applications of this model is the study of traffic equilibrium on a network, however, similarly to the classic minimum cost flow problems, further applications can be found in economic or computer networks. Variational inequality models have been considered by several authors: we refer to [FP03, Pat99] and references therein, for a detailed description of the development of this topic in the literature.

Let $K := \{x \in \mathbb{R}^n : g(x) \leq 0,\ h(x) = 0\}$ and consider the variational inequality which consists in finding $y \in K$ such that

[1] This work has been supported by the National Research Program FIRB/RBNE01WBBB "Large Scale Nonlinear Optimization"

$$\langle F(y), x - y \rangle \geq 0, \quad \forall x \in K, \qquad VI(F,K)$$

where $F : \mathbb{R}^n \longrightarrow \mathbb{R}^n$, $g : \mathbb{R}^n \longrightarrow \mathbb{R}^p$ and $h : \mathbb{R}^n \longrightarrow \mathbb{R}^m$.

In our analysis we will make use of the well-known Lagrangian-type optimality conditions for $VI(F,K)$ which state, under suitable constraint qualifications and in the hypothesis that K is a convex set, that $y \in \mathbb{R}^n$ is a solution of $VI(F,K)$ if and only if there exist $\lambda^* \in \mathbb{R}^m$, $\mu^* \in \mathbb{R}^p$ such that (y, λ^*, μ^*) is a solution of the system

$$\begin{cases} F(x) + \sum_{i=1}^{p} \mu_i \nabla g_i(x) + \sum_{j=1}^{m} \lambda_j \nabla h_j(x) = 0 \\ \langle \mu, g(x) \rangle = 0 \\ \mu \geq 0, \ g(x) \leq 0, \ h(x) = 0 \end{cases} \tag{1}$$

In Section 2 we consider a generalized minimum cost flow problem, formulated by means of the variational inequality $VI(F,K)$, where the operator F represents the cost associated to the arcs of the network. In such a case, it has been shown [MP04] that the multipliers (λ^*, μ^*) can be interpreted in terms of potentials associated with the nodes and the arcs of the network. In particular, we obtain a generalization of the Bellman dual optimality conditions for the classic minimum cost flow problem, that we will use as stopping criterion in an iterative algorithm for solving $VI(F,K)$ that will be presented in Section 3. This algorithm is a slight variant of the Fukushima method for the minimization of a gap function associated with $VI(F,K)$ [Fuk92] and turns out to be closely related to the auxiliary problem principle for $VI(F,K)$ [Coh88]. The last section is devoted to the computational considerations related to the practical implementation of the proposed algorithm.
Let us recall the main definitions that will be used in the sequel.

A function $h : \mathbb{R}^n \longrightarrow \mathbb{R}$ is said strongly convex on the convex set $K \subseteq \mathbb{R}^n$, with modulus $a > 0$, iff, $\forall x_1, x_2 \in K$ and $\forall \lambda \in [0,1]$,

$$h(\lambda x_1 + (1-\lambda)x_2) \leq \lambda h(x_1) + (1-\lambda)h(x_2) - a[\lambda(1-\lambda)/2]\|x_1 - x_2\|^2.$$

We will say that the mapping $F : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ is monotone on K iff:

$$\langle F(y) - F(x), y - x \rangle \geq 0, \quad \forall x, y \in K;$$

it is strictly monotone if strict inequality holds $\forall x \neq y$.
We will say that the mapping F is strongly monotone on K, with modulus $a > 0$, iff:

$$\langle F(y) - F(x), y - x \rangle \geq a\|y - x\|^2, \quad \forall x, y \in K;$$

F is Lipschitz continuous on K, with modulus $L > 0$, iff

$$\|F(x) - F(y)\| \leq L\|x - y\|, \quad \forall x, y \in K.$$

2 The variational model

Given a network represented by the graph $G = (N, A)$, where $N := \{1, \ldots, m\}$ is the set of the nodes and $A := \{A_1, \ldots, A_n\}$ is the set of the arcs, we consider a variational inequality model formulated in terms of the flows on the arcs. In general, in the literature, such models are formulated considering the flows on the paths. Our model is a generalization of the minimum cost flow problem to which it collapses when the operator of the variational inequality is a constant. Let us introduce the following assumptions and notations:

- f_i is the flow on the arc $A_i := (r, s)$ and $f := (f_1, \ldots, f_n)^T$ is the vector of the flows on all arcs.
- We assume that each arc A_i is associated with an upper bound d_i on its capacity, $d := (d_1, \ldots, d_n)^T$.
- $c_i(f)$ is the unit cost on the arc A_i as function of the flows, $c(f) := (c_1(f), \ldots, c_n(f))^T$; we assume that $c(f) \geq 0$.
- q_j is the balance at the node j, $q := (q_1, \ldots, q_m)^T$.
- $\Gamma = (\gamma_{ij}) \in \mathbb{R}^m \times \mathbb{R}^n$ is the node-arc incidence matrix whose elements are

$$\gamma_{ij} = \begin{cases} -1, \text{ if } i \text{ is the initial node of the arc } A_j, \\ +1, \text{ if } i \text{ is the final node of the arc } A_j, \\ 0, \quad \text{otherwise.} \end{cases} \tag{2}$$

The equilibrium model is defined by a variational inequality having c as operator and feasible set given by the classic flow conservation equations at the nodes and capacities on the arcs.

Definition 1. *f^* is an equilibrium flow iff*

$$\langle c(f^*), f - f^* \rangle \geq 0, \quad \forall f \in K_f, \tag{$VI(c, K_f)$}$$

where

$$K_f := \{f \in \mathbb{R}^n \ : \ \Gamma f = q, \ 0 \leq f \leq d\}.$$

$VI(c, K_f)$ is equivalent to the problem of finding $f^* \in K_f$ s.t. f^* is an optimal solution of

$$\min_{f \in K_f} \langle c(f^*), f \rangle \tag{3}$$

The problem (3) collapses to the minimum cost network flow problem when the function $c(f)$ is independent of f, namely, $c(f) := (c_{ij}, (i, j) \in A)$.

In the classic approach, the equilibrium model is defined by the optimization problem

$$\min_{f \in K_f} \langle c(f), f \rangle \tag{4}$$

The solution f^* of $VI(c, K_f)$ can be considered as the user equilibrium, while the optimal solution of (4) as the network equilibrium perceived by an external observer. To this end, we recall that, if we suppose that the (unit) path cost function is additive, then the Wardrop equilibrium principle [War52] can be expressed in terms of a variational inequality having $c(f)$ as operator, and defined on a suitable feasible set that takes into account the demands between the origin–destination pairs [Daf80, FP03]. For example, consider the particular case where the set of origins consists of only one node, say it $\bar{i}$, and the set of destinations is given by $N \setminus \{\bar{i}\}$. If we let $q_j \geq 0$ be the demand of the couple $(\bar{i}, j)$, $j = 1, \ldots, m$, $j \neq \bar{i}$, $q_{\bar{i}} = -\sum_{j \neq \bar{i}} q_j$ and $d_i = +\infty$, $i = 1, \ldots, n$, then it is easy to show that a solution of $VI(c, K_f)$ is a Wardrop equilibrium flow.

The following well-known results provide a primal-dual formulation of $VI(c, K_f)$.

Proposition 1. *f^* is a solution of $VI(c, K_f)$ if and only if there exists $(\lambda^*, \mu^*) \in \mathbb{R}^{m \times n}$ such that (f^*, λ^*, μ^*) is a solution of the system*

$$\begin{cases} c(f) + \Gamma^T \lambda + \mu \geq 0 \\ \langle c(f) + \Gamma^T \lambda + \mu, f \rangle = 0 \\ \langle f - d, \mu \rangle = 0 \\ 0 \leq f \leq d \ , \ \Gamma f = q, \ \mu \geq 0 \end{cases} \tag{5}$$

Proof. It is enough to observe that, at $f := f^*$, the system (5) coincides with the Kuhn-Tucker conditions for the problem (3), that, in our hypotheses, are necessary and sufficient for the optimality of f^*. □

By means of system (5), we can immediately derive the following equilibrium principle.

Theorem 1. *f^* is a solution of $VI(c, K_f)$ if and only if there exist $\lambda^* \in \mathbb{R}^m$ and $\mu^* \in \mathbb{R}^n_+$ such that, $\forall (i,j) \in A$:*

$$0 < f^*_{ij} < d_{ij} \Longrightarrow c_{ij}(f^*) = \lambda^*_i - \lambda^*_j, \quad \mu^*_{ij} = 0, \tag{6}$$

$$f^*_{ij} = 0 \Longrightarrow c_{ij}(f^*) \geq \lambda^*_i - \lambda^*_j, \quad \mu^*_{ij} = 0, \tag{7}$$

$$f^*_{ij} = d_{ij} \Longrightarrow c_{ij}(f^*) = \lambda^*_i - \lambda^*_j - \mu^*_{ij}. \tag{8}$$

Remark 1. The dual variables corresponding to the flow conservation constraints can be interpreted in terms of potentials at the nodes of the network. If we assume that $c(f) \geq 0$,then, from (6) and (8) we deduce that

$$f^*_{ij} > 0 \Longrightarrow \lambda^*_i - \lambda^*_j \geq 0.$$

that is, a positive flow on the arc (i, j) implies that the difference of potential between the nodes i and j is positive.

We observe that conditions (6), (7), (8) are a generalization to VI of the classic Bellman conditions for the linear minimum cost network flow problem. Actually, given $f \in K_f$, define the so called "reduced costs" by $\bar{c}_{ij} := c_{ij}(f) - \lambda_i + \lambda_j$, $(i,j) \in A$, where $\lambda \in \mathbb{R}^m$ is chosen in order to fulfil the system

$$\bar{c}_{ij} = 0, \quad (i,j) \in \{(i,j) \in A : 0 < f_{ij} < d_{ij}\}, \tag{9}$$

so that Theorem 1 can be reformulated in the following way.

Proposition 2. *$f^* \in K_f$ is an optimal solution of $VI(c, K_f)$ if*

$$\bar{c}_{ij} \geq 0, \quad \forall (i,j) \in \{(i,j) \in A : f^*_{ij} = 0\}, \tag{10}$$

$$\bar{c}_{ij} \leq 0, \quad \forall (i,j) \in \{(i,j) \in A : f^*_{ij} = d_{ij}\}. \tag{11}$$

Proof. It is enough to eliminate μ from (7) and (8), taking into account that $\mu \geq 0$. □

When $c_{ij}(f)$ is independent of f, then (10), (11) collapse to the well-known Bellman conditions. We will use these relations as a stopping criterion in an iterative method for $VI(c, K_f)$ that will be presented in the next section.

3 An algorithm for $VI(c, K_f)$

Two important classes of algorithms for VI are those based on the auxiliary problem principle [Coh88] and those related to the minimization of a gap function [Fuk92]. Both methods have the peculiarity of using the solution of the same subproblem, at the current iteration, in order to define the next one. We present a line search algorithm for the minimization of a gap function which is closely related to the previous ones and collapses to one or to the other depending on the choice of suitable parameters and of the step used in the line search.

We recall the notion of gap function associated to $VI(F, K)$.

Definition 2. *Let $K \subseteq \mathbb{R}^n$. $p : \mathbb{R}^n \longrightarrow \mathbb{R}$ is a gap function for $VI(F, K)$ iff:*
i) $p(x) \geq 0, \quad \forall x \in K$;
ii) $p(x) = 0$ iff x is a solution for $VI(F, K)$.

A class of continuously differentiable gap functions has been defined by Fukushima [Fuk92] and subsequently generalized by Zhu and Marcotte [ZM94].

Proposition 3. [ZM94] *Let K be a convex set in $\mathbb{R}^n$ and let $G(x, y) : \mathbb{R}^n \times \mathbb{R}^n \longrightarrow \mathbb{R}$ be a non-negative, continuously differentiable, strongly convex function on K, with respect to y, $\forall x \in K$, such that*

$$G(y,y) = 0, \quad \text{and} \quad \nabla_y G(y,y) = 0, \quad \forall y \in K.$$

Then

$$p(x) := max_{y \in K}\{\langle F(x), x - y\rangle - G(x,y)\} \tag{12}$$

is a gap function for VI(F,K). Moreover, if $F \in C^1$, *then* $p \in C^1$ *and*

$$\nabla p(x) = F(x) - (\nabla F(x))^T (y(x) - x) - \nabla_x G(x, y(x)) \tag{13}$$

where $y(x)$ *is the unique solution of the maximization problem appearing on the right-hand side of (12).*

Consider the problem $VI(c, K_f)$ and the gap function p defined by

$$p(f) := \max_{y \in K_f}\{\langle c(f), f - y\rangle - \frac{\alpha}{2}\|f - y\|^2\}. \tag{14}$$

If c is continuously differentiable, then p is a continuously differentiable gap function [Fuk92]. Let us analyse more in detail the properties of the gap function (14). The following results have been either established by Fukushima or can be easily derived from his work.
Denote by $P(f)$ the extremum problem defined in (14).

Lemma 1. *f^* is an optimal solution of $VI(c, K_f)$ if and only if $y(f^*) = f^*$, where $y(f)$ is the optimal solution of $P(f)$.*

Proof. Let

$$\phi(y) := \langle c(f^*), f^* - y\rangle - \frac{\alpha}{2}\|f^* - y\|^2.$$

If f^* is an optimal solution of $VI(c, K_f)$, then $\phi(y) \leq 0$, $\forall y \in K_f$. Since ϕ is a strongly concave function then the problem

$$\max_{y \in K_f} \phi(y) \qquad P(f^*)$$

admits a unique solution that coincides with f^* because $\phi(f^*) = 0$.

Vice versa, suppose that $y(f^*) = f^*$. Since K_f is a convex set, then the variational inequality which represents the first order optimality condition for $P(f^*)$, at the point $y(f^*)$, holds:

$$\langle -c(f^*) + \alpha(f^* - y(f^*)), z - y(f)\rangle \leq 0, \quad \forall z \in K_f.$$

It follows that f^* is a solution of $VI(c, K_f)$. □

Proposition 4. *Suppose that c is a continuously differentiable operator on K_f. Let f^* be an optimal solution of $VI(c, K_f)$ and (λ^*, μ^*) be the Lagrange multipliers associated with the problem $P(f^*)$.*

Then

i) $\nabla p(f^*) = c(f^*)$;

ii) $(f^*, -\lambda^*, -\mu^*)$ *is a solution of system (5).*

Proof. Put $G(f,y) := \frac{\alpha}{2}\|f-y\|^2$.
i) By Lemma 1, f^* is an optimal solution of $VI(c,K_f)$ if and only if $y(f^*) = f^*$. Observing that $\nabla_f G = \alpha(f-y)$, by the formula (13) we have that

$$\nabla p(f^*) = c(f^*) - (\nabla c(f^*))^T(y(f^*) - f^*) - \alpha(f^* - y(f^*)),$$

which proves the statement *i)*.

ii) Since f^* is an optimal solution of $VI(c,K_f)$, then f^* is optimal for $P(f^*)$. The gradient $\nabla_y\{\langle c(f^*), f^*-y\rangle - \frac{\alpha}{2}\|f^*-y\|^2\}$, evaluated at f^* is equal to $-c(f^*)$, so that the Kuhn-Tucker conditions for $P(f^*)$ coincide, up to a change of the sign of the multipliers, with those related to $VI(c,K_f)$ (defined by system (1)) and, therefore, with (5).

□

The optimal solution of $P(f)$ allows us to determine a descent direction for p at the point $f \in K_f$.

Proposition 5. [Fuk92] *Let $y(f)$ be the optimal solution of $P(f)$. Then $d := y(f) - f$ is a descent direction for p at the point $f \in K_f$, provided that c is a continuously differentiable strictly monotone operator on K_f and $y(f) \neq f$.*

Proof. By (13), we have that

$$\nabla p(f) = c(f) - (\nabla c(f))^T(y(f) - f) - \alpha(f - y(f)),$$

so that

$$\langle \nabla p(f), y(f)-f\rangle = \langle c(f), y(f)-f\rangle - \langle (\nabla c(f))^T(y(f)-f), y(f)-f\rangle + \alpha\|(y(f)-f)\|^2.$$

Since $y(f)$ is the optimal solution of $P(f)$ and K_f is convex, then the following variational inequality holds:

$$\langle -c(f) + \alpha(f - y(f)), z - y(f)\rangle \leq 0, \quad \forall z \in K_f. \tag{15}$$

Computing (15) for $z = f$, we obtain

$$\langle c(f), y(f) - f\rangle + \alpha\|y(f) - f\|^2 \leq 0. \tag{16}$$

Taking into account that c is a differentiable strictly monotone operator, we have that $\nabla c(f)$ is positive definite, $\forall f \in K_f$ [OR70], which, together with (16), implies that $\langle \nabla p(f), y(f) - f\rangle < 0$. □

These remarkable properties allow us to consider the following line-search algorithm in order to minimize the gap function p.

Algorithm.
Let p be defined by (14) and $\alpha, \epsilon, \xi > 0$ be fixed positive parameters.

Step 1. Let $k = 0$, $f^0 \in K_f$;

Step 2. $f^{k+1} := f^k + t_k d^k$, $k = 1, 2, \ldots$
where $d^k := y(f^k) - f^k$, $y(f^k)$ is the optimal solution of the problem

$$\max_{y \in K_f} \{\langle c(f^k), f^k - y\rangle - \frac{\alpha}{2}\|f^k - y\|^2\}. \qquad P(f^k)$$

Selected a random number $\bar{t}_k \in (0, 1]$, put $t_k := \bar{t}^k$ if
$$p(f^k + \bar{t}^k d^k) < p(f^k) - \epsilon,$$
otherwise t_k is chosen as a positive random number sufficiently close to zero.

Step 3. If $p(f^{k+1}) < \xi$, then go to Step 4, otherwise put $k = k + 1$ and go to Step 2.

Step 4. Let (λ^*, μ^*) be the vector of the Lagrange multipliers associated to the solution $y(f^{k+1})$ of the problem $P(f^{k+1})$. If $(f^{k+1}, -\lambda^*, -\mu^*)$ fulfils the relations (6),(7),(8), then f^{k+1} is an optimal solution of $VI(c, K_f)$, otherwise put $\xi = \xi/2$, $k = k + 1$ and go to Step 2.

Step 2 requires the solution of a strongly concave quadratic maximization problem. With regard to Step 4, Proposition 4 shows that the opposite of the Lagrange multipliers associated to the solution $y(f^{k+1})$ of the problem $P(f^{k+1})$ are also solutions of the system (5) if f^{k+1} is a solution of $VI(c, K_f)$: recalling Proposition 1 and Theorem 1, they can be employed for checking if the relations (6),(7),(8) are fulfilled.

Remark 2. Suppose that c is a strongly monotone operator, with modulus a, and Lipschitz continuous, with modulus L, on K_f. If we choose $\alpha > L^2/2a$ and $t_k = 1, \forall k = 1, \ldots$, then the algorithm coincides with the one based on the auxiliary problem principle stated by Cohen [Coh88].
If t_k is chosen in order to be a solution (exact or inexact) of the problem

$$\min_{0 \le t \le 1} p(f^k + t d^k), \qquad (17)$$

then the algorithm collapses to the gap function method proposed by Fukushima [Fuk92].
Both methods are globally convergent under the above mentioned hypotheses.

4 Computational considerations and concluding remarks

We observe that the method based on the "auxiliary problem principle" and the "gap function" method are very similar. The crucial point of the two

methods lies in the fact that the search direction $d^k := y(f^k) - f^k$ is a descent direction for the gap function g at the point f^k. In the former, we set $f^{k+1} := y(f^k)$, while, in the latter, the stepsize is given by the optimal solution of problem (17). Therefore, the methods coincide if the stepsize t_k can be taken always equal to one: this is possible if the parameter α is chosen according to the rules mentioned in Remark 2. The auxiliary principle method has the drawback that the constant α must be determined with a sufficient degree of accuracy, which is not an easy task, since the Lipschitz constant or the modulus of strong monotonicity of the operator c may not be known. On the contrary, gap function methods do not need the evaluation of any constant but require an inexact (or exact) line-search at each step, which increases the computational cost. Numerical experiments show that, at least in the first iterations, it is not convenient to make too much effort in performing the line search, since the value of the gap function may be very high.
Taking into account these observations, in our implementation we consider the descent direction d^k for the gap function p, but without performing the exact minimization (17): first, we select a point $\bar{y}(f^k)$ in the segment $(f^k, y(f^k))$ close to $y(f^k)$ and, in case $p(\bar{y}(f^k)) > p(f^k) - \epsilon$, we make a null step, and choose f^{k+1} sufficiently close to f^k, in order to sligthly perturb the current point.

We have applied the Algorithm to the problem $VI(c, K_f)$ with the following features:

- the incidence matrix $\Gamma \in \mathbb{R}^{m \times n}$, the balance vector $q \in \mathbb{R}^m_+$ and the vector of the capacities $d \in \mathbb{R}^n_+$ are randomly chosen, provided that $\sum_{i=1}^m q_i = 0$.
- Three different choices have been made for the operator c: in the first two choices, c is a linear operator, $c(f) := C_1 f + \beta$, or $c(f) := C_2 f + \beta$ where C_1 and C_2 are non symmetric $n \times n$ randomly chosen non negative matrices with C_1 positive definite, C_2 indefinite and $\beta \in \mathbb{R}^n_+$. In the third choice c is a non linear operator $c(f) := \bar{c}(f) = (\bar{c}_i(f_i), i = 1, \ldots, n)$, where $\bar{c}_i(f_i) := a_i + k_i(f_i/b_i)^s$ and a_i, k_i, b_i, s are suitable positive parameters, for $i = 1, \ldots, n$. Observe that if $c(f) = C_1 f + \beta$ or $c(f) = \bar{c}(f)$, then the operator is strongly monotone or monotone, respectively.

Preliminary numerical tests have been performed using the Matlab software. Table 1 reports the results obtained with the following choices for the parameters: $f^0 \in K_f$ is a random starting point, $\xi = 0.1$, $\epsilon = 10^{-5}$, $\alpha = 1.5$, $\beta_i = 1$, $a_i = 1$, $k_i \in (0, 1)$ a random positive number and $b_i = d_i$, for $i = 1, \ldots, n$, $s = 2$.

For each couple of values, m and n, of the nodes and the arcs, we report the average number of the iterations and the computation times, obtained by performing the algorithm a sufficiently large number of times. The number of iterations includes null steps and coincides with the number of evaluations of the gap function p.

Table 1: Number of iterations and cpu time in seconds

		$c(f) = C_1 f + \beta$		$c(f) = C_2 f + \beta$		$c(f) = \bar{c}(f)$	
m	n	iterations	cpu	iterations	cpu	iterations	cpu
10	25	45	3.6	53	5.1	16	0.9
15	50	54	7.1	51	7.0	26	1.8
15	80	62	14.9	66	18.2	28	8.8
20	100	67	27.3	79	35.1	38	19.9
25	150	74	106.6	78	107.2	46	31.7

For $n \geq 200$, computational time increases drastically, although the convergence is still reached in a number of iterations generally not greater than one hundred. We observe that the lack of monotonicity of the operator does not seem to affect the convergence of the algorithm, but only causes an increase in the number of null steps. Since the number of iterations is not very sensitive to the problem size, the algorithm might be suitable for large scale problems provided that a large scale subroutine is used for solving the problem $P(f)$, which enables us to evaluate $p(f)$.

As regards further developments of the analysis, we remark that gap function theory can also be employed in the study of vector variational inequalities and equilibrium problems [Mas00,CYG00,BO94,Mas03]. Future research may be devoted to the extension of the proposed algorithm to vector variational inequalities and, following the line developed in [Mas03], to the applications to equilibrium problems. Moreover, it is of interest to deepen the analysis of the relationships with similarly structured methods as proximal point or trust region methods.

References

[BO94] Blum E., Oettly W.: From Optimization and Variational Inequalities to Equilibrium Problems. Math. Student, **63**, 123–145 (1994).

[CYG00] Chen, G.Y., Goh, C.J., Yang, X.Q.: On Gap Functions for Vector Variational Inequalities. In: F. Giannessi, F. (ed) Vector Variational Inequalities and Vector Equilibria. Mathematical Theories, Kluwer, Dordrecht, 55–72 (2000).

[Coh88] Cohen, G: Auxiliary Problem Principle Extended to Variational Inequalities. Journal of Optimization Theory and Applications, **59** 325–333 (1988).

[Daf80] Dafermos, S: Traffic equilibria and variational inequalities. Mathematical Programming, **26** 40–47 (1980).

[FP03] Facchinei, F.,Pang, J.S: Finite–dimensional variational inequalities and complementarity problems. Springer, Berlin Heidelberg New York (2003).

[Fuk92] Fukushima, M.: Equivalent differentiable optimization problems and descent methods for asymmetric variational inequality problems. Mathematical Programming, **53**, 99–110 (1992).

[Mas00] Mastroeni, G.: Separation methods for Vector Variational Inequalities. Saddle point and gap function. In: Di Pillo, G., Giannessi, F. (eds), Nonlinear Optimization and Related Topics, Kluwer, Dordrecht, 207–218 (2000).

[Mas03] Mastroeni, G.: Gap functions for Equilibrium Problems. Journal of Global Optimization, **27**, 411–426 (2003).

[MP04] Mastroeni, G., Pappalardo, M.: A Variational Model for Equilibrium Problems in a Traffic Network. RAIRO Operations Research, **38**, 3–12 (2004).

[OR70] Ortega, J.M., Rheinboldt, W.C.: Iterative solutions of nonlinear equations in several variables, Academic Press, New York (1970).

[Pat99] Patriksson, M.: Nonlinear Programming and Variational Inequality Problems. Kluwer, Dordrecht (1999).

[War52] Wardrop, J.G.: Some theoretical aspects of road traffic research. Proceedings of the Institute of Civil Engineers, Part II, **1** 325-378 (1952).

[ZM94] Zhu, D.L., Marcotte, P.: An Extended Descent Framework for Variational Inequalities. Journal of Optimization Theory and Applications, **80**, 349–366 (1994).

Multi-Objective Optimisation of Expensive Objective Functions with Variable Fidelity Models

Daniele Peri, Antonio Pinto, and Emilio F. Campana

INSEAN, Via di Vallerano, 139 - 00128 - Roma, ITALY.
({D.Peri,A.Pinto,E.Campana}@insean.it)

Summary. For the major part of real-life application, the formulation of an optimisation problem involves a lot of different objective functions, often coming from different disciplines or areas. In this context, the optimisation represents a meeting point for many specialists, each one focused his proper requirements, that is, criteria constraints and objective functions. Different disciplines could be involved, like Computational Fluid Dynamics (CFD), structural analysis etc.

Moreover, being different criteria involved, Multi-Objective (MO) techniques must be adopted, in order to control the enhancements of all the objective functions. By the way, designers are not interested in marginal improvements of the starting design, and only Global Optimisation (GO) techniques are able to guarantee a wide and exhaustive exploration of the design space. In conjunction to that, high-fidelity models must be applied during the optimisation process, in order to ensure the quality of the optimising design. This last feature is conflicting with the desiderata of the GO algorithms, that usually require a large amount of evaluations on the objective function in order to qualify the global optimum. Moreover, the design team needs a solution in a short time, and the total time needed by the application of reliable solvers in conjunction with GO algorithms may be unpractical if a single objective function evaluation takes hours or days, as for CFD computations.

In this context, the only way to make the process feasible is to perform a strong reduction on the number of calls to the high-fidelity models, adopting a cheaper one to be substituted to the high-fidelity solver for the most of the calls, without loosing the accuracy of the high-fidelity model. This goal can be obtained by different strategies, all referring to the concept of Variable Fidelity Model (VFM): solvers with different complexity (and cost) are applied together, in a framework in which the exchange of information between the models makes possible to correct the evaluations of the low-fidelity one, substituting efficiently the high-fidelity model.

Here an algorithm for the solution of optimum ship design problems is presented. The procedure, illustrated in the framework of multi-objective optimisation problems, make use of high-fidelity, CPU time expensive computational models, like a free surface capturing RANSE solver, coupled with analytical meta-models of the objective functions (low-fidelity).

The optimisation is composed by global and local phases. In the global stage of the search, few computationally expensive simulations (high-fidelity) are applied

and surrogate models (metamodels) of the objective functions are produced (low-fidelity). After that, a large number of tentative design, placed uniformly on the Feasible Solution Set (F_{SS}), are evaluated with the low-fidelity model. The most promising designs are clustered, then locally minimized and eventually verified with high-fidelity simulations. New exact values are used to enlarge the training points for the low-fidelity model and repeated cycles of the algorithm are performed. A Decision Maker strategy is adopted to select the most promising designs.

Key words: global optimization, simulation based design, viscous flows, ship hydrodynamics, yacht design.

1 Introduction

Simulation-Based Design (SBD) in the naval hydrodynamic context still suffers from three major limitations: first, even if real ship design problems are multi-objective, attention is largely confined to single objective problems; second, it is relying exclusively on local optimisers, mostly gradient-based, either with adjoint formulations or finite-differences approaches; third, the use of high-fidelity, CPU time expensive solvers is still reduced by the large computational effort needed in the optimisation cycles.

Recently, the availability of fast computing platforms and the development of new and efficient analysis algorithms is alleviating the third limitation, shifting the focusing of high-fidelity models, such as the Navier-Stokes equations or those based on fine computational meshes, from the simple analysis of alternative configurations to optimal design. Examples in ship hydrodynamics are [Tahara97], [Hino], [Tahara02], [Peri01], [Newman], [Stern]. However, their use is confined to single objective problems solved with local optimisers.

Some recent work ([Peri01b], [Peri03b], [Peri03], [Minami]) were dedicated to expand optimisation applications from single- to multi-objective problems.

Our present goal is instead the development of a new optimisation procedure to solve multi-objective problems searching for the global optimum, overcoming the aforementioned limitations through the use of metamodels and of an alternate global-local stage in the algorithm. The intent of the present paper is to illustrate the procedure and to give numerical evidence of its capability.

2 Description of the GO algorithm

This section is devoted to the description of the developed algorithm and its definition in the group of GO techniques. For an extensive coverage of various methods of GO useful references are [Torn] and [Horst].

The algorithm is illustrated in the case of a multi-objective problem but it is also applicable to solve single objective problems. Different ways of classifying Global Optimisation methods exist. The proposed method belongs to the

class of deterministic optimiser, and to the family of Covering Methods, but with some features similar to the Adaptive Clustering COvering (ACCO) - Adaptive Clustering covering with Descent (ACD) schemes proposed in [Solomatine]. It is basically founded on the consideration that the only way to find out the global minimum of an unknown objective function, whose global characteristics of continuity and boundedness are not available, is to search uniformly the design variable space.

The algorithm hence consists of two main stages: (i) a global search phase, where a GO algorithm is used to explore the design space avoiding local minima and trying to locate regions where promising solutions are found and (ii) a local refinement phase, where best configurations (according to the Decision Maker) are grouped in clusters and then locally optimised with a multi-objective local method. The fundamental elements of the global algorithm are described in the following.

Formulation of the GO problem: In the most general way, the GO problem for a single objective function can be stated as follows. Let us consider a function $f : Z \rightarrow R$, where $Z \subset \mathcal{R}^N$:

$$\begin{gathered} \text{find } \mathbf{x}^* \in Z \\ \text{such that} \\ f(\mathbf{x}^*) \;=\; minf(\mathbf{x}),\ \mathbf{x} \in Z. \end{gathered}$$

In the constrained problem, bounds on the N design variables x_i and M functional constraints should be considered:

$$\begin{gathered} l_i \leq x_i \leq u_i \; i = 1, ..., N \\ g_j(\mathbf{x}) \leq 0, \; j = 1, ..., M \end{gathered}$$

In general $f(\mathbf{x})$ and $g(\mathbf{x})$ may be non-convex, non-smooth functions. The multi-objective optimisation problem can now be easily defined:

$$\min\{\ f_1(\mathbf{x}),\ f_2(\mathbf{x}),\ \ldots\ ,\ f_K(\mathbf{x})\ \}$$

where we have $K(\geq 2)$ different objective functions $f_k : Z \rightarrow R$.

Manipulation of the ship geometry and discrete mesh: In the implementation of an algorithm for shape optimisation there is the obvious need for a geometry modification procedure. Several attempts have been made to deal with this item. We decided not to rely on the use of a specific commercial CAD program but to induce modification in the ship's geometry by controlling a perturbation polynomial surface, which is added to the unmodified original geometry (details in [Peri01]). The control points of this polynomial surface will become the design variables x_i of the design problem. General guidelines for this procedure are the following: (i) when only a part of the ship is directly involved by shape optimisation, the modified region should join the original design without discontinuities and should be generally smooth; (ii) the number of design variables should be kept as small as possible to reduce

the complexity of the optimisation problem, but (iii) the algorithm should be as flexible as possible in order to achieve the largest number of possible solutions.

The above requirements have been obtained by using Béziér patches gradually reducing to a zero level while approaching the unmodified hull shape. In this way, geometric continuity between grid boundaries is guaranteed, and if the number of control points is kept sufficiently small, realistic geometry can be obtained that do not need major refinements prior to construction. Once the geometry is modified, the volume grid is adjusted accordingly.

Metamodel identification from CFD analysis results: As recalled before, when the number of the objective function evaluations needed by the algorithm increases, the application of high-fidelity models for the evaluation of the objective functions is discouraged. Anyway, the possible existence of a (unknown) relationship between the results coming from CFD and a much simpler analysis tool could help in reducing the number of evaluations addressed to the high-fidelity models. An interesting possibility is to assign an analytical structure to the data coming from the CFD over the whole design variable space (or a smaller part of it) trying to derive a metamodel. We call it "metamodel" because we are building a "model of the model", that is, an analytical approximation of the available data coming from the numerical simulation available at that time. Obviously, a specific metamodel must be constructed for each different objective function considered in the multi-objective problem.

A variety of metamodelling techniques exist (for an analysis of their performances see [Jin]). Polynomial regression models [Myers] is a widely known approach for the design and analysis of computer experiments. The coefficients of the polynomial functions may be computed by using a least square technique, or a more sophisticated identification parameter technique, as the Levemberg-Marquardt method. Moreover, performing the Analysis Of the VAriance (ANOVA) of the response surface it is also possible to enhance their quality, deleting terms not really affecting the approximation ([Giunta97] and [Giunta97b]). Obviously, this method is capable to correctly describe only objective functions whose behaviour is approximated by a polynomial function up to a certain order. If the objective function is much more complicated, the degree of the polynomial may be increased, but the metamodel quality may decrease because of numerical instabilities.

Artificial neural networks [Smith], [Cheng] are well-known approaches for identifying approximations of complex simulation codes and to fit a wide class of objective functions. In the following, a neural network of radial basis function (RBF neural network, [Powell]) has been selected as metamodel to solve the GO problem. Given a set of T points, the interpolating function has the form:

$$y(\mathbf{x}) = \sum_{t=1}^{T} w_t \Phi(||\mathbf{x} - \mathbf{x}^t||)$$

where Φ is a continuous function, which can be chosen among radial functions. The most used function in an RBF network is a Gaussian

$$\Phi = e^{-r^2}$$

RBF networks are feedforward with only one hidden layer. RBF hidden layer units have a receptive field, which has a center (a particular input value at which they have a maximal output). Their output tails off as the input moves away from this point. An example of the performances of the RBF neural network is reported in Fig. 1.

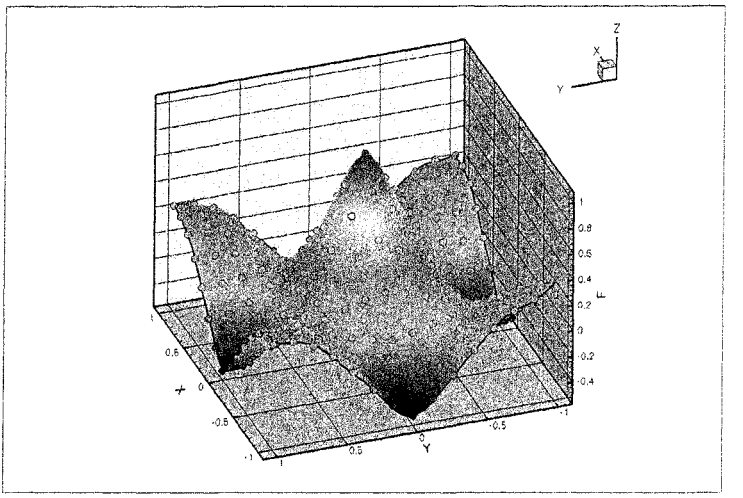


Fig. 1: An example showing the training phase of the metamodel approximating a f(x,y) function. A RBF Neural Network is trained with 15*15 points. Points are coloured with the function value, while the surface represents the final metamodel.

To construct the metamodel one has to select the T training points that have to be computed with the high-fidelity model. This can be performed using a Design Of Experiment (DOE) technique. A complete factorial design usually requires at least L^N solutions to be computed with the high-fidelity model, where N is the number of design variables and L is the number of levels in which each design variable interval is subdivided. The minimum value is 2^N, the vertexes of a hypercube constructed in the design variable space around the initial design. Consequently, the number of solutions needed to build the metamodel with a complete factorial design rapidly grows (exponentially)

with the number of design variables. Hence, an incomplete factorial design, in which some extremes of the design variable space are discharged, is usually applied. The criterion for the vertex's elimination generates a huge number of different methodologies. In this paper we have selected an Orthogonal Array (OA) technique [Hedayat], for which only $N+1$ points (i.e. CFD solutions) are requested to build the metamodel.

The search in the design variable space: Once an interval for the design variables has been fixed, trial points (i.e. solutions to be evaluated with the metamodel) must be distributed into the design space. For a fixed density of trial points, uniformity of the distribution is a crucial characteristic for the success of the optimisation, since an under-sampling of some region could deceive the optimiser forcing to discharge that portion of the design space. A regular sampling (cubic grid) would produce an uniform hexahedrical mesh on the hypercube defined in the design variable space, with two major drawbacks: too many points are needed when the number of design variables increases, and a marked shadow effect is produced (i.e. the coincidence of the projections of some points on the coordinate axes). LP_τ grids [Statnikov], belonging to the family of the Uniformly Distributed Sequences (UDS), have some attractive features, like an high degree of uniformity with a reduced set of trial points and a moderate shadow effect. In [Statnikov] the maximum number of points in the LP_τ distribution is 2^{16} and this value has been selected in the numerical test.

Once an UDS is placed into the design variable space, geometrical constraints are verified on these configurations and the Feasible Solution Set (F_{SS}) is identified. Obviously, the density of trial points in the F_{SS} is initially connected with the starting distribution and it is clear that the number of points in the F_{SS} must be as high as possible to be able to find a global optimum. On the other hand, CPU time needed for the constraints verification may be not small for real applications and a very long time for constructing the F_{SS} may be necessary in the case of an excessive sampling. Anyway, local refinement techniques described below aid to reduce this problem, allowing for an increase of the number of points in the F_{SS} near the best configurations during the process of optimisation.

Pareto optimality: In multi-objective problems, the problem of finding optimal solutions among those belonging to the feasible set is solved employing the concept of Pareto optimality:

Definition: a configuration identified by the objective vector $\mathbf{x_o} \in F_{SS}$ is called *optimal in the Pareto sense* if there does not exists another design $\mathbf{x} \in \mathrm{F}_{SS}$ such that $f_k(\mathbf{x}) \leq f_k(\mathbf{x_o}) \forall k = 1, ..., K$, where K is the number of objective functions, and $f_k(\mathbf{x}) < f_k(\mathbf{x_o})$ for at least a single $k \in [1, ..., K]$.

By applying the Pareto definition on the feasible solutions it is possible to find all the design vectors where none of the components can be improved without deterioration of at least one of the other components (non-dominated solutions). These designs belong to the Pareto optimal set $\mathcal{P}$.

Decision maker: Mathematically, all $\mathbf{x} \in \mathcal{P}$ are optimal solutions of the multi-objective problem. However, the final task is to order the design vectors belonging to $\mathcal{P}$ according to some preference rules indicated by the designer and select one optimal configuration among them. In general, one needs the cooperation between the decision maker and the analyst. A decision maker may be defined [Miettinen] as the designer who is supposed to have better insight into the problem and can express preference relations between different solutions. On the contrary, the analyst can be a computer program that gives the information to the decision maker. A wide number of different methodologies exist in literature (see [Miettinen] for an extensive summary of the subject) depending on the role of the decision maker in the optimisation process. In no-preference methods the opinions of the decision maker are not taken into consideration and the selection is accomplished by measuring (in the objective function space) the distance between some reference points (the "ideal" objective vector) and the Pareto solutions. It is the simple Global criterion, which can use different metrics. A powerful and classical way to solve this problem from the standpoint of practical application is the use of the ideas of goal programming. The procedure is simple: a list of hypothetical alternatives with assigned values (aspiration levels) of the objective functions is ranked by the designer according to his preference and experience (these are the goals). The problem is then modified into the minimization of the distance from these goals. Designer data may be used for constructing a metamodel of the preference order. In this way, the optimisation process will be mainly driven by the real needs of the designer, and the portion of Pareto set explored by the optimiser will contain the subset of the most preferable solution in the opinion of the designer.

Local refinement of the best solution: At the beginning, all the points belonging to F_{SS} have the same probability to be optimal points, but during the development of the optimisation process some designs show better characteristics than others, and the probability that the optimal solution is located in the vicinity of these good points is higher. This part of the F_{SS} may be deemed more interesting than others. In multi-objective GO the local refinement may follow two different strategies: (i) use a simple local method, able to give small improvements for all the objective functions in the nearby of a promising point, and/or (ii) adopt a clustering technique [Becker], in order to identify the region for which a deeper investigation is required and an increased density of trial points is wanted.

Cluster analysis of the promising solutions and refinement of the clusters: The task of any clustering procedure is the recognition of the regions of attraction, i.e. those regions R_i of the objective space such that for any starting point $\mathbf{x} \in R_i$, an infinitely small step steepest descent method will converge the same local minimum [Torn]. In a multi-objective reformulation of this approach (it is not possible to apply a steepest-descent method to these problem) the points belonging to the Pareto optimal set are the most promis-

ing points for the problem under consideration. The clustering algorithm may be easily summarized:

- taken the $\mathbf{x}$ vectors in the design space that correspond to the Pareto solutions, let a_{nn} be the average distance for the nearest neighbour and a_{all} the average distance between all the points $\mathbf{x} \in \mathcal{P}$ to be clustered.
- two hyperspheres of radii a_{nn} and a_{all} are centred in the selected Pareto point $\mathbf{x_1}$;
- compute $r_{12} = ||\mathbf{x_1} - \mathbf{x_2}||$ being $\mathbf{x_2}$ another Pareto point;
- if $r_{12} < a_{nn}$ then $\mathbf{x_2}$ is averaged with $\mathbf{x_1}$ and this become the new center of the cluster;
- if $a_{all} \geq r_{12} \geq a_{nn}$, $\mathbf{x_2}$ lies between the two hyperspheres and is returned;
- if $r_{12} > a_{all}$ then two new hyperspheres of radii a_{nn} and a_{all} are centred in $\mathbf{x_2}$.
- the remaining points are assigned to the clusters with the same rules.

The local refinement is then obtained by placing around the center of the clusters another LP_τ net, with small radius and few points. The radius of the investigated region in the nearby of the Pareto point decreases during the optimisation process, and the distribution is also rotated at each step, in order to spread out points in all the directions.

The algorithm for the GO problem: Main steps of the algorithm may be summarized as follows:

1. Initial exploration of the design space - Orthogonal Array is adopted for the initial exploration of the design space and trial points are distributed.
2. Model Identification from CFD results - Trial points are evaluated using the CFD and then used for the construction of the metamodels (one for each objective function);
3. The search in the design variable space - Trial designs ($2^{16} = 65536$) are uniformly distributed in the design variable space by using the LP_τ-grid;
4. Derive the feasible set - Enforcing the geometrical and functional constraints: the trial points not respecting the constraints are discharged and the feasible solution set is derived;
5. Identify the Pareto front - Analyse feasible points using the metamodels and find all $\mathbf{x} \in \mathcal{P}$;
6. Adopt a Decision Maker strategy for ordering the designs and finding non-dominated solutions;
7. Local refinement of the best solution with a multi-objective local method based on the metamodels (or with a scalarisation of the problem according to the DM);
8. Verification of the best solution using the CFD solvers. The new solution will be added to the metamodel training set for its improvement;
9. Clustering of the Pareto solutions is performed in the design space around dominating solutions. A reduced number of sets are obtained;

10. Refinement around the center of the clusters: new trial designs are uniformly distributed with smaller LP_τ-grids centred around the clusters;
11. go to step 4 until no more regions of attraction are found;

An important feature of this algorithm is that, as a consequence of both the refinements (steps 7 and 10), new added points may fall in a region of the design space which was not adequately covered. Even more, portion of the design space not considered at all in the initial distribution of trial points could be included dynamically. This last feature is a really useful quality of the method: in fact, in our particular case, since there is not a strong connection between design variables (the control points of the Béziér patches) and geometrical constraints, a correct estimation of the boundaries of the design parameters is non trivial. For this reason, the initial distribution sometimes does not cover the whole F_{SS}, since the investigated volume must be as small as possible in order to retain the point density of the F_{SS}. The local refinement technique automatically corrects the underestimation of the design space extension: the optimisation problem is still constrained, but bounds on the design variables may change dynamically within the course of the optimisation problem solution.

3 Multi-Objective Optimisation Test

A multi-objective problem for a race yacht is here presented. The original geometry has been selected among the ones of the systematic series of the "Il Moro di Venezia". This ship belongs to International America's Cup Class (IACC), and was racing during the edition of the America's Cup in 1992, winning the Louis Vuitton Cup, being the first European challenger for the Cup. This represents an old design, and it can be probably improved.

Differently from usual ships, like cargo ships or cruise ships, for which an autonomous propulsion system is available, here the wind gives the power for the motion of the ship, and it could be different during the months and also during a single regatta. Since this kind of ships are designed for the particular region in which the regatta will take place, some assumptions are made about the weather and wind conditions much more probable in the period of the race. Anyway, there is a great uncertainty on this data, and the ship could be completely wrong if it is designed for a very narrow wind conditions and the predictive method for the meteorological predictions fail.

Another big point is the different sailing conditions during the race. Due to the particular rules here applied, the boat is supposed to sail through a path oriented upwind and than downwind exactly: if the wind direction changes during the race, the markers of the regatta field are moved in order to accomplish this rule, and the position of the markers is aligned with the wind again. As a consequence, the ship is travelling in only two different positions with respect to the wind. But, depending on the strength of the

wind and the adopted sails, the yacht will be inclined differently on the sea surface. Moreover, since the yacht is not able to travel exactly upwind, the course of the yacht in such a situation is like a zigzag path, and the wind comes partly form one side of the boat. Furthermore, in this position the generation of a force in the direction of advance is paid at the price of a side force that tends to shift the boat in the direction of the wind. In order to contrast this side force, the yacht yaws of an angle depending on the speed and the wind/sails condition. As a consequence, the shape of the immersed part of the ship will be different than the one of the steady yacht. Summarizing, during the race the yacht is travelling in different positions, depending on the weather conditions the yacht encounters during the race. This situation makes the number of objective functions to be accounted for increasing, because each sailing position will drive to a different immersed shape of the yacht and consequently to a different objective function, depending from the wind speed, the boat speed and also the tactical choices of the skipper, who is in charge of the selection of the sails and their trim. It is now clear that a multi-objective optimisation problem must be solved. Great attention must be paid on the definition of the interesting sailing conditions: in fact, if a "general purpose" boat is designed, this will lose the race against a boat designed for the real sailing conditions. On the other hand, due to the uncertainty on the meteorological predictions, to design a boat for a very narrow range of conditions could be a mess if the weather conditions are strongly different than the expected ones. This is also the reason why two different boats are usually built by each participant (the rules limit the number of boats to be less than three).

Now, since there are no clear objectives, and there is not a clear understanding about the mutual influence on the potential objective functions, ship designers are used to produce different alternatives in order to evaluate the trade-off between different objective functions: as a consequence, the practice of assembling all the objective functions into a single merit function and than solving a single objective optimisation problem is not the recommended one, because this kind of approach makes the designer to loose the alternatives, forcing him team to accept or reject a single hull configuration. The Pareto front approach is preferable: the designer will select its best configuration among the alternatives $\mathbf{x} \in \mathcal{P}$. As a consequence, the wideness of the produced Pareto front and its resolution is of paramount importance, since it means a wider range of alternatives.

3.1 Selection of the Objective Functions

Since the selected design is a race boat, it is intuitive to ask to the optimisation team a faster boat than the present one. As a consequence, the total resistance of the hull will be one of the objective functions.

On the other hand, the boat is travelling on the sea surface powered by the wind, and the ship will assume different positions as a consequence of the wind

direction and the requested path. The position of the ship could be identified by three angles: the heel angle, (the angle around the longitudinal axis), the leeway angle or yaw angle (the angle around the vertical axis) and the trim angle (the angle around the side direction axis). For this class of ships, the regatta field is defined by two marks (windward and leeward). The race course is 18.5 nautical miles (or the equivalent of 34 kilometres) and consists of three laps of a windward-leeward type course with starboard rounding.

The boats begin from the starting line between two marks laid at right angles to the wind's direction and sail upwind to the first mark to be rounded to the starboard (right side).

Once the ships turn around the windward marker, the race continues downwind to the second mark, that is laid in close proximity to the starting line 100-meters further up the course. The boats round the marks 3 times before the winner crosses the Finish line.

Scrolling the course of the race, it is possible to identify two main relative positions of the wind with respect to the ship. During the first part of the track, the boat is travelling with the wind from the front: as a consequence, the boat is describing a zigzag trajectory, and the wind is coming from one side mainly. In this condition, the boat is heeled, that is, the ship is inclined around the advancing (X) direction, and also around the vertical (Z) direction. In the second side of the track, the wind is coming essentially from the rear, and the boat is just a little bit heeled and yawed because of the asymmetry of the sails.

As a consequence, two positions of the ship, defined by the three angles of inclination, have been accounted for:

- 10 degrees of heel, 4 degrees of leeway;
- 2 degrees of heel, 0 degrees of leeway.

A sketch of the reference frame here adopted is reported in figure 2.

For the first condition (namely *heeled*), both the total resistance and the side force are assumed as the first two objective functions for the problem. In fact, since the ship is travelling with a side wind, a side force must be generated in order to maintain the prescribed path. Since this force is negative in the adopted reference frame, we have a minimization problem also for this objective function. If a higher side force is generated, a lower yaw angle is needed: since a drag is connected with a yaw angle, the lower the aw angle the lower the induced drag. As a consequence, a lower yaw angle results also in a lower resistance.

In the second condition (namely *upright*), the total resistance is assumed as the third (last) objective function of the problem.

Side force is mainly generated by the appendages, while the hull gives a low contribution on it. Anyway, the flow condition for the appendages is depending from the hull shape: this means that there is a strong interaction between the hull and the appendages. Since the hull will be the only modified part during the optimisation process, an increase of the side force can be

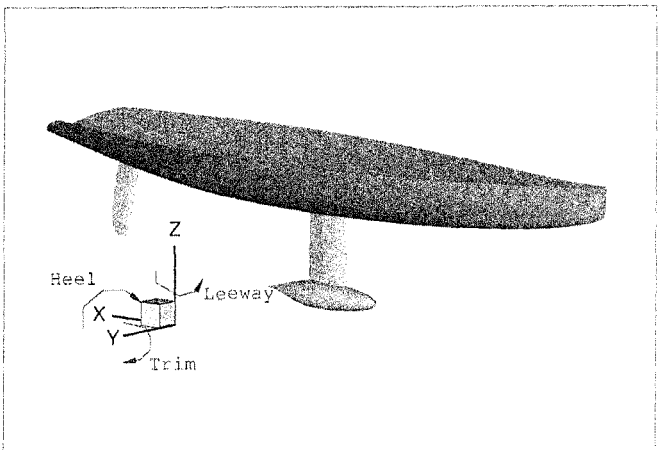


Fig. 2: Reference frame adopted for the sailing yacht. Main hull is the darkest part, appendages are the brightest part. The underwater part only is shown.

obtained only through a modification of the interaction factors between the hull and the appendages. The obtained results will focus on the importance of these interaction factors, usually considered by the designers in a simplified way.

3.2 Hull Shape Modification

In previous works [Peri01], a superimposition of some analytical surfaces has been adopted for the modification of the hull shape. This strategy allows the control of the connections between the different blocks of the computational grid, and a faired surface is still obtained at the end of the optimisation process. In this particular application, due to the complexity of the topology of the computational grid, it is not possible to apply the continuity condition between the grid blocks located in the nearby of the hull-rudder connections, and a large part of the stern cannot be modified if this strategy is applied.

In order to overcome this difficulty, a different strategy has been developed. Here a single rectangular patch has been placed in the physical domain, surrounding the part of the hull to be modified. This patch is still described by a Béziér surface. The surface has an offset, and it has the unit value if all the variables are set to zero. The patch represents the map of the multiplicative factor to be applied to the local lateral wideness of the hull: once the design variables are moved, the control surface moves and assumes different values in different regions of the space. Consequently, the Y coordinate is multiplied by the local value of the control surface, and a new hull is obtained. Control points represent the design parameters of the hull, moving the Béziér surface.

Continuity is enforced as explained in [Peri01]. The patch is here covering the entire hull surface, no matter about any particular structure for the numerical grid. The application of this kind of approach is encouraged by the fact that the shape of the hull has single curvature, and there is a biunivocal correspondence between a couple of coordinates in the XZ plane and a point of the hull. Moreover, the points laying on the symmetry plane are not moved because their side coordinate is just zero.

3.3 Active Constraints

In order to produce an hull still respecting the IACC class rules, all the class constraints must be applied. On the other hand, since we are not forced to produce a new design respecting the rules, only few of the rule have been followed. In particular, the active constraints for the problem are listed below:

- Fixed displacement (obtained by a change in the immersion of the boat).
- Beam in between 3.5 and 5.0 meters.
- Maximum displacement of the grid points: 2.0 meters.

The reduction of number of the constraints is not equivalent to a large simplification of the problem to be solved. In fact, constraints are non-linear, and no assumption about the shape of the feasible set is possible. In particular, convexity, continuity and connection of the feasible set are not guaranteed at all. As a consequence, all the difficulties of a non-linear programming problem are still present in this application. It is important to stress how there is not a parameter able to fix, for example, the beam of the ship, and also the displacement of a grid point depends on the movement of all the design parameters. This is a consequence of the parameterisation strategy, whose flexibility gives guarantees on the variety of the so produced shapes, but increase the complexity of the optimisation problem. This results also in difficulties in defining the correct initial range of variation of the design parameters.

3.4 Numerical results

Each hull shape evaluation requires two different runs of a RANSE solver, one for each position of the yacht. Since the time needed for the computation of a single configuration is about 8 hours on a Intel P4 2.8GHz, this means that for a single hull about 16 hours are required before obtaining the needed solutions. As a consequence, only 100 different hull shapes, including the ones required for the training of the metamodels, have been allowed, in order to limit the resources available for this study.

One single configuration is evaluated at each step of the algorithm, and 13 points have been distributed in the design space as training points of the metamodels. As a consequence, 87 different configurations have been evaluated during the course of the algorithm. At the end of the optimisation cycle, 12 different hulls compose the Pareto front, reported in table 1.

ID. #	F_1	F_2	F_3
36	0.88468	0.91041	1.08172
54	0.87730	0.90526	1.08048
71	0.93423	0.94711	1.10979
72	0.92344	0.96320	1.12452
74	0.96062	1.00322	1.14720
76	0.90208	0.92414	1.08534
77	0.90265	0.92462	1.08557
80	0.92781	0.95877	1.12915
93	0.91226	0.94200	1.10647
96	0.91789	0.94825	1.11527
99	0.90030	0.92471	1.09599
100	0.92884	0.95752	1.13995

Table 1: Pareto Optimal Set for the here depicted optimisation problem. All the objective functions are non-dimensionalised by their initial value.

Good solutions are indicated just at the start of the process, as the first column of the table 1 shows (one Pareto point is selected after 36 solutions). This means that, while the optimiser is running, designers are able to perform some preliminary analysis on the obtained results, and some modifications on the problem formulation are allowed without waiting for the end of the process. Even more, some new designs could be added on line, exploring with higher details some regions of the design space the designer is now interested in, as a consequence of the preliminary analysis. This is a feature connected with the fact that the algorithm is fully sequential, and can be interrupted and restarted in any moment, without losing any information produced at the time. Anyway, the longer the algorithm runs, the more the number of Pareto points discovered.

Hull number 54 is the best for the total resistance in upright and heeled condition, while hull number 74 is the best for the side force. This does not means that the first two objective functions are highly connected, because if we rank the hulls for these two objective functions the position in the list changes. Anyway, they do not change so much. On the other hand, the rank is quite reversed looking at the third objective function. This is the classical situation of a multi-objective optimisation problem, where some objective functions are conflicting each other. The rank for the different objective functions is reported in table 2. Here some considerations could be drawn about the best hull to be selected, if the choice were made on the basis of the rank position. In fact, if we select the sum of the ranking position as the selection parameter, best hull is number 54, who is the best for two objective functions, but is nearly the worst for the third objective function. If we assume the square of the ranks as the selection parameter, hull number 99 becomes the best: it is not the best for a single objective function, but it ranks quite well for all the different objective functions. Anyway, due to the small connections between the first

and the second objective function, this selection strategy penalize the third objective function: as a consequence, the final selection must be performed in a different way.

ID. #	Rank F_1	Rank F_2	Rank F_3	S	SSQ
36	2	2	11	15	129
54	1	1	12	14	146
71	11	7	6	24	206
72	8	11	4	23	201
74	12	12	1	25	289
76	4	3	10	17	125
77	5	4	9	18	122
80	9	10	3	22	190
93	6	6	7	19	121
96	7	8	5	20	138
99	3	5	8	16	98
100	10	9	2	21	185

Table 2: Pareto Optimal Set for the here depicted optimisation problem: column number 2, 3 and 4 report the ranking with respect to objective function number 1, 2 and 3 respectively. Column labelled with S reports the sum of the ranks, while column SSQ reports the sum of the square of the ranks.

Maximum improvements are 12% for the total resistance in upright, 10% for the total resistance in heeled condition and 15% for the side force in heeled condition. If we consider that the difference between the best hull and the worse one was about 3% in the last America's Cup edition, and a difference on the performances of less than 1% gives the victory, it is clear the significance of the obtained improvements. The wideness of the Pareto front is quite large: the wideness of the variation is about 8% for the first objective function, 10% for the second one and 6% for the third one. This gives the opportunity of a wide choice for the designer, obtaining a good balance between the performances as needed. If this analysis is not considered to be enough, deeper analysis on the Pareto points is possible, evaluating different objective functions in order to better assess the behaviour of the yacht in off-design conditions.

As a final consideration, the increase in the side force generation is only due to the hull shape modifications. Since the grater part of the side force is generated by the appendages, whose geometry is unchanged during the course of the optimisation problem solution, it is clear that enhancements in this direction are obtained only by changing the working conditions of the fin, that is, by changing their inflow through the modification in the shape of the hull. As a consequence, there is a strong influence of the hull shape in the performances of the appendages, and a simple linear superimposition of the forces generated by the hull and the fin separately, eventually corrected

by some empirical correlation factors, is not enough. Concluding, the design of the appendages and the hull must be performed together, and not in a separate way, as in common practice.

4 Conclusions

Optimisation tools could help the designer, and GO techniques can lead to new design concepts. The final goal of the authors was to develop an useful GO tool for ship design, plus techniques for reducing CPU-time requirements, a fundamental step if a GO problem has to be solved. To this aim, a GO problem in a multi-objective context has been formulated and solved with an original algorithm. Although the numerical results are still preliminary, strong reductions on the interesting quantities have been obtained, although the main difficulties of a classical multi-objective optimisation problem have been encountered, with highly conflicting objective functions. The applied numerical solvers are able to give reliable information on the flow field, allowing improvements otherwise difficult to be obtained in the absence of correlations law between main geometrical parameters and local flow variables. The optimisation tool seems to be able to coordinate the different objectives and the analysis tools used in the procedure are used in a rational way. The inclusion of this approach into the spiral design cycle is recommended, in particular when some special requests are present in the design specifications.

Acknowledgments

This work has been supported by the U.S. *Office of Naval Research* under the grant No. 000140210489, through Dr. Pat Purtell. The Authors also wish to thank Natalia Alexandrova for her technical advices and the anonymous referees for their comments and suggestions.

References

[Barthelemy] Barthelemy, J.-F.M.; Haftka, R.T.: Approximation concepts for optimum structural design - a review. Structural Optim. **5**, 129-144 (1993).

[Bassanini] Bassanini P., Bulgarelli U., Campana E.F., Lalli F.: The wave resistance problem in a boundary integral formulation. Surv Math Ind, **4**, (1994).

[Becker] Becker R.W., Lago G.V.: A global optimization algorithm. Proceedings of the 8th Allerton Conference on Circuits and Systems Theory, 3-12 (1970).

[Chang] Chang, K.J., Haftka, R.T., Giles, G.L. and Kao, P.-J.: Sensitivity-based scaling for approximating structural response. Journal of Aircraft, **30**(2), 283-288 (1993).

[Cheng] Cheng, B., Titterington, D.M.: Neural networks: a review from a statistical perspective. Statistical Sci. **9**, 2-54 (1994).

[DiMascio] Di Mascio A., Broglia R., Favini B.: A second-order Godunov-type scheme for Naval Hydrodynamics. Godunov (Ed) Methods: Theory and application, Singapore, Kluwer Academic/Plenum (2000).

[Dixon] Dixon L.C.W., Szegö G.P.: Towards global optimization. North-Holland (1975).

[Giering98] Giering, R., and Kamnski, T.: Recipes for Adjoint Code Construction. ACM Trans. on Math. Software, **24** No. 4 437-474 (1998).

[Giering02] Giering, R., Kamnski, T. and Slavig, T.: Applying TAF to a Navier-Stokes solver that simulates an Euler flow around an airfoil. Future generation computer systems, in press (2002), Elsevier, Amsterdam.

[Giunta97] Giunta A.A.: Aircraft multidisciplinary design optimization using design of experiments theory and response surface modeling methods. MAS Center Report 97-05-01i (1997), Virginia Polytechnic Institute and State University, Blacksburg, USA.

[Giunta97b] Giunta A.A., Balabanov V., Kaufman M., Burgee S., Grossman B., Haftka R.T., Mason W.H., Watson L.T.: Variable-Complexity Response Surface Design of an HSCT configuration. In: Multidisciplinary Design Optimization, Alexandrov N.M. and Hussaini M.Y. eds, SIAM, Philadelphia, USA (1997).

[Haftka] Haftka R.T., Vitali R., Sankar B.: Optimization of Composite Structures Using Response Surface Approximations. NATO ASI meeting on Mechanics of Composite Materials and Structures, Troia, Portugal (1998).

[Haimes] Haimes Y.Y., Li D.: Hierarchical multiobjective analysis for large-scale systems: review and current status. Automatica, **24**, No.1, 53-69 (1988).

[Hedayat] Hedayat A.S., Sloane N.J.A., Stufken J.: Orthogonal Arrays: Theory and Applications. (Springer Series in Statistics), Springer-Verlag, Berlin, Germany (1999).

[Hino] Hino, T., Kodama , Y., and Hirata , N., 1998: Hydrodynamic shape optimisation of ship hull forms using CFD. Third Osaka Colloquium on Advanced CFD Applications to Ship Flow and Hull Form Design, Osaka Prefecture Univ. and Osaka Univ., Japan (1998).

[Horst] Horst, R. and Pardalos, P.M.: Handbook of Global Optimization. Kluwer, Dordrecht (1994).

[Iwashita] Iwashita, H., Nechita, M, Colagrossi, A., Landrini, M., Bertram, V.: A Critical Assessment of Potential Flow Models for Ship Seakeeping. Osaka Colloquium on Seakeeping, pp. 37-64 (Osaka), Japan (2000).

[Jin] Jin R., Chen W., Simpson T.W.: Comparative studies of metamodelling techniques under multiple modelling criteria. Struct. Multidisc. Optim. **23** (2001).

[Kleijen] Kleijnen, J.P.C.: Statistical tools for simulation practitioners. NewYork, Marcel Dekker (1997).

[Knill] Knill, D.L., Giunta, A.A., Baker, C.A., Grossman, B., Mason, W.H., Haftka, R.T., Watson, L.T.: Response surface models combining linear and Euler aerodynamics for supersonic transport design. Journal of Aircraft **36**, 1, 75-86 (1999).

[Minami] Minami,Y. , Hinatsu M.: Multi Objective Optimization of Ship Hull Form Design by Response Surface Methodology. 24th Symposium on Naval Hydrodynamics, Fukuoka, JAPAN (2002).

[Newman] Newman III, J.C., Pankajakshan, R., Whitfield, D.L., and Taylor, L.K.: Computational Design Optimization Using RANS. 24th Symposium on Naval Hydrodynamics, Fukuoka, JAPAN (2002).

[Miettinen] Miettinen K.M.: Nonlinear multiobjective optimization. Kluwer Academic Publisher, (1999).

[Myers] Myers R.H., Montgomery D.C.: Response Surface Methodology. Wiley, USA (1997).

[Pareto] Pareto V.: Manuale di economia politica, Società editrice libraria, Milano (1906), Italy. Also in Course d'Economie Politique, Lausanne, Switzerland, Rouge (1896).

[Peri01] Peri D., Rossetti M., Campana E.F.: Design optimization of ship hulls via CFD techniques. Journal of Ship Research, **45** 2, 140-149 (2001).

[Peri01b] Peri D., Campana E.F., Di Mascio A.: Development of CFD-based design optimization architecture. 1st MIT conference on fluid and solid mechanics, Cambridge, MA, USA (2001).

[Peri03] Peri D., Campana E.F.: High fidelity models in the Multi-disciplinary Optimization of a frigate ship. 2nd MIT conference on fluid and solid mechanics, Cambridge, MA, USA (2003).

[Peri03b] Peri D., Campana E.F.: Multidisciplinary Design Optimization of a Naval Surface Combatant Journal of Ship Research, **47**, 1, 1-12 (2003).

[Powell] Powell, M.J.D.: Radial basis functions for multivariable interpolation: a review. In: Mason, J.C.: Cox, M.G. (eds.) Algorithms for approximation, Oxford University Press (1983).

[Sacks] Sacks J., Welch W.J., Mitchell, .T.J, Wynn H.P.: Design and analysis of computer experiments. Statistical Science **4**, 409-435 (1989).

[Smith] Smith M.: Neural networks for statistical modeling. Von Nostrand Reinhold, New York (1993).

[Sobie] Sobieszczanski-Sobieski, J., Haftka, R.T.: Multidisciplinary aerospace design optimisation: survey of recent developments. Struct. Optim. **14**, 1-23 (1997).

[Solomatine] Solomatine D.P.: Two strategies of adaptive cluster covering with descent and their comparison to other algorithms. Journal of Global Optimization **14**, 1, 55-78 (1999).

[Stern] Stern, F., Wilson, R., Longo, J., Carrica, P., Xing T., Tahara, Y., Simonsen, C., Kim, J., Shao, J., Irvine, M., Kandysamy, M., Gosh, S., Weymouth, G.: Paradigm for development of simulation based design for ship hydrodynamics. 3rd International Conference: Navy And Shipbuilding Nowadays (NSN'2003), Krylov Shipbuilding Research Institute, St. Petersburg, Russia (2003).

[Statnikov] Statnikov R.B., Matusov J.B.: Multicriteria optimization and engineering. Chapman & Hall, USA (1995).

[Tahara97] Tahara Y., Himeno Y.: An application of computational fluid dynamics to tanker hull form optimization problem. Third Osaka Colloquium on Advanced CFD Applications to Ship Flow and Hull Form Design, Osaka Prefecture Univ. and Osaka Univ., Japan (1997).

[Tahara02] Tahara, Y., Patterson, E., Stern, F., and Himeno, Y.: Flow- and wavefield optimization of surface combatants using CFD-based optimization, methods. 23rd ONR Symposium on Naval Hydrodynamics, Val de Reuil, France (2002).

[Thomas] Thomas J.P., Hall K.C., Dowell E.H.: A disrete adjoint approach for modeling unsteady aerodynamic design sensitivities. 41-th Aerospace Science Meeting and Exibit, Reno, Nevada, USA (2003).

[Torn] Törn A.A., Žilinskas A.: Global optimization. Springer-Verlag, Berlin, Germany (1989).

Towards the Numerical Solution of a Large Scale PDAE Constrained Optimization Problem Arising in Molten Carbonate Fuel Cell Modeling

Hans Josef Pesch, Kati Sternberg, and Kurt Chudej

Lehrstuhl für Ingenieurmathematik, Univ. Bayreuth, 95440 Bayreuth, Germany
({hans-josef.pesch,kati.sternberg,kurt.chudej}@uni-bayreuth.de)

Summary. Molten carbonate fuel cells (MCFCs) allow an efficient and environmentally friendly energy production by converting the chemical energy contained in the fuel gas in virtue of electro-chemical reactions. Their dynamical behavior can be described by large scale embedded systems of 1D or 2D nonlinear partial differential algebraic equations (PDAEs) up to dimension 28. They are of mixed parabolic-hyperbolic type with integral terms in the right hand side and initial and nonlinear boundary conditions, the latter governed by a system of ordinary differential equations.

In this paper a new 2D model together with results of its numerical simulation is presented. The numerical results show a good correspondence with the expected dynamical behavior of MCFCs. The ultimate goal is to optimize this large scale nonlinear PDAE system to increase efficiency and service life of MCFCs.

Key words: partial differential algebraic equations, numerical simulation, PDE constrained optimization, fuel cells.

1 Introduction

Molten carbonate fuel cells (MCFCs) are especially well suited for stationary power plants if their process heat is used to increase their efficiency. MCFCs seem to become soon competitive compared to traditional power plants. In order to enhance service life, high temperature gradients inside the fuel cells must be avoided. Therefore control strategies are currently under development. Admittedly, not only the avoidance of high temperature gradients is of interest, but also the optimization of the efficiency of fuel cell systems.

The dynamic behavior of MCFCs can be modeled mathematically by a hierarchy of systems of partial differential algebraic equations; see Heidebrecht and Sundmacher [5,6] and, in particular, Heidebrecht [4]. A detailed description of a certain 1D model together with an index analysis concerning the differential time and spatial index as well as the MOL index of the 1D PDAE

system and some numerical results can be found in Chudej et al. [3]. An index analysis of several 1D and 2D models can be found in Chudej [2]. In addition, for a simplified linearized 2D model also a perturbation index analysis can be found in Rang and Chudej [8].

The present paper focuses on a new enhanced 2D cross flow model of a MCFC as well as on its numerical simulation and optimization.

2 2D Molten Carbonate Fuel Cell Model

The model investigated in the present paper describes an MCFC with a cross flow mode for the guidance of the anode and cathode gas streams. Such a design has been realized for the so-called *HotModule* produced by the MTU CFC Solutions GmbH, Munich, and operated for example by the IPF-Heizkraftwerksbetriebsgesellschaft mbH, Magdeburg, at the power plant of the University Hospital of Magdeburg. See Fig. 1 for a schematic picture of the main devices of the HotModule, which has been taken into account in the mathematical model. The anode inlet is on the south-west side of the fuel cell. The anode outlet gas is fed into the cathode inlet on the south-east side of the fuel cell via a catalytic combustor and a reversal chamber. The cathode exhaust gas is partially led back to the catalytic combustor and goes then again into the cathode inlet. The remaining part of the exhaust gas at the cathode outlet finally goes to the cell exhaust device.

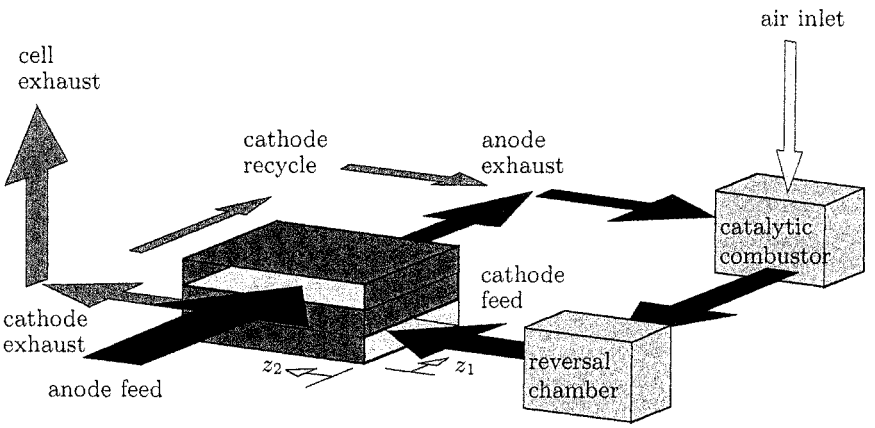


Fig. 1: Cross flow configuration of the HotModule

In the anode and cathode gas channels, resp., the following chemical reactions take place. Within the gas stream of the anode channel an internal

reforming can be performed due to the high operating temperature to produce the necessary hydrogen

$$\begin{aligned} CH_4 + H_2O &\rightleftharpoons CO + 3\,H_2\,, \\ CO + H_2O &\rightleftharpoons CO_2 + H_2\,. \end{aligned} \tag{1}$$

Simultaneously, an electro-chemical reaction, the oxidation reaction, takes place at the anode electrode

$$\begin{aligned} H_2 + CO_3^{2-} &\rightleftharpoons H_2O + CO_2 + 2e^-\,, \\ CO + CO_3^{2-} &\rightleftharpoons 2\,CO_2 + 2e^-\,. \end{aligned} \tag{2}$$

In contrast, at the cathode electrode we have the reduction reaction

$$\frac{1}{2}O_2 + CO_2 + 2e^- \rightleftharpoons CO_3^{2-}\,. \tag{3}$$

Hereby, the carbonate ions are transferred through the electrolyte from the cathode electrode to the anode electrode. Bipolar plates serve as heat conducting material and separate the single cells of the stack. From the cell's outside, the electric current can be collected at the electrodes; see Fig. 2. For the sake of simplicity, we use here 1D figures for a cell with a counter flow configuration in order to describe the gas flows and the dominating chemical reactions (by Fig. 2) and the concrete physical quantities involved in the model (by Fig. 3). The model however treats the cross flow configuration and the chemical reactions of Eqs. (1–3).

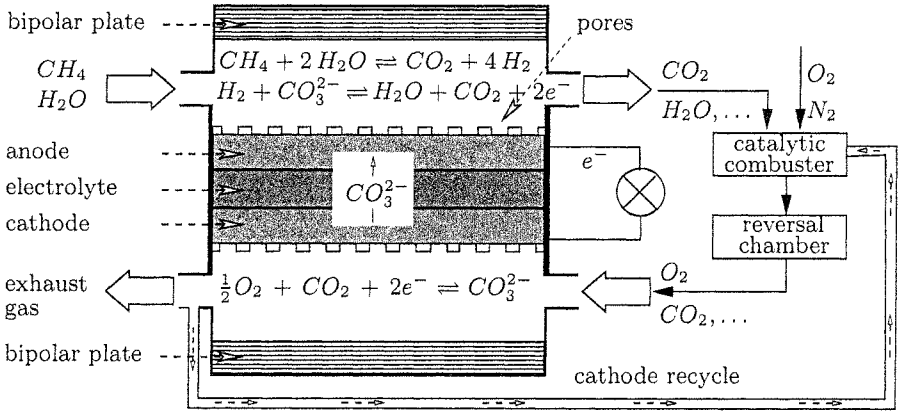


Fig. 2: Gas flow and (simplified) chemical reactions

In the anode and cathode channels 7 substances arise in the chemical reactions: methane CH_4, hydrogen H_2, water H_2O, oxygen O_2, carbon monoxide CO, carbon dioxide CO_2, and nitrogen N_2. The molar fractions of all these

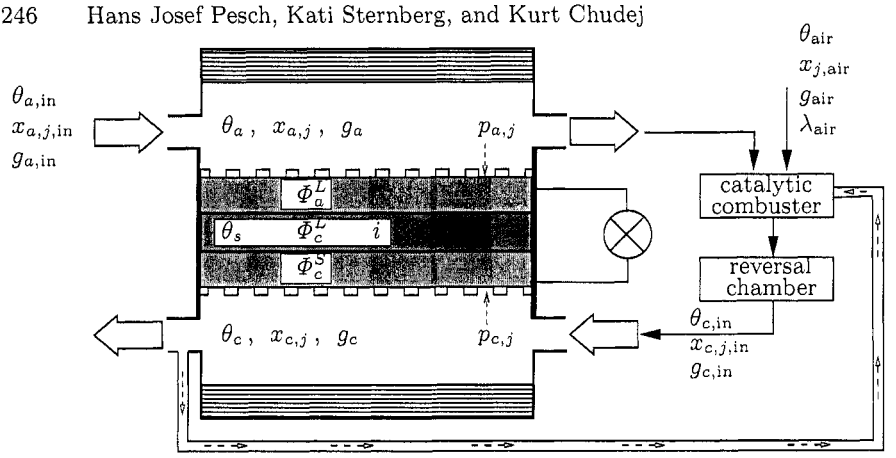


Fig. 3: Variables and boundary conditions

species in the gas flows of the two channels are denoted by $x_{a,j}$ and $x_{c,j}$, $j = 1, \ldots, 7$, resp. In contrast, the molar fractions in the pores of the electrodes are denoted by $p_{a,j}$ and $p_{c,j}$; see Fig. 3. For the numerical treatment we have to include only four variables $p_{a,j}$ associated with H_2O, H_2, CO, CO_2 in the pores of the anode and two variables $p_{c,j}$ associated with CO, CO_2 in the pores of the cathode. Note that the distinction between the molar fractions in the gas streams, where the internal reforming (1) takes place, and the pores, where the anode (2) and the cathode (3) reaction take place, is a more detailed description compared to the model treated in a previous paper of Sternberg et al. [9]. Both models are due to Heidebrecht [4]. Further, we have to take into account the molar flows g_a and g_c as well as the temperatures θ_a and θ_c in the anode and cathode gas channels. These temperatures θ_a and θ_c are dominated by convection and are to be distinguished from the solid temperature θ_s, which is distributed through the solid by diffusion, i.e. heat conduction. Moreover, the electrical potentials Φ_a^L, Φ_c^L, and Φ_c^S at which anode ion layer, cathode ion layer and cathode electrode are relative to a reference potential, and the current density i are introduced into the model. Finally, the total cell current $I_{\text{cell}} = I_{\text{cell}}(t)$ represents a degree of freedom in the model, that can be used for controlling the fuel cell.

Altogether, we end up with a system of time t dependent nonlinear partial differential algebraic equations in two space dimensions $z := (z_1, z_2) \in \Omega := [0, 1]^2$ corresponding to the two gas flow directions due to the cross flow configuration of the stack. We have the following parts of a large scale coupled system:

Heat equation in the solid:

$$\frac{\partial \theta_s}{\partial t} = \lambda\, \Delta\theta_s + \varphi_1(\theta_s, \theta_a, \theta_c, x_a, x_c, p_a, p_c, \Phi_a^L, \Phi_c^L, \Phi_c^S)\,, \quad \lambda > 0 \text{ constant} \tag{4}$$

Advection equations in the gas streams:

$$\frac{\partial x_{a,j}}{\partial t} = -\, g_a \theta_a \frac{\partial x_{a,j}}{\partial z_1} + \varphi_{2,j}(\theta_s, \theta_a, x_a, p_a, \Phi_a^L)\,, \qquad j = 1, \ldots, 7\,, \tag{5}$$

$$\frac{\partial x_{c,j}}{\partial t} = -\, g_c \theta_c \frac{\partial x_{c,j}}{\partial z_2} + \varphi_{3,j}(\theta_s, \theta_c, x_c, p_c, \Phi_c^L, \Phi_c^S)\,, \qquad j = 1, \ldots, 7\,, \tag{6}$$

$$\frac{\partial \theta_a}{\partial t} = -\, g_a \theta_a \frac{\partial \theta_a}{\partial z_1} + \varphi_4(\theta_s, \theta_a, x_a, p_a, \Phi_a^L)\,, \tag{7}$$

$$\frac{\partial \theta_c}{\partial t} = -\, g_c \theta_c \frac{\partial \theta_c}{\partial z_2} + \varphi_5(\theta_s, \theta_c, x_c, p_c, \Phi_c^L, \Phi_c^S)\,. \tag{8}$$

Algebraic equations in the pores:

$$p_{a,j} = \varphi_{6,j}(p_a, \theta_s, x_a, \Phi_a^L)\,, \qquad j = 1, \ldots, 4\,, \tag{9}$$

$$p_{c,j} = \varphi_{7,j}(p_c, \theta_s, x_c, \Phi_c^L, \Phi_c^S)\,, \qquad j = 1, \ldots, 2\,. \tag{10}$$

Degenerated PDEs for the molar flows:

$$0 = \frac{\partial (g_a \theta_a)}{\partial z_1} + \varphi_8(\theta_s, \theta_a, x_a, p_a, \Phi_a^L)\,, \tag{11}$$

$$0 = \frac{\partial (g_c \theta_c)}{\partial z_2} + \varphi_9(\theta_s, \theta_c, x_c, p_c, \Phi_c^L, \Phi_c^S)\,. \tag{12}$$

Integro-PDAE system for the potentials:

$$\frac{\partial \Phi_a^L}{\partial t} = [i_a(p_a, \theta_s, \Phi_a^L) - i]/c_a\,, \tag{13}$$

$$\frac{\partial \Phi_c^L}{\partial t} = [i_a(p_a, \theta_s, \Phi_a^L) - i]/c_a + [i_e(\Phi_a^L, \Phi_c^L) - i]/c_c\,, \tag{14}$$

$$\begin{aligned} \frac{\mathrm{d}\Phi_c^S}{\mathrm{d}t} = \varphi_{10}\Big(& \int_\Omega i_a(p_a, \theta_s, \Phi_a^L)\,\mathrm{d}z, \int_\Omega i_c(p_c, \theta_s, \Phi_c^L, \Phi_c^S)\,\mathrm{d}z, \\ & \int_\Omega i_e(\Phi_a^L, \Phi_c^L)\,\mathrm{d}z, I_{\text{cell}}\Big)\,. \end{aligned} \tag{15}$$

where the current density $i = i(t, z)$ is given by

$$i = \varphi_{11}\left(i_a, i_c, i_e, \int_\Omega i_a\,\mathrm{d}z, \int_\Omega i_c\,\mathrm{d}z, \int_\Omega i_e\,\mathrm{d}z, I_{\text{cell}}\right)$$

and the total cell current I_{cell} can be used for control purposes as mentioned above.

All functions determined by the PDAEs are functions of z and t except $\Phi_c^S = \Phi_c^S(t)$. The presentation of all source terms φ_j, $j = 1, \ldots, 11$, as well as of the functions i_a, i_c, and i_e in full detail would be beyond the scope of this paper; they can be found in Heidebrecht [4]. Summarizing we obtain a nonlinear PDAE system of dimension 28. Note that by substituting $v_a := g_a\,\theta_a$ and $v_c := g_c\,\theta_c$ the PDAE system becomes quasi-linear, but the boundary conditions still remain highly nonlinear as we will see subsequently.

For simulations, the boundary conditions for the temperature θ_a, the molar fractions $x_{a,j}$, and the molar flow g_a are prescribed at the anode inlet, i.e., at $z_1 = 0$, $z_2 \in [0,1]$; compare Figs. 1 and 3 (here denoted by the subscript $_{a,\text{in}}$). For control purposes they can also be chosen as time dependent boundary controls. For the temperature θ_c, the molar fractions $x_{c,j}$, and the molar flow g_c the boundary conditions at the cathode inlet (in Fig. 3 denoted by the subscript $_{c,\text{in}}$), i.e., at $z_2 = 0$, $z_1 \in [0,1]$, depend on the corresponding output values at the anode outlet, i.e., at $z_1 = 1$, $z_2 \in [0,1]$ and from the environment; see Fig. 3. These values are given via the solution of a system of ordinary differential algebraic equations. It describes the time dependent behavior of the gas temperature, the molar fractions and the molar flow in the catalytic combustor and the reversal chamber. Note that these quantities depend on the averaged values of the associated quantities at the anode outlet. Again, without giving the details for the right hand sides φ_j, $j = 12, \ldots, 14$, of the ordinary DAE system, the boundary conditions at the cathode inlet can be read as follows:

$$x_{c,j}(t,z)\Big|_{z_1\in[0,1],\, z_2=0} = x_{m,j}(t), \quad j = 1, \ldots, 7\,, \tag{16}$$

$$\theta_c(t,z)\Big|_{z_1\in[0,1],\, z_2=0} = \theta_m(t)\,, \quad g_c(t,z)\Big|_{z_1\in[0,1],\, z_2=0} = g_m(t) \tag{17}$$

where

$$\frac{\mathrm{d}x_{m,j}}{\mathrm{d}t} = \varphi_{12,j}\Big(x_{m,j}, \theta_m, \int_0^1 x_a\Big|_{z_1=1}\mathrm{d}z_2, \int_0^1 \theta_a\Big|_{z_1=1}\mathrm{d}z_2, \int_0^1 g_a\Big|_{z_1=1}\mathrm{d}z_2, x_{j,\text{air}}, \lambda_{\text{air}}\Big)\,, \tag{18}$$

$$\frac{\mathrm{d}\theta_m}{\mathrm{d}t} = \varphi_{13}\Big(\theta_m, \int_0^1 x_a\Big|_{z_1=1}\mathrm{d}z_2, \int_0^1 \theta_a\Big|_{z_1=1}\mathrm{d}z_2, \int_0^1 g_a\Big|_{z_1=1}\mathrm{d}z_2, \theta_{\text{air}}, x_{j,\text{air}}, \lambda_{\text{air}}\Big)\,, \tag{19}$$

$$g_m = \varphi_{14}\Big(\theta_m, \int_0^1 x_a\Big|_{z_1=1}\mathrm{d}z_2, \int_0^1 \theta_a\Big|_{z_1=1}\mathrm{d}z_2, \int_0^1 g_a\Big|_{z_1=1}\mathrm{d}z_2, \theta_{\text{air}}, x_{j,\text{air}}, \lambda_{\text{air}}\Big)\,. \tag{20}$$

Here, θ_{air}, $x_{j,\text{air}}$, and λ_{air} denote the ambient temperature, the molar fractions of oxygen and nitrogen in air and the air number, resp. The latter defines the amount of air fed into the burner. All these quantities can be subject to changes in time and thus can play the roles of boundary controls.

So far, the re-feeding is not modeled. It will introduce further dependencies of the right hand sides of the ordinary DAE system. In this case, the function φ_{12} will also depend on the integrals $\int_0^1 x_c|_{z_2=1}\,\mathrm{d}z_1$, $\int_0^1 \theta_c|_{z_2=1}\,\mathrm{d}z_1$, and $\int_0^1 g_a|_{z_2=1}\,\mathrm{d}z_1$.

Moreover, the boundary conditions for the solid temperature θ_s are, because of the insulation, given by zero Neumann conditions at the boundary of the electrolyte, i. e. at $\partial\Omega$. Finally, consistent initial conditions at time $t = 0$ for all $z \in \Omega$ must be suitably imposed.

This completes the mathematical model.

3 Simulation Results

The entire PDAE system can be written in compact form as follows

$$A\,\frac{\partial u}{\partial t} = B\,\Delta u + C_1\,\frac{\partial u}{\partial z_1} + C_2\,\frac{\partial u}{\partial z_2} + f(u)\,. \tag{21}$$

Hereby A is a singular diagonal matrix containing only zeros and ones, and $B = \mathrm{diag}(\lambda, 0, \ldots, 0)$, if we choose the vector solution u containing all the unknown functions of the Eqs. (4)–(15) as $u := (\theta_s, \ldots, \Phi_c^S)$. Here the Laplacian has to be understood componentwise. The singular diagonal matrices C_1, resp. C_2 depend on the auxiliary variables $v_a := g_a\,\theta_a$, resp. $v_c := g_c\,\theta_c$ previously defined making the system quasi-linear. Finally the term $f(u)$ contains all further nonlinear dependencies.

In order to solve this complex nonstandard PDAE constrained optimization problem numerically in the future we choose the approach to *first discretize, then optimize* in contrast to *first optimize, then discretize*, i. e. to derive firstly the optimality conditions in function spaces which are then discretized. We favor this approach because of the various model updates in the past being typical when dealing with real world problems. This fact makes it (almost) impossible to apply the mathematically more safeguarded latter approach.

Thus, we apply, as the method of choice, the (vertical) method of lines, and we obtain a large scale ordinary DAE system of the form

$$A_h\,\frac{\partial u_h}{\partial t} = g(u_h)\,, \tag{22}$$

for which now standard techniques can be used, if the dimension remains moderately large, e.g., a transcription to a very large scale nonlinear programming problem.

The numerical approximation $u_h(t)$ of the exact solution $u(t, z_1, z_2)$ is accordingly obtained by semi-discretization in the spatial variables z_1 and z_2. Hereby, the Laplace operator is approximated by the standard five point difference star. For the convection terms upwind formulas are used taking into account the known wind direction in the anode and cathode gas channels, resp. Despite the complicated boundary conditions they can be handled in the usual way like Dirichlet and Neumann conditions with values partly obtained via the solution of an ordinary DAE system.

In Eq. (22), A_h is a constant singular diagonal matrix containing again only zeros and ones, and the nonlinear function g stands for the discretized right hand side of (21).

In the following we present the numerical simulation results. The software package NUDOCCCS [1] developed for the numerical solution of large scale ODE and DAE constrained optimal control problems can be used for simulation purposes, too. The numerical integration of the DAE system is performed by a fifth order implicit Runge-Kutta method (RADAU 5).

The numerical solution of some selected components is depicted in the Figs. 4–11. The figures show the two-dimensional spatial distribution at a time, at which we assume that the steady state has been reached. About 10 000 time steps had to be performed. Further note that all quantities are dimensionless.

In each diagram, the inlet of the anode is on the south-west side ($z_1 = 0$, $z_2 \in [0, 1]$), the cathode inlet is on the south-east side ($z_1 \in [0, 1]$, $z_2 = 0$). The gas flows in z_1-direction through the anode channel, in z_2-direction through the cathode channel.

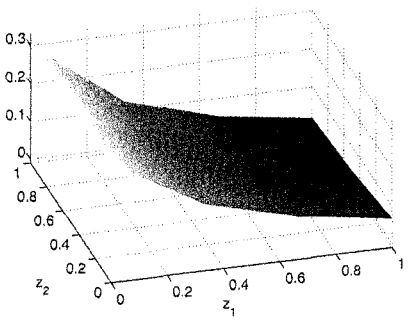


Fig. 4: Methane in anode channel

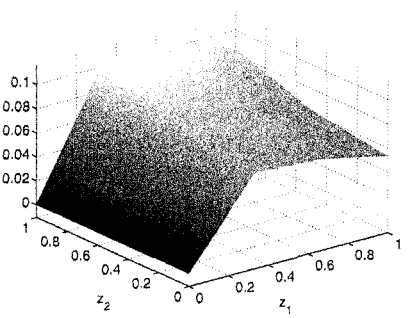


Fig. 5: Hydrogen in anode channel

The results obtained make sound sense as we will show. The methane depicted in Fig. 4 decreases over the entire width due to the methane consuming endothermic reforming reaction (1). The concentration is slightly lower on that side of the anode channel where the cathode outlet is located. This is due to

the higher temperatures in this region leading to higher reforming reaction rates there.

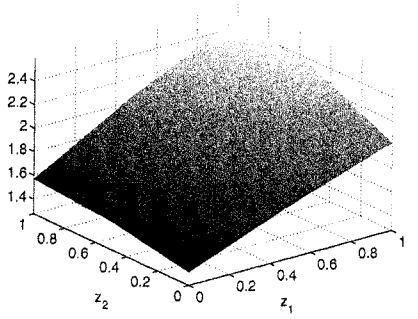


Fig. 6: Molar flow in anode channel

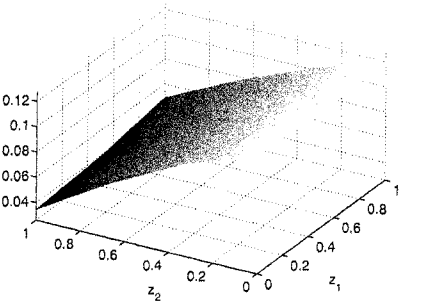


Fig. 7: Carbon dioxide in cathode channel

In accordance with the methane behavior hydrogen is produced in the endothermic reforming reaction (1) immediately after the anode inlet to be subsequently consumed in the exothermic oxidation reaction (2); see Fig. 5. Near $z_1 = 1$ and $z_2 = 1$, we see higher concentrations of hydrogen due to higher temperatures and consequently higher reforming reaction rates.

The molar flow in the anode, as depicted in Fig. 6, is increasing due to both the reforming reaction and the oxidation reaction.

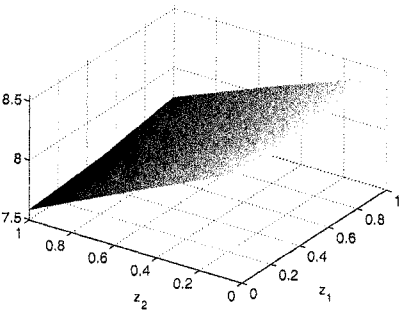


Fig. 8: Molar flow in cathode channel

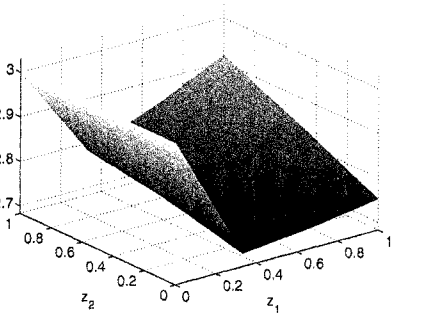


Fig. 9: Temperature in anode channel

As can be seen from Figs. 7 and 8 both the concentration of the carbon dioxide and the molar flow in the cathode decreases from the cathode inlet on the south-east side to its outlet on the north-west side due to the reduction reaction (3).

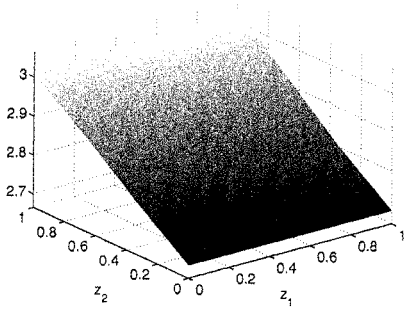


Fig. 10: Temperature in cathode channel

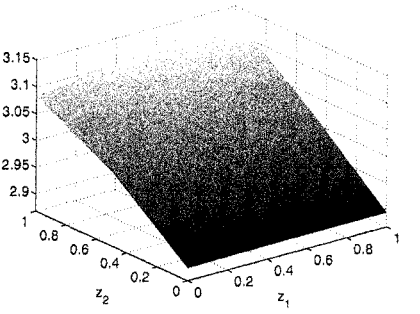


Fig. 11: Temperature in solid

The most important results are concerned with the temperatures; see Figs. 9–11. They directly influence the reaction rates. Moreover, the solid temperature, see Fig. 11, must be particularly observed to avoid so-called hot spots leading to material corrosion and consequently to a reduced service life.

The temperature distribution in the anode channel coincides with the heat demand and the heat release of the reactions therein. Initially we have the endothermic reaction (1). Thus the temperature declines. In the following, the anode gas temperature is increased again by heat exchange with the solid, which is heated by the electrochemical reactions (Eqs. 2, 3).

Since the solid temperature is not evenly distributed, see Fig. 11, the temperature in the anode channel has a local maximum near $z_1 = 1$ and $z_2 = 1$ due to heat conduction, see Fig. 9. The cathode gas is heated up by the solid phase all along the channel, so the temperature continuously increases along z_2.

4 Conclusions

A complicated mathematical model, describing the dynamical behavior of a molten carbonate fuel cell, has been presented. The semi-discretization in space of the large scale partial differential algebraic equation system with integral terms in the right hand side together with its nonstandard boundary conditions including an ODE system yields a very large scale DAE system. The obtained numerical results correspond to the practical experiences of engineers with real fuel cells of the type investigated here. This approach enables further investigations with respect to parameter identification as well as optimal control purposes, e.g. the optimization of the fuel cell when working in certain operation modes such as load changes. For those purposes we can use the same mathematical model as well as the same software NUDOCCCS used here, since NUDOCCCS can solve optimal control problems with large scale DAE constraints.

Acknowledgement

This work has been supported by the German Federal Ministry of Education and Research (BMBF) within the project *Optimierte Prozessführung von Brennstoffzellensystemen mit Methoden der Nichtlinearen Dynamik*. The authors greatfully acknowledge this support. The authors also greatfully appreciate the contributions of Prof. Dr.-Ing. Kai Sundmacher and Dr.-Ing. Peter Heidebrecht teaching us the necessary technical background and, in particular, for providing us with this challenging model.

References

1. Büskens, C.: Optimierungsmethoden und Sensitivitätsanalyse für optimale Steuerprozesse mit Steuer- und Zustands-Beschränkungen. Dissertation, Univ. Münster, Germany, 1998.
2. Chudej, K.: Index Analysis for Singular PDE Models of Fuel Cells. to appear in Proc. of ECMI 2004.
3. Chudej, K.; Heidebrecht, P.; Petzet, V.; Scherdel, S.; Schittkowski, K.; Pesch, H.J.; Sundmacher, K.: Index Analysis and Numerical Solution of a Large Scale Nonlinear PDAE System Describing the Dynamical Behaviour of Molten Carbonate Fuel Cells. Zeitschrift für Angewandte Mathematik und Mechanik 85, 2, 132-140 (2005).
4. Heidebrecht, P.: *Modelling, Analysis and Optimisation of a Molten Carbonate Fuel Cell with Direct Internal Reforming (DIR-MCFC)*. Dissertation, Fortschritt-Berichte, Reihe 3, Nr. 826, VDI-Verlag, Düsseldorf, 2005.
5. Heidebrecht, P., Sundmacher, K.: *Molten carbonate fuel cell (MCFC) with internal reforming: model-based analysis of cell dynamics*, Chemical Engineering Science **58** (2003) 1029–1036.
6. Heidebrecht, P., Sundmacher, K.: *Dynamic Modeling and Simulation of a Countercurrent Molten Carbonate Fuel Cell (MCFC) with Internal Reforming*, Fuel Cells **3–4** (2002), 166–180.
7. Pesch, H.J.; Chudej, K.; Petzet, V.; Scherdel, S.; Schittkowski, K.; Heidebrecht, P.; Sundmacher, K.: Numerical Simulation of a 1D Model of a Molten Carbonate Fuel Cell. Proc. Appl. Math. Mech. 3 (2003) 521–522.
8. Rang, J.; Chudej, K.: A perturbation index for a singular PDE model of a fuel cell. Mathematik-Bericht Nr. 2004/6, Institut für Mathematik, Technische Universität Clausthal (2004).
9. Sternberg, K., Chudej, K., Pesch, H.J.: A partial differential algebraic dynamic model of a molten carbonate fuel cell. Proc. Appl. Math. Mech. 4 (2004) 584-585.

The NEWUOA software for unconstrained optimization without derivatives

M.J.D. Powell

Department of Applied Mathematics and Theoretical Physics, Centre for Mathematical Sciences, Wilberforce Road, Cambridge CB3 0WA, England.
(mjdp@cam.ac.uk)

Summary. The NEWUOA software seeks the least value of a function $F(\underline{x})$, $\underline{x} \in \mathcal{R}^n$, when $F(\underline{x})$ can be calculated for any vector of variables $\underline{x}$. The algorithm is iterative, a quadratic model $Q \approx F$ being required at the beginning of each iteration, which is used in a trust region procedure for adjusting the variables. When Q is revised, the new Q interpolates F at m points, the value $m = 2n+1$ being recommended. The remaining freedom in the new Q is taken up by minimizing the Frobenius norm of the change to $\nabla^2 Q$. Only one interpolation point is altered on each iteration. Thus, except for occasional origin shifts, the amount of work per iteration is only of order $(m+n)^2$, which allows n to be quite large. Many questions were addressed during the development of NEWUOA, for the achievement of good accuracy and robustness. They include the choice of the initial quadratic model, the need to maintain enough linear independence in the interpolation conditions in the presence of computer rounding errors, and the stability of the updating of certain matrices that allow the fast revision of Q. Details are given of the techniques that answer all the questions that occurred. The software was tried on several test problems. Numerical results for nine of them are reported and discussed, in order to demonstrate the performance of the software for up to 160 variables.

Key words: direct search, quadratic models, trust regions, unconstrained minimization, updating.

1 Introduction

Quadratic approximations to the objective function are highly useful for obtaining a fast rate of convergence in iterative algorithms for unconstrained optimization, because usually some attention has to be given to the curvature of the objective function. On the other hand, each quadratic model has $\frac{1}{2}(n+1)(n+2)$ independent parameters, and this number of calculations of values of the objective function is prohibitively expensive in many applications with large n. Therefore the new algorithm tries to construct suitable

quadratic models from fewer data. The model $Q(\underline{x})$, $\underline{x} \in \mathcal{R}^n$, at the beginning of a typical iteration, has to satisfy only m interpolation conditions

$$Q(\underline{x}_i) = F(\underline{x}_i), \qquad i=1,2,\ldots,m, \tag{1.1}$$

where $F(\underline{x})$, $\underline{x} \in \mathcal{R}^n$, is the objective function, where the number m is prescribed by the user, and where the positions of the different points $\underline{x}_i$, $i=1,2,\ldots,m$, are generated automatically. We require $m \geq n+2$, in order that the equations (1.1) always provide some conditions on the second derivative matrix $\nabla^2 Q$, and we require $m \leq \frac{1}{2}(n+1)(n+2)$, because otherwise no quadratic model Q can satisfy all the equations (1.1) for general right hand sides. The numerical results in the last section of this paper give excellent support for the choice $m=2n+1$.

The success of the new algorithm is due to a technique that is suggested by the symmetric Broyden method for updating $\nabla^2 Q$ when first derivatives of F are available (see pages 195–198 of Dennis and Schnabel, 1983, for instance). Let an old model $Q_{\rm old}$ be present, and let the new model $Q_{\rm new}$ be required to satisfy some conditions that are compatible and that leave some freedom in the parameters of $Q_{\rm new}$. The technique takes up this freedom by minimizing $\|\nabla^2 Q_{\rm new} - \nabla^2 Q_{\rm old}\|_F$, where the subscript "$F$" denotes the Frobenius norm

$$\|A\|_F = \Big\{ \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij}^2 \Big\}^{1/2}, \qquad A \in \mathcal{R}^{n \times n}. \tag{1.2}$$

Our conditions on the new model $Q = Q_{\rm new}$ are the interpolation equations (1.1). Thus $\nabla^2 Q_{\rm new}$ is defined uniquely, and $Q_{\rm new}$ itself is also unique, because the automatic choice of the points $\underline{x}_i$ excludes the possibility that a nonzero linear polynomial $p(\underline{x})$, $\underline{x} \in \mathcal{R}^n$, has the property $p(\underline{x}_i)=0$, $i=1,2,\ldots,m$. In other words, the algorithm ensures that the rows of the $(n+1)\times m$ matrix

$$X = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \underline{x}_1 - \underline{x}_0 & \underline{x}_2 - \underline{x}_0 & \cdots & \underline{x}_m - \underline{x}_0 \end{pmatrix} \tag{1.3}$$

are linearly independent, where $\underline{x}_0$ is any fixed vector.

The strength of this updating technique can be explained by considering the case when the objective function F is quadratic. Guided by the model $Q = Q_{\rm old}$ at the beginning of the current iteration, a new vector of variables $\underline{x}_{\rm new} = \underline{x}_{\rm opt} + \underline{d}$ is chosen, where $\underline{x}_{\rm opt}$ is such that $F(\underline{x}_{\rm opt})$ is the least calculated value of F so far. If the error $|F(\underline{x}_{\rm new}) - Q_{\rm old}(\underline{x}_{\rm new})|$ is relatively small, then the model has done well in predicting the new value of F, even if the errors of the approximation $\nabla^2 Q \approx \nabla^2 F$ are substantial. On the other hand, if $|F(\underline{x}_{\rm new}) - Q_{\rm old}(\underline{x}_{\rm new})|$ is relatively large, then, by satisfying $Q_{\rm new}(\underline{x}_{\rm new}) = F(\underline{x}_{\rm new})$, the updating technique should improve the accuracy of the model significantly, which is a win/win situation. Numerical results show that these welcome alternatives provide excellent convergence in the vectors of variables

that are generated by the algorithm, although usually the second derivative error $\|\nabla^2 Q - \nabla^2 F\|_F$ is big for every Q that occurs. Thus the algorithm seems to achieve automatically the features of the quadratic model that give suitable changes to the variables, without paying much attention to other features of the approximation $Q \approx F$. This suggestion is made with hindsight, after discovering experimentally that the number of calculations of F is only $\mathcal{O}(n)$ in many cases that allow n to be varied. Further discussion of the efficiency of the updating technique can be found in Powell (2004b).

The first discovery of this kind, made in January 2002, is mentioned in Powell (2003). Specifically, by employing the least Frobenius norm updating method, an unconstrained minimization problem with 160 variables was solved to high accuracy, using only 9688 values of F, although quadratic models have 13122 independent parameters in the case $n=160$. Then the author began to develop a Fortran implementation of the new procedure for general use, but that task was not completed until December, 2003, because, throughout the first 18 months of the development, computer rounding errors caused unacceptable loss of accuracy in a few difficult test problems. A progress report on that work, with some highly promising numerical results, was presented at the conference in Hangzhou, China, that celebrated the tenth anniversary of the journal *Optimization Methods and Software* (Powell, 2004b). The author resisted pressure from the editor and referees of that paper to include a detailed description of the algorithm that calculated the given results, because of the occasional numerical instabilities. The loss of accuracy occurred in the part of the Fortran software that derives Q_{new} from Q_{old} in only $\mathcal{O}(m^2)$ computer operations, the change to Q being defined by an $(m+n+1)\times(m+n+1)$ system of linear equations. Let W be the matrix of this system. The inverse matrix $H=W^{-1}$ was stored and updated explicitly. In theory the rank of Ω, which is the leading $m\times m$ submatrix of H, is only $m-n-1$, but this property was lost in practice. Now, however, a factorization of Ω is stored instead of Ω itself, which gives the correct rank in a way that is not damaged by computer rounding errors. This device corrected the unacceptable loss of accuracy (Powell, 2004c), and then the remaining development of the final version of NEWUOA became straightforward. The purpose of the present paper is to provide details and some numerical results of the new algorithm.

An outline of the method of NEWUOA is given in Section 2, but m (the number of interpolation conditions) and the way of updating Q are not mentioned, so most of the outline applies also to the UOBYQA software of the author (Powell, 2002), where each quadratic model is defined by interpolation to $\frac{1}{2}(n+1)(n+2)$ values of F. The selection of the initial interpolation points and the construction of the first quadratic model are described in Section 3, with formulae for the initial matrix H and the factorization of Ω, as introduced in the previous paragraph. Not only Q but also H and the factorization of Ω are updated when the positions of the interpolation points are revised, which is the subject of Section 4. On most iterations, the change in variables $\underline{d}$ is an approximate solution to the trust region subproblem

$$\text{Minimize} \quad Q(\underline{x}_{\text{opt}}+\underline{d}) \quad \text{subject to} \quad \|\underline{d}\| \leq \Delta, \tag{1.4}$$

which receives attention in Section 5, the parameter $\Delta > 0$ being available with Q. Section 6 addresses an alternative way of choosing $\underline{d}$, which may be invoked when trust region steps fail to yield good reductions in F. Other details of the algorithm are considered in Section 7, including shifts of the origin of $\mathcal{R}^n$, which are necessary to avoid huge losses of accuracy when H is revised. Several numerical results are presented and discussed in Section 8. The first of these experiments suggests a modification to the procedure for updating the quadratic model, which was made to NEWUOA before the calculation of the other results. It seems that the new algorithm is suitable for a wide range of unconstrained minimization calculations. Proofs of some of the assertions of Section 3 are given in an appendix.

2 An outline of the method

The user of the NEWUOA software has to define the objective function by a Fortran subroutine that computes $F(\underline{x})$ for any vector of variables $\underline{x} \in \mathcal{R}^n$. An initial vector $\underline{x}_0 \in \mathcal{R}^n$, the number m of interpolation conditions (1.1), and the initial and final values of a trust region radius, namely ρ_{beg} and ρ_{end}, are required too. It is mentioned in Section 1 that m is a fixed integer from the interval

$$n+2 \leq m \leq \tfrac{1}{2}(n+1)(n+2), \tag{2.1}$$

and that often the choice $m = 2n+1$ is good for efficiency. The initial interpolation points $\underline{x}_i$, $i = 1, 2, \ldots, m$, include $\underline{x}_0$, while the other points have the property $\|\underline{x}_i - \underline{x}_0\|_\infty = \rho_{\text{beg}}$, as specified in Section 3. The choice of ρ_{beg} should be such that the computed values of F at these points provide useful information about the behaviour of the true objective function near $\underline{x}_0$, especially when the computations may include some spurious contributions that are larger than rounding errors. The parameter ρ_{end}, which has to satisfy $\rho_{\text{end}} \leq \rho_{\text{beg}}$, should have the magnitude of the required accuracy in the final values of the variables.

An outline of the method is given in Figure 1. The details of the operations of Box 1 are addressed in Section 3. The parameter ρ is a lower bound on the trust region radius Δ from the interval $[\rho_{\text{end}}, \rho_{\text{beg}}]$. The value of Δ is revised on most iterations, but the purpose of ρ is to maintain enough distance between the interpolation points $\underline{x}_i$, $i = 1, 2, \ldots, m$, in order to restrict the damage to Q from the interpolation conditions (1.1) when there are substantial errors in each computation of F. Therefore ρ is altered only when the constraint $\Delta \geq \rho$ seems to be preventing futher reductions in the objective function. Each change to ρ is a decrease by about a factor of ten, except that the iterations are terminated in the case $\rho = \rho_{\text{end}}$, as shown in Boxes 11-13 of the figure.

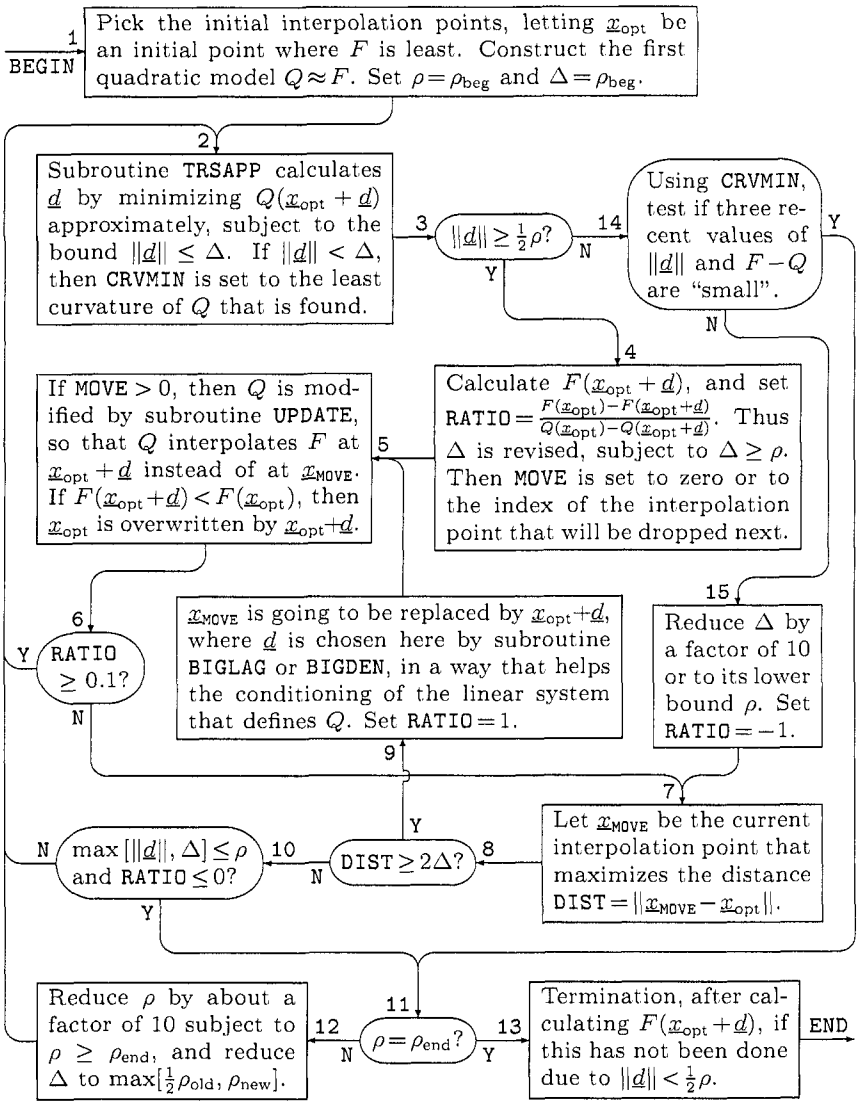


Fig. 1: An outline of the method, where Y = Yes and N = No

Boxes 2-6 of Figure 1 are followed in sequence when the algorithm performs a trust region iteration that calculates a new value of F. The step $\underline{d}$ from $\underline{x}_{\rm opt}$ is derived from the subproblem (1.4) in Box 2 by the truncated conjugate gradient procedure of Section 5. If $\|\underline{d}\| < \Delta$ occurs here, then Q has positive curvature along every search direction of that procedure, and CRVMIN is set to the least of those curvatures, for use in Box 14 when the N branch is taken from Box 3, which receives attention later. Box 4 is reached in the present case, however, where Δ is revised in a way that depends on the ratio

$$\texttt{RATIO} = \{F(\underline{x}_{\rm opt}) - F(\underline{x}_{\rm opt}+\underline{d})\} \,/\, \{Q(\underline{x}_{\rm opt}) - Q(\underline{x}_{\rm opt}+\underline{d})\}, \qquad (2.2)$$

as described in Section 7. The other task of Box 4 is to pick the m interpolation points of the next quadratic model. Usually one of the current points $\underline{x}_i$, $i=1,2,\ldots,m$, is replaced by $\underline{x}_{\rm opt}+\underline{d}$, and all the other points are retained. In this case the integer MOVE is set in Box 4 to the index of the interpolation point that is dropped. The only other possibility is no change to the interpolation equations, and then MOVE is set to zero. Details of the choice of MOVE are also given in Section 7, the case MOVE > 0 being mandatory when the strict reduction $F(\underline{x}_{\rm opt}+\underline{d}) < F(\underline{x}_{\rm opt})$ is achieved, in order that the best calculated value of F so far is among the new interpolation conditions. The updating operations of Box 5 are the subject of Section 4. Box 6 branches back to Box 2 for another trust region iteration if the ratio (2.2) is sufficiently large.

The N branch is taken from Box 6 of the figure when Box 4 has provided a change $F(\underline{x}_{\rm opt}) - F(\underline{x}_{\rm opt}+\underline{d})$ in the objective function that compares unfavourably with the predicted reduction $Q(\underline{x}_{\rm opt}) - Q(\underline{x}_{\rm opt}+\underline{d})$. Usually this happens because the positions of the points $\underline{x}_i$ in the interpolation equations (1.1) are unsuitable for maintaining a good quadratic model, especially when the trust region iterations have caused some of the distances $\|\underline{x}_i - \underline{x}_{\rm opt}\|$, $i=1,2,\ldots,m$, to be much greater than Δ. Therefore the purpose of Box 7 is to identify the current interpolation point, $\underline{x}_{\texttt{MOVE}}$ say, that is furthest from $\underline{x}_{\rm opt}$. We take the view that, if $\|\underline{x}_{\texttt{MOVE}} - \underline{x}_{\rm opt}\| \geq 2\Delta$ holds, then Q can be improved substantially by replacing the interpolation condition $Q(\underline{x}_{\texttt{MOVE}}) = F(\underline{x}_{\texttt{MOVE}})$ by $Q(\underline{x}_{\rm opt}+\underline{d}) = F(\underline{x}_{\rm opt}+\underline{d})$, for some step $\underline{d}$ that satisfies $\|\underline{d}\| \leq \Delta$. We see in the figure that the actual choice of $\underline{d}$ is made in Box 9, details being given in Section 6, because they depend on the updating formulae of Section 4. Then Box 5 is reached from Box 9, in order to update Q as before, after the calculation of the new function value $F(\underline{x}_{\rm opt}+\underline{d})$. In this case the branch from Box 6 to Box 2 is always followed, due to the setting of the artificial value RATIO $= 1$ at the end of Box 9. Thus the algorithm makes use immediately of the new information in the quadratic model.

The N branch is taken from Box 8 when the positions of the current points $\underline{x}_i$, $i=1,2,\ldots,m$, are under consideration, and when they have the property

$$\|\underline{x}_i - \underline{x}_{\rm opt}\| \; < \; 2\Delta, \qquad i=1,2,\ldots,m. \qquad (2.3)$$

Then the tests in Box 10 determine whether the work with the current value of ρ is complete. We see that the work continues if and only if one or more

of the conditions $\|\underline{d}\| > \rho$, $\Delta > \rho$ or RATIO > 0 holds. Another trust region iteration is performed with the same ρ in the first two cases, because ρ has not restricted the most recent choice of $\underline{d}$. In the third case, RATIO > 0 implies $F(\underline{x}_{\rm opt}+\underline{d}) < F(\underline{x}_{\rm opt})$ in Box 4, and we prefer to retain the old ρ while strict reductions in the objective function are being obtained. Thus an infinite loop with ρ fixed may happen in theory. In practice, however, the finite precision of the computer arithmetic provides an upper bound on the number of different values of F that can occur.

Finally, we consider the operations of Figure 1 when the step $\underline{d}$ of Box 2 satisfies $\|\underline{d}\| < \frac{1}{2}\rho$. Then Box 14 is reached from Box 3, and often $F(\underline{x}_{\rm opt}+\underline{d})$ is not going to be calculated, because, as mentioned already, the computed difference $F(\underline{x}_{\rm opt})-F(\underline{x}_{\rm opt}+\underline{d})$ tends to give misleading information about the true objective function when $\|\underline{d}\|$ becomes small. If Box 14 branches to Box 15, a big reduction is made in Δ if allowed by $\Delta \geq \rho$, and then, beginning at Box 7, there is a choice as before between replacing the interpolation point $\underline{x}_{\rm MOVE}$, or performing a trust region iteration with the new Δ, or going to Box 11 because the work with the current ρ is complete. Alternatively, we see that Box 14 can branch directly to Box 11, the reason being as follows.

Let $\hat{\underline{x}}_{\rm opt}$ and $\check{\underline{x}}_{\rm opt}$ be the first and last values of $\underline{x}_{\rm opt}$ during all the work with the current ρ, and let $\hat{\underline{x}}_i$, $i = 1, 2, \ldots, m$, be the interpolation points at the start of this part of the computation. When ρ is less than $\rho_{\rm beg}$, the current ρ was selected in Box 12, and, because it is much smaller than its previous value, we expect the points $\hat{\underline{x}}_i$ to satisfy $\|\hat{\underline{x}}_i - \hat{\underline{x}}_{\rm opt}\| \geq 2\rho$, $i \neq {\rm opt}$. On the other hand, because of Boxes 7 and 8 in the figure, Box 11 can be reached from Box 10 only in the case $\|\underline{x}_i - \check{\underline{x}}_{\rm opt}\| < 2\rho$, $i = 1, 2, \ldots, m$. These remarks suggest that at least $m-1$ new values of the objective function may be calculated for the current ρ. It is important to efficiency, however, to include a less laborious route to Box 11, especially when m is large and $\rho_{\rm end}$ is tiny. Details of the tests that pick the Y branch from Box 14 are given in Section 7. They are based on the assumption that there is no need for further improvements to the model Q, if the differences $|F(\underline{x}_{\rm opt}+\underline{d}) - Q(\underline{x}_{\rm opt}+\underline{d})|$ of recent iterations compare favourably with the current second derivative term $\frac{1}{8}\rho^2$CRVMIN.

When the Y branch is taken from Box 14, we let $\underline{d}_{\rm old}$ be the vector $\underline{d}$ that has satisfied $\|\underline{d}\| < \frac{1}{2}\rho$ in Box 3 of the current iteration. Often $\underline{d}_{\rm old}$ is an excellent step to take from $\underline{x}_{\rm opt}$ in the space of the variables, so we wish to allow its use after leaving Box 11. If Box 2 is reached from Box 11 via Box 12, then $\underline{d} = \underline{d}_{\rm old}$ is generated again, because the quadratic model is the same as before, and the change to Δ in Box 12 preserves the property $\Delta \geq \frac{1}{2}\rho_{\rm old} > \|\underline{d}_{\rm old}\|$. Alternatively, if the Y branches are taken from Boxes 14 and 11, we see in Box 13 that $F(\underline{x}_{\rm opt}+\underline{d}_{\rm old})$ is computed. The NEWUOA software returns to the user the first vector of variables that gives the least of the calculated values of the objective function.

3 The initial calculations

We write the quadratic model of the first iteration in the form

$$Q(\underline{x}_0+\underline{d}) \;=\; Q(\underline{x}_0)+\underline{d}^T\underline{\nabla}Q(\underline{x}_0)+\tfrac{1}{2}\,\underline{d}^T\nabla^2Q\,\underline{d}, \qquad \underline{d}\in\mathcal{R}^n, \tag{3.1}$$

$\underline{x}_0$ being the initial vector of variables that is provided by the user. When the number of interpolation conditions (1.1) satisfies $m\geq 2n+1$, the first $2n+1$ of the points $\underline{x}_i$, $i=1,2,\ldots,m$, are chosen to be the vectors

$$\underline{x}_1=\underline{x}_0 \qquad \text{and} \qquad \left.\begin{aligned}\underline{x}_{i+1}&=\underline{x}_0+\rho_{\rm beg}\,\underline{e}_i\\ \underline{x}_{i+n+1}&=\underline{x}_0-\rho_{\rm beg}\,\underline{e}_i\end{aligned}\right\}, \quad i=1,2,\ldots,n, \tag{3.2}$$

where $\rho_{\rm beg}$ is also provided by the user as mentioned already, and where $\underline{e}_i$ is the i-th coordinate vector in $\mathcal{R}^n$. Thus $Q(\underline{x}_0)$, $\underline{\nabla}Q(\underline{x}_0)$ and the diagonal elements $(\nabla^2Q)_{ii}$, $i=1,2,\ldots,n$, are given uniquely by the first $2n+1$ of the equations (1.1). Alternatively, when m satisfies $n+2\leq m\leq 2n$, the initial interpolation points are the first m of the vectors (3.2). It follows that $Q(\underline{x}_0)$, the first $m-n-1$ components of $\underline{\nabla}Q(\underline{x}_0)$ and $(\nabla^2Q)_{ii}$, $i=1,2,\ldots,m-n-1$, are defined as before. The other diagonal elements of ∇^2Q are set to zero, so the other components of $\underline{\nabla}Q(\underline{x}_0)$ take the values $\{F(\underline{x}_0+\rho_{\rm beg}\,\underline{e}_i)-F(\underline{x}_0)\}/\rho_{\rm beg}$, $m-n\leq i\leq n$.

In the case $m>2n+1$, the initial points $\underline{x}_i$, $i=1,2,\ldots,m$, are chosen so that the conditions (1.1) also provide $2(m-2n-1)$ off-diagonal elements of ∇^2Q, the factor 2 being due to symmetry. Specifically, for $i\in[2n+2,m]$, the point $\underline{x}_i$ has the form

$$\underline{x}_i \;=\; \underline{x}_0+\sigma_p\,\rho_{\rm beg}\,\underline{e}_p+\sigma_q\,\rho_{\rm beg}\,\underline{e}_q, \tag{3.3}$$

where p and q are different integers from $[1,n]$, and where σ_p and σ_q are included in the definitions

$$\sigma_j \;=\; \begin{cases} -1, & F(\underline{x}_0-\rho_{\rm beg}\,\underline{e}_j)<F(\underline{x}_0+\rho_{\rm beg}\,\underline{e}_j)\\ +1, & F(\underline{x}_0-\rho_{\rm beg}\,\underline{e}_j)\geq F(\underline{x}_0+\rho_{\rm beg}\,\underline{e}_j),\end{cases} \qquad j=1,2,\ldots,n, \tag{3.4}$$

which biases the choice (3.3) towards smaller values of the objective function. Thus the element $(\nabla^2Q)_{pq}=(\nabla^2Q)_{qp}$ is given by the equations (1.1), since every quadratic function $Q(\underline{x})$, $\underline{x}\in\mathcal{R}^n$, has the property

$$\begin{aligned}\rho_{\rm beg}^{-2}\,\{Q(\underline{x}_0)-Q(\underline{x}_0+\sigma_p\,\rho_{\rm beg}\,\underline{e}_p)-Q(\underline{x}_0+\sigma_q\,\rho_{\rm beg}\,\underline{e}_q)\\ +Q(\underline{x}_0+\sigma_p\,\rho_{\rm beg}\,\underline{e}_p+\sigma_q\,\rho_{\rm beg}\,\underline{e}_q)\}\;=\;\sigma_p\,\sigma_q\,(\nabla^2Q)_{pq}.\end{aligned} \tag{3.5}$$

For simplicity, we pick p and q in formula (3.3) in the following way. We let j be the integer part of the quotient $(i-n-2)/n$, which satisfies $j\geq 1$ due to $i\geq 2n+2$, we set $p=i-n-1-jn$, which is in the interval $[1,n]$, and we let q have the value $p+j$ or $p+j-n$, the latter choice being made in the case

$p+j>n$. Hence, if $n=5$ and $m=20$, for example, there are 9 pairs $\{p,q\}$, generated in the order $\{1,2\}$, $\{2,3\}$, $\{3,4\}$, $\{4,5\}$, $\{5,1\}$, $\{1,3\}$, $\{2,4\}$, $\{3,5\}$ and $\{4,1\}$. All the off-diagonal elements of $\nabla^2 Q$ that are not provided by the method of this paragraph are set to zero, which completes the specification of the initial quadratic model (3.1).

The preliminary work of NEWUOA includes also the setting of the initial matrix $H=W^{-1}$, where W occurs in the linear system of equations that defines the change to the quadratic model. We recall from Section 1 that, when Q is updated from $Q_{\rm old}$ to $Q_{\rm new}=Q_{\rm old}+D$, say, the quadratic function D is constructed so that $\|\nabla^2 D\|_F^2$ is least subject to the constraints

$$D(\underline{x}_i) \;=\; F(\underline{x}_i)-Q_{\rm old}(\underline{x}_i), \qquad i=1,2,\ldots,m, \tag{3.6}$$

these constraints being equivalent to $Q_{\rm new}(\underline{x}_i)=F(\underline{x}_i)$, $i=1,2,\ldots,m$. We see that the calculation of D is a quadratic programming problem, and we let λ_j, $j=1,2,\ldots,m$, be the Lagrange multipliers of its KKT conditions. They have the properties

$$\sum_{j=1}^m \lambda_j \;=\; 0 \qquad \text{and} \qquad \sum_{j=1}^m \lambda_j\,(\underline{x}_j-\underline{x}_0) \;=\; 0, \tag{3.7}$$

and the second derivative matrix of D takes the form

$$\nabla^2 D \;=\; \sum_{j=1}^m \lambda_j\,\underline{x}_j\,\underline{x}_j^T \;=\; \sum_{j=1}^m \lambda_j\,(\underline{x}_j-\underline{x}_0)\,(\underline{x}_j-\underline{x}_0)^T \tag{3.8}$$

(Powell, 2004a), the last part of expression (3.8) being a consequence of the equations (3.7). This form of $\nabla^2 D$ allows D to be the function

$$D(\underline{x}) \;=\; c+(\underline{x}-\underline{x}_0)^T\underline{g}+\tfrac{1}{2}\sum_{j=1}^m \lambda_j\,\{(\underline{x}-\underline{x}_0)^T(\underline{x}_j-\underline{x}_0)\}^2, \qquad \underline{x}\in\mathcal{R}^n, \tag{3.9}$$

and we seek the values of the parameters $c\in\mathcal{R}$, $\underline{g}\in\mathcal{R}^n$ and $\underline{\lambda}\in\mathcal{R}^m$. The conditions (3.6) and (3.7) give the square system of linear equations

$$\left(\begin{array}{c|c} A & X^T \\ \hline X & 0 \end{array}\right)\left(\begin{array}{c} \underline{\lambda} \\ \hline c \\ \underline{g} \end{array}\right) \;=\; \left(\begin{array}{c} \underline{r} \\ \hline 0 \end{array}\right) \begin{array}{l} \updownarrow\, m \\ \updownarrow\, n+1\,, \end{array} \tag{3.10}$$

where A has the elements

$$A_{ij} \;=\; \tfrac{1}{2}\,\{(\underline{x}_i-\underline{x}_0)^T(\underline{x}_j-\underline{x}_0)\}^2, \qquad 1\le i,j\le m, \tag{3.11}$$

where X is the matrix (1.3), and where $\underline{r}$ has the components $F(\underline{x}_i)-Q_{\rm old}(\underline{x}_i)$, $i=1,2,\ldots,m$. Therefore W and H are the matrices

$$W = \left(\begin{array}{c|c} A & X^T \\ \hline X & 0 \end{array}\right) \qquad \text{and} \qquad H = W^{-1} = \left(\begin{array}{c|c} \Omega & \Xi^T \\ \hline \Xi & \Upsilon \end{array}\right), \tag{3.12}$$

say. It is straightforward to derive the elements of W from the vectors $\underline{x}_i$, $i = 1,2,\ldots,m$, but we require the elements of Ξ and Υ explicitly, with a factorization of Ω. Fortunately, the chosen positions of the initial interpolation points provide convenient formulae for all of these terms, as stated below. Proofs of the correctness of the formulae are given in the appendix.

The first row of the initial $(n+1)\times m$ matrix Ξ has the very simple form

$$\Xi_{1j} = \delta_{1j}, \qquad j=1,2,\ldots,m. \tag{3.13}$$

Further, for integers i that satisfy $2 \le i \le \min[n+1, m-n]$, the i-th row of Ξ has the nonzero elements

$$\Xi_{ii} = (2\,\rho_{\rm beg})^{-1} \qquad \text{and} \qquad \Xi_{i\,i+n} = -(2\,\rho_{\rm beg})^{-1}, \tag{3.14}$$

all the other entries being zero, which defines the initial Ξ in the cases $m \ge 2n+1$. Otherwise, when $m-n+1 \le i \le n+1$ holds, the i-th row of the initial Ξ also has just two nonzero elements, namely the values

$$\Xi_{i1} = -(\rho_{\rm beg})^{-1} \qquad \text{and} \qquad \Xi_{ii} = (\rho_{\rm beg})^{-1}, \tag{3.15}$$

which completes the definition of Ξ for the given interpolation points. Moreover, the initial $(n+1)\times(n+1)$ matrix Υ is amazingly sparse, being identically zero in the cases $m \ge 2n+1$. Otherwise, its only nonzero elements are the last $2n-m+1$ diagonal entries, which take the values

$$\Upsilon_{ii} = -\tfrac{1}{2}\,\rho_{\rm beg}^2, \qquad m-n+1 \le i \le n+1. \tag{3.16}$$

The factorization of Ω, mentioned in Section 1, guarantees that the rank of Ω is at most $m-n-1$, by having the form

$$\Omega = \sum_{k=1}^{m-n-1} s_k\,\underline{z}_k\,\underline{z}_k^T = \sum_{k=1}^{m-n-1} \underline{z}_k\,\underline{z}_k^T = Z\,Z^T, \tag{3.17}$$

the second equation being valid because each s_k is set to one initially. When $1 \le k \le \min[n, m-n-1]$, the components of the initial vector $\underline{z}_k \in \mathcal{R}^m$, which is the k-th column of Z, are given the values

$$\left.\begin{array}{ll} Z_{1k} = -\sqrt{2}\,\rho_{\rm beg}^{-2}, & Z_{k+1\,k} = \frac{1}{2}\sqrt{2}\,\rho_{\rm beg}^{-2}, \\ Z_{k+n+1\,k} = \frac{1}{2}\sqrt{2}\,\rho_{\rm beg}^{-2}, & Z_{jk} = 0 \ \text{otherwise}, \end{array}\right\} \tag{3.18}$$

so each of these columns has just three nonzero elements. Alternatively, when $m > 2n+1$ and $n+1 \le k \le m-n-1$, the initial $\underline{z}_k$ depends on the choice (3.3) of $\underline{x}_i$ in the case $i = k+n+1$. We let p, q, σ_p and σ_q be as before, and we define $\hat{p}$ and $\hat{q}$ by the equations

$$\underline{x}_{\hat{p}} = \underline{x}_0 + \sigma_p \rho_{\rm beg} \underline{e}_p \qquad \text{and} \qquad \underline{x}_{\hat{q}} = \underline{x}_0 + \sigma_q \rho_{\rm beg} \underline{e}_q. \tag{3.19}$$

It follows from the positions of the interpolation points that $\hat{p}$ is either $p{+}1$ or $p{+}n{+}1$, while $\hat{q}$ is either $q{+}1$ or $q{+}n{+}1$. Now there are four nonzero elements in the k-th column of Z, the initial $\underline{z}_k$ being given the components

$$\left.\begin{aligned} Z_{1k} &= \rho_{\rm beg}^{-2}, & Z_{\hat{p}k} = Z_{\hat{q}k} &= -\rho_{\rm beg}^{-2}, \\ Z_{k+n+1\,k} &= \rho_{\rm beg}^{-2}, & Z_{jk} &= 0 \ \text{ otherwise.} \end{aligned}\right\} \tag{3.20}$$

All the given formulae for the nonzero elements of $H = W^{-1}$ are applied in only $\mathcal{O}(m)$ operations, due to the convenient choice of the initial interpolation points, but the work of setting the zero elements of Ξ, Υ and Z is $\mathcal{O}(m^2)$. The description of the preliminary work of NEWUOA is complete.

4 The updating procedures

In this section we consider the change that is made to the quadratic model Q on each iteration of NEWUOA that alters the set of interpolation points. We let the new points have the positions

$$\left.\begin{aligned} \underline{x}_t^+ &= \underline{x}_{\rm opt} + \underline{d} = \underline{x}^+, \quad \text{say,} \\ \underline{x}_i^+ &= \underline{x}_i, \qquad i \in \{1,2,\ldots,m\}\backslash\{t\}, \end{aligned}\right\} \tag{4.1}$$

which agrees with the outline of the method in Figure 1, because now we write t instead of MOVE. The change $D = Q_{\rm new} - Q_{\rm old}$ has to satisfy the analogue of the conditions (3.6) for the new points, and $Q_{\rm old}$ interpolates F at the old interpolation points. Thus D is the quadratic function that minimizes $\|\nabla^2 D\|_F$ subject to the constraints

$$D(\underline{x}_i^+) = \left\{F(\underline{x}^+) - Q_{\rm old}(\underline{x}^+)\right\} \delta_{it}, \qquad i = 1,2,\ldots,m. \tag{4.2}$$

Let W^+ and H^+ be the matrices

$$W^+ = \left(\begin{array}{c|c} A^+ & (X^+)^T \\ \hline X^+ & 0 \end{array}\right) \quad \text{and} \quad H^+ = (W^+)^{-1} = \left(\begin{array}{c|c} \Omega^+ & (\Xi^+)^T \\ \hline \Xi^+ & \Upsilon^+ \end{array}\right), \tag{4.3}$$

where A^+ and X^+ are defined by replacing the old interpolation points by the new ones in equations (1.3) and (3.11). It follows from the derivation of the system (3.10) and from the conditions (4.2) that D is now the function

$$D(\underline{x}) = c^+ + (\underline{x} - \underline{x}_0)^T \underline{g}^+ + \tfrac{1}{2} \sum_{j=1}^m \lambda_j^+ \{(\underline{x} - \underline{x}_0)^T (\underline{x}_j^+ - \underline{x}_0)\}^2, \qquad \underline{x} \in \mathcal{R}^n, \tag{4.4}$$

the parameters being the components of the vector

$$\begin{pmatrix} \underline{\lambda}^+ \\ c^+ \\ \underline{g}^+ \end{pmatrix} = \left\{F(\underline{x}^+) - Q_{\rm old}(\underline{x}^+)\right\} H^+ \underline{e}_t, \tag{4.5}$$

where $\underline{e}_t$ is now in $\mathcal{R}^{m+n+1}$. Expressions (4.5) and (4.4) are used by the NEWUOA software to generate the function D for the updating formula

$$Q_{\rm new}(\underline{x}) = Q_{\rm old}(\underline{x}) + D(\underline{x}), \qquad \underline{x} \in \mathcal{R}^n. \tag{4.6}$$

The matrix $H = W^{-1}$ is available at the beginning of the current iteration, the submatrices Ξ and Υ being stored explicitly, with the factorization $\sum_{k=1}^{m-n-1} s_k \underline{z}_k \underline{z}_k^T$ of Ω that has been mentioned, but H^+ occurs in equation (4.5). Therefore Ξ and Υ are overwritten by the submatrices Ξ^+ and Υ^+ of expression (4.3), and also the new factorization

$$\Omega^+ = \sum_{k=1}^{m-n-1} s_k^+ \underline{z}_k^+ (\underline{z}_k^+)^T \tag{4.7}$$

is required. Fortunately, the amount of work of these tasks is only $\mathcal{O}(m^2)$ operations, by taking advantage of the simple change (4.1) to the interpolation points. Indeed, we deduce from equations (4.1), (1.3), (3.11), (3.12) and (4.3) that all differences between the elements of W and W^+ are confined to the t-th row and column. Thus $W^+ - W$ is a matrix of rank two, which implies that the rank of $H^+ - H$ is also two. Therefore Ξ^+, Υ^+ and the factorization (4.7) are constructed from H by an extension of the Sherman–Morrison formula. Details and some relevant analysis are given in Powell (2004c), so only a brief outline of these calculations is presented below, before considering the implementation of formula (4.6). The updating of H in $\mathcal{O}(m^2)$ operations is highly important to the efficiency of the NEWUOA software, since an *ab initio* calculation of the change (4.4) to the quadratic model would require $\mathcal{O}(m^3)$ computer operations.

In theory, H^+ is the inverse of the matrix W^+ that has the elements

$$\left.\begin{aligned} W_{it}^+ &= W_{ti}^+ = (W^+ \underline{e}_t)_i, \quad i = 1, 2, \ldots, m+n+1, \\ W_{ij}^+ &= W_{ij} = H_{ij}^{-1}, \quad \text{otherwise}, \quad 1 \le i, j \le m+n+1. \end{aligned}\right\} \tag{4.8}$$

It follows from the right hand sides of this expression that H and the t-th column of W^+ provide enough information for the derivation of H^+. The definitions (1.3) and (3.11) show that $W^+ \underline{e}_t$ has the components

$$\left.\begin{aligned} W_{it}^+ &= \tfrac{1}{2} \{(\underline{x}_i^+ - \underline{x}_0)^T (\underline{x}^+ - \underline{x}_0)\}^2, \qquad i = 1, 2, \ldots, m \\ W_{m+1\,t}^+ &= 1 \quad \text{and} \quad W_{i+m+1\,t}^+ = (\underline{x}^+ - \underline{x}_0)_i, \quad i = 1, 2, \ldots, n \end{aligned}\right\}, \tag{4.9}$$

the notation $\underline{x}^+$ being used instead of $\underline{x}_t^+$, because $\underline{x}^+ = \underline{x}_{\rm opt} + \underline{d}$ is available before $t =$ MOVE is picked in Box 4 of Figure 1. Of course t must have the

property that W^+ is nonsingular, which holds if no divisions by zero occur when H^+ is calculated. Therefore we employ a formula for H^+ that gives conveniently the dependence of H^+ on t. Let the components of $\underline{w}\in\mathcal{R}^{m+n+1}$ take the values

$$\left.\begin{aligned} w_i &= \tfrac{1}{2}\{(\underline{x}_i-\underline{x}_0)^T(\underline{x}^+-\underline{x}_0)\}^2, && i=1,2,\ldots,m \\ w_{m+1} &= 1 \quad \text{and} \quad w_{i+m+1} = (\underline{x}^+-\underline{x}_0)_i, && i=1,2,\ldots,n \end{aligned}\right\}, \tag{4.10}$$

so $\underline{w}$ is independent of t. Equations (4.1), (4.9) and (4.10) imply that $W^+\underline{e}_t$ differs from $\underline{w}$ only in its t-th component, which allows H^+ to be written in terms of H, $\underline{w}$ and $\underline{e}_t$. Specifically, Powell (2004a) derives the formula

$$\begin{aligned} H^+=H+\sigma^{-1}\Big[&\alpha\,(\underline{e}_t-H\,\underline{w})\,(\underline{e}_t-H\,\underline{w})^T-\beta\,H\,\underline{e}_t\,\underline{e}_t^T\,H \\ &+\tau\,\{H\,\underline{e}_t\,(\underline{e}_t-H\,\underline{w})^T+(\underline{e}_t-H\,\underline{w})\,\underline{e}_t^T\,H\}\Big], \end{aligned} \tag{4.11}$$

the parameters being the expressions

$$\left.\begin{aligned} &\alpha=\underline{e}_t^T\,H\,\underline{e}_t, \quad \beta=\tfrac{1}{2}\,\|\underline{x}^+-\underline{x}_0\|^4-\underline{w}^T\,H\,\underline{w}, \\ &\tau=\underline{e}_t^T\,H\,\underline{w} \quad \text{and} \quad \sigma=\alpha\,\beta+\tau^2. \end{aligned}\right\} \tag{4.12}$$

We see that $H\underline{w}$ and β can be calculated before t is chosen, so it is inexpensive in practice to investigate the dependence of the denominator σ on t, in order to ensure that $|\sigma|$ is sufficiently large. The actual selection of t is addressed in Section 7.

Formula (4.11) was applied by an early version of NEWUOA, before the introduction of the factorization of Ω. The bottom left $(n+1)\times m$ and bottom right $(n+1)\times(n+1)$ submatrices of this formula are still used to construct Ξ^+ and Υ^+ from Ξ and Υ, respectively, the calculation of $H\underline{w}$ and $H\underline{e}_t$ being straightforward when the terms s_k and $\underline{z}_k$, $k=1,2,\ldots,m-n-1$, of the factorization (3.17) are stored instead of Ω.

The purpose of the factorization is to reduce the damage from rounding errors to the identity $W=H^{-1}$, which holds in theory at the beginning of each iteration. It became obvious from numerical experiments, however, that huge errors may occur in H in practice, including a few negative values of H_{ii}, $1\le i\le m$, although Ω should be positive semi-definite. Therefore we consider the updating of H when H is very different from W^{-1}, assuming that the calculations of the current iteration are exact. Then H^+ is the inverse of the matrix that has the elements on the right hand side of expression (4.8), which gives the identities

$$\left.\begin{aligned} &(H^+)^{-1}_{it}=W^+_{it} \quad \text{and} \quad (H^+)^{-1}_{ti}=W^+_{ti}, \quad i=1,2,\ldots,m+n+1, \\ &W^+_{ij}-(H^+)^{-1}_{ij}=W_{ij}-H^{-1}_{ij}, \quad \text{otherwise}, \quad 1\le i,j\le m+n+1. \end{aligned}\right\} \tag{4.13}$$

In other words, the overwriting of W and H by W^+ and H^+ makes no difference to the elements of $W-H^{-1}$, except that the t-th row and column of this

error matrix become zero. It follows that, when all the current interpolation points have been discarded by future iterations, then all the current errors in the first m rows and columns of $W-H^{-1}$ will have been annihilated. Equation (4.13) suggests, however, that any errors in the bottom right $(n+1)\times(n+1)$ submatrix of H^{-1} are retained. The factorization (3.17) provides the perfect remedy to this situation. Indeed, if H is any nonsingular $(m+n+1)\times(m+n+1)$ matrix of the form (3.12), and if the rank of the leading $m\times m$ submatrix Ω is $m-n-1$, then the bottom right $(n+1)\times(n+1)$ submatrix of H^{-1} is zero, which can be proved by expressing the elements of H^{-1} as cofactors of H divided by $\det H$ (Powell, 2004c). Thus the very welcome property

$$(H^+)^{-1}_{ij} = W^+_{ij} = 0, \qquad m+1 \le i,j \le m+n+1, \tag{4.14}$$

is guaranteed by the factorization (4.7), even in the presence of computer rounding errors.

The updating of the factorization of Ω by NEWUOA depends on the fact that the values

$$s^+_k = s_k \quad \text{and} \quad \underline{z}^+_k = \underline{z}_k, \qquad k \in \mathcal{K}, \tag{4.15}$$

are suitable in expression (4.7), where k is in $\mathcal{K}$ if and only if the t-th component of $\underline{z}_k$ is zero. Before taking advantage of this fact, an elementary change is made if necessary to the terms of the sum

$$\Omega = \sum_{k=1}^{m-n-1} s_k \, \underline{z}_k \, \underline{z}_k^T, \tag{4.16}$$

which forces the number of integers in $\mathcal{K}$ to be at least $m-n-3$. Specifically, NEWUOA employs the remark that, when $s_i = s_j$ holds in equation (4.16), then the equation remains true if $\underline{z}_i$ and $\underline{z}_j$ are replaced by the vectors

$$\cos\theta \, \underline{z}_i + \sin\theta \, \underline{z}_j \qquad \text{and} \qquad -\sin\theta \, \underline{z}_i + \cos\theta \, \underline{z}_j, \tag{4.17}$$

respectively, for any $\theta \in [0, 2\pi]$. The choice of θ allows either i or j to be added to $\mathcal{K}$ if both i and j were not in $\mathcal{K}$ previously. Thus, because $s_k = \pm 1$ holds for each k, only one or two of the new vectors $\underline{z}^+_k$, $k = 1, 2, \ldots, m-n-1$, have to be calculated after retaining the values (4.15). When $|\mathcal{K}| = m-n-2$, we let $\underline{z}^+_{m-n-1}$ be the required new vector, which is the usual situation as the theoretical positive definiteness of Ω should exclude negative values of s_k. Then the last term of the new factorization (4.7) is defined by the equations

$$\left.\begin{aligned} s^+_{m-n-1} &= \text{sign}\,(\sigma)\, s_{m-n-1} \\ \underline{z}^+_{m-n-1} &= |\sigma|^{-1/2} \left\{ \tau \, \underline{z}_{m-n-1} + Z_{t\,m-n-1} \, \text{chop}\,(\underline{e}_t - H\,\underline{w}) \right\} \end{aligned}\right\}, \tag{4.18}$$

where τ, σ and $\underline{e}_t - H\,\underline{w}$ are taken from the updating formula (4.11), where $Z_{t\,m-n-1}$ is the t-th component of $\underline{z}_{m-n-1}$, and where chop $(\underline{e}_t - H\,\underline{w})$ is the

vector in $\mathcal{R}^m$ whose components are the first m components of $\underline{e}_t - H\,\underline{w}$. These assertions and those of the next paragraph are justified in Powell (2003c).

In the alternative case $|\mathcal{K}| = m-n-3$, we simplify the notation by assuming that only $\underline{z}_1^+$ and $\underline{z}_2^+$ are not provided by equation (4.15), and that the signs $s_1 = +1$ and $s_2 = -1$ occur. Then the t-th components of $\underline{z}_1$ and $\underline{z}_2$, namely Z_{t1} and Z_{t2}, are nonzero. Many choices of the required new vectors $\underline{z}_1^+$ and $\underline{z}_2^+$ are possible, because of the freedom that corresponds to the orthogonal rotation (4.17). We make two of them available to NEWUOA, in order to avoid cancellation. Specifically, if the parameter β of expression (4.12) is nonnegative, we define $\zeta = \tau^2 + \beta\, Z_{t1}^2$, and NEWUOA applies the formulae

$$\left.\begin{aligned} s_1^+ &= s_1 = +1, \qquad s_2^+ = \mathrm{sign}\,(\sigma)\, s_2 = -\mathrm{sign}\,(\sigma), \\ \underline{z}_1^+ &= |\zeta|^{-1/2} \{\, \tau\, \underline{z}_1 + Z_{t1}\, \mathrm{chop}\,(\underline{e}_t - H\,\underline{w}) \,\}, \\ \underline{z}_2^+ &= |\zeta\,\sigma|^{-1/2} \{\, -\beta\, Z_{t1}\, Z_{t2}\, \underline{z}_1 + \zeta\, \underline{z}_2 + \tau\, Z_{t2}\, \mathrm{chop}\,(\underline{e}_t - H\,\underline{w}) \,\}. \end{aligned}\right\} \tag{4.19}$$

Otherwise, when $\beta < 0$, we define $\zeta = \tau^2 - \beta\, Z_{t2}^2$, and NEWUOA sets the values

$$\left.\begin{aligned} s_1^+ &= \mathrm{sign}\,(\sigma)\, s_1 = \mathrm{sign}\,(\sigma), \qquad s_2^+ = s_2 = -1, \\ \underline{z}_1^+ &= |\zeta\,\sigma|^{-1/2} \{\, \zeta\, \underline{z}_1 + \beta\, Z_{t1}\, Z_{t2}\, \underline{z}_2 + \tau\, Z_{t1}\, \mathrm{chop}\,(\underline{e}_t - H\,\underline{w}) \,\}, \\ \underline{z}_2^+ &= |\zeta|^{-1/2} \{\, \tau\, \underline{z}_2 + Z_{t2}\, \mathrm{chop}\,(\underline{e}_t - H\,\underline{w}) \,\}. \end{aligned}\right\} \tag{4.20}$$

The technique of the previous paragraph is employed only if at least one of the signs s_k, $k = 1, 2, \ldots, m-n-1$, is negative, and then $\sigma < 0$ must have occurred in equation (4.18) on an earlier iteration, because every s_k is set to $+1$ initially. Moreover, any failure in the conditions $\alpha \geq 0$ and $\beta \geq 0$ is due to computer rounding errors. Therefore Powell (2004a) suggests that the parameter σ in formula (4.11) be given the value

$$\sigma_{\mathrm{new}} \;=\; \max[0, \alpha]\, \max[0, \beta] + \tau^2, \tag{4.21}$$

instead of $\alpha\beta + \tau^2$. If the new value is different from before, however, then the new matrix (4.11) may not satisfy any of the conditions (4.8), except that the factorizations (4.16) and (4.7) ensure that the bottom right $(n+1) \times (n+1)$ submatrices of H^{-1} and $(H^+)^{-1}$ are zero. Another way of keeping σ positive is to retain $\alpha = \underline{e}_t^T H \underline{e}_t$, $\tau = \underline{e}_t^T H\, \underline{w}$ and $\sigma = \alpha\beta + \tau^2$ from expression (4.12), and to define β by the formula

$$\beta_{\mathrm{new}} \;=\; \max\left[\, 0,\; \tfrac{1}{2}\, \|\underline{x}^+ - \underline{x}_0\|^4 - \underline{w}^T H\, \underline{w} \,\right]. \tag{4.22}$$

In this case any change to β alters the element $(H^+)_{tt}^{-1}$, but every other stability property (4.13) is preserved, as proved in Lemma 2.3 of Powell (2004c). Therefore equation (4.21) was abandoned, and the usefulness of the value (4.22) instead of the definition (4.12) of β was investigated experimentally. Substantial differences in the numerical results were found only when the

damage from rounding errors was huge, and then the recovery that is provided by *all* of the conditions (4.13) is important to efficiency. Therefore the procedures that have been described already for updating Ξ, Υ and the factorization of Ω are preferred, although in practice α, β, σ and some of the signs s_k may become negative occasionally. Such errors are usually corrected automatically by a few more iterations of NEWUOA.

Another feature of the storage and updating of H by NEWUOA takes advantage of the remark that, when $\underline{d}$ is calculated in Box 2 of Figure 1, the constant term of Q is irrelevant. Moreover, the constant term of $Q_{\rm old}$ is not required in equation (4.5), because the identities $Q_{\rm old}(\underline{x}_{\rm opt}) = F(\underline{x}_{\rm opt})$ and $\underline{x}^+ = \underline{x}_{\rm opt} + \underline{d}$ allow this equation to be written in the form

$$\begin{pmatrix} \underline{\lambda}^+ \\ \hline c^+ \\ \underline{g}^+ \end{pmatrix} = \left\{ [\, F(\underline{x}_{\rm opt}+\underline{d}) - F(\underline{x}_{\rm opt}) \,] - [\, Q_{\rm old}(\underline{x}_{\rm opt}+\underline{d}) - Q_{\rm old}(\underline{x}_{\rm opt}) \,] \right\} H^+ \underline{e}_t. \tag{4.23}$$

Therefore NEWUOA does not store the constant term of any quadratic model. It follows that c^+ in expression (4.23) is ignored, which makes the $(m+1)$-th row of H^+ unnecessary for the revision of Q by formula (4.6). Equation (4.23) shows that the $(m+1)$-th column of H^+ is also unnecessary, t being in the interval $[1,m]$. Actually, the $(m+1)$-th row and column of every H matrix are suppressed by NEWUOA, which is equivalent to removing the first row of every submatrix Ξ and the first row and column of every submatrix Υ, but the other elements of these submatrices are retained. Usually this device gains some accuracy by diverting attention from actual values of F and $\underline{x} \in \mathcal{R}^n$ to the changes that occur in the objective function and the variables, as shown on the right hand side of equation (4.23) for example. The following procedure is used by NEWUOA to update H without its $(m+1)$-th row and column.

Let "opt" be the integer in $[1,m]$ such that $i = {\rm opt}$ gives the best of the interpolation points $\underline{x}_i$, $i = 1,2,\ldots,m$, which agrees with the notation in Sections 1 and 2, and let $\underline{v} \in \mathcal{R}^{m+n+1}$ have the components

$$\left.\begin{aligned} v_i &= \tfrac{1}{2} \{(\underline{x}_i - \underline{x}_0)^T (\underline{x}_{\rm opt} - \underline{x}_0)\}^2, && i = 1,2,\ldots,m \\ v_{m+1} &= 1 \quad \text{and} \quad v_{i+m+1} = (\underline{x}_{\rm opt} - \underline{x}_0)_i, && i = 1,2,\ldots,n \end{aligned}\right\}. \tag{4.24}$$

Therefore $\underline{v}$ is the opt-th column of the matrix W, so expression (3.12) implies $H\underline{v} = \underline{e}_{\rm opt}$ in theory, where $\underline{e}_{\rm opt}$ is the opt-th coordinate vector in $\mathcal{R}^{m+n+1}$. Thus the terms $H\underline{w}$ and $\underline{w}^T H \underline{w}$ of equations (4.11) and (4.12) take the values

$$H\underline{w} = H(\underline{w} - \underline{v}) + \underline{e}_{\rm opt} \tag{4.25}$$

and

$$\underline{w}^T H \underline{w} = (\underline{w} - \underline{v})^T H (\underline{w} - \underline{v}) + 2\, \underline{w}^T \underline{e}_{\rm opt} - \underline{v}^T \underline{e}_{\rm opt}. \tag{4.26}$$

These formulae allow the parameters (4.12) to be calculated without the $(m+1)$-th row and column of H, because the $(m+1)$-th component of $\underline{w} - \underline{v}$

is zero. Similarly, the first m and last n components of $H\underline{w}$ are given by formula (4.25), and these components of $H\underline{e}_t$ are known. Thus all the terms of expression (4.11) are available for generating the required parts of Ξ^+ and Υ^+. Moreover, after constructing chop $(\underline{e}_t - H\underline{w})$, the updating of the factorization of Ω is unchanged. It is proved in Lemma 3 of Powell (2004a) that, when this version of the updating procedure is applied, and when H has been damaged by rounding errors, then the new H^+ enjoys stability properties that are analogous to the conditions (4.13).

We see that the given procedures for updating H require only $\mathcal{O}(m^2)$ computer operations, which is highly favourable in the recommended case $m = 2n+1$. On the other hand, the function (4.4) has the second derivative matrix

$$\nabla^2 D = \sum_{j=1}^{m} \lambda_j^+ \, (\underline{x}_j^+ - \underline{x}_0) \, (\underline{x}_j^+ - \underline{x}_0)^T, \tag{4.27}$$

so the calculation of its elements would take $\mathcal{O}(mn^2)$ operations. Therefore $\nabla^2 Q_{\rm new}$ is not derived explicitly from formula (4.6). Instead, as suggested at the end of Section 3 of Powell (2004a), the NEWUOA software employs the forms

$$\left. \begin{aligned} \nabla^2 Q_{\rm old} &= \Gamma + \textstyle\sum_{j=1}^{m} \gamma_j \, (\underline{x}_j - \underline{x}_0) \, (\underline{x}_j - \underline{x}_0)^T \\ \nabla^2 Q_{\rm new} &= \Gamma^+ + \textstyle\sum_{j=1}^{m} \gamma_j^+ \, (\underline{x}_j^+ - \underline{x}_0) \, (\underline{x}_j^+ - \underline{x}_0)^T \end{aligned} \right\}, \tag{4.28}$$

overwriting the symmetric matrix Γ and the real coefficients γ_j, $j = 1, 2, \ldots, m$, by Γ^+ and γ_j^+, $j = 1, 2, \ldots, m$, respectively. At the beginning of the first iteration, each γ_j is set to zero, and we let Γ be the second derivative matrix of the initial quadratic model, its elements being specified in the first two paragraphs of Section 3. When the change (4.6) is made to the quadratic model, conditions (4.1), (4.27) and (4.28) allow the choices

$$\left. \begin{aligned} &\Gamma^+ = \Gamma + \gamma_t \, (\underline{x}_t - \underline{x}_0) \, (\underline{x}_t - \underline{x}_0)^T, \qquad \gamma_t^+ = \lambda_t^+ \\ &\text{and} \qquad \gamma_j^+ = \gamma_j + \lambda_j^+, \quad j \in \{1, 2, \ldots, m\} \backslash \{t\} \end{aligned} \right\}, \tag{4.29}$$

which are included in NEWUOA, because they can be implemented in only $\mathcal{O}(n^2)$ operations. Finally, the gradient of the quadratic model (3.1) is revised by the formula

$$\underline{\nabla} Q_{\rm new}(\underline{x}_0) = \underline{\nabla} Q_{\rm old}(\underline{x}_0) + \underline{g}^+, \tag{4.30}$$

in accordance with expressions (4.4) and (4.6), where $\underline{g}^+$ is taken from equation (4.23). The description of the updating of Q, without the unnecessary constant term $Q(\underline{x}_0)$, is complete, except that some of the numerical results of Section 8 suggested a recent modification that is described there.

5 The trust region subproblem

We recall from Box 2 of Figure 1 that subroutine TRSAPP generates a step $\underline{d}$ from $\underline{x}_{\rm opt}$ that is an approximate solution of the subproblem

$$\text{Minimize} \quad Q(\underline{x}_{\rm opt}+\underline{d}) \quad \text{subject to} \quad \|\underline{d}\| \leq \Delta. \tag{5.1}$$

The method of the subroutine is explained below. Figure 1 shows that the trust region radius Δ and the quadratic model Q are available when the subroutine is called, but, as mentioned at the end of Section 4, the matrix $\nabla^2 Q$ is stored in the form

$$\nabla^2 Q \;=\; \Gamma + \sum_{k=1}^{m} \gamma_k \, (\underline{x}_k - \underline{x}_0) \, (\underline{x}_k - \underline{x}_0)^T, \tag{5.2}$$

because it would be too onerous to work with all the elements of $\nabla^2 Q$ explicitly when n is large. Expression (5.2) implies the identity

$$\nabla^2 Q \, \underline{u} \;=\; \Gamma \underline{u} + \sum_{k=1}^{m} \eta_k \, (\underline{x}_k - \underline{x}_0), \tag{5.3}$$

where $\eta_k = \gamma_k \, (\underline{x}_k - \underline{x}_0)^T \underline{u}$, $k = 1, 2, \ldots, m$, and where $\underline{u}$ is a general vector in $\mathcal{R}^n$. Thus the product $\nabla^2 Q \, \underline{u}$ can be calculated in $\mathcal{O}(mn)$ operations for any choice of $\underline{u}$. Therefore it is suitable to generate $\underline{d}$ by a version of the truncated conjugate gradient method (see Conn, Gould and Toint, 2000, for instance).

This method produces a piecewise linear path in $\mathcal{R}^n$, starting at $\underline{x}_{\rm opt} = \underline{x}_{\rm opt} + \underline{d}_0$, where $\underline{d}_0 = 0$. For $j \geq 1$, we let $\underline{x}_{\rm opt} + \underline{d}_j$ be the point on the path at the end of the j-th line segment. It has the form

$$\underline{x}_{\rm opt} + \underline{d}_j \;=\; \underline{x}_{\rm opt} + \underline{d}_{j-1} + \alpha_j \, \underline{s}_j, \qquad j \geq 1, \tag{5.4}$$

where $\underline{s}_j$ is the direction of the line segment and α_j is now a steplength. We do not include any preconditioning, because the norm of the bound $\|\underline{d}\| \leq \Delta$ in expression (5.1) is Euclidean. Moreover, the path is truncated at $\underline{x}_{\rm opt} + \underline{d}_{j-1}$ if $\|\underline{\nabla} Q(\underline{x}_{\rm opt} + \underline{d}_{j-1})\|$ is sufficiently small, if $\|\underline{d}_{j-1}\| = \Delta$ holds, or if some other test is satisfied, as specified later. The complete path has the property that, if one moves along it from $\underline{x}_{\rm opt}$, then the Euclidean distance from $\underline{x}_{\rm opt}$ in $\mathcal{R}^n$ increases monotonically.

When the j-th line segment of the path is constructed, its direction is defined by the formula

$$\underline{s}_j \;=\; \begin{cases} -\underline{\nabla} Q(\underline{x}_{\rm opt}), & j = 1, \\ -\underline{\nabla} Q(\underline{x}_{\rm opt} + \underline{d}_{j-1}) + \beta_j \, \underline{s}_{j-1}, & j \geq 2, \end{cases} \tag{5.5}$$

where β_j is the ratio $\|\underline{\nabla} Q(\underline{x}_{\rm opt} + \underline{d}_{j-1})\|^2 / \|\underline{\nabla} Q(\underline{x}_{\rm opt} + \underline{d}_{j-2})\|^2$, this convenient value being taken from Fletcher and Reeves (1964). Then the steplength α_j

of equation (5.4) is chosen to minimize $Q(\underline{x}_{\rm opt}+\underline{d}_j)$ subject to $\alpha_j \geq 0$ and $\|\underline{d}_j\| \leq \Delta$ for each j. Formula (5.5) provides the well-known descent condition

$$\underline{s}_j^T \underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_{j-1}) \;=\; -\|\underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_{j-1})\|^2 \;<\; 0, \qquad j \geq 1, \tag{5.6}$$

which depends on the choice of α_{j-1} when $j \geq 2$. It follows from $\|\underline{d}_{j-1}\| < \Delta$ that α_j is positive.

The form (5.3) of the product $\nabla^2 Q\,\underline{u}$ assists the calculation of the gradients $\underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_j)$, $j \geq 0$, and the steplengths α_j, $j \geq 1$. The initial vector $\underline{u}$ is the difference $\underline{x}_{\rm opt}-\underline{x}_0$, in order to obtain from expression (3.1) the gradient

$$\underline{\nabla} Q(\underline{x}_{\rm opt}) \;=\; \underline{\nabla} Q(\underline{x}_0) + \nabla^2 Q\,\{\underline{x}_{\rm opt}-\underline{x}_0\}. \tag{5.7}$$

The other choices of $\underline{u}$ are just all the vectors (5.5) that occur. The availability of $\underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_{j-1})$ and $\nabla^2 Q\,\underline{s}_j$ allows α_j to be found cheaply, because it is the value of α in the interval $[0, \hat{\alpha}_j]$ that minimizes the function

$$Q(\underline{x}_{\rm opt}+\underline{d}_{j-1}+\alpha\,\underline{s}_j) = Q(\underline{x}_{\rm opt}+\underline{d}_{j-1}) + \alpha\,\underline{s}_j^T \underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_{j-1}) + \tfrac{1}{2}\alpha^2 \underline{s}_j^T \nabla^2 Q\,\underline{s}_j, \tag{5.8}$$

where $\hat{\alpha}_j$ is the positive root of the equation $\|\underline{x}_{\rm opt}+\underline{d}_{j-1}+\hat{\alpha}_j \underline{s}_j\| = \Delta$. Therefore we ask whether $Q(\underline{x}_{\rm opt}+\underline{d}_{j-1}+\alpha\underline{s}_j)$, $0 \leq \alpha \leq \hat{\alpha}_j$, decreases monotonically. Equations (5.6) and (5.8) imply that the answer is affirmative in the case

$$-\|\underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_{j-1})\|^2 + \hat{\alpha}_j\,\underline{s}_j^T \nabla^2 Q\,\underline{s}_j \;\leq\; 0, \tag{5.9}$$

and then $\alpha_j = \hat{\alpha}_j$ is selected. Otherwise, $\underline{s}_j^T \nabla^2 Q\,\underline{s}_j$ is positive, and the subroutine picks the value

$$\alpha_j \;=\; \|\underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_{j-1})\|^2 \,/\, \underline{s}_j^T \nabla^2 Q\,\underline{s}_j \;<\; \hat{\alpha}_j. \tag{5.10}$$

After finding α_j, the gradient $\underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_j)$ is constructed by the formula

$$\underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_j) \;=\; \underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_{j-1}) + \alpha_j\,\nabla^2 Q\,\underline{s}_j, \tag{5.11}$$

which is derived from the relation (5.4), the product $\nabla^2 Q\,\underline{s}_j$ being employed again. The techniques of this paragraph are applied for each line segment of the path.

The path is truncated at $\underline{x}_{\rm opt}+\underline{d}_j$ in the case $\alpha_j = \hat{\alpha}_j$, because then $\underline{d} = \underline{d}_j$ is on the boundary of the trust region $\|\underline{d}\| \leq \Delta$. Moreover, it is truncated at its starting point $\underline{x}_{\rm opt}+\underline{d}_0 = \underline{x}_{\rm opt}$ in the unusual case when the initial gradient $\underline{\nabla} Q(\underline{x}_{\rm opt})$ is identically zero. Otherwise, we try to truncate the path when the ratio

$$\Big[\, Q(\underline{x}_{\rm opt}) - Q(\underline{x}_{\rm opt}+\underline{d}_j) \,\Big] \,/\, \Big[\, Q(\underline{x}_{\rm opt}) - \min\,\{Q(\underline{x}_{\rm opt}+\underline{d}) : \|\underline{d}\| \leq \Delta\} \,\Big] \tag{5.12}$$

is sufficiently close to one, in order to avoid conjugate gradient iterations that improve only slightly the reduction in the objective function that is predicted

by the quadratic model. The implementation of this aim is empirical. Specifically, the iterations are terminated if at least one of the conditions

$$\left.\begin{aligned} \|\underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_j)\| &\le 10^{-2}\,\|\underline{\nabla} Q(\underline{x}_{\rm opt})\| \\ \left[\,Q(\underline{x}_{\rm opt}+\underline{d}_{j-1})-Q(\underline{x}_{\rm opt}+\underline{d}_j)\,\right] &\le 10^{-2}\left[\,Q(\underline{x}_{\rm opt})-Q(\underline{x}_{\rm opt}+\underline{d}_j)\,\right] \end{aligned}\right\} \quad (5.13)$$

is satisfied, the change in Q for each line segment being derived from expression (5.8), and $Q(\underline{x}_{\rm opt})-Q(\underline{x}_{\rm opt}+\underline{d}_j)$ is the sum of the changes so far. The path is also truncated if j reaches the theoretical upper bound on the number of iterations, namely n, but we expect this test to be redundant for $n \ge 10$.

Let $\underline{x}_{\rm opt}+\underline{d}_j$ be the final point of the path. The step $\underline{d}=\underline{d}_j$ is returned by subroutine TRSAPP in the case $\|\underline{d}_j\|<\Delta$, because then there is no interference with the conjugate gradient iterations from the trust region boundary. Further, the parameter CRVMIN, introduced in Box 2 of Figure 1, is given the value

$$\texttt{CRVMIN} \;=\; \min\left\{\,\underline{s}_i^T\nabla^2 Q\,\underline{s}_i \,/\, \|\underline{s}_i\|^2 : i=1,2,\ldots,j\,\right\}. \quad (5.14)$$

Otherwise, CRVMIN is set to zero, and, because of the possibility that the ratio (5.12) may be substantially less than one, the following iterative procedure is applied. It also calculates $\underline{d}_j$ from $\underline{d}_{j-1}$, the initial point $\underline{x}_{\rm opt}+\underline{d}_{j-1}$ being the final point of the truncated piecewise linear path, so $\underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_{j-1})$ is available. The conditions $\|\underline{d}_j\| = \|\underline{d}_{j-1}\| = \Delta$ are going to hold on every iteration of the additional procedure.

At the beginning of an iteration, we decide, using only $\underline{d}_{j-1}$ and $\underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_{j-1})$, whether $\underline{d}=\underline{d}_{j-1}$ is acceptable as an approximate solution of the subproblem (5.1). If $\underline{d}_{j-1}$ were the true solution, then, by the KKT conditions of the subproblem, $\underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_{j-1})$ would be a nonpositive multiple of $\underline{d}_{j-1}$, and we also give attention to the first of the conditions (5.13). Indeed, subroutine TRSAPP picks $\underline{d}=\underline{d}_{j-1}$ if one or both of the inequalities

$$\left.\begin{aligned} \|\underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_{j-1})\| &\le 10^{-2}\,\|\underline{\nabla} Q(\underline{x}_{\rm opt})\| \\ \underline{d}_{j-1}^T\,\underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_{j-1}) &\le -0.99\,\|\underline{d}_{j-1}\|\,\|\underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_{j-1})\| \end{aligned}\right\} \quad (5.15)$$

is achieved. Otherwise, $\underline{d}_{j-1}$ and $\underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_{j-1})$ span a two dimensional subspace of $\mathcal{R}^n$, and $\underline{d}_j$ is calculated to be the vector in this subspace that minimizes $Q(\underline{x}_{\rm opt}+\underline{d}_j)$ subject to $\|\underline{d}_j\|=\Delta$. Therefore $\underline{d}_j$ has the form

$$\underline{d}_j \;=\; \underline{d}(\theta) \;=\; \cos\theta\,\underline{d}_{j-1}+\sin\theta\,\underline{s}_j, \qquad \theta\in[0,2\pi], \quad (5.16)$$

where now the search direction $\underline{s}_j$ is chosen to be a vector in the two dimensional subspace that has the properties

$$\underline{s}_j^T\underline{d}_{j-1}=0 \qquad \text{and} \qquad \|\underline{s}_j\|=\Delta. \quad (5.17)$$

Equation (5.16) implies that $Q(\underline{x}_{\rm opt}+\underline{d}(\theta))$ is the expression

$$Q(\underline{x}_{\rm opt}) + \big(\cos\theta\,\underline{d}_{j-1} + \sin\theta\,\underline{s}_j\big)^T\,\underline{\nabla} Q(\underline{x}_{\rm opt}) + \big(\tfrac{1}{2}\cos^2\theta\,\underline{d}_{j-1} + \cos\theta\,\sin\theta\,\underline{s}_j\big)^T$$
$$\big\{\underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_{j-1}) - \underline{\nabla} Q(\underline{x}_{\rm opt})\big\} + \tfrac{1}{2}\sin^2\theta\,\underline{s}_j^T\nabla^2 Q\,\underline{s}_j, \quad 0\le\theta\le 2\pi, \qquad (5.18)$$

because the last term in braces is the product $\nabla^2 Q\,\underline{d}_{j-1}$. Again $\nabla^2 Q\,\underline{s}_j$ is constructed by formula (5.3), after which the minimization of the function (5.18) takes only $\mathcal{O}(n)$ operations. Thus $\underline{d}_j$ is determined, and the subroutine returns $\underline{d}=\underline{d}_j$ if the second of the conditions (5.13) holds, or if j is at least n. Alternatively, $\underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_j)$ is calculated for the next iteration, by applying the remark that equation (5.16) gives the gradient

$$\underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_j) \;=\; (1-\cos\theta)\,\underline{\nabla} Q(\underline{x}_{\rm opt}) + \cos\theta\,\underline{\nabla} Q(\underline{x}_{\rm opt}+\underline{d}_{j-1}) + \sin\theta\,\nabla^2 Q\,\underline{s}_j. \qquad (5.19)$$

Then j is increased by one, in order that the procedure of this paragraph can be applied recursively until termination occurs.

6 Subroutines BIGLAG and BIGDEN

We recall from Section 2 that, if Box 9 of Figure 1 is reached, then the condition (1.1) with index $i=$ MOVE is going to be replaced by the interpolation condition $Q(\underline{x}_{\rm opt}+\underline{d}) = F(\underline{x}_{\rm opt}+\underline{d})$, where $\underline{d}$ is calculated by the procedure of this section. In theory, given the index MOVE, the choice of $\underline{d}$ is derived from the positions $\underline{x}_i$, $i=1,2,\ldots,m$, of the current interpolation points, but in practice it depends also on the errors that occur in the matrices that are stored and updated, namely the submatrices Ξ and Υ of expression (3.12) and the factorization (4.16). We write t instead of MOVE, in order to retain the notation of Section 4. In particular, equation (4.1) shows the new positions of the interpolation points.

The t-th Lagrange function of the current interpolation points is important. It is the quadratic polynomial $\ell_t(\underline{x})$, $\underline{x}\in\mathcal{R}^n$, that satisfies the Lagrange conditions

$$\ell_t(\underline{x}_i) = \delta_{it}, \qquad i=1,2,\ldots,m, \qquad (6.1)$$

where the remaining freedom in the usual case $m<\frac{1}{2}(n+1)(n+2)$ is taken up by minimizing the Frobenius norm $\|\nabla^2\ell_t\|_F$. Therefore ℓ_t is the function

$$\ell_t(\underline{x}) \;=\; c + (\underline{x}-\underline{x}_0)^T\underline{g} + \tfrac{1}{2}\sum_{k=1}^{m}\lambda_k\,\{(\underline{x}-\underline{x}_0)^T(\underline{x}_k-\underline{x}_0)\}^2, \qquad \underline{x}\in\mathcal{R}^n, \quad (6.2)$$

the parameters c, $\underline{g}$ and λ_k, $k=1,2,\ldots,m$, being defined by the linear system of equations (3.10), where the right hand side is now the coordinate vector $\underline{e}_t\in\mathcal{R}^{m+n+1}$. Thus the parameters are the elements of the t-th column of the matrix H of expression (3.12). For each $\underline{x}\in\mathcal{R}^n$, we let $\underline{w}(\underline{x})$ be the vector in $\mathcal{R}^{m+n+1}$ that has the components

$$\left.\begin{aligned} w(\underline{x})_k &= \tfrac{1}{2}\{(\underline{x}-\underline{x}_0)^T(\underline{x}_k-\underline{x}_0)\}^2, && k=1,2,\ldots,m \\ w(\underline{x})_{m+1} &= 1 \quad \text{and} \quad w(\underline{x})_{i+m+1} = (\underline{x}-\underline{x}_0)_i, && i=1,2,\ldots,n \end{aligned}\right\}. \tag{6.3}$$

It follows that expression (6.2) can be written in the form

$$\ell_t(\underline{x}) = \sum_{k=1}^{m} \lambda_k w(\underline{x})_k + c\, w(\underline{x})_{m+1} + \sum_{i=1}^{n} g_i w(\underline{x})_{i+m+1} = (H\underline{e}_t)^T \underline{w}(\underline{x}). \tag{6.4}$$

Therefore, when the symmetric matrix H is updated by formula (4.11), because of the change (4.1) to the interpolation points, expression (4.12) includes the value

$$\tau = \underline{e}_t^T H \underline{w} = \underline{e}_t^T H \underline{w}(\underline{x}^+) = (H\underline{e}_t)^T \underline{w}(\underline{x}_{\rm opt}+\underline{d}) = \ell_t(\underline{x}_{\rm opt}+\underline{d}). \tag{6.5}$$

Thus the Lagrange function (6.2) gives the dependence of τ on the choice of $\underline{d}$.

As mentioned in Section 4, we expect a relatively large modulus of the denominator $\sigma = \alpha\beta+\tau^2$ to be beneficial when formula (4.11) is applied. Usually $\sigma > \tau^2$ holds in practice, because in theory both α and β are positive. Thus we deduce from the previous paragraph that it may be advantageous to let $\underline{d}$ be an approximate solution of the subproblem

$$\text{Maximize} \quad |\ell_t(\underline{x}_{\rm opt}+\underline{d})| \quad \text{subject to} \quad \|\underline{d}\| \le \overline{\Delta}, \tag{6.6}$$

where $\overline{\Delta} > 0$ is prescribed. This calculation is performed by subroutine BIGLAG, details being given in the next paragraph. There is an excellent reason for a large value of $|\ell_t(\underline{x}_{\rm opt}+\underline{d})|$ in the case $m = \frac{1}{2}(n+1)(n+2)$. Specifically, one picks a convenient basis of the space of quadratic polynomials, in order that the construction of Q from the interpolation conditions (1.1) reduces to the solution of an $m \times m$ system of linear equations. Let B and B^+ be the old and new matrices of the system when the change (4.1) is made to the interpolation points. Then, as shown in Powell (2001), the dependence of the ratio $\det B^+ / \det B$ on $\underline{d} \in \mathcal{R}^n$ is just a quadratic polynomial, which is exactly $\ell_t(\underline{x}_{\rm opt}+\underline{d})$, $\underline{d} \in \mathcal{R}^n$, because of the Lagrange conditions (6.1). In this case, therefore, the subproblem (6.6) is highly suitable for promoting the nonsingularity of B^+.

The method of BIGLAG is iterative, and is like the procedure of the last paragraph of Section 5. As in equation (5.16), the j-th iteration generates the vector

$$\underline{d}_j = \underline{d}(\theta) = \cos\theta\, \underline{d}_{j-1} + \sin\theta\, \underline{s}_j, \tag{6.7}$$

where $\underline{d}_{j-1}$ is the best estimate of the required $\underline{d}$ at the beginning of the current iteration, where $\underline{d}_{j-1}$ and $\underline{s}_j$ have the properties

$$\|\underline{d}_{j-1}\| = \|\underline{s}_j\| = \overline{\Delta} \qquad \text{and} \qquad \underline{s}_j^T \underline{d}_{j-1} = 0, \tag{6.8}$$

and where the angle θ of equation (6.7) is calculated to maximize $|\ell_t(\underline{x}_{\rm opt}+\underline{d}_j)|$. The choice

$$\underline{d}_0 \;=\; \pm\,\overline{\Delta}\,(\underline{x}_t-\underline{x}_{\rm opt})\,/\,\|\underline{x}_t-\underline{x}_{\rm opt}\| \tag{6.9}$$

is made for the first iteration, with the sign that provides the larger value of $|\ell_t(\underline{x}_{\rm opt}+\underline{d}_0)|$, which implies $\underline{\nabla}\ell_t(\underline{x}_{\rm opt}+\underline{d}_0)\neq 0$, because ℓ_t is a quadratic polynomial that satisfies the Lagrange conditions $\ell_t(\underline{x}_{\rm opt})=0$ and $\ell_t(\underline{x}_t)=1$. The vector $\underline{s}_1$ of the first iteration is taken from the two dimensional subspace that is spanned by $\underline{d}_0$ and $\underline{\nabla}\ell_t(\underline{x}_{\rm opt})$, provided that both the inequalities

$$\left.\begin{aligned} |\underline{d}_0^T\underline{\nabla}\ell_t(\underline{x}_{\rm opt})|^2 \;&\leq\; 0.99\,\overline{\Delta}^{\,2}\,\|\underline{\nabla}\ell_t(\underline{x}_{\rm opt})\|^2 \\ \text{and}\quad \|\underline{\nabla}\ell_t(\underline{x}_{\rm opt})\| \;&\geq\; 0.1\,|\ell_t(\underline{x}_{\rm opt}+\underline{d}_0)|\,/\,\overline{\Delta} \end{aligned}\right\} \tag{6.10}$$

hold, because this use of $\underline{\nabla}\ell_t(\underline{x}_{\rm opt})$ is unattractive if the subspace is nearly degenerate, or if the bound $\overline{\Delta}\,\|\underline{\nabla}\ell_t(\underline{x}_{\rm opt})\|$ on the first order term of the identity

$$\ell_t(\underline{x}_{\rm opt}+\underline{d}) \;=\; \underline{d}^T\underline{\nabla}\ell_t(\underline{x}_{\rm opt})+\tfrac{1}{2}\,\underline{d}^T\,\underline{\nabla}^2\ell_t\,\underline{d}, \qquad \|\underline{d}\|\leq\overline{\Delta}, \tag{6.11}$$

compares unfavourably with $|\ell_t(\underline{x}_{\rm opt}+\underline{d}_0)|$. Alternatively, if at least one of the conditions (6.10) fails, then $\underline{s}_1$ is defined by the technique that gives $\underline{s}_j$ for $j\geq 2$. Specifically, $\underline{s}_j$ is a linear combination of $\underline{d}_{j-1}$ and $\underline{\nabla}\ell_t(\underline{x}_{\rm opt}+\underline{d}_{j-1})$ that has the properties (6.8), except that the subroutine returns the vector $\underline{d}=\underline{d}_{j-1}$ in the unlikely situation

$$|\,\underline{d}_{j-1}^T\underline{\nabla}\ell_t(\underline{x}_{\rm opt}+\underline{d}_{j-1})\,|^2 \;\geq\; (1-10^{-8})\,\overline{\Delta}^{\,2}\,\|\underline{\nabla}\ell_t(\underline{x}_{\rm opt}+\underline{d}_{j-1})\|^2. \tag{6.12}$$

The usual test for termination is the condition

$$|\ell_t(\underline{x}_{\rm opt}+\underline{d}_j)| \;\leq\; 1.1\,|\ell_t(\underline{x}_{\rm opt}+\underline{d}_{j-1})|, \tag{6.13}$$

because the iteration has not improved very much the objective function of the subproblem (6.6). Then $\underline{d}=\underline{d}_j$ is returned, which happens too if j reaches the value n. Otherwise, as in equation (5.19), the gradient

$$\underline{\nabla}\ell_t(\underline{x}_{\rm opt}+\underline{d}_j) \;=\; (1-\cos\theta)\,\underline{\nabla}\ell_t(\underline{x}_{\rm opt})+\cos\theta\,\underline{\nabla}\ell_t(\underline{x}_{\rm opt}+\underline{d}_{j-1})+\sin\theta\,\nabla^2\ell_t\,\underline{s}_j \tag{6.14}$$

is calculated, and then j is increased by one for the next iteration. Because the second derivative matrix of the function (6.2) is not constructed explicitly, the remarks on $\nabla^2 Q$ in Section 5 apply also to $\nabla^2\ell_t$, including the use of the formula

$$\nabla^2\ell_t\,\underline{u} \;=\; \Big\{\sum_{k=1}^{m}\lambda_k\,(\underline{x}_k-\underline{x}_0)\,(\underline{x}_k-\underline{x}_0)^T\Big\}\,\underline{u} \;=\; \sum_{k=1}^{m}\eta_k\,(\underline{x}_k-\underline{x}_0), \tag{6.15}$$

where $\eta_k=\lambda_k(\underline{x}_k-\underline{x}_0)^T\underline{u}$, $k=1,2,\ldots,m$. Now the vectors $\underline{u}$ that occur are just $\underline{x}_{\rm opt}-\underline{x}_0$, $\underline{d}_0$ and each $\underline{s}_j$, so the amount of work of BIGLAG is similar to that of subroutine TRSAPP.

The parameter $\overline{\Delta}$ of the subproblem (6.6) is set automatically to a value that depends on three considerations. Firstly, because of the purpose of ρ, as described in the second paragraph of Section 2, the bound $\overline{\Delta} \geq \rho$ is imposed. Secondly, the Y-branch has been taken from Box 8 of Figure 1 because $\texttt{DIST} = \|\underline{x}_t - \underline{x}_{\text{opt}}\|$ is unacceptably large, so the condition $\overline{\Delta} \leq 0.1\,\texttt{DIST}$ is reasonable. Thirdly, $\overline{\Delta}$ should be no greater than the current Δ of the trust region subproblem of Section 5, and we anticipate that Δ may be halved. These remarks provide the choice

$$\overline{\Delta} = \max\left[\min\{0.1\,\texttt{DIST},\ 0.5\,\Delta\},\ \rho\right], \tag{6.16}$$

which seems to be suitable in practice, even if the given ρ_{beg} causes ρ to be much less than the required changes to the variables.

After the construction of $\underline{d}$ by subroutine BIGLAG, the parameters (4.12) are calculated, $\underline{x}^+$ being the vector $\underline{x}_{\text{opt}} + \underline{d}$. It has been mentioned already that in theory α and β are positive, but that negative values of $\sigma = \alpha\beta + \tau^2$ may occur occasionally, due to computer rounding errors. We recall also that formula (4.11) is applied even if σ is negative, but the updating would be unhelpful if σ were too close to zero. Therefore the $\underline{d}$ from BIGLAG is rejected if and only if the current parameters have the property

$$|\,\sigma\,| = |\,\alpha\beta + \tau^2\,| \leq 0.8\,\tau^2. \tag{6.17}$$

The alternative choice of $\underline{d}$ is made by calling subroutine BIGDEN, which seeks a big value of the denominator $|\sigma|$ instead of a big value of $|\tau|$. The dependence of σ on $\underline{x} = \underline{x}_{\text{opt}} + \underline{d}$ is obtained by substituting $\underline{x}^+ = \underline{x}$ and $\underline{w} = \underline{w}(\underline{x})$ into expression (4.12), using the definition (6.3). Then BIGDEN sets $\underline{d}$ to an approximation to the solution of the subproblem

$$\text{Maximize} \quad |\sigma(\underline{x}_{\text{opt}} + \underline{d})| \quad \text{subject to} \quad \|\underline{d}\| \leq \overline{\Delta}, \tag{6.18}$$

where $\overline{\Delta}$ still has the value (6.16). This task is much more laborious than the calculation of BIGLAG, because $\sigma(\underline{x})$, $\underline{x} \in \mathcal{R}^n$, is a quartic polynomial. Fortunately, numerical experiments show that the situation (6.17) is very rare in practice.

The methods of subroutines BIGLAG and BIGDEN are similar, except for obvious changes due to the differences between their objective functions. Indeed, BIGDEN also picks initial vectors $\underline{d}_0$ and $\underline{s}_1$ that satisfy the equations (6.8), in order to begin an iterative procedure. Again the j-th iteration lets $\underline{d}_j$ have the form (6.7), but now θ is calculated to maximize $|\sigma(\underline{x}_{\text{opt}} + \underline{d}_j)|$. When $j \geq 2$, the vector $\underline{d} = \underline{d}_j$ is returned by BIGDEN if it has the property

$$|\sigma(\underline{x}_{\text{opt}} + \underline{d}_j)| \leq 1.1\,|\sigma(\underline{x}_{\text{opt}} + \underline{d}_{j-1})|, \tag{6.19}$$

or if j has reached the value n, the test (6.19) being analogous to condition (6.13). Otherwise, after increasing j by one, the gradient $\underline{\nabla}\sigma(\underline{x}_{\text{opt}} + \underline{d}_{j-1})$ is

constructed, using some numbers that are known already, as described at the end of this section. If the inequality

$$|\,\underline{d}_{j-1}^T\,\underline{\nabla}\sigma(\underline{x}_{\rm opt}+\underline{d}_{j-1})\,|^2 \;<\; (1-10^{-8})\,\overline{\Delta}^{\,2}\,\|\underline{\nabla}\sigma(\underline{x}_{\rm opt}+\underline{d}_{j-1})\|^2 \qquad (6.20)$$

holds, then $\mathcal{S}_j=\mathrm{span}\{\underline{d}_{j-1},\underline{\nabla}\sigma(\underline{x}_{\rm opt}+\underline{d}_{j-1})\}$ is a well-defined two dimensional subspace of $\mathcal{R}^n$. Then another iteration is performed, $\underline{s}_j$ being set to a vector in $\mathcal{S}_j$ with the properties (6.8). If the test (6.20) fails, however, the first order conditions for the solution of the subproblem (6.18) are nearly achieved, so BIGDEN returns the vector $\underline{d}=\underline{d}_{j-1}$.

The choice of $\underline{d}_0$ in BIGDEN is the $\underline{d}$ that has just been picked by BIGLAG, because we expect $|\sigma(\underline{x}_{\rm opt}+\underline{d})|$ to be large when $|\ell_t(\underline{x}_{\rm opt}+\underline{d})|$ is large, although rounding errors have caused the unwelcome situation (6.17). The direction $\underline{s}_1$ is taken from the space $\mathcal{S}_1=\mathrm{span}\{\underline{d}_0,\underline{u}\}$, where $\underline{u}$ is the step $\underline{x}_k-\underline{x}_{\rm opt}$ from $\underline{x}_{\rm opt}$ to one of the other interpolation points. The value of k depends on the ratios

$$\omega_i \;=\; \frac{|\,(\underline{x}_i-\underline{x}_{\rm opt})^T\underline{d}_0\,|^2}{\|\underline{x}_i-\underline{x}_{\rm opt}\|^2\,\|\underline{d}_0\|^2}, \qquad i\in\{1,2,\ldots,m\}\backslash\{\rm opt\}. \qquad (6.21)$$

Priority is given to $k=t$, this selection being made in the case $\omega_t\le 0.99$. Otherwise, k is such that ω_k is the least of the ratios (6.21). A criticism of this procedure is that it ignores the objective function σ, which is why the test (6.19) for termination is not tried on the first iteration. The possibility $\underline{u}=\underline{\nabla}\sigma(\underline{x}_{\rm opt})$ is unattractive, because $\underline{\nabla}\sigma(\underline{x}_{\rm opt})$ is zero in exact arithmetic, and it would be inconvenient to pick $\underline{u}=\underline{\nabla}\sigma(\underline{x}_{\rm opt}+\underline{d}_0)$, because the numbers that assist the construction of this gradient, mentioned in the previous paragraph, are not yet available.

Let $\hat{\sigma}(\theta)$, $\theta\in\mathcal{R}$, be the value of $\sigma(\underline{x}_{\rm opt}+\underline{d})$, when $\underline{d}=\underline{d}(\theta)$ is the vector (6.7). The main task of an iteration of BIGDEN is to assemble the coefficients $\check{\sigma}_\ell$, $\ell=1,2,\ldots,9$, such that $\hat{\sigma}$ is the function

$$\hat{\sigma}(\theta) \;=\; \check{\sigma}_1+\sum_{k=1}^{4}\left\{\check{\sigma}_{2k}\cos(k\theta)+\check{\sigma}_{2k+1}\sin(k\theta)\right\}, \qquad \theta\in\mathcal{R}. \qquad (6.22)$$

Because the right hand side of equation (4.25) is used in the calculation of σ, matrices U and V of dimension $(m+n)\times 5$ are constructed, that provide the dependence of the relevant parts of $\underline{w}-\underline{v}$ and $H(\underline{w}-\underline{v})$, respectively, on θ. We define $\underline{w}$ by putting the vector

$$\underline{x} \;=\; \underline{x}_{\rm opt}+\underline{d}(\theta) \;=\; \underline{x}_{\rm opt}+\cos\theta\,\underline{d}_{j-1}+\sin\theta\,\underline{s}_j, \qquad \theta\in\mathcal{R}, \qquad (6.23)$$

into expression (6.3), but the definition (4.24) of $\underline{v}$ is independent of θ. Thus we find the components

$$\begin{aligned}(\underline{w}-\underline{v})_i&=\tfrac{1}{2}\left\{(\underline{x}-\underline{x}_0)^T(\underline{x}_i-\underline{x}_0)\right\}^2-\tfrac{1}{2}\left\{(\underline{x}_{\rm opt}-\underline{x}_0)^T(\underline{x}_i-\underline{x}_0)\right\}^2\\&=\tfrac{1}{2}\left\{(\underline{x}-\underline{x}_{\rm opt})^T(\underline{x}_i-\underline{x}_0)\right\}\left\{(\underline{x}+\underline{x}_{\rm opt}-2\,\underline{x}_0)^T(\underline{x}_i-\underline{x}_0)\right\}\\&=\left\{\hat{v}_i\cos\theta+\hat{w}_i\sin\theta\right\}\left\{\hat{u}_i+\tfrac{1}{2}\hat{v}_i\cos\theta+\tfrac{1}{2}\hat{w}_i\sin\theta\right\}, \quad 1\le i\le m, \quad (6.24)\end{aligned}$$

and

$$(\underline{w}-\underline{v})_{i+m+1} \;=\; \cos\theta\,(\underline{d}_{j-1})_i + \sin\theta\,(\underline{s}_j)_i, \qquad i=1,2,\ldots,n, \tag{6.25}$$

where $\hat{u}_i$, $\hat{v}_i$ and $\hat{w}_i$ are the scalar products $(\underline{x}_{\rm opt}-\underline{x}_0)^T(\underline{x}_i-\underline{x}_0)$, $\underline{d}_{j-1}^T(\underline{x}_i-\underline{x}_0)$ and $\underline{s}_j^T(\underline{x}_i-\underline{x}_0)$, respectively. We construct the rows of U by regarding these components of $\underline{w}-\underline{v}$ as functions of θ, writing them in sequence in the form

$$U_{i1}+U_{i2}\cos\theta+U_{i3}\sin\theta+U_{i4}\cos(2\theta)+U_{i5}\sin(2\theta), \quad i=1,2,\ldots,m+n. \tag{6.26}$$

Then we define V by the property that the terms

$$V_{i1}+V_{i2}\cos\theta+V_{i3}\sin\theta+V_{i4}\cos(2\theta)+V_{i5}\sin(2\theta), \quad i=1,2,\ldots,m+n, \tag{6.27}$$

are the first m and last n components of $H(\underline{w}-\underline{v})$. In other words, because $(\underline{w}-\underline{v})_{m+1}$ is zero, V is the product $H_{\rm red}U$, where $H_{\rm red}$ is the matrix H without its $(m+1)$-th row and column, which receives attention in the paragraph that includes equation (4.23). The product of the displays (6.26) and (6.27) is expressed as a constant plus a linear combination of $\cos(k\theta)$ and $\sin(k\theta)$, $k=1,2,3,4$, and the results are summed over i. Thus we find the coefficients $\check{\beta}_\ell$, $\ell=1,2,\ldots,9$, of the function

$$(\underline{w}-\underline{v})^T H\,(\underline{w}-\underline{v}) \;=\; \check{\beta}_1 + \sum_{k=1}^{4}\{\check{\beta}_{2k}\cos(k\theta)+\check{\beta}_{2k+1}\sin(k\theta)\}, \quad \theta\in\mathcal{R}. \tag{6.28}$$

The contribution from these coefficients to expression (6.22) is explained below.

The definitions (6.3) and (4.24) provide $\underline{w}^T\underline{e}_{\rm opt}=\frac{1}{2}\{(\underline{x}_{\rm opt}-\underline{x}_0)^T(\underline{x}-\underline{x}_0)\}^2$ and $\underline{v}^T\underline{e}_{\rm opt}=\frac{1}{2}\|\underline{x}_{\rm opt}-\underline{x}_0\|^4$ in formula (4.26). Hence equations (4.12), (4.26) and (4.25), with $t\neq$opt, allow $\hat{\sigma}$ to be written in the form

$$\begin{aligned}\hat{\sigma}(\theta)=\alpha\,\big[\,\tfrac{1}{2}\,\|\underline{x}-\underline{x}_0\|^4 - \{(\underline{x}_{\rm opt}-\underline{x}_0)^T(\underline{x}-\underline{x}_0)\}^2 + \tfrac{1}{2}\,\|\underline{x}_{\rm opt}-\underline{x}_0\|^4\,\big] \\ -\;\alpha\,(\underline{w}-\underline{v})^T H\,(\underline{w}-\underline{v}) + \big[\,\underline{e}_t^T H\,(\underline{w}-\underline{v})\,\big]^2.\end{aligned} \tag{6.29}$$

Therefore, because $\alpha=\underline{e}_t^T H\,\underline{e}_t$ is independent of $\underline{x}=\underline{x}_{\rm opt}+\underline{d}(\theta)$, subroutine BIGDEN sets the required coefficients of the function (6.22) to $\check{\sigma}_\ell=-\alpha\check{\beta}_\ell$, $\ell=1,2,\ldots,9$, initially, and then it makes the adjustments that provide the square bracket terms of equation (6.29).

The adjustment for the last term of this equation begins with the remark that $\underline{e}_t^T H(\underline{w}-\underline{v})$ is the function (6.27) of θ in the case $i=t$. Therefore BIGDEN expresses the square of this function as a constant plus a linear combination of $\cos(k\theta)$ and $\sin(k\theta)$, $k=1,2,3,4$, and it adds the resultant coefficients to the corresponding values of $\check{\sigma}_\ell$, $\ell=1,2,\ldots,9$. Moreover, one can deduce from the conditions (6.23) and (6.8) that the first square brackets of equation (6.29) contain the function

$$\left(\overline{\Delta}^2+\hat{v}_{\rm opt}\cos\theta+\hat{w}_{\rm opt}\sin\theta\right)^2+\overline{\Delta}^2\left(\hat{u}_{\rm opt}-\tfrac{1}{2}\overline{\Delta}^2\right),\qquad\theta\in\mathcal{R},\qquad(6.30)$$

where $\hat{u}_{\rm opt}$, $\hat{v}_{\rm opt}$ and $\hat{w}_{\rm opt}$ are taken from expression (6.24). It follows that the final adjustment of the $\check{\sigma}_\ell$ coefficients is elementary. Next, BIGDEN computes the values $\hat{\sigma}(2\pi i/50)$, $i=0,1,\ldots,49$, directly from equation (6.22), identifying the integer $\hat{i}\in[0,49]$ that maximizes $|\hat{\sigma}(2\pi\hat{i}/50)|$. Then the quadratic polynomial $\hat{q}(\theta)$, $\theta\in\mathcal{R}$, is constructed by interpolation to $\hat{\sigma}$ at the points $\theta=2\pi i/50$, $i=\hat{i}-1,\hat{i},\hat{i}+1$. The choice of θ for the definition (6.7) of $\underline{d}_j$ is completed by giving it the value that maximizes $|\hat{q}(\theta)|$ within the range of its interpolation points.

After calculating $\underline{d}_j$, and then increasing j by one if the test (6.19) fails, the gradient $\underline{\nabla}\sigma(\underline{x}_{\rm opt}+\underline{d}_{j-1})$ is required, as mentioned already. We are going to derive it from expression (6.29), the right hand side being the function $\sigma(\underline{x})$, $\underline{x}\in\mathcal{R}^n$, where $\underline{w}$ depends on $\underline{x}$ through equation (6.3). We consider the equivalent task of finding $\underline{\nabla}\sigma(\underline{x}_{\rm opt}+\underline{d}_j)$ for the old value of j, in order to retain the notation of the previous three paragraphs.

The gradient of the first line of the function (6.29) at $\underline{x}=\underline{x}_{\rm opt}+\underline{d}_j$ is the vector

$$\begin{aligned}2\alpha\,\big[\,\|\underline{x}-\underline{x}_0\|^2(\underline{x}-\underline{x}_0)-\{(\underline{x}_{\rm opt}-\underline{x}_0)^T(\underline{x}-\underline{x}_0\}\,(\underline{x}_{\rm opt}-\underline{x}_0)\,\big]\\ =2\alpha\,\Big[\,\|\underline{x}-\underline{x}_0\|^2\,\underline{d}_j+\Big\{\underline{d}_j^T(\underline{x}-\underline{x}_0)\Big\}\,(\underline{x}_{\rm opt}-\underline{x}_0)\,\Big],\end{aligned}\qquad(6.31)$$

the right hand side being given by the relation $(\underline{x}-\underline{x}_0)=\underline{d}_j+(\underline{x}_{\rm opt}-\underline{x}_0)$. It is employed by BIGDEN, in order to avoid some cancellation when $\|\underline{d}_j\|$ is relatively small. The remainder of the gradient of the function (6.29) is the sum

$$-2\alpha\sum_{i=1}^{m+n+1}\{H(\underline{w}-\underline{v})\}_i\,\underline{\nabla}\{w(\underline{x})_i\}+2\{\underline{e}_t^TH(\underline{w}-\underline{v})\}\sum_{i=1}^{m+n+1}H_{ti}\,\underline{\nabla}\{w(\underline{x})_i\}.\qquad(6.32)$$

An advantage of the work so far is that the terms (6.27) for the chosen θ are the first m and last n components of $H(\underline{w}-\underline{v})$. Thus expression (6.27) provides the numbers $\hat{\eta}_i=\{H(\underline{w}-\underline{v})\}_i$, $i=1,2,\ldots,m$, and $\check{\eta}_i=\{H(\underline{w}-\underline{v})\}_{i+m+1}$, $i=1,2,\ldots,n$. We recall from equations (4.12) and (4.25), with $t\neq{\rm opt}$, that $\underline{e}_t^TH(\underline{w}-\underline{v})$ is the current value of τ. Therefore, because the definition (6.3) shows that $w(\underline{x})_{m+1}$ is constant, the sum (6.32) can be written in the form

$$2\sum_{i=1}^{m}(\tau H_{ti}-\alpha\,\hat{\eta}_i)\,\underline{\nabla}\{w(\underline{x})_i\}+2\sum_{i=1}^{n}(\tau H_{t\,i+m+1}-\alpha\,\check{\eta}_i)\,\underline{\nabla}\{w(\underline{x})_{i+m+1}\}.\qquad(6.33)$$

Equation (6.3) gives $\underline{\nabla}\{w(\underline{x})_i\}=\{(\underline{x}-\underline{x}_0)^T(\underline{x}_i-\underline{x}_0)\}(\underline{x}_i-\underline{x}_0)$, $i=1,2,\ldots,m$, and $\underline{\nabla}\{w(\underline{x})_{i+m+1}\}=\underline{e}_i$, $i=1,2,\ldots,n$. It follows that the required gradient of $\sigma(\underline{x})$ is the sum of three vectors, namely expression (6.31), the sum

$$2\sum_{i=1}^{m}\left\{(\tau H_{ti}-\alpha\hat{\eta}_i)\,(\underline{x}-\underline{x}_0)^T(\underline{x}_i-\underline{x}_0)\right\}\,(\underline{x}_i-\underline{x}_0), \tag{6.34}$$

and the vector in $\mathcal{R}^n$ with the components $2\,(\tau H_{t\,i+m+1}-\alpha\check{\eta}_i)$, $i=1,2,\ldots,n$. The description of the method of BIGDEN is complete.

7 Other details of NEWUOA

We see in Figure 1 of Section 2 that Δ is revised and MOVE is set in Box 4, that ρ is reduced in Box 12, and that a test is made in Box 14. We recall also from the end of Section 1 that shifts of origin are important to the accuracy of the H matrix. The details of these operations are addressed in this section.

Let $\Delta_{\rm old}$ and $\Delta_{\rm new}$ be the old and new values of Δ that occur in Box 4. As mentioned already, the choice of $\Delta_{\rm new}$ depends on the ratio (2.2), and also the Euclidean length of the step $\underline{d}$ receives attention. Possible values of $\Delta_{\rm new}$ are $\frac{1}{2}\|\underline{d}\|$, $\|\underline{d}\|$ and $2\|\underline{d}\|$ in the cases RATIO ≤ 0.1, $0.1<$ RATIO ≤ 0.7 and RATIO > 0.7, respectively, but we take the view that, if RATIO > 0.1, then a large reduction in Δ may be too restrictive on the next iteration. Moreover, we observe the bound $\Delta\ge\rho$, and we prefer to sharpen the test in Box 10 by avoiding trust region radii that are close to ρ. Therefore NEWUOA sets $\Delta_{\rm new}$ to ρ or to $\Delta_{\rm int}$ in the cases $\Delta_{\rm int}\le 1.5\rho$ or $\Delta_{\rm int}>1.5\rho$, respectively, where $\Delta_{\rm int}$ is the intermediate value

$$\Delta_{\rm int}\;=\;\begin{cases}\frac{1}{2}\,\|\underline{d}\|, & \text{RATIO}\le 0.1,\\ \max\{\,\|\underline{d}\|,\,\frac{1}{2}\,\Delta_{\rm old}\}, & 0.1<\text{RATIO}\le 0.7,\\ \max\{\,2\,\|\underline{d}\|,\,\frac{1}{2}\,\Delta_{\rm old}\}, & \text{RATIO}>0.7.\end{cases} \tag{7.1}$$

The selection of MOVE in Box 4 provides a relatively large denominator for the updating formula (4.11), as stated after expression (4.12). We recall that $H\underline{w}$ and β in this expression are independent of t. Let $\mathcal{T}$ be the set $\{1,2,\ldots,m\}$, except that the integer "opt" is excluded from $\mathcal{T}$ in the case $F(\underline{x}_{\rm opt}+\underline{d})\ge F(\underline{x}_{\rm opt})$, in order to prevent the removal of $\underline{x}_{\rm opt}$ from the set of interpolation points. The numbers

$$\sigma_t\;=\;(\underline{e}_t^T H\,\underline{e}_t)\,\beta+(\underline{e}_t^T H\,\underline{w})^2,\qquad t\in\mathcal{T}, \tag{7.2}$$

are calculated, σ_t being the denominator that would result from choosing MOVE$=t$. There is a strong disadvantage in making $|\sigma_{\rm MOVE}|$ as large as possible, however, as we prefer to retain interpolation points that are close to $\underline{x}_{\rm opt}$. The disadvantage occurs, for instance, when at least $n+1$ of the points $\underline{x}_i$, $i=1,2,\ldots,m$, are within distance Δ of $\underline{x}_{\rm opt}$, but $\underline{x}_t$ is much further away. Then the Lagrange conditions (6.1) suggest that ℓ_t may be not unlike the function $\|\underline{x}-\underline{x}_{\rm opt}\|^2/\|\underline{x}_t-\underline{x}_{\rm opt}\|^2$, $\underline{x}\in\mathcal{R}^n$, which, because of the bound $\|\underline{d}\|\le\Delta$, would imply the property

$$|\ell_t(\underline{x}_{\rm opt}+\underline{d})| \;=\; \mathcal{O}\left(\Delta^2/\,\|\underline{x}_t-\underline{x}_{\rm opt}\|^2\right). \tag{7.3}$$

Now the equations (6.5) include $\underline{e}_t^T H\underline{w} = \ell_t(\underline{x}_{\rm opt}+\underline{d})$, and it is usual for $(\underline{e}_t^T H\underline{e}_t)\beta$ and $(\underline{e}_t^T H\underline{w})^2$ to be positive numbers of similar magnitudes in expression (7.2). Thus, for general $t\in\mathcal{T}$, it may happen that $|\sigma_t|$ is $\mathcal{O}(1)$ or $\mathcal{O}(\Delta^4/\|\underline{x}_t-\underline{x}_{\rm opt}\|^4)$ in the case $\|\underline{x}_t-\underline{x}_{\rm opt}\|\le\Delta$ or $\|\underline{x}_t-\underline{x}_{\rm opt}\|>\Delta$, respectively. Therefore NEWUOA sets `MOVE` either to zero or to the integer $t^*\in\mathcal{T}$ that satisfies the equation

$$w_{t^*}|\sigma_{t^*}| \;=\; \max\{w_t\,|\sigma_t| : t\in\mathcal{T}\}, \tag{7.4}$$

where w_t is a weighting factor that is necessary for the automatic removal of interpolation points that are far from $\underline{x}_{\rm opt}$. This removal is encouraged by using a sixth power of $\|\underline{x}_t-\underline{x}_{\rm opt}\|$ instead of the fourth power that is indicated above. Another consideration is that interpolation points tend to cluster near $\underline{x}_{\rm opt}$ only when Δ is either being reduced or is at its lower bound ρ, so the weights are given the values

$$w_t \;=\; \max\left[1,\left\{\|\underline{x}_t-\underline{x}^*\|\;\Big/\max[0.1\,\Delta,\,\rho]\right\}^6\right], \qquad t\in\mathcal{T}, \tag{7.5}$$

where $\underline{x}^*$ is the $\underline{x}_{\rm opt}$ that is going to be selected in Box 5 of Figure 1. The `MOVE` $=0$ alternative preserves the old interpolation points, so it is available only in the case $F(\underline{x}_{\rm opt}+\underline{d})\ge F(\underline{x}_{\rm opt})$. We wish to avoid applications of formula (4.11) that cause abnormal growth in the elements of H, taking into consideration that some growth is usual when a remote interpolation point is dropped. Therefore `MOVE` is set to zero instead of to t^* if and only if both the conditions $F(\underline{x}_{\rm opt}+\underline{d})\ge F(\underline{x}_{\rm opt})$ and $w_{t^*}|\sigma_{t^*}|\le 1$ hold.

The value of ρ is decreased from $\rho_{\rm old}$ to $\rho_{\rm new}$ in Box 12 of Figure 1. The reduction is by a factor of 10, unless only one or two changes to ρ are going to attain the final value $\rho=\rho_{\rm end}$. The equation $\rho_{\rm old}/\rho_{\rm new}=\rho_{\rm new}/\rho_{\rm end}$ gives a balance between the two reductions in the latter case. These remarks and some choices of parameters provide the formula

$$\rho_{\rm new} \;=\; \begin{cases} \rho_{\rm end}, & \rho_{\rm old}\le 16\,\rho_{\rm end}, \\ (\rho_{\rm old}\,\rho_{\rm end})^{1/2}, & 16\,\rho_{\rm end}<\rho_{\rm old}\le 250\,\rho_{\rm end}, \\ 0.1\,\rho_{\rm old}, & \rho_{\rm old}>250\,\rho_{\rm end}, \end{cases} \tag{7.6}$$

for the adjustment of ρ by NEWUOA.

The reason for Box 14 in Figure 1 is explained in the penultimate paragraph of Section 2, the calculations with the current value of ρ being complete if the "`Y`" branch is taken. We see that Box 14 is reached when the trust region subproblem of Box 2 yields a step $\underline{d}$ that has the property $\|\underline{d}\|<\frac{1}{2}\rho$, which suggests that the current quadratic model Q is convex. Therefore, assuming that `CRVMIN` is a useful estimate of the least eigenvalue of $\nabla^2 Q$, we prefer not to calculate $F(\underline{x}_{\rm opt}+\underline{d})$ when the predicted reduction in F, namely

$Q(\underline{x}_{\rm opt})-Q(\underline{x}_{\rm opt}+\underline{d})$, is less than $\frac{1}{8}\rho^2$CRVMIN. Further, if the values of the error $|Q(\underline{x}_{\rm opt}+\underline{d})-F(\underline{x}_{\rm opt}+\underline{d})|$ on recent iterations are also less than this amount, then we take the view that trying to improve the accuracy of the model would be a waste of effort. Specifically, the test in Box 14 is satisfied if at least 3 new values of F have been computed for the current ρ, and if all the conditions

$$\|\underline{d}^{(j)}\| \le \rho \quad \text{and} \quad |Q_j(\underline{x}_{\rm opt}^{(j)}+\underline{d}^{(j)}) - F(\underline{x}_{\rm opt}^{(j)}+\underline{d}^{(j)})| \le \tfrac{1}{8}\rho^2\texttt{CRVMIN}, \quad j\in\mathcal{J}, \tag{7.7}$$

hold, where Q_j, $\underline{d}^{(j)}$ and $\underline{x}_{\rm opt}^{(j)}$ are Q, $\underline{d}$ and $\underline{x}_{\rm opt}$ at the beginning of Box 5 on the j-th iteration, where CRVMIN is generated on the current iteration, and where $\mathcal{J}$ contains 3 integers, namely the iteration numbers of the 3 most recent visits to Box 5 before the current iteration. Thus the work of NEWUOA with the current ρ is terminated often, although some of the distances $\|\underline{x}_i-\underline{x}_{\rm opt}\|$, $i=1,2,\ldots,m$, may exceed 2ρ.

In order to show the importance of $\underline{x}_0$ in practice to the rounding errors of the updating formula (4.11), we assume that all the distances $\|\underline{x}_i-\underline{x}_j\|$, $1\le i<j\le m$, between interpolation points are of magnitude one, that $\|\underline{d}\| = \|\underline{x}^+-\underline{x}_{\rm opt}\|$ is also of magnitude one, but that $\|\underline{x}_{\rm opt}-\underline{x}_0\|=M$, say, is large. In theory, the parameters α, β, τ and σ of expression (4.12), and also the leading $m\times m$ submatrix of H, are independent of $\underline{x}_0$ (Powell, 2004a), but the definition (4.10) implies that each of the first m components of $\underline{w}$ is approximately $\frac{1}{2}M^4$. Thus much cancellation occurs in the formula

$$\beta \;=\; \tfrac{1}{2}\,\|\underline{x}^+-\underline{x}_0\|^4 - \underline{w}^T H\,\underline{w}. \tag{7.8}$$

Further, if there were an error of ε in H_{11}, and if there were no other errors on the right hand side of equation (7.8), then β would include an error of magnitude $M^8\varepsilon$, this power of M being so large that $M>100$ could be disastrous. The substitution of expression (4.26) into formula (7.8) is less unfavourable, because H_{11} is multiplied by $-(w_1-v_1)^2$, and the middle line of equation (6.24) provides the value

$$w_1-v_1 \;=\; \tfrac{1}{2}\,\{(\underline{x}^+-\underline{x}_{\rm opt})^T(\underline{x}_1-\underline{x}_0)\}\,\{(\underline{x}^++\underline{x}_{\rm opt}-2\,\underline{x}_0)^T(\underline{x}_1-\underline{x}_0)\}. \tag{7.9}$$

Thus the error in β is now of magnitude $M^6\varepsilon\cos 2\theta$, where θ is the angle between $\underline{x}_1-\underline{x}_0$ and $\underline{d}=\underline{x}^+-\underline{x}_{\rm opt}$. The factorization (4.16) also helps the attainment of adequate accuracy. Nevertheless, we found from numerical experiments in REAL*8 arithmetic, using some difficult objective functions, that sequences of iterations may cause unacceptable errors if $\|\underline{x}_{\rm opt}-\underline{x}_0\|\ge 10^{2.5}\|\underline{d}\|$ is allowed in the updating calculations of Section 4. Therefore NEWUOA tests the condition

$$\|\underline{d}\|^2 \le 10^{-3}\,\|\underline{x}_{\rm opt}-\underline{x}_0\|^2 \tag{7.10}$$

before replacing $\underline{x}_{\texttt{MOVE}}$ by $\underline{x}_{\rm opt}+\underline{d}$ in Box 5 of Figure 1. If this condition holds, then $\underline{x}_0$ is overwritten by the $\underline{x}_{\rm opt}$ that occurs at the beginning of Box 5, which alters the last n rows of the matrix (1.3) and all the elements (3.11). In

practice, however, the matrix $H=W^{-1}$ of expression (3.12) is stored instead of W. Therefore H is revised in the way that is implied by the change to W, except that the $(m+1)$-th row and column of H are not required. Details of this task are considered in Section 5 of Powell (2004a), so only a brief outline is given below of the changes that are made to H when $\underline{x}_0$ is shifted.

Let $\underline{x}_{\rm av}$ and $\underline{s}$ be the vectors $\frac{1}{2}(\underline{x}_0+\underline{x}_{\rm opt})$ and $\underline{x}_{\rm opt}-\underline{x}_0$, respectively, before $\underline{x}_0$ is overwritten by $\underline{x}_{\rm opt}$, let Y be the $n\times m$ matrix that has the columns

$$\underline{y}_j \;=\; \{\underline{s}^T(\underline{x}_j-\underline{x}_{\rm av})\}\,(\underline{x}_j-\underline{x}_{\rm av})+\tfrac{1}{4}\,\|\underline{s}\|^2\,\underline{s}, \qquad j=1,2,\ldots,m, \tag{7.11}$$

and let $\Theta_{\rm old}$ and $\Theta_{\rm new}$ be the old and new H matrices without their $(m+1)$-th rows and columns. Then, according to equations (5.11) and (5.12) of Powell (2004a), $\Theta_{\rm new}$ is defined by the formula

$$\Theta_{\rm new} \;=\; \left(\begin{array}{c|c} I & 0 \\ \hline Y & I \end{array}\right) \Theta_{\rm old} \left(\begin{array}{c|c} I & Y^T \\ \hline 0 & I \end{array}\right). \tag{7.12}$$

Thus, as mentioned already, the submatrix Ω of expression (3.12) is undisturbed, and we keep its factorization (4.16). It follows also from expressions (3.12) and (7.12) that the product $Y\Omega$ and the sum $Y\Xi_{\rm red}^T+\Xi_{\rm red}Y^T+Y\Omega Y^T$ are added to the last n rows of Ξ and to the trailing $n\times n$ submatrix of Υ, respectively, where $\Xi_{\rm red}$ is the original matrix Ξ without its first row.

When $\underline{x}_0$ is overwritten by $\underline{x}_{\rm opt}$, the gradient $\underline{\nabla}Q(\underline{x}_0)$ has to be revised too. Specifically, because the function (3.1) can be written in the form

$$Q(\underline{x}_{\rm opt}+\underline{d}) \;=\; Q(\underline{x}_{\rm opt})+\underline{d}^T\underline{\nabla}Q(\underline{x}_{\rm opt})+\tfrac{1}{2}\,\underline{d}^T\nabla^2Q\,\underline{d}, \qquad \underline{d}\in\mathcal{R}^n, \tag{7.13}$$

and because $\underline{\nabla}Q(\underline{x}_{\rm opt})=\underline{\nabla}Q(\underline{x}_0)+\nabla^2Q\,\underline{s}$ follows from $\underline{s}=\underline{x}_{\rm opt}-\underline{x}_0$, the vector $\nabla^2Q\,\underline{s}$ is added to $\underline{\nabla}Q(\underline{x}_0)$. The constant term of Q is unnecessary, as stated at the end of Section 4, and ∇^2Q is independent of $\underline{x}_0$, except that, as in equation (4.28), it is expressed as the sum

$$\begin{aligned}\nabla^2Q &= \Gamma+\textstyle\sum_{j=1}^m\gamma_j\,(\underline{x}_j-\underline{x}_0)\,(\underline{x}_j-\underline{x}_0)^T\\ &= \Gamma+\textstyle\sum_{j=1}^m\gamma_j\,(\underline{x}_j-\underline{x}_{\rm opt}+\underline{s})\,(\underline{x}_j-\underline{x}_{\rm opt}+\underline{s})^T\\ &= \Gamma+\underline{v}\,\underline{s}^T+\underline{s}\,\underline{v}^T+\textstyle\sum_{j=1}^m\gamma_j\,(\underline{x}_j-\underline{x}_{\rm opt})\,(\underline{x}_j-\underline{x}_{\rm opt})^T,\end{aligned} \tag{7.14}$$

where $\underline{v}=\sum_{j=1}^m\gamma_j\,(\underline{x}_j-\underline{x}_{\rm opt}+\frac{1}{2}\underline{s})=\sum_{j=1}^m\gamma_j\,(\underline{x}_j-\underline{x}_{\rm av})$. Therefore the shift in $\underline{x}_0$ requires $\underline{v}\,\underline{s}^T+\underline{s}\,\underline{v}^T$ to be added to Γ, although the parameters γ_j, $j=1,2,\ldots,m$, are unchanged.

The amount of work in the previous paragraph is only $\mathcal{O}(mn)$, but the implementation of the product (7.12) takes $\mathcal{O}(m^2n)$ operations. Therefore we hope that condition (7.10) holds on only a small fraction of the total number of iterations, especially when n is large. Rough answers to this question are provided by the running times of the numerical experiments of the next section. They suggest that usually the average work per iteration of NEWUOA is close to $\mathcal{O}(mn)$.

8 Numerical results

In December, 2003, the author released the Fortran software of the version of NEWUOA that has been described, having tested it on a range of problems with up to 200 variables. Then, at the conference in Erice of these proceedings, he discussed with Nick Gould some other problems that might be tried, which led to more experiments. It became clear from one of them that a further modification would be advantageous occasionally. It has now been made, and is the first subject of this section, because the numerical results that follow were calculated by the new version of NEWUOA.

The experiment that suggested the modification is the VARDIM test problem on page 98 of Buckley (1989). The objective function is the quartic polynomial

$$F(\underline{x}) = \sum_{\ell=1}^{n} (x_\ell - 1)^2 + \Big\{ \sum_{\ell=1}^{n} \ell\,(x_\ell - 1) \Big\}^2 + \Big\{ \sum_{\ell=1}^{n} \ell\,(x_\ell - 1) \Big\}^4, \qquad \underline{x} \in \mathcal{R}^n, \quad (8.1)$$

which takes its least value of zero at $\underline{x} = \underline{e}$, the vector of ones. Analytic differentiation gives the second derivative matrix

$$\nabla^2 F(\underline{x}) \;=\; 2\,I + \big[\,2 + 12\,\{\textstyle\sum_{\ell=1}^{n} \ell\,(x_\ell - 1)\}^2\,\big]\,\Theta, \qquad \underline{x} \in \mathcal{R}^n, \quad (8.2)$$

where I is the $n \times n$ unit matrix and where Θ is the rank one matrix that has the elements $\theta_{ij} = ij$, $1 \le i, j \le n$. Thus $\nabla^2 F(\underline{x})$ has $n-1$ eigenvalues of 2 and one of $2 + [\,\frac{1}{3} + 2\{\sum_{\ell=1}^{n} \ell(x_\ell - 1)\}^2]\,n(n+1)(2n+1)$. When NEWUOA is employed with $m = 2n+1$, however, the initial quadratic model has a diagonal second derivative matrix, the diagonal elements of $\nabla^2 Q$ being approximately those of $\nabla^2 F(\underline{x}_0)$, where $\underline{x}_0$ is the given starting vector of variables, which has the components $1 - i/n$, $i = 1, 2, \ldots, n$, in the VARDIM test problem. Thus initially the eigenvalues of $\nabla^2 Q$ are about $2 + [\,2 + 12\{\sum_{\ell=1}^{n} \ell(x_\ell - 1)\}^2]\,i^2$, $i = 1, 2, \ldots, n$, the term in square brackets being $2 + \frac{1}{3}(n+1)^2(2n+1)^2$. It follows that, at the start of the calculation, $\nabla^2 Q$ is a very bad estimate of $\nabla^2 F$. Further, if $n = 80$ for instance, the range of eigenvalues of $\nabla^2 Q$ initially is from about 5.7×107 to 3.6×10^{11}, but the large eigenvalue of $\nabla^2 F$ at the solution $\underline{x} = \underline{e}$ is only 347762. Therefore NEWUOA cannot perform satisfactorily unless huge improvements are made to $\nabla^2 Q$ by the updating formulae of the sequence of iterations.

Unfortunately, however, each application of the least Frobenius norm updating method makes the smallest change to $\nabla^2 Q$ that is allowed by the new interpolation conditions, so the basic method of NEWUOA is not suitable for the VARDIM test problem. Therefore the recent modification tries to recognise when the elements of $\nabla^2 Q$ are much too large, and, if there is strong evidence for this possibility, then Q is replaced by $Q_{\rm int}$, which is the quadratic model that minimizes $\|\nabla^2 Q_{\rm int}\|_F$, instead of the Frobenius norm of the change to $\nabla^2 Q$, subject to the conditions $Q_{\rm int}(\underline{x}_i) = F(\underline{x}_i)$, $i = 1, 2, \ldots, m$, the interpolation points $\underline{x}_i$ being the updated ones at the exit from Box 5 of

Figure 1. When $Q_{\rm int}$ is preferred, the gradient $\underline{\nabla} Q_{\rm int}(\underline{x}_0)$ and the parameters γ_j, $j=1,2,\ldots,m$, of the expression

$$\nabla^2 Q_{\rm int} \;=\; \textstyle\sum_{j=1}^{m} \gamma_j \, (\underline{x}_j - \underline{x}_0) \, (\underline{x}_j - \underline{x}_0)^T \tag{8.3}$$

are required. It follows from the definition of $Q_{\rm int}$ that they are the vector $\underline{g}$ and the components of $\underline{\lambda}$ in the system (3.10), where $\underline{r}$ has the components $r_i = F(\underline{x}_i) - \phi$, $i=1,2,\ldots,m$, for any $\phi \in \mathcal{R}$. Some damage from rounding errors is avoided by the choice $\phi = F(\underline{x}_{\rm opt})$. We deduce from the notation (3.12) that $\underline{g}$ and $\underline{\lambda}$ are the products $\Xi_{\rm red}\,\underline{r}$ and $\Omega\,\underline{r}$, respectively, where $\Xi_{\rm red}$ is still the matrix Ξ without its first row. Thus NEWUOA constructs a useful form of $Q_{\rm int}$ in $\mathcal{O}(m^2)$ operations.

When the elements of $\nabla^2 Q$ are much too large, the interpolation equations (1.1) imply that $\|\underline{\nabla} Q(\underline{x})\|$ is also much too large for most vectors of variables. Usually a huge value of $\|\underline{\nabla} Q(\underline{x}_{\rm opt})\|$ causes the ratio (2.2) to be tiny. Moreover, because $\underline{\nabla} Q(\underline{x}_0)$ is available, and because we have found that $\underline{\nabla} Q_{\rm int}(\underline{x}_0)$ is the product $\Xi_{\rm red}\,\underline{r}$, it is easy to compare $\|\underline{\nabla} Q_{\rm int}(\underline{x}_0)\|$ with $\|\underline{\nabla} Q(\underline{x}_0)\|$. On the iterations of the new version of NEWUOA that reach Box 5 of Figure 1 from Box 4, a flag is set to `YES` or `NO`, the `YES` being chosen when the conditions

$$\texttt{RATIO} \le 0.01 \qquad \text{and} \qquad \|\underline{\nabla} Q_{\rm int}(\underline{x}_0)\| \le 0.1 \, \|\underline{\nabla} Q(\underline{x}_0)\| \tag{8.4}$$

hold at the end of Box 5. Then Q is replaced by $Q_{\rm int}$ if and only if three consecutive settings of the flag are all `YES`.

n	Original NEWUOA		Modified NEWUOA	
	$\#F$	$F(\underline{x}_{\rm fin})$	$\#F$	$F(\underline{x}_{\rm fin})$
20	12018 : 11517	2×10^{-11} : 8×10^{-11}	5447 : 4610	4×10^{-11} : 3×10^{-11}
40	45510 : 56698	7×10^{-10} : 3×10^{-10}	17106 : 17853	1×10^{-10} : 8×10^{-11}
80	196135 : 234804	7×10^{-9} : 3×10^{-9}	60305 : 55051	1×10^{-10} : 3×10^{-10}

Table 1: Two versions of NEWUOA applied to VARDIM with $m=2n+1$

The VARDIM test problem with 80 variables can be solved by the older version of NEWUOA, in spite of the deficiencies in $\nabla^2 Q$ that have been noted. Results for the unmodified and modified versions, using $\rho_{\rm beg} = (2n)^{-1}$ and $\rho_{\rm end} = 10^{-6}$, are displayed on the left and right hand sides, respectively, of Table 1. The heading $\#F$ denotes the total number of calculations of the objective function, and $\underline{x}_{\rm fin}$ is the vector of variables that is returned by NEWUOA, because it gives the least calculated value of F. In theory, a reordering of the variables makes no difference, the initial set of interpolation points being unchanged for $m=2n+1$, so this device can be used to investigate some effects of computer rounding errors. The entries to the left and right of the colons in

Table 1 were obtained with different orderings. We see that rounding errors are highly influential, that the values of $F(\underline{x}_{\rm fin})$ are satisfactory, and that the modification is successful in reducing $\#F$.

During the development of NEWUOA, the objective function that was used most is the trigonometric sum of squares

$$F(\underline{x}) \;=\; \sum_{i=1}^{2n} \Big\{ b_i - \sum_{j=1}^{n} \Big(S_{ij}\, \sin(\theta_j x_j) + C_{ij}\, \cos(\theta_j x_j) \Big) \Big\}^2, \qquad \underline{x} \in \mathcal{R}^n, \quad (8.5)$$

namely TRIGSSQS. The elements of the matrices S and C are random integers from $[-100, 100]$, each scaling factor θ_j is sampled from the logarithmic distribution on $[0.1, 1]$, and the parameters b_i, $i = 1, 2, \ldots, 2n$, are defined by $F(\underline{x}^*) = 0$, where $\underline{x}^*$ has the components $x_j^* = \hat{x}_j^*/\theta_j$, $j = 1, 2, \ldots, n$, each $\hat{x}_j^*$ being picked from the uniform distribution on $[-\pi, \pi]$. The initial vector $\underline{x}_0$ has the components $(\hat{x}_j^* + 0.1\hat{y}_j^*)/\theta_j$, $j = 1, 2, \ldots, n$, where every $\hat{y}_j^*$ is also taken at random from $[-\pi, \pi]$. The function (8.5) has saddle points and maxima, due to periodicity, and the values of the scaling factors θ_j provide a tougher problem than the case $\theta_j = 1$, $j = 1, 2, \ldots, n$. For each n, we generate five different objective functions and starting points by choosing different random numbers. We let the number of interpolation conditions, namely m, be $2n+1$, $m^{(\rm av)}$ or $\frac{1}{2}(n+1)(n+2)$, where $m^{(\rm av)}$ is the integer that is nearest to $\{(n+\frac{1}{2})(n+1)(n+2)\}^{1/2}$. Results of the NEWUOA software for some of these cases, with four values of n and the parameters $\rho_{\rm beg} = 10^{-1}$ and $\rho_{\rm end} = 10^{-6}$, are reported in Table 2, the entries in the main part of the table being averages for the five different test problems that have been mentioned. Both $\#F$ and $\underline{x}_{\rm fin}$ have been defined already. Again the results are sensitive to the effects of computer rounding errors. The dashes in the table indicate that the problems were not tried, because of the running times that would be required on a Sun Ultra 10 workstation. The values of $\#F$ in the $m = 2n+1$ part of the table are much smaller than the author had expected originally, because they become less than the number of degrees of freedom in a quadratic model when n is large. This highly welcome situation provides excellent support for the least Frobenius norm updating technique. The accuracy of the calculations is satisfactory, the $\|\underline{x}_{\rm fin} - \underline{x}^*\|_\infty$ entries in the table being comparable to $\rho_{\rm end}$.

The method of NEWUOA, in particular the use of the bound $\Delta \geq \rho$ in Figure 1, is intended to be suitable for the minimization of functions that have first derivative discontinuities. Therefore Table 3 gives some results for the objective function TRIGSABS, which has the form

$$F(\underline{x}) \;=\; \sum_{i=1}^{2n} \Big| \, b_i - \sum_{j=1}^{n} \Big(S_{ij}\, \sin x_j + C_{ij}\, \cos x_j \Big) \Big|, \qquad \underline{x} \in \mathcal{R}^n. \quad (8.6)$$

The parameters b_i, S_{ij} and C_{ij}, and the initial vector $\underline{x}_0$, are generated randomly as in the previous paragraph, except that we employ the scaling factors $\theta_j = 1$, $j = 1, 2, \ldots, n$. Different random numbers provide five test problems

n	$m=2n+1$		$m=m^{(\mathrm{av})}$		$m=\frac{1}{2}(n+1)(n+2)$	
	$\#F$	$\|\underline{x}_{\mathrm{fin}}-\underline{x}^*\|_\infty$	$\#F$	$\|\underline{x}_{\mathrm{fin}}-\underline{x}^*\|_\infty$	$\#F$	$\|\underline{x}_{\mathrm{fin}}-\underline{x}^*\|_\infty$
20	931	1.4×10^{-6}	833	6.9×10^{-7}	649	2.0×10^{-7}
40	1809	4.2×10^{-6}	1716	1.3×10^{-6}	2061	5.5×10^{-7}
80	3159	3.8×10^{-6}	3471	2.1×10^{-6}	—	—
160	6013	5.8×10^{-6}	—	—	—	—

Table 2: Averages for NEWUOA applied to 5 versions of TRIGSSQS

for each n as before. We retain $\rho_{\mathrm{beg}}=0.1$, but we set $\rho_{\mathrm{end}}=10^{-8}$, in order to take advantage of the sharpness of the minimum of F at $\underline{x}=\underline{x}^*$. The entries in Table 3 are analogous to those of Table 2. We see that, for each n, the least value of $\#F$ occurs in the $m=2n+1$ column, those results being very encouraging. If ρ_{end} is reduced to 10^{-6}, the figures for $m=2n+1$ and $n=160$ become $\#F=12007$ and $\|\underline{x}_{\mathrm{fin}}-\underline{x}^*\|_\infty=1.6\times10^{-6}$, so again $\#F$ is less than the number of degrees of freedom in a quadratic model.

n	$m=2n+1$		$m=m^{(\mathrm{av})}$		$m=\frac{1}{2}(n+1)(n+2)$	
	$\#F$	$\|\underline{x}_{\mathrm{fin}}-\underline{x}^*\|_\infty$	$\#F$	$\|\underline{x}_{\mathrm{fin}}-\underline{x}^*\|_\infty$	$\#F$	$\|\underline{x}_{\mathrm{fin}}-\underline{x}^*\|_\infty$
20	1454	1.0×10^{-8}	2172	6.6×10^{-9}	4947	4.8×10^{-9}
40	3447	1.6×10^{-8}	6232	7.7×10^{-9}	24039	5.9×10^{-9}
80	7626	1.2×10^{-8}	16504	7.2×10^{-9}	—	—
160	16496	2.2×10^{-8}	—	—	—	—

Table 3: Averages for NEWUOA applied to 5 versions of TRIGSABS

We consider next a test problem that was invented by the author recently, namely SPHRPTS. Here n is even, and $n/2$ points have to be placed on the surface of the unit sphere in three dimensions at positions that are far apart. We let the k-th point $\underline{p}_k\in\mathcal{R}^3$ have the coordinates

$$\underline{p}_k \;=\; \begin{pmatrix} \cos x_{2k-1}\,\cos x_{2k} \\ \sin x_{2k-1}\,\cos x_{2k} \\ \sin x_{2k} \end{pmatrix}, \qquad k=1,2,\ldots,n/2, \tag{8.7}$$

where $\underline{x}\in\mathcal{R}^n$ is still the vector of variables. The problem is to minimize the function

$$F(\underline{x}) \;=\; \sum_{k=2}^{n/2}\,\sum_{\ell=1}^{k-1}\,\|\underline{p}_\ell-\underline{p}_k\|^{-2}, \qquad \underline{x}\in\mathcal{R}^n, \tag{8.8}$$

where initially the points $\underline{p}_k$ are equally spaced on the equator of the sphere, the vector $\underline{x}_0$ having the components $(\underline{x}_0)_{2k-1}=4\pi k/n$ and $(\underline{x}_0)_{2k}=0$, $k=$

$1, 2, \ldots, n/2$. The NEWUOA software was applied to this problem, taking m and n from Tables 2 and 3, with $\rho_{\rm beg} = n^{-1}$ and $\rho_{\rm end} = 10^{-6}$. The resultant values of $\#F$ are shown to the left of the colons in Table 4. We found also that $F(\underline{x}_{\rm fin})$ agrees with the minimum value of $F(\underline{x})$, $\underline{x} \in \mathcal{R}^n$, to more than 10 decimal places, although there is much freedom in the optimal vector of variables, because permutations of the points and rotations of the unit sphere do not alter the value of the double sum (8.8). Therefore many adjustments of the variables in practice cause only a tiny reduction in the objective function. Indeed, after computing each $F(\underline{x}_{\rm fin})$, we inspected the sequence of values of F calculated by NEWUOA, in order to note the position in the sequence of the first value that satisfies $F(\underline{x}) \leq 1.001\, F(\underline{x}_{\rm fin})$. These positions are given to the right of the colons in Table 4. We see that, for the SPHRPTS problem, most of the work is spent on marginal improvements to F, especially during the calculations of the $m = 2n+1$ column.

n	$m = 2n+1$	$m = m^{(\rm av)}$	$m = \frac{1}{2}(n+1)(n+2)$
20	2077 : 351	1285 : 513	1161 : 627
40	7245 : 1620	4775 : 2884	6636 : 2924
80	9043 : 3644	18679 : 13898	—
160	24031 : 8193	—	—

Table 4: Values of $\#F$ for the SPHRPTS problem

The NEWUOA software has also been tested on several problems that have been proposed by other authors. The final table presents results in the following five cases using $m = 2n+1$. The ARWHEAD problem (see the Appendix of Conn *et al*, 1994) has the objective function

$$F(\underline{x}) \;=\; \sum_{i=1}^{n-1} \left\{(x_i^2 + x_n^2)^2 - 4\,x_i + 3\right\}, \qquad \underline{x} \in \mathcal{R}^n, \tag{8.9}$$

and the starting point $\underline{x}_0$ is $\underline{e} \in \mathcal{R}^n$, which is still the vector of ones. In the CHROSEN problem (see page 45 of Buckley, 1989), we let F be the function

$$F(\underline{x}) \;=\; \sum_{i=1}^{n-1} \left\{4\,(x_i - x_{i+1}^2)^2 + (1 - x_{i+1})^2\right\}, \qquad \underline{x} \in \mathcal{R}^n, \tag{8.10}$$

and the starting point $\underline{x}_0$ is $-\underline{e} \in \mathcal{R}^n$. The PENALTY1 problem (see page 79 of Buckley, 1989) includes two parameters, and we pick the objective function

$$F(\underline{x}) \;=\; 10^{-5} \sum_{i=1}^{n} (x_i - 1)^2 + \Big(\tfrac{1}{4} - \sum_{i=1}^{n} x_i^2\Big)^2, \qquad \underline{x} \in \mathcal{R}^n, \tag{8.11}$$

with the starting point $(\underline{x}_0)_i = i$, $i = 1, 2, \ldots, n$. Our choice of parameters for the PENALTY2 problem (see page 80 of Buckley, 1989) gives the function

$$F(\underline{x}) = \sum_{i=2}^{n} \left\{ (e^{x_{i-1}/10} + e^{x_i/10} - e^{(i-1)/10} - e^{i/10})^2 + (e^{x_i/10} - e^{-1/10})^2 \right\}$$
$$+ \left\{ 1 - \sum_{i=1}^{n} (n-i+1)\, x_i^2 \right\}^2 + (x_1 - \tfrac{1}{5})^2, \qquad \underline{x} \in \mathcal{R}^n, \quad (8.12)$$

and the starting point $\underline{x}_0$ is $\frac{1}{2}\underline{e} \in \mathcal{R}^n$. The PENALTY3 problem (see page 81 of Buckley, 1989) has the objective function

$$F(\underline{x}) = 10^{-3} \left(1 + R\, e^{x_n} + S\, e^{x_{n-1}} + R\, S \right)$$
$$+ \left\{ \sum_{i=1}^{n} (x_i^2 - n) \right\}^2 + \sum_{i=1}^{n/2} (x_i - 1)^2, \qquad \underline{x} \in \mathcal{R}^n, \quad (8.13)$$

where R and S are the sums

$$R = \sum_{i=1}^{n-2} (x_i + 2\, x_{i+1} + 10\, x_{i+2} - 1)^2 \quad \text{and} \quad S = \sum_{i=1}^{n-2} (2\, x_i + x_{i+1} - 3)^2, \quad (8.14)$$

and we let the starting point $\underline{x}_0 \in \mathcal{R}^n$ be the zero vector. We set $\rho_{\text{end}} = 10^{-6}$ in every case, while ρ_{beg} is given the value 0.5, 0.5, 1.0, 0.1 and 0.1 for ARWHEAD, CHROSEN, PENALTY1, PENALTY2 and PENALTY3, respectively. Table 5 shows the numbers of function evaluations that occurred when NEWUOA was applied to these problems with our usual choices of n, except that $\star$ indicates that $\#F$ exceeded 500,000.

n	ARWHEAD	CHROSEN	PENALTY1	PENALTY2	PENALTY3
20	404	845	7476	2443	3219
40	1497	1876	14370	2455	16589
80	3287	4314	32390	5703	136902
160	8504	9875	72519	$\star$	$\star$

Table 5: Values of $\#F$ for 5 problems with $m = 2n+1$

All the ARWHEAD, CHROSEN and PENALTY1 calculations were completed successfully, the greatest distance $\|\underline{x}_{\text{fin}} - \underline{x}^*\|_\infty$ being 6.1×10^{-6}, where $\underline{x}_{\text{fin}}$ and $\underline{x}^*$ are still the final and optimal vectors of variables. Good accuracy was also achieved in the PENALTY2 calculations with $n \leq 80$, the values of $F(\underline{x}_{\text{fin}})$ agreeing to 13 decimal places with other values that were obtained for permutations of the variables and other choices of m. When $n = 160$ is selected, however, the constants $e^{i/10}$, $i = 1, 2, \ldots, n$, vary from 1.1 to 9×106, so the magnitudes of the terms under the first summation sign of expression (8.12) vary from 1 to 10^{13}, which causes the PENALTY2 problem to be too difficult

in REAL*8 arithmetic. We compared the given results of the PENALTY3 calculations with those that occurred after permuting the variables. The $\#F$ entries became 4336, 18209 and 125884 for $n=20$, $n=40$ and $n=80$, respectively, which agrees well with the last column of Table 5. Further, for each n, the two values of $F(\underline{x}_{\rm fin})$ were slightly less than n^2, and they agreed to about 11 decimal places. A feature of PENALTY3, however, is that the minimum value of the objective function is close to 10^{-3} and is hard to find. This magnitude is exposed by picking the variables $x_i=1$, $i=1,2,\ldots,n-1$, and $x_n=-(n^2-n+1)^{1/2}$, because then e^{x_n} is tiny and both S and the second line of expression (8.13) are zero, which provides $F(\underline{x})=10^{-3}(1+R\,e^{x_n})\approx 10^{-3}$. When NEWUOA was applied to PENALTY3 with $n=160$, the original ordering of the variables yielded $\#F=629582$ and $F(\underline{x}_{\rm fin})=25447.688$, while the new ordering yielded $\#F=16844$ and $F(\underline{x}_{\rm fin})=0.001002$. We had not expected the new ordering to be so favourable, because the differences in the results are due entirely to computer rounding errors.

The average amount of work per iteration is mentioned at the end of Section 7, being at best $\mathcal{O}(n^2)$ in the case $m=2n+1$. We tested this possibility in the ARWHEAD and PENALTY1 experiments of Table 5. The total time in seconds of each calculation on a Sun Ultra 10 workstation was divided by the product of n^2 and $\#F$. The resultant quotients for ARWHEAD are 8.4×10^{-6}, 8.0×10^{-6}, 8.5×10^{-6} and 8.8×10^{-6} in the cases $n=20$, $n=40$, $n=80$ and $n=160$, respectively, and the corresponding quotients for PENALTY1 are 9.2×10^{-6}, 8.5×10^{-6}, 8.6×10^{-6} and 9.3×10^{-6}, the running time in the last case being nearly 5 hours, while ARWHEAD with $n=20$ was solved in only 1.36 seconds. These findings suggest that the average complexity of each iteration is proportional to n^2, which is most welcome.

The development of NEWUOA has taken nearly three years. The work was very frustrating, due to severe damage from computer rounding errors in difficult cases, before the factorization (4.16) of Ω was introduced. Therefore the author has had doubts about the use of the explicit inverse matrix $H=W^{-1}$, instead of using a factored form of W that allows the system (3.10) to be solved in $\mathcal{O}(m^2)$ operations. The numerical results are still highly sensitive to computer rounding errors, but the experiments of this section show that good accuracy is achieved eventually, which confirms the stability of the given techniques. Thus we conclude that the least Frobenius norm method for updating quadratic models is highly successful in unconstrained minimization calculations without derivatives. Readers are invited to request a free copy of the NEWUOA Fortran software by sending an e-mail to mjdp@cam.ac.uk.

Appendix: Proofs for Section 3

The assertions of the last two paragraphs of Section 3 are presented below as lemmas with proofs. The positions of the relevant interpolation points are

described at the beginning of Section 3, followed by the definitions of the matrices (3.12).

Lemma 1. *The first row of the initial matrix Ξ has the elements (3.13), and, for every integer i that satisfies $2 \le i \le \min[n+1, m-n]$, the i-th row includes the elements (3.14). When $m \le 2n$, the nonzero elements of the remaining rows of Ξ take the values (3.15), where i is any integer from the interval $[m-n+1, n+1]$. All other elements of the initial matrix Ξ are zero.*

Proof: For each integer j in $[1, m]$, we let the quadratic polynomial

$$\ell_j(\underline{x}) = \ell_j(\underline{x}_0) + (\underline{x}-\underline{x}_0)^T \underline{\nabla} \ell_j(\underline{x}_0) + \tfrac{1}{2} (\underline{x}-\underline{x}_0)^T \nabla^2 \ell_j (\underline{x}-\underline{x}_0), \quad \underline{x} \in \mathcal{R}^n, \quad \text{(A.1)}$$

be the j-th Lagrange function of the initial interpolation points, which means that $\|\nabla^2 \ell_j\|_F$ is as small as possible subject to the conditions

$$\ell_j(\underline{x}_i) = \delta_{ij}, \qquad i = 1, 2, \ldots, m, \qquad \text{(A.2)}$$

as stated in the second paragraph of Section 6. The construction of ℓ_j is the same as the construction of D in Section 3, if the constraints (3.6) have the right hand sides $F(\underline{x}_i) - Q_{\rm old}(\underline{x}_i) = \delta_{ij}$, $i = 1, 2, \ldots, m$. Therefore $\ell_j(\underline{x}_0)$ and $\underline{\nabla} \ell_j(\underline{x}_0)$ are the same as c and $\underline{g}$, respectively, in the system (3.10), when $\underline{r}$ is the coordinate vector $\underline{e}_j \in \mathcal{R}^m$. In this case, the partitioned vector on the left hand side of equation (3.10) is the j-th column of W^{-1}. It follows from the notation (3.12) that $\ell_j(\underline{x}_0)$ and $\underline{\nabla} \ell_j(\underline{x}_0)$ provide the j-th column of Ξ, as shown in the expression

$$\Xi = \begin{pmatrix} \ell_1(\underline{x}_0) & \ell_2(\underline{x}_0) & \cdots & \ell_m(\underline{x}_0) \\ \underline{\nabla} \ell_1(\underline{x}_0) & \underline{\nabla} \ell_2(\underline{x}_0) & \cdots & \underline{\nabla} \ell_m(\underline{x}_0) \end{pmatrix}. \qquad \text{(A.3)}$$

The remainder of the proof depends on the positions of the initial interpolation points. In particular, because of the choice $\underline{x}_1 = \underline{x}_0$ with the first of the conditions (A.2) for each j, the first row of the matrix (A.3) has the elements (3.13). Moreover, when k satisfies $1 \le k \le \min[n, m-n-1]$, the points $\underline{x}_{k+1} = \underline{x}_0 + \rho_{\rm beg} \underline{e}_k$ and $\underline{x}_{k+n+1} = \underline{x}_0 - \rho_{\rm beg} \underline{e}_k$ have been chosen, so the k-th component of $\underline{\nabla} \ell_j(\underline{x}_0)$ is the divided difference

$$\begin{aligned} \Big(\underline{\nabla} \ell_j(\underline{x}_0) \Big)_k &= (2\,\rho_{\rm beg})^{-1} \Big(\ell_j(\underline{x}_{k+1}) - \ell_j(\underline{x}_{k+n+1}) \Big) \\ &= (2\,\rho_{\rm beg})^{-1} \, (\delta_{k+1\,j} - \delta_{k+n+1\,j}), \quad j = 1, 2, \ldots, m, \qquad \text{(A.4)} \end{aligned}$$

because ℓ_j is a quadratic that takes the values (A.2). We replace $k+1$ by i, and then expression (A.3) gives $(\underline{\nabla} \ell_j(\underline{x}_0))_k = \Xi_{k+1\,j} = \Xi_{ij}$. It follows from equation (A.4) that formula (3.14) does provide all the nonzero elements of the i-th row of Ξ for $2 \le i \le \min[n+1, m-n]$. Finally, if k satisfies $m-n \le k \le n$, then only the first two of the vectors $\underline{x}_1 = \underline{x}_0$, $\underline{x}_{k+1} = \underline{x}_0 + \rho_{\rm beg} \underline{e}_k$ and $\underline{x}_0 - \rho_{\rm beg} \underline{e}_k$

are interpolation points. Further, the minimization of $\|\nabla^2\ell_j\|_F$ subject to the conditions (A.2) yields $(\nabla^2\ell_j)_{kk}=0$, $j=1,2,\ldots,m$, so the univariate function $\ell_j(\underline{x}_0+\alpha\underline{e}_k)$, $\alpha\in\mathcal{R}$, is a linear polynomial for each j. Therefore the $(k+1)$-th row of the matrix (A.3) contains the divided differences

$$\begin{aligned}\Xi_{k+1\,j} &= \Big(\underline{\nabla}\ell_j(\underline{x}_0)\Big)_k = (\rho_{\rm beg})^{-1}\Big(\ell_j(\underline{x}_{k+1})-\ell_j(\underline{x}_1)\Big)\\ &= (\rho_{\rm beg})^{-1}\,(\delta_{k+1\,j}-\delta_{1j}),\quad j=1,2,\ldots,m.\end{aligned}\tag{A.5}$$

Again we replace $k+1$ by i, so equation (A.5) establishes that the nonzero elements of the i-th row of Ξ have the values (3.15) when i satisfies $m-n+1\le i\le n+1$. The proof of the lemma is complete.

Lemma 2. *When $m\ge 2n+1$ holds, the initial matrix Υ is identically zero. Otherwise, Υ is a diagonal matrix, and expression (3.16) gives all the elements of Υ that are nonzero.*

Proof: Let $\tilde{m}$ be the integer $\min[m,2n+1]$, and let $\tilde{\Xi}$, $\tilde{A}$ and $\tilde{X}$ be the leading $(n+1)\times\tilde{m}$, $\tilde{m}\times(n+1)$ and $(n+1)\times(n+1)$ submatrices of Ξ, A and X, respectively. The definitions (3.12) provide the matrix equation $\Xi A+\Upsilon X=0$, and its first $n+1$ columns give the identity $\tilde{\Xi}\tilde{A}+\Upsilon\tilde{X}=0$, which depends on the property in Lemma 1 that, if $m>2n+1$, then the last $m-2n-1$ columns of Ξ are zero. We deduce from equations (3.2) and (3.11) that $\tilde{A}$ has the elements

$$\left.\begin{aligned}\tilde{A}_{ii}=A_{ii}&=\tfrac{1}{2}\,\rho_{\rm beg}^4, && i=2,3,\ldots,n+1\\ \tilde{A}_{i+n\,i}=A_{i+n\,i}&=\tfrac{1}{2}\,\rho_{\rm beg}^4, && i=2,3,\ldots,\tilde{m}-n\\ \tilde{A}_{ij}=A_{ij}&=0, && \text{otherwise}\end{aligned}\right\},\quad \begin{aligned}&i=1,2,\ldots,\tilde{m},\\ &j=1,2,\ldots,n+1.\end{aligned}\tag{A.6}$$

We seek the elements of the product $\tilde{\Xi}\tilde{A}$, which is a square matrix. For each integer j in $[1,n+1]$, equations (3.13), (3.14) and (3.15) give the formula

$$(\tilde{\Xi}\tilde{A})_{ij}=\begin{cases}\tilde{A}_{1j} & i=1,\\ (2\,\rho_{\rm beg})^{-1}\,(\tilde{A}_{ij}-\tilde{A}_{i+n\,j}), & 2\le i\le\min[n+1,m-n],\\ (\rho_{\rm beg})^{-1}\,(\tilde{A}_{ij}-\tilde{A}_{1j}), & m-n+1\le i\le n+1,\end{cases}\tag{A.7}$$

the last line being void in the case $m\ge 2n+1$. It follows from equation (A.6) that $\tilde{\Xi}\tilde{A}$ is a diagonal matrix, and that its first row and column are zero. Further, because $\min[n+1,m-n]$ is the same as $\tilde{m}-n$, we find the diagonal elements

$$\left.\begin{aligned}(\tilde{\Xi}\tilde{A})_{ii}&=0, && 1\le i\le\tilde{m}-n\\ (\tilde{\Xi}\tilde{A})_{ii}&=\tfrac{1}{2}\,\rho_{\rm beg}^3, && m-n+1\le i\le n+1\end{aligned}\right\}.\tag{A.8}$$

We now consider the identity $\tilde{\Xi}\tilde{A}+\Upsilon\tilde{X}=0$. The definition (1.3) of X with $\underline{x}_1=\underline{x}_0$ imply $\tilde{\Xi}\underline{e}_1=\underline{e}_1$, where $\underline{e}_1$ is the first coordinate vector in $\mathcal{R}^{n+1}$, and

we recall $\tilde{\Xi}\tilde{A}\underline{e}_1=0$. It follows from $(\tilde{\Xi}\tilde{A}+\Upsilon\tilde{X})\underline{e}_1=0$ that the first column of Υ is also zero. Thus $\tilde{\Xi}\tilde{A}+\Upsilon\tilde{X}=0$ remains true if any change is made to the first row of $\tilde{X}$. Expressions (1.3) and (3.2) allow the new $\tilde{X}$ to be $\rho_{\rm beg}$ times the $(n+1)\times(n+1)$ unit matrix. Hence Υ is the matrix $-\rho_{\rm beg}^{-1}\,\tilde{\Xi}\tilde{A}$, which we know is diagonal. Further, we deduce from equation (A.8) that Υ is the zero matrix in the cases $m\geq 2n+1$, and that otherwise the nonzero elements of Υ take the values (3.16). Therefore the lemma is true.

Lemma 3. *The initial matrix Ω has the factorization*

$$\Omega \;=\; \sum_{k=1}^{m-n-1} \underline{z}_k\,\underline{z}_k^T \;=\; Z\,Z^T, \tag{A.9}$$

where the vectors $\underline{z}_k\in\mathcal{R}^m$, $k=1,2,\ldots,m-n-1$, are the columns of Z. Further, the first $\min[n,m-n-1]$ of these vectors have the components (3.18), and, if $m>2n+1$, the remaining vectors have the components (3.20), the subscripts $\hat{p}$ and $\hat{q}$ being introduced in the last paragraph of Section 3.

Proof: Let $Q(\underline{x})$, $\underline{x}\in\mathcal{R}^n$, be the initial quadratic model, given at the beginning of Section 3. Each element of $\nabla^2 Q$ is either defined by the equations (1.1) or is set to zero. Therefore the choice of Q minimizes $\|\nabla^2 Q\|_F$ subject to the interpolation conditions. It follows from the derivation of the system (3.10) that, if we let $\underline{r}$ have the components $r_i=F(\underline{x}_i)$, $i=1,2,\ldots,m$, and if we set $\underline{\lambda}=\Omega\underline{r}$, where Ω is taken from expression (3.12), then $\underline{\lambda}$ is the unique vector satisfying the constraints (3.7), such that $\nabla^2 Q$ is the matrix (3.8). These remarks characterise Ω uniquely, because they are valid for all right hand sides $r_i=F(\underline{x}_i)$, $i=1,2,\ldots,m$. Hence it is sufficient to verify that, if we put the matrix (A.9) into the equation $\underline{\lambda}=\Omega\underline{r}$ for general $\underline{r}$, then $\underline{\lambda}$ has the properties that have been mentioned.

The first of the constraints (3.7) is $\underline{\lambda}^T\underline{e}=0$, where $\underline{e}\in\mathcal{R}^m$ is the vector of ones. Substituting $\underline{\lambda}=\Omega\underline{r}$ and $\Omega=ZZ^T$, this condition becomes $\underline{r}^T ZZ^T\underline{e}=0$, which is achieved, because each column $\underline{z}_k$ of Z has the components (3.18) or (3.20), and both sets of components provide $\underline{z}_k^T\underline{e}=0$. Similarly, the relation $\underline{\lambda}=ZZ^T\underline{r}$ implies that the other constraint (3.7) also holds if Z satisfies the equations

$$\sum_{i=1}^{m} Z_{ik}\,(\underline{x}_i-\underline{x}_0) \;=\; 0, \qquad k=1,2,\ldots,m-n-1. \tag{A.10}$$

For $1\leq k\leq\min[n,m-n-1]$, the values (3.18) and (3.2) imply that the left hand side of this expression is a multiple of the difference $\rho_{\rm beg}\underline{e}_k-\rho_{\rm beg}\underline{e}_k=0$. Alternatively, for $n+1\leq k\leq m-n-1$, the values (3.20), (3.19) and (3.3), with $i=k+n+1$ and $\underline{x}_1=\underline{x}_0$, give the condition

$$\sum_{i=1}^{m} Z_{ik}\,(\underline{x}_i-\underline{x}_0) = \rho_{\rm beg}^{-2}\,(-\sigma_p\,\rho_{\rm beg}\,\underline{e}_p-\sigma_q\,\rho_{\rm beg}\,\underline{e}_q+\sigma_p\,\rho_{\rm beg}\,\underline{e}_p+\sigma_q\,\rho_{\rm beg}\,\underline{e}_q)=0. \tag{A.11}$$

Thus, for general $\underline{r}\in\mathcal{R}^m$, the vector $\underline{\lambda}=\Omega\underline{r}$ does obey the constraints (3.7).

By substituting $\underline{\lambda}=ZZ^T\underline{r}$, we write the matrix (3.8) in the form

$$\nabla^2 D \;=\; \sum_{k=1}^{m-n-1} (\underline{z}_k^T\underline{r}) \left\{ \sum_{j=1}^{m} Z_{jk}\,(\underline{x}_j-\underline{x}_0)\,(\underline{x}_j-\underline{x}_0)^T \right\}, \tag{A.12}$$

and we complete the proof by establishing $\nabla^2 Q=\nabla^2 D$. For $1\le k\le \min\,[n, m-n-1]$, the components (3.18) provide the equations

$$\left.\begin{aligned} \underline{z}_k^T\underline{r} \;&=\; \sqrt{2}\,\rho_{\rm beg}^{-2}\left\{-F(\underline{x}_0)+\tfrac{1}{2}F(\underline{x}_0+\rho_{\rm beg}\,\underline{e}_k)+\tfrac{1}{2}F(\underline{x}_0-\rho_{\rm beg}\,\underline{e}_k)\right\} \\ \textstyle\sum_{j=1}^{m} Z_{jk}\,(\underline{x}_j-\underline{x}_0)\,(\underline{x}_j-\underline{x}_0)^T \;&=\; \sqrt{2}\,\rho_{\rm beg}^{-2}\left\{\rho_{\rm beg}^2\,\underline{e}_k\,\underline{e}_k^T\right\} \;=\; \sqrt{2}\,\underline{e}_k\,\underline{e}_k^T \end{aligned}\right\}. \tag{A.13}$$

Moreover, the construction in the first paragraph of this section employs the divided difference

$$(\nabla^2 Q)_{kk} \;=\; \rho_{\rm beg}^{-2}\left\{F(\underline{x}_0-\rho_{\rm beg}\,\underline{e}_k)-2\,F(\underline{x}_0)+F(\underline{x}_0+\rho_{\rm beg}\,\underline{e}_k)\right\}. \tag{A.14}$$

It follows that the first $\min[n, m-n-1]$ terms of the sum over k in expression (A.12) provide a diagonal matrix, whose diagonal elements are the same as those of $\nabla^2 Q$. Thus $\nabla^2 Q=\nabla^2 D$ is achieved in the cases $m\le 2n+1$. It remains to show that, if $m>2n+1$, then the last $m-2n-1$ values of k in expression (A.12) generate the off-diagonal elements of $\nabla^2 Q$ without disturbing the diagonal elements.

For each k in the interval $[n+1, m-n-1]$, the interpolation points (3.3) and (3.19) are relevant with $i=k+n+1$. Indeed, the components (3.20) imply that $\underline{z}_k^T\underline{r}$ is just the left hand side of equation (3.5), while the term in the braces of expression (A.12) is the matrix

$$-\underline{e}_p\,\underline{e}_p^T-\underline{e}_q\,\underline{e}_q^T+(\sigma_p\,\underline{e}_p+\sigma_q\,\underline{e}_q)\,(\sigma_p\,\underline{e}_p+\sigma_q\,\underline{e}_q)^T \;=\; \sigma_p\,\sigma_q\,(\underline{e}_p\,\underline{e}_q^T+\underline{e}_q\,\underline{e}_p^T). \tag{A.15}$$

Therefore the k-th term of the sum (A.12) contributes to $(\nabla^2 D)_{pq}$ and $(\nabla^2 D)_{qp}$ the amount that is required by equation (3.5), and it does not alter any other element of $\nabla^2 D$. Thus all the different elements of $\nabla^2 Q$ that can be nonzero are provided by the different values of k in expression (A.12). The justification of the initial choice of Z is complete.

Acknowledgements

The author is very grateful for the facilities he has enjoyed, throughout the development of NEWUOA, as an Emeritus Professor at the Centre for Mathematical Sciences of the University of Cambridge. He has also received excellent support for this research from the City University of Hong Kong and from the University of Minnesota. The first numerical experiments on the given method for updating Q were run during a two month stay in Hong Kong, and the investigations of several auxiliary techniques were helped greatly by discussions with other visitors during the IMA Program on Optimization in Minneapolis.

References

[Buckley (1989)] A.G. Buckley, Test functions for unconstrained minimization, Technical Report CS-3, Dalhousie University, Canada, (1989).

[Conn et al. (1994)] A.R. Conn, N.I.M. Gould, M. Lescrenier and Ph.L. Toint, Performance of a multifrontal scheme for partially separable optimization, in *Advances in Optimization and Numerical Analysis*, eds. Susana Gomez and Jean-Pierre Hennart, Kluwer Academic (Dordrecht), pp. 79–96, (1994).

[Conn, Gould and Toint (2000)] A.R. Conn, N.I.M. Gould and Ph.L. Toint, *Trust-Region Methods*, MPS–SIAM Series on Optimization (Philadelphia), (2000).

[Dennis and Schnabel (1983)] J.E. Dennis and R.B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice Hall (Englewood Cliffs), (1983).

[Fletcher and Reeves (1964)] R. Fletcher and C.M. Reeves, Function minimization by conjugate gradients, *Computer J.*, Vol. 7, pp. 149–154, (1964).

[Powell (2001)] M.J.D. Powell, On the Lagrange functions of quadratic models that are defined by interpolation, *Optim. Meth. Software*, Vol. 16, pp. 289–309, (2001).

[Powell (2002)] M.J.D. Powell, UOBYQA: unconstrained optimization by quadratic approximation, *Math. Programming*, Vol. 92, pp. 555–582, (2002).

[Powell (2003)] M.J.D. Powell, On trust region methods for unconstrained minimization without derivatives, *Math. Programming*, Vol. 97, pp. 605–623, (2003).

[Powell (2004a)] M.J.D. Powell, Least Frobenius norm updating of quadratic models that satisfy interpolation conditions, *Math. Programming*, Vol. 100, pp. 183–215, (2004).

[Powell (2004b)] M.J.D. Powell, On the use of quadratic models in unconstrained minimization without derivatives, *Optim. Meth. Software*, Vol. 19, pp. 399–411, (2004).

[Powell (2004c)] M.J.D. Powell, On updating the inverse of a KKT matrix, in *Numerical Linear Algebra and Optimization*, ed. Ya-xiang Yuan, Science Press (Beijing), pp. 56–78, (2004).