Springer Proceedings in Mathematics & Statistics

Mehiddin Al-Baali
Anton Purnama
Lucio Grandinetti *Editors*

Numerical Analysis and Optimization

NAO-V, Muscat, Oman, January 2020

Springer

Springer Proceedings in Mathematics & Statistics

Volume 354

Mehiddin Al-Baali · Anton Purnama · Lucio Grandinetti
Editors

Numerical Analysis and Optimization

NAO-V, Muscat, Oman, January 2020

Springer

Editors
Mehiddin Al-Baali
Department of Mathematics
Sultan Qaboos University
Muscat, Oman

Anton Purnama
Department of Mathematics
Sultan Qaboos University
Muscat, Oman

Lucio Grandinetti
DIMES
University of Calabria
Arcavacada di Rende, Cosenza, Italy

ISSN 2194-1009 ISSN 2194-1017 (electronic)
Springer Proceedings in Mathematics & Statistics
ISBN 978-3-030-72039-1 ISBN 978-3-030-72040-7 (eBook)
https://doi.org/10.1007/978-3-030-72040-7

Mathematics Subject Classification (2020): 49Mxx, 65-XX, 35-XX, 34-XX, 49-XX, 65-XX, 68Nxx, 90-XX, 93-XX, 97Cxx, 97Nxx, 97Pxx

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

This special edited book series of Springer Proceedings in Mathematics and Statistics contains 12 selected keynote papers presented at the *Fifth International Conference on Numerical Analysis and Optimization: Theory, Methods, Applications and Technology Transfer* (NAOV-2020) held during January 6–9, 2020, at Sultan Qaboos University (SQU), Muscat, Oman. The NAOV-2020 conference was sponsored by SQU, AMPL (USA), and Weierstrass Institute for Applied Analysis and Stochastics (WIAS, Germany). Twenty-four world leading researchers gave keynote lectures. In total, forty-three international participants contributed talks. After the conference, selected contributed papers were invited to be submitted for publication in a special issue of the following international journals: *Optimization Methods and Software*, Springer Nature *Operations Research Forum* and *SQU Journal for Science*. More information on the conference is available at the Website conferences.squ.edu.om/naov-2020/Home. Each of the keynote papers was accepted for this edited proceedings volume after a stringent peer review process by independent reviewers. We wish to express our gratitude to all contributors. We are also indebted to many anonymous referees for the care taken in reviewing the papers submitted for publication.

The NAO conference series is held once every 3 years at SQU: the first conference (NAO-2008) was held on April 6–8, 2008; the second conference (NAOII-2011) was held on January 3–6, 2011; the third conference (NAOIII-2014) was held on January 5–9, 2014; and the fourth conference (NAOIV-2017) was held on January 2–5, 2017. The NAO conference will hopefully remain a forum where prominent mathematicians, worldwide experts, and active researchers gather and meet to share their knowledge on new scientific methodologies and stimulate the communication of new innovative ideas, promote scientific exchange, and discuss possibilities of further cooperation, networking, and promotion of mobility of senior and young researchers and research students.

For the previous NAOIV-2017 conference, a total of 13 keynote papers were published in an edited book of Springer series on *Proceedings in Mathematics and Statistics* (volume 235, 2018). This volume is dedicated to the late Profs. Mike JD Powell and Roger Fletcher (passed away on 2015 and on 2016, respectively), who were the pioneers and leading figures in the mathematics of nonlinear optimization.

Seven papers were published in the volume 23(1) for the (2018) special issue of the *SQU Journal for Science* (free download from the Website https://journals.squ.edu.om/index.php/squjs/issue/view/197).

For the past NAOIII-2014 conference, a total of 13 keynote papers were published in an edited book of Springer series on *Proceedings in Mathematics and Statistics* (volume 134, 2015), and eight papers were published in the volume 20(2) for the (2015) special issue of the *SQU Journal for Science* (free download from the Website https://journals.squ.edu.om/index.php/squjs/issue/view/50).

For the past NAOII-2011 conference, nineteen papers were selected for two special issues of the *SQU Journal for Science* highlighting the two themes of the conference *Numerical Optimization* and *Numerical Analysis*. Eleven papers were published in the volume 17(1) for the (2012) special issue on *Numerical Optimization* (free download from the Website https://journals.squ.edu.om/index.php/squjs/issue/view/44) and eight papers in the volume 17(2) for the (2012) special issue on *Numerical Analysis* (free download from the Website https://journals.squ.edu.om/index.php/squjs/issue/view/45).

For the first NAO-2008 conference, Mike Powell was the first plenary speaker.

Muscat, Oman	Mehiddin Al-Baali
Muscat, Oman	Anton Purnama
Rende, Cosenza, Italy	Lucio Grandinetti

A Personal Perspective on Numerical Analysis and Optimization

Desmond J. Higham
School of Mathematics, University of Edinburgh, Edinburgh, UK

Summary I give a brief, non-technical, historical perspective on numerical analysis and optimization. I also touch on emerging trends and future challenges. This content is based on the short presentation that I made at the opening ceremony of *The International Conference on Numerical Analysis and Optimization*, which was held at Sultan Qaboos University, Muscat, Oman, on January 6–9, 2020. Of course, the material covered here is necessarily incomplete and biased towards my own interests and comfort zones. My main aim is to give a feel for how the area has developed over the past few decades and how it may continue. I hope that this material will also provide some context that motivates and complements the research described in the main body of the conference proceedings.

1. Definitions

Mathematicians love to make definitions. But defining an area of mathematics is a thankless task. The best one-sentence definitions that I can come up with for numerical analysis and optimization are as follows.

Numerical Analysis: the design, analysis, and implementation of *computational algorithms* to deliver approximate solutions to problems arising in *applied mathematics.*

Optimization: the design, analysis, and implementation of *computational algorithms* to approximate the *best solution* to a problem arising in *applied mathematics* when there may be *many feasible solutions.*

For alternative versions, I refer to the references [3, 24, 29].

2. Emergence

The unstoppable growth of interest in the use of computational algorithms can be attributed to two main factors. First, technology has advanced rapidly and consistently since the digital computing era began in the 1950s. The CDC6600, widely acknowledged to be the world's first "supercomputer", was introduced in 1964, achieving a speed of 3 megaflops (that is, 3×10^6 floating operations per second) [17]. Today's fastest supercomputers can achieve petaflop speeds (10^{15} floating operations per second). By contrast, in his 1970 Turing Award Lecture [28, 1971], James Wilkinson discussed the use of mechanical desk calculators:

> "It happened that some time after my arrival [at the National Physical Laboratory in 1946], a system of 18 equations arrived in Mathematics Division and after talking around it for some time we finally decided to abandon theorizing and to solve it... The operation was manned by Fox, Goodwin, Turing, and me, and we decided on Gaussian elimination with complete pivoting."

Leslie Fox [12] noted that the computation referred to in this quotation took about two weeks to complete. By my estimation, this corresponds to around 0.003 floating operations per second!

A second, and equally important, factor behind the rise of scientific computation is the availability of ever-increasing sources of data, caused by improvements in experimental techniques and, perhaps most notably, by the inexorable sensorization and digitization of our everyday lives. Hence, although numerical analysis and optimization build on classical ideas that can be attributed to the likes of Newton, Euler, Lagrange, and Gauss, they continue to be shaped by current developments. Of course, many disciplines make extensive use of computational techniques. For example, the word "Computational" often appears before the words Biology, Chemistry, Physics, and Social Science. Furthermore, Computational Science and Engineering [23] is a well-established discipline that is often referred to as the third leg of the science and engineering stool, equal in stature to observation and theory. In addition, many graduate schools now offer courses with titles such as "Data Analytics" and "Data Science." Although there is clearly much overlap, my view is that numerical analysis and optimization have a distinct role of focusing on the *design* and *analysis* of algorithms for problems in applied mathematics, in terms of complexity, accuracy, and stability, and hence they are informed by, but not driven by, application fields.

3. Reflections

My own exposure to numerical analysis and optimization dates back to the mid 1980s when I enrolled on an MSc course on "Numerical Analysis and Programming" at the University of Manchester. The course, in which optimization was treated as a branch of numerical analysis, made heavy use of the series of textbooks published by Wiley that was written by members of the highly influential numerical analysis group at the University of Dundee [27]. Here are the topics, and most recent versions of these books: approximation theory [26], numerical methods for ODEs [18], numerical methods for PDEs [22, 25], and optimization [11]. An additional topic was numerical linear algebra [14]. Each of these topics remains active and highly relevant. Numerical linear algebra and optimization are often important building blocks within larger computational tasks, and hence their popularity has never waned. Partial differential equations lie at the heart of most models in the natural and engineering sciences, and they come in many varieties, giving rise to an ever-expanding problem set. Timestepping methods for ODEs gained impetus through the concept of geometric integration [15] and now play a prominent role in the development of tools for statistical sampling [6]. ODE simulation also forms a key component in certain classes of neural network, as described in [7], which received a Best Paper Award at NeurIPS 2018, a leading conference in machine learning. Approximation theory lies at the heart of the current deep learning revolution [16, 19], and, in particular, understanding very high-dimensional data spaces and/or parameter spaces remains a fundamental challenge.

4. Impact

In a special issue of the journal Computing in Science and Engineering, Jack Dongarra and Francis Sullivan published their top-ten list of algorithms that had the "greatest influence on the development and practice of science and engineering in the 20th century" [9]. These were as follows:

- Metropolis Algorithm for Monte Carlo.
- Simplex Method for Linear Programming.
- Krylov Subspace Iteration Methods.
- The Decompositional Approach to Matrix Computations.
- The Fortran Optimizing Compiler.
- QR Algorithm for Computing Eigenvalues.
- Quicksort Algorithm for Sorting.
- Fast Fourier Transform.
- Integer Relation Detection.
- Fast Multipole Method.

Here, the word "algorithm" is being used in a very general sense, but it is clear that most of these achievements have ideas from numerical analysis and optimization at their heart.

Researchers in numerical analysis and optimization are in the advantageous position that their work is not only recorded in journals and textbooks, but may also be made operational through public domain software. Many authors now deposit code alongside their academic publications, and state-of-the-art code is available in a wide range of languages and platforms, including FORTRAN, C, R, MATLAB, Maple, (Scientific) Python, and the more recent Julia [4].

5. Momentum

Judging by the level of activity around graduate classes, seminar series, conferences, and journals, there is a strong pull for further research in numerical analysis and optimization. Particularly active, and overlapping, directions include

- dealing with, or exploiting, randomness, both in the simulation of mathematical models that are inherently stochastic [20] and in the use of randomization in solving deterministic problems [8, 21],
- accurately and efficiently simulating mathematical models that operate over a vast range of temporal or spatial scales [10, 13],

- tackling problems of extremely high dimension, notably large-scale optimization and inverse problems in machine learning and imaging [2, 5], and
- exploiting the latest computer architectures and designing algorithms that efficiently trade off between issues such as memory bandwidth, data access, communication, and, perhaps most topically, the use of low precision special function units [1].

Such items, and may others, emphasize that important challenges remain for researchers in numerical analysis and optimization in the design, evaluation, and extension of the modern computational scientist's toolbox.

Acknowledgements The author is supported by Programme Grant EP/P020720/1 from the Engineering and Physical Sciences Research Council of the UK.

References

1. Abdelfattah, H. Anzt, E. G. Boman, E. Carson, T. Cojean, J. Dongarra, M. Gates, T. Grützmacher, N. J. Higham, S. Li, N. Lindquist, Y. Liu, J. Loe, P. Luszczek, P. Nayak, S. Pranesh, S. Rajamanickam, T. Ribizel, B. Smith, K. Swirydowicz, S. Thomas, S. Tomov, Y. M. Tsai, I. Yamazaki, and U. M. Yang, *A survey of numerical methods utilizing mixed precision arithmetic*, The International Journal of High Performance Computing Applications, 35 (2021), pp. 344–369.
2. S. Arridge, P. Maass, O. Öktem, and C.-B. Schönlieb, *Solving inverse problems using data-driven models*, Acta Numerica, 28 (2019), pp. 1–174.
3. J. Barrow-Green and R. Siegmund-Schultze, *The history of applied mathematics*, in The Princeton Companion to Applied Mathematics, N. J. Higham, M. R. Dennis, P. Glendinning, P. A. Martin, F. Santosa, and J. Tanner, eds., Princeton University Press, Princeton, NJ, USA, 2015, pp. 55–79.
4. J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, *Julia: A fresh approach to numerical computing*, SIAM Review, 59 (2017), pp. 65–98.
5. L. Bottou, F. E. Curtis, and J. Nocedal, *Optimization methods for largescale machine learning*, SIAM Review, 60 (2018), pp. 223–311.
6. S. Brooks, A. Gelman, G. L. Jones, and X.-L. Meng, eds., *Handbook of Markov Chain Monte Carlo*, CRC Press, Boca Raton, 2011.
7. R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, *Neural ordinary differential equations*, in Advances in Neural Information Processing Systems 31, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, eds., Curran Associates, Inc., 2018, pp. 6571–6583.
8. M. P. Connolly, N. J. Higham, and T. Mary, *Stochastic rounding and its probabilistic backward error analysis*, SIAM J. Sci. Comput., 43 (2021), pp. A566–A585.
9. J. Dongarra and F. Sullivan, *Guest editors' introduction to the top 10 algorithms*, Computing in Science and Engineering, 2 (2000), pp. 22–23.
10. W. E, *Principles of Multiscale Modeling*, Cambridge University Press, Cambridge, 2011.
11. R. Fletcher, *Practical Methods of Optimization*, John Wiley and Sons, Chichester, 2000.
12. L. Fox, *James Hardy Wilkinson, 1919–1986*, Biographical Memoirs of Fellows of the Royal Society, 33 (1987), pp. 671–708.
13. M. G. D. Geers, V. G. Kouznetsova, K. Matouš, and J. Yvonnet, *Homogenization methods and multiscale modeling: Nonlinear problems*, in Encyclopedia of Computational Mechanics, Second Edition, Wiley, 2017, pp. 1–34.
14. G. H. Golub and C. F. Van Loan, *Matrix Computations*, The Johns Hopkins University Press, 3rd ed., 1996.

15. E. Hairer, C. Lubich, and G. Wanner, *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*, Springer, Berlin, 2nd ed., 2006.
16. C. F. Higham and D. J. Higham, *Deep learning: An introduction for applied mathematicians*, SIAM Review, 61 (2019), pp. 860–891.
17. S. Hongwei, *Seymour Cray: The father of world supercomputer*, History Research, 7 (2019), pp. 1–6.
18. J. D. Lambert, *Numerical Methods for Ordinary Differential Systems: The Initial Value Problem*, John Wiley and Sons, Chichester, 1991.
19. Y. LeCun, Y. Bengio, and G. Hinton, *Deep learning*, Nature, 521 (2015), pp. 436–444.
20. G. J. Lord, C. E. Powell, and T. Shardlow, *An Introduction to Computational Stochastic PDEs*, Cambridge University Press, Cambridge, 2014.
21. P.-G. Martinsson and J. Tropp, *Randomized numerical linear algebra: Foundations and algorithms*, Acta Numerica, 29 (2020), pp. 403–572.
22. A. R. Mitchell and D. F. Griffiths, *The Finite Difference Method in Partial Differential Equations*, John Wiley and Sons, Chichester, 1980.
23. U. Rüde, K. Willcox, L. C. McInnes, H. D. Sterck, G. Biros, H. Bungartz, J. Corones, E. Cramer, J. Crowley, O. Ghattas, M. Gunzburger, M. Hanke, R. Harrison, M. Heroux, P. J. Jan Hesthaven, C. Johnson, K. E. Jordan, D. E. Keyes, R. Krause, V. Kumar, S. Mayer, J. Meza, K. M. Mørken, J. T. Oden, L. Petzold, P. Raghavan, S. M. Shontz, A. Trefethen, P. Turner, V. Voevodin, B. Wohlmuth, and C. S. Woodward, *Research and education in computational science and engineering*, SIAM Review, 60 (2018), pp. 707–754.
24. L. N. Trefethen, *The definition of numerical analysis*, SIAM News, 25 (1992).
25. R. Wait and A. Mitchell, *Finite Element Analysis and Applications*, John Wiley and Sons, Chichester, 1985.
26. G. A. Watson, *Approximation Theory and Numerical Methods*, John Wiley and Sons, Chichester, 1980.
27. G. A. Watson, *The history and development of numerical analysis in Scotland: a personal perspective*, in The Birth of Numerical Analysis, World Scientific, London, 2009, pp. 161–177.
28. J. H. Wilkinson, *Some comments from a numerical analyst*, J. Assoc. Comput. Mach., 18 (1971), pp. 137–147.
29. S. J. Wright, *Continuous optimization (nonlinear and linear programming)*, in The Princeton Companion to Applied Mathematics, N. J. Higham, M. R. Dennis, P. Glendinning, P. A. Martin, F. Santosa, and J. Tanner, eds., Princeton University Press, Princeton, NJ, USA, 2015, pp. 281–293.

A Personal View of Numerical Analysis and Optimization

David M. Gay
AMPL Optimization

Now and then I need to fill out a form on which I must specify my occupation. I always say "numerical analyst". My introduction to numerical analysis came in a couple of college courses and a stint at the numerical summer school that used to be run at the University of Michigan. When it came time for graduate studies, I sought places where I could learn more numerical analysis, as I liked the mix of practicalities one must deal with in that discipline. I ended up doing my graduate work in Computer Science at Cornell University. At that time the Computer Science Department had three numerical professors, Jim Bunch, John Dennis, and Jorge Moré—an emphasis on numerical linear algebra and optimization. I took courses or had other interactions with all three and ended up with John Dennis as my thesis advisor. There was also a lot of work on algorithms at that time at Cornell, and I absorbed much material on algorithms.

Since graduate school, I've been interested in practical scientific computing. My optimization interests for a while were drawn to data fitting, starting with work with John Dennis and Roy Welsch on NL2SOL, though I was also interested in good linear algebra for the simplex method. When Bob Fourer spent a sabbatical at Bell Labs, he, Brian Kernighan and I came up with the AMPL modeling language, on which I have since spent much time. I was aware of forward automatic differentiation via the AUGMENT pre-processor and thought of implementing forward AD in the interface to nonlinear solvers, but when I explained this to Andreas Griewank at a Mathematical Programming Symposium, he told me about backwards AD, which is much more efficient for gradient computations, and I ended up writing several AD papers and providing a backwards AD implementation in the AMPL/Solver Interface Library (ASL).

In general I am still drawn to practical scientific computing. I like ideas of using good algorithms for practical scientific computations.

Contents

Contributors

Hani Ahmadzadeh Faculty of Mathematical Sciences, Sharif University of Technology, Tehran, Iran

Azam Asl University of Chicago Booth School of Business, Chicago, IL, USA

A. M. Bagirov School of Engineering, Information Technology and Physical Sciences, Federation University Australia, Ballarat, Australia

Yanqin Bai Department of Mathematics, Shanghai University, Shanghai, China

Yu-Hong Dai State Key Laboratory of Scientific and Engineering Computing, Institute of Computational Mathematics and Scientific/Engineering Computing, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, China

Francisco Facchinei Department of Computer, Control, and Management Engineering Antonio Ruberti, Sapienza University of Rome, Rome, Italy

Craig Gilmour Department of Mathematics and Statistics, University of Strathclyde, Glasgow, UK

Desmond J. Higham School of Mathematics, University of Edinburgh, Edinburgh, UK

Yakui Huang Institute of Mathematics, Hebei University of Technology, Tianjin, China

Vyacheslav Kungurtsev Department of Computer Science, Faculty of Electrical Engineering, Czech Technical University in Prague, Prague, Czech Republic

Lorenzo Lampariello Department of Business Studies, Roma Tre University, Rome, Italy

Xin-Wei Liu Institute of Mathematics, Hebei University of Technology, Tianjin, China

Ding Ma Department of Management Science and Department of Marketing, College of Business, City University of Hong Kong, Kowloon, Hong Kong

Nezam Mahdavi-Amiri Faculty of Mathematical Sciences, Sharif University of Technology, Tehran, Iran

Giovanni Micheli Department of Management, Information and Production Engineering, University of Bergamo, Dalmine, BG, Italy

Dominique Orban GERAD and Department of Mathematics and Industrial Engineering, Ecole Polytechnique de Montréal, Montreal, QC, Canada

Michael L. Overton Courant Institute of Mathematical Sciences, New York University, New York, USA

G. Ozturk Department of Industrial Engineering, Eskisehir Technical University, Eskisehir, Turkey

Noémi Petra Department of Applied Mathematics, University of California, Merced, CA, USA

János D. Pintér Department of Management Science and Information Systems, Rutgers University, Piscataway, NJ, USA

Kees Roos Department of Electrical Engineering, Mathematics and Computer Science, Technical University Delft, CD Delft, The Netherlands

Ekkehard W. Sachs Department of Mathematics, Trier University, Trier, Germany

Michael A. Saunders Systems Optimization Laboratory, Department of Management Science and Engineering, Stanford University, Stanford, CA, USA

Gesualdo Scutari School of Industrial Engineering, Purdue University, West-Lafayette, IN, USA

Trond Steihaug Department of Informatics, University of Bergen, Bergen, Norway

N. Sultanova School of Engineering, Information Technology and Physical Sciences, Federation University Australia, Ballarat, Australia

S. Taheri School of Science, RMIT University, Melbourne, Australia

Maria Teresa Vespucci Department of Management, Information and Production Engineering, University of Bergamo, Dalmine, BG, Italy

A New Inexact Nonmonotone Filter Sequential Quadratic Programming Algorithm

Hani Ahmadzadeh and Nezam Mahdavi-Amiri

Abstract An inexact nonmonotone filter sequential quadratic programming algorithm is presented for solving general constrained nonlinear programming problems. At every iteration, a steering direction is computed as a minimizer of a linear model of the constraint violation over a trust region. A possible infeasible stationary point can be detected using the steering direction. If the current iterate is not an infeasible stationary point, a strongly convex feasible quadratic programming subproblem is defined to compute a search direction as an inexact solution satisfying some loose and achievable conditions. We prove that the search direction is a descent direction for the constraint violation or the objective function. Moreover, we use a penalty parameter updating strategy to yield the search direction as a descent direction for the penalty function. To attain a superlinear local convergence, an accelerating direction is computed in certain iterations. We use a nonmonotone filter strategy to compute the step-length along the accelerating direction or the search direction. The global convergence and superlinear local convergence of the algorithm can be obtained under some standard assumptions. Competitive numerical results obtained by an implementation of our algorithm are reported.

Keywords Nonlinear optimization · Sequential quadratic programming · Inexact methods · Nonmonotone filter · Penalty function

H. Ahmadzadeh · N. Mahdavi-Amiri (✉)
Faculty of Mathematical Sciences, Sharif University of Technology, Tehran, Iran
e-mail: nezamm@sharif.edu

H. Ahmadzadeh
e-mail: hani.ahmadzadeh@gmail.com

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*, Springer Proceedings in Mathematics & Statistics 354, https://doi.org/10.1007/978-3-030-72040-7_1

1 Introduction

Nonlinear optimization problems appear in various areas of science and engineering, such as finance [4, 21], economics [17], optimal control [7, 30, 41], process engineering [8, 42, 44] and many other branches of engineering [5, 15]. Sequential quadratic programming (SQP) comprises a well-known class of algorithms for solving nonlinear optimization problems [22], specially when a sequence of related problems such as mixed integer nonlinear programming [19, 34, 35], optimal control [3], and partial differential equations (PDE)-constrained optimization [9, 31] need to be solved.

Despite the many advantages of SQP algorithms including the warm-start ability and fast detection of infeasibility, they often suffer from the heavy computational cost in every iteration. This major disadvantage of SQP methods served to be a key motivation for researchers to develop inexact/truncated SQP algorithms [16, 39]. A recent foray into this area is the algorithm proposed by Curtis et al. [14]. The most striking characteristic of their algorithm is the computation of the search direction as a convex combination of inexact solutions of two quadratic programming (QP) subproblems: "penalty QP" and "feasibility QP". The authors provided a set of complicated but loose conditions for inexact solutions of QP subproblems so that the algorithm is globally convergent. Nevertheless, they could not provide a reliable implementation of the inexact SQP algorithm, because the available QP solvers at the time were not capable of finding an inexact solution of the QP subproblem satisfying the desired conditions.

Commonly, filter [20, 43] or merit functions [13, 28] are used in the context of line-search or trust region framework to guarantee the global convergence of SQP algorithms. An undesirable phenomenon in the traditional filter methods is the need for use of a restoration phase when infeasible subproblems are faced. Gould et al. [24] replaced the restoration phase in traditional filter methods with a penalty mode to reduce infeasibility, thereby obtaining a new filter line-search method for solving nonlinear programs. Aiming to prevent the occurrence of the Maratos effect [38] and loss of superlinear local convergence, they proposed a nonmonotone version of their algorithm [25]. On theoretical front, the global and local superlinear rate of convergence of the nonmonotone algorithm were shown; on the practical side, the reliability and efficiency of the algorithm were shown on some small- and medium-scale test problems.

Here, we propose an inexact nonmonotone filter SQP algorithm, based upon the ideas of the algorithms proposed in [14, 24, 25]. In every iteration, we compute a steering direction by solving a linear optimization problem. Using the optimal solution of the linear problem, a possible infeasible stationary point is figured out. If the current iterate is not an infeasible stationary point, a feasible strongly convex QP subproblem is built. The search direction is computed using an inexact solution of the QP subproblem satisfying some loose and achievable conditions. Solving the QP subproblems inexactly turns to reduce the computational cost of the algorithm significantly, without adversely affecting either the global convergence or the rate of convergence of the nonmonotone SQP algorithm, and thus the superlinear rate of convergence is preserved.

The remainder of our work is organized as follows. Section 2 provides some preliminaries needed for describing our algorithm. In Sect. 3, a brief overview of the nonmonotone filter SQP algorithm of Gould et al. [24, 25] is given. The process of computing the search direction as an inexact solution of a feasible strongly convex QP subproblem and complete description of the algorithm are presented in Sect. 4. We report some encouraging comparative numerical results in Sect. 5. Finally, we conclude in Sect. 6.

Notations. Here, $\mathbb{R}$ denotes the set of real numbers and $\bar{\mathbb{R}} := \mathbb{R} \cup \{\pm\infty\}$ is the extended real number system. The jth component of a vector $x \in \bar{\mathbb{R}}^n$ is shown by $[x]_j$. For two vectors $a, b \in \bar{\mathbb{R}}^n$, the notation $a \le b$ means that $[a]_j \le [b]_j$, for all $j = 1, \dots, n$, $\min\{a, b\}$ is an n-vector with the jth component being $\min\{[a]_j, [b]_j\}$, and the n-vector $\max\{a, b\}$ is defined similarly. Moreover, $[\cdot]^+ = \max\{\cdot, 0\}$ and $[\cdot]^- = \max\{-\cdot, 0\}$. The ℓ_p-norm is denoted by $\|\cdot\|_p$, for $p = 1, 2, \infty$, and $\mathbf{1}_n$ and $\mathbf{0}_n$ are the n-vectors with all entries being one and zero, respectively. A superscript is used to declare an iteration number; for example, $x^{(k)}$ is the kth iterate, or $f^{(k)}$ is the value of a function f at a point $x^{(k)}$.

2 Preliminaries

Consider the general nonlinear optimization problem,

$$\begin{aligned} \min_x \quad & f(x) \\ \text{s.t.} \quad & c_l \le c(x) \le c_u, \\ & x_l \le x \le x_u, \end{aligned} \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ and $c : \mathbb{R}^n \to \mathbb{R}^m$ are twice continuously differentiable functions, $c_l, c_u \in \bar{\mathbb{R}}^m$ and $x_l, x_u \in \bar{\mathbb{R}}^n$ are constant vectors so that $c_l \le c_u$ and $x_l < x_u$. The ith major constraint is an equality constraint if $[c_l]_i = [c_u]_i$, is unbounded from below if $[c_l]_i = -\infty$, and is unbounded from above if $[c_u]_i = \infty$. Similarly, the jth decision variable $[x]_j$ is unbounded from below (from above) if $[x_l]_j = -\infty$ ($[x_u]_j = \infty$). The measure of constraint violation is defined as follows:

$$h(x) := \|[c(x) - c_l]^-\|_1 + \|[c(x) - c_u]^+\|_1 + \|[x - x_l]^-\|_1 + \|[x - x_u]^+\|_1.$$

Obviously, a point $x \in \mathbb{R}^n$ satisfies all the constraints of problem (1), or simply is a feasible point of problem (1), if and only if $h(x) = 0$. The Lagrangian function related to (1) is defined to be

$$\mathcal{L}(x, y_l, y_u, z_l, z_u) := f(x) - y_l^T(c(x) - c_l) - y_u^T(c_u - c(x)) - z_l^T(x - x_l) - z_u^T(x_u - x),$$

where $y_l \in \mathbb{R}^m$, $y_u \in \mathbb{R}^m$, $z_l \in \mathbb{R}^n$ and $z_u \in \mathbb{R}^n$ are the Lagrange multiplier vectors of the constraints $c(x) \geq c_l$, $c(x) \leq c_u$, $x \geq x_l$ and $x \leq x_u$, respectively. If $[c_l]_i = -\infty$ or $[c_u]_i = \infty$, then the related term is omitted in the Lagrangian function by setting $[y_l]_i = 0$ or $[y_u]_i = 0$. Similarly, the jth component of z_l (z_u) is fixed to be zero whenever $[x_l]_j = -\infty$ ($[x_u]_j = \infty$).

The KKT conditions for problem (1) are:

$$\nabla_x \mathcal{L}(x, y_l, y_u, z_l, z_u) = \mathbf{0}_n, \tag{2a}$$

$$c(x) - c_l \geq \mathbf{0}_m, \quad y_l \geq \mathbf{0}_m, \tag{2b}$$

$$c_u - c(x) \geq \mathbf{0}_m, \quad y_u \geq \mathbf{0}_m, \tag{2c}$$

$$x - x_l \geq \mathbf{0}_n, \quad z_l \geq \mathbf{0}_n, \tag{2d}$$

$$x_u - x \geq \mathbf{0}_n, \quad z_u \geq \mathbf{0}_n, \tag{2e}$$

$$y_l^T (c(x) - c_l) = 0, \tag{2f}$$

$$y_u^T (c_u - c(x)) = 0, \tag{2g}$$

$$z_l^T (x - x_l) = 0, \tag{2h}$$

$$z_u^T (x_u - x) = 0. \tag{2i}$$

Considering $g(x) := \nabla f(x)$ as the gradient of the objective function and $J(x) := \nabla^T c(x)$ as the Jacobian matrix of the function $c(x)$, we have

$$\nabla_x \mathcal{L}(x, y_l, y_u, z_l, z_u) = g(x) - J(x)^T (y_l - y_u) - (z_l - z_u).$$

So, the condition (2a) implies $z = g(x) - J(x)^T y$, where $z := z_l - z_u$ and $y := y_l - y_u$. Consequently, from the complementary slackness conditions (2f)–(2i) and non-negativity of the vectors y_l, y_u, z_l and z_u, we have $y_l = [y]^+$, $y_u = [y]^-$, $z_l = [z]^+$ and $z_u = [z]^-$. Hence, the KKT conditions (2) can be compactly written as follows:

$$\Psi(x, y) := \begin{bmatrix} \min\{c_u - c(x), [y]^-\} \\ \min\{c(x) - c_l, [y]^+\} \\ \min\{x_u - x, [\nabla_x \mathcal{L}_0(x, y)]^-\} \\ \min\{x - x_l, [\nabla_x \mathcal{L}_0(x, y)]^+\} \end{bmatrix} = \begin{bmatrix} \mathbf{0}_m \\ \mathbf{0}_m \\ \mathbf{0}_n \\ \mathbf{0}_n \end{bmatrix}, \tag{3}$$

where $\mathcal{L}_0(x, y) := f(x) - y^T c(x)$ is the reduced Lagrangian function corresponding to (1). The vector $\Psi(x, y)$ is the KKT residual of (1) at a primal-dual pair (x, y). If $\Psi(x, y) = \mathbf{0}_{2m+2n}$, then (x, y) is called a KKT pair and x is called a KKT point of problem (1).

We say that x is a stationary point of h if and only if all of its directional derivatives are non-negative, that is,

$$D_p h(x) := \lim_{t \downarrow 0} \frac{h(x + tp) - h(x)}{t} \geq 0, \quad \forall p \in \mathbb{R}^n.$$

If x is a stationary point of the constraint violation function and $h(x) > 0$, then x is called an infeasible stationary point (see [40, Definition 17.1]), and the nonlinear optimization problem (1) is said to be locally infeasible at the point x [13]. Infeasibility is an important feature to be detected by nonlinear programming algorithms [10, 12], specially when used in branch and bound methods for solving mixed integer nonlinear programs [35].

The ℓ_1-penalty function is denoted by $P(x; \mu) := \mu f(x) + h(x)$, for a penalty parameter $\mu > 0$. It is known (e.g., see [6]) that if some suitable assumptions are held and the penalty parameter μ is chosen to be small enough, then any local solution of (1) is a local minimizer of $P(x; \mu)$. Furthermore, if x is a stationary point of $P(x; \mu)$, for all $\mu > 0$ small enough, then x is a KKT point or an infeasible stationary point, whenever $h(x) = 0$ or $h(x) > 0$, respectively (see [40, Theorem 17.4]).

Linear models of the functions f, h, and P at a point x, along a direction d, are defined by

$$\bar{f}(d; x) := f(x) + d^T g(x), \tag{4a}$$

$$\begin{aligned} \bar{h}(d; x) := & \|[c(x) + J(x)d - c_l]^-\|_1 + \|[c(x) + J(x)d - c_u]^+\|_1 \\ & + \|[x + d - x_l]^-\|_1 + \|[x + d - x_u]^+\|_1, \end{aligned} \tag{4b}$$

$$\bar{P}(d; x, \mu) := \mu \bar{f}(d; x) + \bar{h}(d; x). \tag{4c}$$

For a given symmetric matrix $B \in \mathbb{R}^{n \times n}$, quadratic models of $f(x)$ and $P(x; \mu)$ along a direction d are specified to be

$$\breve{f}(d; x, B) := f(x) + d^T g(x) + \frac{1}{2} d^T B d, \tag{5a}$$

$$\breve{P}(d; x, B, \mu) := \mu \breve{f}(d; x, B) + \bar{h}(d; x). \tag{5b}$$

Moreover, linear predicted changes in the objective function, the constraint violation function, and the penalty function are specified by

$$\Delta \bar{f}(d; x) := f(x) - \bar{f}(d; x) = -d^T g(x), \tag{6a}$$

$$\Delta \bar{h}(d; x) := h(x) - \bar{h}(d; x), \tag{6b}$$

$$\Delta \bar{P}(d; x, \mu) := P(x; \mu) - \bar{P}(d; x, \mu) = \mu \Delta \bar{f}(d; x) + \Delta \bar{h}(d; x). \tag{6c}$$

Quadratic predicted changes in the objective function and the penalty function are defined by

$$\Delta \breve{f}(d; x, B) := f(x) - \breve{f}(d; x, B) = -d^T g(x) - \frac{1}{2} d^T B d, \tag{7a}$$

$$\Delta \breve{P}(d; x, B, \mu) := P(x, \mu) - \breve{P}(d; x, B, \mu) = \mu \Delta \breve{f}(d; x, B) + \Delta \bar{h}(d; x) \tag{7b}$$

$$= \Delta \bar{P}(d; x, \mu) - \frac{\mu}{2} d^T B d. \tag{7c}$$

Lemma 2.5 of [24] expresses a relationship between the linear predicted changes in the constraint violation function and its directional derivative. Moreover, a relationship between the linear predicted changes in the penalty function and its directional derivative is delineated in [24, Lemma 2.9]. Here, we merge these results into the following lemma.

Lemma 1 *For a point x, a direction d, and a penalty parameter $\mu > 0$, we have*

$$D_d h(x) \leq -\Delta \bar{h}(d; x), \tag{8a}$$

$$D_d P(x; \mu) \leq -\Delta \bar{P}(d; x, \mu). \tag{8b}$$

Using these notations, we first provide a review of nonmonotone filter SQP algorithm of Gould et al. [24, 25] (FiSQP) in the next section.

3 FiSQP Algorithm

This section gives a brief description of the FiSQP algorithm, proposed by Gould et al. [24, 25]. At the kth iteration of this algorithm, with a certain trust region radius $\rho_k > 0$, a steering subproblem is defined as

$$\min_s \; \bar{h}(s; x^{(k)}) \quad \text{s.t.} \quad \|s\|_\infty \leq \rho_k. \tag{9}$$

Because $\bar{h}(s; x^{(k)})$ is a convex function, (9) is a convex subproblem. Consequently, if $s^{(k)}$ is an optimal solution of the steering subproblem, then we have

$$\bar{h}(s^{(k)}; x^{(k)}) \leq \bar{h}(s; x^{(k)}),$$

for all s in the trust region, that is, $\{s : \|s\|_\infty \leq \rho_k\}$. Specially, for $s = \mathbf{0}_n$, we have

$$h^{(k)} := h(x^{(k)}) = \bar{h}(\mathbf{0}_n; x^{(k)}) \geq \bar{h}(s^{(k)}; x^{(k)}),$$

and therefore, $\Delta \bar{h}(s^{(k)}; x^{(k)}) := h^{(k)} - \bar{h}(s^{(k)}; x^{(k)}) \geq 0$. If $\Delta \bar{h}(s^{(k)}; x^{(k)}) = 0$ and $h^{(k)} > 0$, then $x^{(k)}$ is recognized as an infeasible stationary point (see [13, Theorem 3.2]) and the algorithm terminates. In this situation, problem (1) is decided to be locally infeasible at the point $x^{(k)}$ (see [10]–[13]).

If $x^{(k)}$ is not an infeasible stationary point, then a matrix $B^{(k)}$ is chosen as a symmetric positive-definite approximation of $H^{(k)} := \nabla^2_{xx} \mathcal{L}_0(x^{(k)}, y^{(k)})$. Afterwards, depending on whether $\bar{h}(s^{(k)}; x^{(k)}) = 0$ or $\bar{h}(s^{(k)}; x^{(k)}) > 0$, two different feasible convex QP subproblems are defined to compute a predictor direction. In the case $\bar{h}(s^{(k)}; x^{(k)}) = 0$, the predictor direction $p^{(k)}$ is computed as the unique optimal solution of the following feasible strongly convex QP subproblem:

$$\begin{aligned} \min_{p} \quad & \frac{1}{2} p^T B^{(k)} p + p^T g^{(k)} \\ \text{s.t.} \quad & c_l \le c^{(k)} + J^{(k)} p \le c_u, \\ & x_l \le x^{(k)} + p \le x_u, \end{aligned} \tag{10}$$

where $g^{(k)} := g(x^{(k)})$, $J^{(k)} := J(x^{(k)})$ and $c^{(k)} := c(x^{(k)})$. If $\bar{h}(s^{(k)}; x^{(k)}) > 0$, then the QP subproblem (10) is infeasible. Hence, in this case the predictor direction is defined as $p^{(k)} := \operatorname{argmin}_{p\in\mathbb{R}^n} \breve{P}(p; x^{(k)}, B^{(k)}, \mu_k)$, where $\mu_k > 0$ is the penalty parameter at the current iterate.

The search direction is computed as a convex combination of the predictor direction and the steering direction, that is,

$$d^{(k)} := d(\tau_k) := \tau_k p^{(k)} + (1 - \tau_k) s^{(k)},$$

where

$$\tau_k = \max \left\{ \tau^j \mid \Delta\bar{h}(d(\tau^j); x^{(k)}) \ge \zeta_1 \Delta\bar{h}(s^{(k)}; x^{(k)}), \quad j = 0, 1, 2, \ldots \right\}, \tag{11}$$

for predesignated scalars $\tau, \zeta_1 \in (0, 1)$. Hence, Lemma 1 ensures that the search direction $d^{(k)}$ is a descent direction for the constraint violation function provided that $\Delta\bar{h}(s^{(k)}; x^{(k)}) > 0$. Moreover, if $\Delta\bar{h}(s^{(k)}; x^{(k)}) = 0$ and $x^{(k)}$ is not an infeasible stationary point, then $d^{(k)}$ is a descent direction for the objective function.

To have the search direction $d^{(k)}$ as a descent direction for the penalty function, the penalty parameter is updated as follows:

$$\mu_{k+1} = \begin{cases} \mu_k, & \text{if } \Delta\bar{P}(d^{(k)}; x^{(k)}, \mu_k) \\ & \qquad \ge \zeta_2 \Delta\bar{h}(s^{(k)}; x^{(k)}), \\ \min\left\{ \frac{1}{2}\mu_k, \frac{\zeta_2 \Delta\bar{h}(s^{(k)}; x^{(k)}) - \Delta\bar{h}(d^{(k)}; x^{(k)})}{\Delta\bar{f}(d^{(k)}; x^{(k)})} \right\}, & \text{otherwise}, \end{cases} \tag{12}$$

where $\zeta_2 \in (0, \zeta_1)$ is a predesignated constant. This penalty parameter updating formula along with Lemma 1 guarantees that

$$D_{d^{(k)}} P(x^{(k)}; \mu_{k+1}) \le -\Delta\bar{P}(d^{(k)}; x^{(k)}, \mu_{k+1}) \le -\zeta_2 \Delta\bar{h}(s^{(k)}; x^{(k)}).$$

Consequently, the search direction $d^{(k)}$ is a descent direction for $P(x; \mu_{k+1})$ at the point $x = x^{(k)}$ (see [24, Lemma 2.10]).

In order to accelerate the convergence of the algorithm, Gould et al. [24, 25] compute an accelerating direction in every iteration. Active sets of general constraints and bound constraints are respectively defined as follows:

$$\mathcal{A}_G(p; x^{(k)}) := \{i = 1, \ldots, m \mid [c^{(k)} + J^{(k)} p]_i = [c_l]_i \text{ or } [c^{(k)} + J^{(k)} p]_i = [c_u]_i\}, \tag{13a}$$

$$\mathcal{A}_B(p; x^{(k)}) := \{j = 1, \ldots, n \mid [x^{(k)} + p]_j = [x_l]_j \text{ or } [x^{(k)} + p]_j = [x_u]_j\}. \tag{13b}$$

The accelerating direction is computed to be

$$d_A^{(k)} := p^{(k)} + a^{(k)}, \tag{14}$$

where $a^{(k)}$ is an optimal solution of the accelerating subproblem, posed as

$$\begin{aligned} \min_{a} \quad & \frac{1}{2} a^T H^{(k)} a + a^T (g^{(k)} + H^{(k)} p^{(k)}) \\ \text{s.t.} \quad & [c^{(k)} + J^{(k)} a]_i = 0, \quad i \in \mathcal{A}_G^{(k)}, \\ & [a]_j = 0, \qquad\qquad\quad j \in \mathcal{A}_B^{(k)}, \\ & \|a\|_2 \le \rho_k^A, \end{aligned} \tag{15}$$

where $\mathcal{A}_G^{(k)} := \mathcal{A}_G(p^{(k)}; x^{(k)})$ and $\mathcal{A}_B^{(k)} := \mathcal{A}_B(p^{(k)}; x^{(k)})$, with $\rho_k^A > 0$ being a trust-region radius.

Next, we describe the nonmonotone filter strategy of Gould et al. [25] for computing the step-length along the search direction $d^{(k)}$ or the accelerating direction $d_A^{(k)}$. We first need to state some basic definitions of filter methods.

Definition 1 $x^{(k)}$ is called to be acceptable to $x^{(i)}$, if either one of the following conditions holds:

$$h^{(k)} \le \max\left\{\beta h^{(i)},\ h^{(i)} - \alpha_i \zeta_1 \Delta\bar{h}(s^{(i)}; x^{(i)})\right\}, \tag{16a}$$

$$f^{(k)} \le f^{(i)} - \gamma \min\left\{\beta h^{(i)},\ h^{(i)} - \alpha_i \zeta_1 \Delta\bar{h}(s^{(i)}; x^{(i)})\right\}, \tag{16b}$$

where $\gamma, \beta \in (0, 1)$ are constants, ζ_1 is defined as in (11), and α_i is the step-length in the ith iteration. If neither (16a) nor (16b) holds, then we say that $x^{(k)}$ is dominated by $x^{(i)}$.

At the kth iteration ($k \ge 1$), the filter $\mathcal{F}_k$ is taken to be a subset of the set $\{0, \ldots, k - 1\}$ such that for every $i, j \in \mathcal{F}_k$, we have $x^{(i)}$ acceptable to $x^{(j)}$. This clearly means that $i, j \in \mathcal{F}_k$ if and only if $x^{(i)}$ is acceptable to $x^{(j)}$. For $k = 0$, we have $\mathcal{F}_0 = \emptyset$. In the following definition, we define acceptability of a trial point x to a filter $\mathcal{F}_k$.

Definition 2 If x is acceptable to $x^{(i)}$, for every $i \in \mathcal{F}_k$, then x is called acceptable to the filter $\mathcal{F}_k$. We say x is acceptable to $\mathcal{F}_k$ augmented by $x^{(k)}$ if and only if x is acceptable to $\mathcal{F}_k$ and also to $x^{(k)}$.

Consider a search direction $d^{(k)}$, and possibly an accelerating direction $d_A^{(k)}$, computed at a point $x^{(k)}$. Using a backtracking line-search procedure, a step-length $\alpha \in (0, 1]$

is found so that a trial point $x^{(k)} + \alpha\hat{d}^{(k)}$, where $\hat{d}^{(k)} \in \{d_A^{(k)}, d^{(k)}\}$, satisfies one of the four possible sets of conditions. The filter mode and the penalty mode are the two alternative modes in which these conditions are checked. The algorithm starts at the filter mode; this implies that the quality of a trial point is to be assessed by the filter. According to certain conditions being satisfied, the pair $(\alpha, \hat{d}^{(k)})$ is called an h-, an f- or a b-(blocking)-pair, and the related iterate is correspondingly called an h-, an f-, or a b-type iterate, respectively. The algorithm switches to the penalty mode when a b-type iterate is encountered. There, a step-length α is found so that $x^{(k)} + \alpha\hat{d}^{(k)}$ sufficiently reduces the penalty function. The pair $(\alpha, \hat{d}^{(k)})$ is then called a p-type pair and the related iterate is called a p-type iterate. The penalty mode continues until an acceptable point to the filter is found. In order to describe those conditions, the Cauchy steps are employed. Gould and Robinson [28, 29] used the Cauchy steps to utilize the exact curvature of the problem in their SQP algorithms.

Definition 3 (*Cauchy steps*) At iteration k, the Cauchy-f step $d_{cf}^{(k)}$ is defined to be

$$d_{cf}^{(k)} = \alpha_k^f d^{(k)}, \quad \text{where} \quad \alpha_k^f = \text{argmin}_{0\le\alpha\le 1} \breve{f}(\alpha d^{(k)}; x^{(k)}, H^{(k)}), \tag{17}$$

and the Cauchy-P step $d_{cP}^{(k)}$ is defined to be

$$d_{cP}^{(k)} = \alpha_k^P d^{(k)}, \quad \text{where} \quad \alpha_k^P = \text{argmin}_{0\le\alpha\le 1} \breve{P}(\alpha d^{(k)}; x^{(k)}, H^{(k)}, \mu_{k+1}). \tag{18}$$

The iterations of the nonmonotone algorithm are partitioned into a set of successful iterations $\mathcal{S}$ and a set of unsuccessful iterations $\mathcal{U}$. Let $R(k) := \max\{i \mid 1 \le i \le k,\ i \in \mathcal{S}\}$ be the largest successful iteration index $i \le k$. The following four definitions describe the four possible scenarios for updating an iterate.

Definition 4 (*h-type pair/iterate*) A pair (α, d) forms an h-type pair whenever $x^{(k)} + \alpha p$ is acceptable to $\mathcal{F}_{R(k)}$ augmented by $x^{(R(k))}$, and

$$\Delta\bar{f}(d^{(R(k))}; x^{(R(k))}) < \kappa_1 \Delta\bar{h}(d^{(R(k))}; x^{(R(k))}), \tag{19}$$

for some constant $\kappa_1 \in (0, 1)$. If $(\alpha_k, d^{(k)})$ is an h-type pair, then $x^{(R(k))}$ is called an P-type iterate, and $k + 1$ is added to the set of successful iterations.

Definition 5 (*f-type pair/iterate*) A pair (α, d) forms an f-type pair whenever $x^{(R(k))} + \alpha d$ is acceptable to $\mathcal{F}_{R(k)}$, inequality (19) does not hold, and

$$f(x^{(k)} + \alpha d) \le f^{(R(k))} - \kappa_2 \alpha \omega_{R(k)}^f, \tag{20}$$

where $\kappa_2 \in (0, 1)$ is a constant, and

$$\omega_{R(k)}^f := \min\left\{\Delta\bar{f}(d^{(R(k))}; x^{(R(k))}),\ \Delta\breve{f}(d_{cf}^{(R(k))}; x^{(R(k))}, H^{(R(k))})\right\}. \tag{21}$$

If $(\alpha_k, d^{(k)})$ is an f-type pair, then $x^{(R(k))}$ is called an f-type iterate, and $k+1$ is added to the set of successful iterations.

Definition 6 (*b-type pair/iterate*) A pair (α, d) forms a b-type pair whenever we have

$$h(x^{(k)} + \alpha d) < h^{(R(k))} \tag{22}$$

and

$$P(x^{(k)} + \alpha d; \mu_{k+1}) \leq P(x^{(R(k))}; \mu_{R(k)+1}) - \kappa_3 \alpha \omega^P_{R(k)}, \tag{23}$$

where $\kappa_3 \in (0, 1)$ and

$$\begin{aligned}\omega^P_{R(k)} := \min \big\{ &\Delta \bar{P}(d^{(R(k))}; x^{(R(k))}, \mu_{R(k)+1}), \\ &\Delta \breve{P}(d^{(R(k))}_{cP}; x^{(R(k))}, H^{(R(k))}, \mu_{R(k)+1}) \big\}.\end{aligned} \tag{24}$$

If $(\alpha_k, d^{(k)})$ is a b-type pair, then $x^{(R(k))}$ is called a b-type iterate, and $k+1$ is added to the set of successful iterations.

Definition 7 (*p-type pair/iterate*) A pair (α, d) forms a p-type pair whenever (23) holds. If $(\alpha_k, d^{(k)})$ is a p-type pair, then $x^{(R(k))}$ is called an p-type iterate, and $k+1$ is added to the set of successful iterations.

In the FiSQP algorithm, a non-negative integer variable named as *fails* stores the number of consecutive unsuccessful iterations. Moreover, $max_{fails} > 0$ is a predesignated integer that specifies the maximum number of allowed consecutive unsuccessful iterations. The logical variable P_{mode} indicates whether the algorithm is currently in penalty mode or filter mode.

If $fails \leq max_{fails}$, then steering, predictor, accelerating and search directions are computed as stated above, and the penalty parameter is updated by (12). The algorithm sets $\hat{d}^{(k)} \leftarrow d^{(k)}_A$, and checks whether $(1, \hat{d}^{(k)})$ is a successful pair or not. If $(1, \hat{d}^{(k)})$ is a p-type pair (when $P_{mode} =$ **true**) or an h-, an f- or a b-type pair (when $P_{mode} =$ **false**), then $(1, \hat{d}_k)$ is called a successful pair ($k+1$ is added to $\mathcal{S}$), *fails* is set to zero and P_{mode} is updated. Otherwise, $(1, \hat{d}_k)$ is called an unsuccessful pair ($k+1$ is added to $\mathcal{U}$), and *fails* is increased by one. Regardless of $(1, \hat{d}^{(k)})$ being a successful or an unsuccessful pair, the iterate is updated as $x^{(k+1)} = x^{(k)} + \hat{d}^{(k)}$.

If $fails > max_{fails}$, then the algorithm returns to the last successful iteration ($k \leftarrow R(k)$), and a backtracking line-search is performed to compute a step-length α_k so that $(\alpha_k, \hat{d}^{(k)})$ is a successful pair for either $\hat{d}^{(k)} = d^{(k)}_A$ or $\hat{d}^{(k)} = d^{(k)}$. Then, *fails* is set to zero, P_{mode} is updated, and the iterate is updated as $x^{(k+1)} = x^{(k)} + \alpha_k \hat{d}^{(k)}$.

Finally, the penalty parameter is decreased by the factor $\frac{1}{2}$, provided that

$$\Delta \breve{P}(d^{(k)}; x^{(k)}, B^{(k)}, \mu_{k+1}) < \zeta_3 \Delta \breve{P}(p^{(k)}; x^{(k)}, B^{(k)}, \mu_{k+1}), \tag{25}$$

for some $\zeta_3 \in (0, 1)$. Moreover, the last successful iteration index $(R(k))$ and the filter are updated.

On the basis of the FiSQP algorithm, we propose our new inexact nonmonotone filter SQP algorithm in the next section.

4 *i*FiSQP Algorithm

We present a new inexact nonmonotone filter SQP algorithm (*i*FiSQP) for solving the general nonlinear program (1). The main contribution of our work is in computing the search direction as an inexact solution of a feasible strongly convex quadratic program. We first describe the process for computing the search direction.

In practice, by defining the slack variables, the steering subproblem (9) is reformulated as the following linear program:

$$\begin{aligned}
\min_{s,r^+,r^-,t^+,t^-} \quad & \mathbf{1}_m^T(r^+ + r^-) + \mathbf{1}_n^T(t^+ + t^-) \\
\text{s.t.} \quad & c_l \le c^{(k)} + J^{(k)}s + r^+ - r^- \le c_u, \\
& x_l \le x^{(k)} + s + t^+ - t^- \le x_u, \\
& -\rho_k \le [s]_j \le \rho_k, \quad j = 1, \ldots, n, \\
& r^+, r^- \ge \mathbf{0}_m, \quad t^+, t^- \ge \mathbf{0}_n.
\end{aligned} \tag{26}$$

Let $(s^{(k)}, r^{+(k)}, r^{-(k)}, t^{+(k)}, t^{-(k)})$ be an optimal solution of (26). Accordingly, $s^{(k)}$ would be an optimal solution of the steering subproblem (9), which is called the steering direction. If $x^{(k)}$ is not an infeasible stationary point, then $\Delta\bar{h}(s^{(k)}; x^{(k)}) = 0$ implies that $h^{(k)} = 0$. Conversely, if $h^{(k)} = 0$, then $s^{(k)} = \mathbf{0}_n$ is an optimal solution of (9) and we have $\Delta\bar{h}(s^{(k)}; x^{(k)}) = 0$. As a result, if $x^{(k)}$ is not an infeasible stationary point, then

$$h^{(k)} = 0 \Leftrightarrow \Delta\bar{h}(s^{(k)}; x^{(k)}) = 0, \tag{27}$$

for every optimal solution $s^{(k)}$ of the steering subproblem (9). We note that if $h^{(k)} = 0$, then there is no need to solve the subproblem (9). In this case, we set $s^{(k)} = \mathbf{0}_n$ as a trivial optimal solution of the steering subproblem. Algorithm 1 gives an outline of the steps for computing the steering direction.

Algorithm 1 Compute the steering direction.

```
1: procedure STEERING_DIRECTION ($x^{(k)}$).
2:     if $h^{(k)} = 0$ then
3:         Set $s^{(k)} = t^{+(k)} = t^{-(k)} = \mathbf{0}_n$ and $r^{+(k)} = r^{-(k)} = \mathbf{0}_m$;
4:         return $(s^{(k)}, r^{+(k)}, r^{-(k)}, t^{+(k)}, t^{-(k)})$;
5:     else
6:         Choose a trust region radius $\rho_k \in [\underline{\rho}, \overline{\rho}]$;
7:         construct the linear program (26);
8:         obtain $(s^{(k)}, r^{+(k)}, r^{-(k)}, t^{+(k)}, t^{-(k)})$ by solving (26);
9:         return $(s^{(k)}, r^{+(k)}, r^{-(k)}, t^{+(k)}, t^{-(k)})$;
10:    end if
11: end procedure.
```

Using the optimal solution of (26) and defining $r^{(k)} := r^{+(k)} - r^{-(k)}$ and $t^{(k)} := t^{+(k)} - t^{-(k)}$, a feasible strongly convex QP subproblem is defined as follows:

$$\begin{aligned} \min_{d} \quad & \frac{1}{2} d^T B^{(k)} d + d^T g^{(k)} \\ \text{s.t.} \quad & c_l \le c^{(k)} + J^{(k)} d + r^{(k)} \le c_u, \\ & x_l \le x^{(k)} + d + t^{(k)} \le x_u. \end{aligned} \tag{28}$$

Obviously, $d = s^{(k)}$ satisfies all the constraints of (28), and it is a feasible point for this QP subproblem. The KKT conditions for (28) can be compactly stated as

$$\psi_k(d, y) := \begin{bmatrix} \min\{c_u - c^{(k)} - J^{(k)} d - r^{(k)} \quad , \quad [y]^-\} \\ \min\{c^{(k)} + J^{(k)} d + r^{(k)} - c_l \quad , \quad [y]^+\} \\ \min\{x_u - x^{(k)} - d - t^{(k)} \quad , \quad [B^{(k)} d + \nabla_x \mathcal{L}_0(x, y)]^-\} \\ \min\{x^{(k)} + d + t^{(k)} - x_l \quad , \quad [B^{(k)} d + \nabla_x \mathcal{L}_0(x, y)]^+\} \end{bmatrix} = \begin{bmatrix} \mathbf{0}_m \\ \mathbf{0}_m \\ \mathbf{0}_n \\ \mathbf{0}_n \end{bmatrix}, \tag{29}$$

in a similar manner that (3) was obtained for the main problem (1). Because (28) is a strongly convex QP, d would be the optimal solution of this problem if and only if there exists a Lagrange multiplier vector y so that $\psi_k(d, y) = \mathbf{0}_{2m+2n}$. It is worthwhile to note that $\psi_k(\mathbf{0}_n, y) = \Psi(x^{(k)}, y)$.

In order to compute the search direction, a quadratic program solver (QP-solver) is invoked to find an approximate solution $(d^{(k)}, y_d^{(k)})$ of (28) satisfying the following conditions:

$$\zeta_1 \Delta \bar{h}(s^{(k)}; x^{(k)}) \le \Delta \bar{h}(d^{(k)}; x^{(k)}) \le \Delta \bar{h}(s^{(k)}; x^{(k)}), \tag{30a}$$

$$\Delta \breve{P}(d^{(k)}; x^{(k)}, B^{(k)}, \mu_k) \ge \Delta \breve{P}(s^{(k)}; x^{(k)}, B^{(k)}, \mu_k), \tag{30b}$$

$$\|\psi_k(d^{(k)}, y_d^{(k)})\| \le \eta \|\Psi(x^{(k)}, y^{(k)})\|, \tag{30c}$$

where $\eta, \zeta_1 \in (0, 1)$ are some predesignated constants. The steps for computing the search direction is presented in Algorithm 2.

Algorithm 2 Compute the search direction.

```
1: procedure SEARCH_DIRECTION ($x^{(k)}, s^{(k)}, r^{(k)}, t^{(k)}, \mu_k$).
2:     Given constants: $\zeta_1, \eta \in (0, 1)$;
3:     choose a symmetric positive-definite matrix $B^{(k)}$;
4:     construct the QP subproblem (28);
5:     call a QP-solver to find $(d^{(k)}, y_d^{(k)})$ as an approximate primal-dual solution of (28) that
       satisfies conditions (30);
6:     return $(d^{(k)}, y_d^{(k)})$;
7: end procedure.
```

According to the condition (30c), as iterates get closer to a stationary point of the main problem, QP subproblems are solved more accurately. If $h^{(k)} = 0$, then the kth iteration is called a feasible iteration; otherwise, it is called an infeasible iteration. In the following lemma, we investigate descent properties of the computed search direction satisfying the conditions (30a) and (30b).

Lemma 2 *If $x^{(k)}$ is neither an infeasible stationary point nor a stationary point of* (1)*, $B^{(k)}$ is a symmetric positive-definite matrix, and $d^{(k)}$ satisfies the conditions* (30a) *and* (30b)*, then we have the followings:*

(a) If $h^{(k)} = 0$, then

$$D_{d^{(k)}} P(x^{(k)}; \mu) \leq -\frac{\mu}{2} d^{(k)^T} B^{(k)} d^{(k)}, \tag{31}$$

$$D_{d^{(k)}} f(x^{(k)}) \leq -\frac{1}{2} d^{(k)^T} B^{(k)} d^{(k)}, \tag{32}$$

for every penalty parameter $\mu > 0$. So, if $d^{(k)} \neq \mathbf{0}_n$, then $d^{(k)}$ is a descent direction for both the penalty function and the objective function.

(b) If $h^{(k)} > 0$, then $d^{(k)} \neq \mathbf{0}_n$, and

$$D_{d^{(k)}} h(x^{(k)}) \leq -\Delta \bar{h}(d^{(k)}; x^{(k)}) < 0, \tag{33}$$

and thus, $d^{(k)}$ is a descent direction for the constraint violation function.

Proof As stated before, $h^{(k)} = 0$ implies $s^{(k)} = \mathbf{0}_n$, $\Delta \bar{h}(s^{(k)}; x^{(k)}) = 0$, and $\Delta \breve{P}(s^{(k)}; x^{(k)}, B^{(k)}, \mu_k) = 0$. From (8b), (7c) and (30b), we have

$$\begin{aligned}
D_{d^{(k)}} P(x^{(k)}; \mu) &\leq -\Delta \bar{P}(d^{(k)}; x^{(k)}, \mu) \\
&= -\Delta \breve{P}(d^{(k)}; x^{(k)}, B^{(k)}, \mu) - \frac{\mu}{2} d^{(k)^T} B^{(k)} d^{(k)} \\
&\leq -\Delta \breve{P}(s^{(k)}; x^{(k)}, B^{(k)}, \mu) - \frac{\mu}{2} d^{(k)^T} B^{(k)} d^{(k)} \\
&= -\frac{\mu}{2} d^{(k)^T} B^{(k)} d^{(k)},
\end{aligned}$$

for every penalty parameter $\mu > 0$. So, (31) is established.

Moreover, it follows from $\Delta\bar{h}(s^{(k)}; x^{(k)}) = 0$ and (30a) that $\Delta\bar{h}(d^{(k)}; x^{(k)}) = 0$. From f being a continuously differentiable function and attending to the definition of predicted changes in the objective function and the penalty function, we have

$$\begin{aligned}
D_{d^{(k)}} f(x^{(k)}) &= g^{(k)^T} d^{(k)} \\
&= -\Delta\bar{f}(d^{(k)}; x^{(k)}) \\
&= -\frac{1}{\mu}\big(\mu\Delta\bar{f}(d^{(k)}; x^{(k)}) + \Delta\bar{h}(d^{(k)}; x^{(k)})\big) \\
&= -\frac{1}{\mu}\Delta\breve{P}(d^{(k)}; x^{(k)}, B^{(k)}, \mu) - \frac{1}{2}d^{(k)^T} B^{(k)} d^{(k)} \\
&\le -\frac{1}{\mu}\Delta\breve{P}(s^{(k)}; x^{(k)}, B^{(k)}, \mu) - \frac{1}{2}d^{(k)^T} B^{(k)} d^{(k)} \\
&= -\frac{1}{2}d^{(k)^T} B^{(k)} d^{(k)},
\end{aligned}$$

which establishes the inequality (32).

On the other hand, if $h^{(k)} > 0$ and $x^{(k)}$ is not an infeasible stationary point, then according to (27) we have $\Delta\bar{h}(s^{(k)}; x^{(k)}) > 0$. Thus, (30a) yields $\Delta\bar{h}(d^{(k)}; x^{(k)}) > 0$ and this shows that $d^{(k)} \neq \mathbf{0}_n$. Consequently, (33) is obtained from (8a). ■

As Lemma 2 states, in the infeasible iterations the computed search direction is a descent direction for the constraint violation function. In the next lemma we will see that the search direction would be a descent direction for the penalty function using the penalty parameter updating rule (12). Indeed, the following lemma incorporates the results of Lemmas 2.8, 2.9 and 2.10 of [24] for our settings.

Lemma 3 *Assume $x^{(k)}$ is neither a stationary point of* (1) *nor an infeasible stationary point. If $\mu_k \in (0, 1)$, then the penalty parameter update formula* (12) *is well-defined, and immediately we have*

$$\mu_{k+1} \in (0, 1), \ \textit{and} \tag{34}$$

$$\Delta\bar{P}(d^{(k)}; x^{(k)}, \mu_{k+1}) \ge \zeta_2 \Delta\bar{h}(s^{(k)}; x^{(k)}), \tag{35}$$

to yield

$$D_{d^{(k)}} P(x^{(k)}; \mu_{k+1}) \le -\Delta\bar{P}(d^{(k)}; x^{(k)}, \mu_{k+1}) < 0. \tag{36}$$

Proof In the feasible iterations, we have $s^{(k)} = \mathbf{0}_n$, $\Delta\bar{h}(s^{(k)}; x^{(k)}) = 0$ and $\Delta\breve{P}(s^{(k)}; x^{(k)}, B^{(k)}, \mu_k) = 0$. Hence, from (7c), (30b) and positive-definiteness of $B^{(k)}$, we obtain:

$$\begin{aligned}\Delta\bar{P}(d^{(k)}; x^{(k)}, \mu_k) &= \Delta\check{P}(d^{(k)}; x^{(k)}, B^{(k)}, \mu_k) + \frac{\mu_k}{2} d^{(k)^T} B^{(k)} d^{(k)}\\ &\geq \Delta\check{P}(s^{(k)}; x^{(k)}, B^{(k)}, \mu_k) + \frac{\mu_k}{2} d^{(k)^T} B^{(k)} d^{(k)}\\ &\geq 0 = \zeta_2\, \Delta\bar{h}(s^{(k)}; x^{(k)}).\end{aligned}$$

As a result, we get $\mu_{k+1} = \mu_k \in (0, 1)$ from (12). Hence, (34) and (35) are clearly established in the feasible iterations.

On the contrary, consider $x^{(k)}$ being an infeasible iterate which is not an infeasible stationary point. We have $\Delta\bar{h}(s^{(k)}; x^{(k)}) > 0$ in this case. If

$$\Delta\bar{P}(d^{(k)}; x^{(k)}, \mu_k) \geq \zeta_2\, \Delta\bar{h}(s^{(k)}; x^{(k)}),$$

then according to (12) we have $\mu_{k+1} = \mu_k \in (0, 1)$ and (35) is at hand. In the rest of the proof, we assume

$$\Delta\bar{P}(d^{(k)}; x^{(k)}, \mu_k) < \zeta_2\, \Delta\bar{h}(s^{(k)}; x^{(k)}). \tag{37}$$

Thus, from (6c), (37), $\Delta\bar{h}(s^{(k)}; x^{(k)}) > 0$, $\mu_k > 0$, $\zeta_2 \in (0, \zeta_1)$ and (30a), we have

$$\begin{aligned}\Delta\bar{f}(d^{(k)}; x^{(k)}) &= \frac{1}{\mu_k}\big(\Delta\bar{P}(d^{(k)}; x^{(k)}, \mu_k) - \Delta\bar{h}(d^{(k)}; x^{(k)})\big)\\ &< \frac{1}{\mu_k}\big(\zeta_2\Delta\bar{h}(d^{(k)}; x^{(k)}) - \Delta\bar{h}(s^{(k)}; x^{(k)})\big)\\ &< \frac{1}{\mu_k}\big(\zeta_1\Delta\bar{h}(d^{(k)}; x^{(k)}) - \Delta\bar{h}(d^{(k)}; x^{(k)})\big) < 0.\end{aligned} \tag{38}$$

As a consequence of (38), (30a), $\zeta_2 \in (0, \zeta_1)$, and $\mu_k \in (0, 1)$, we have the penalty parameter update, given by

$$\mu_{k+1} = \min\left\{\frac{1}{2}\mu_k, \frac{\zeta_2\Delta\bar{h}(s^{(k)}; x^{(k)}) - \Delta\bar{h}(d^{(k)}; x^{(k)})}{\Delta\bar{f}(d^{(k)}; x^{(k)})}\right\} \in (0, 1),$$

to be well-defined, and to yield (35), whenever (37) holds. Therefore, under the assumptions, (34) and (35) are valid. Finally, (36) is concluded form (8b) and (35). ■

In accordance with the proof of Lemma 3, we note that the penalty parameter should not be changed in the feasible iterations. We also note that conditions (30a)–(30c) hold for $(d_*^{(k)}, y_{d*}^{(k)})$, the optimal primal-dual solution of the QP subproblem (28). Hence, using an appropriate QP-solver, Algorithm 2 is well-defined for computing the search direction.

Similar to [24, 25], and in order to accelerate the convergence of the algorithm, we use an accelerating direction. Gould et al. [24, 25] compute an accelerating direction in every iteration. However, an accelerating direction is only useful when the iterate is near the feasible region of the problem [2]. So, in our algorithm, we compute an accelerating direction only when $\bar{h}(d^{(k)}; x^{(k)}) = 0$, which holds near the feasible region.

Algorithm 3 gives an outline of a procedure to compute the accelerating direction.

Algorithm 3 Compute the accelerating direction.

```
1: procedure ACCELERATING_DIRECTION ($x^{(k)}, d^{(k)}, y_d^{(k)}$).
2:     Compute $H^{(k)} \leftarrow \nabla_{xx}^2 \mathcal{L}_0(x^{(k)}, y_d^{(k)})$;
3:     form the sets $\mathcal{A}_G^{(k)} := \mathcal{A}_G(d^{(k)}; x^{(k)})$ and $\mathcal{A}_B^{(k)} := \mathcal{A}_B(d^{(k)}; x^{(k)})$ by (13);
4:     choose a trust region radius $\rho_k^A \in [\underline{\rho}, \overline{\rho}]$;
5:     construct accelerating subproblem (15);
6:     find $(a^{(k)}, y_A^{(k)})$ as a primal-dual solution of (15);
7:     compute $d_A^{(k)}$ by (14);
8:     return $(d_A^{(k)}, y_A^{(k)})$;
9: end procedure.
```

We also equip our algorithm with the nonmonotone filter strategy of Gould et al. [25]. Algorithm 4 provides the steps for checking whether a trial point $x^+ := x^{(k)} + \alpha_k \hat{d}^{(k)}$ is successful or not. Now, our complete inexact nonmonotone filter SQP algorithm is presented as Algorithm 5. In this algorithm, parameters *fails* and max_{fails} hold the number of successive unsuccessful iterations and the maximum number of allowed successive unsuccessful iterations, respectively. These parameters are used for the nonmonotone aspects of the algorithm.

The main difference between our algorithm and the FiSQP algorithm of [24, 25] is in computing the search direction. In our algorithm, the search direction is computed as an inexact solution of a feasible and strongly convex QP subproblem. But, in the FiSQP algorithm, at first the predictor direction is computed as an exact solution of a feasible convex QP subproblem. Then, using a backtracking procedure, the search direction is computed as a convex combination of the predictor direction and the steering direction. Therefore, our algorithm may consume less computational cost for obtaining the search direction. However, our search direction has the same descent properties as the search direction of the FiSQP algorithm; compare the results of Lemmas 2 and 3 in our work here with results of Lemmas 2.6, 2.7 and 2.10 of [24].

As another difference, the accelerating direction is computed in every iteration of the FiSQP algorithm. But, in our algorithm, the accelerating direction is only computed when $\bar{h}(d^{(k)}; x^{(k)}) = 0$. Let (x^*, y^*) be the optimal primal-dual solution of the main problem (1) that satisfies the following conditions:

1. The linear independence constraint qualification (LICQ) holds,
2. The second order sufficient optimality conditions hold,
3. The strict complementary conditions hold.

Algorithm 4 Check for success of a trial point.

```
1: procedure CHECK $(x^{(k)}, \alpha_k, \hat{d}^{(k)}, \mu_{k+1}, R(k), fails, \mathcal{F}_k, P_{mode})$.
2:     Given constants: $\zeta_1, \beta, \gamma, \kappa_1, \kappa_2, \kappa_3, \in (0, 1)$;
3:     compute the trial point $x^+ \leftarrow x^{(k)} + \alpha_k \hat{d}^{(k)}$;
4:     if $P_{mode}$ then
5:         if $(\alpha_k, \hat{d}^{(k)})$ is a $p$-pair then
6:             if $x^+$ is acceptable to $\mathcal{F}_k$ then
7:                 $P_{mode} \leftarrow$ false;
8:             end if
9:             Set $success \leftarrow$ true, $type \leftarrow p$ and $fails \leftarrow 0$;        ▷ $\mathcal{S} \leftarrow \mathcal{S} \cup \{k+1\}$
10:        else
11:            Set $success \leftarrow$ false and $fails \leftarrow fails + 1$;        ▷ $\mathcal{U} \leftarrow \mathcal{U} \cup \{k+1\}$
12:        end if
13:    else
14:        if $(\alpha_k, \hat{d}^{(k)})$ is an $h$-pair then
15:            Set $success \leftarrow$ true, $type \leftarrow h$ and $fails \leftarrow 0$;        ▷ $\mathcal{S} \leftarrow \mathcal{S} \cup \{k+1\}$;
16:        else if $(\alpha_k, \hat{d}^{(k)})$ is an $f$-pair then
17:            Set $success \leftarrow$ true, $type \leftarrow f$ and $fails \leftarrow 0$;        ▷ $\mathcal{S} \leftarrow \mathcal{S} \cup \{k+1\}$;
18:        else if $(\alpha_k, \hat{d}^{(k)})$ is a $b$-pair then
19:            Set $P_{mode} \leftarrow$ false, $success \leftarrow$ true, $type \leftarrow b$ and $fails \leftarrow 0$;
                                                                        ▷ $\mathcal{S} \leftarrow \mathcal{S} \cup \{k+1\}$;
20:        else
21:            Set $success \leftarrow$ false, $type \leftarrow u$ and $fails \leftarrow fails + 1$;        ▷ $\mathcal{U} \leftarrow \mathcal{U} \cup \{k+1\}$
22:        end if
23:    end if
24:    return $P_{mode}$, $fails$, $success$, and $type$;
25: end procedure.
```

If $(x^{(k)}, y^{(k)})$ is sufficiently close to (x^*, y^*), then the linearized constraints around $x^{(k)}$ are compatible ($\bar{h}(s^{(k)}; x^{(k)}) = 0$) and the optimal active-set can be detected by the search direction (see, for example, [40, Theorem 18.1]). Hence, condition (30a) implies that $\bar{h}(d^{(k)}; x^{(k)}) = 0$, when $x^{(k)}$ is near x^*. Consequently, in our algorithm, the accelerating direction is computed at every iteration in which the iterate is near the optimal solution. Since the optimal active-set can be recognized by the search direction only near the optimal solution, it is reasonable to compute the accelerating direction only near the optimal solution.

According to the above arguments, and with similar arguments given by [24, 25], the global convergence and a superlinear local convergence of the *i*FiSQP algorithm can be obtained under some standard assumptions. For more details, see Ahmadzadeh and Mahdavi-Amiri [1] presenting a more comprehensive discussion of a similar inexact nonmonotone filter SQP algorithm.

Algorithm 5 Inexact filter SQP (*i*FiSQP) algorithm.

```
1: Choose $\zeta_1, \zeta_2, \beta, \gamma, \kappa_1, \kappa_2, \kappa_3, \eta, \tau \in (0,1)$, $0 < \underline{\rho} < \overline{\rho} < \infty$ and $max_{fails} \in \mathbb{N}$;
2: **Initialization:** Give an initial pair $(x^{(0)}, y^{(0)})$, set $k \leftarrow 0$, $\mathcal{F}_0 \leftarrow \emptyset$, $fails \leftarrow 0$,
   $terminate \leftarrow$ **false**, $P_{mode} \leftarrow$ **false**, $R(k) \leftarrow 0$, and choose a penalty parameter $\mu_0 > 0$;
3: **while** not $terminate$ **do**
4:     **if** $\|\Psi(x^{(k)}, y^{(k)})\| = 0$ **then**                ▷ A KKT point is at hand
5:         set $terminate \leftarrow$ **true**, and **go to** line 40;
6:     **end if**
7:     **if** $fails \leq max_{fails}$ **then**
8:         $(s^{(k)}, r^{+(k)}, r^{-(k)}, t^{+(k)}, t^{-(k)})$ = STEERING_DIRECTION $(x^{(k)})$;
9:         **if** $h^{(k)} = \bar{h}(s^{(k)}; x^{(k)}) > 0$ **then**        ▷ An infeasible stationary point is found
10:            $terminate \leftarrow$ **true**, and **go to** line 40;
11:        **end if**
12:        set $r^{(k)} \leftarrow r^{+(k)} - r^{-(k)}$ and $t^{(k)} \leftarrow t^{+(k)} - t^{-(k)}$;
13:        $(d^{(k)}, y_d^{(k)})$ = SEARCH_DIRECTION $(x^{(k)}, s^{(k)}, r^{(k)}, t^{(k)}, \mu_k)$;
14:        update the penalty parameter by (12);
15:        evaluate $H^{(k)} = \nabla^2_{xx}\mathcal{L}_0(x^{(k)}, y_d^{(k)})$;
16:        compute the Cauchy steps $d_{cP}^{(k)}$ and $d_{cf}^{(k)}$ by (18) and (17), respectively;
17:        **if** $\bar{h}(d^{(k)}; x^{(k)}) = 0$ **then**
18:            $(d_A^{(k)}, y_A^{(k)})$ = ACCELERATING_DIRECTION $(x^{(k)}, d^{(k)}, y_d^{(k)})$;
19:            set $\hat{d}^{(k)} \leftarrow d_A^{(k)}$, and $\alpha_k \leftarrow 1$;
20:        **else**
21:            set $\hat{d}^{(k)} \leftarrow d^{(k)}$, and $\alpha_k \leftarrow 1$;
22:        **end if**
23:        $(P_{mode}, fails, success, type)$ = CHECK $(x^{(k)}, \alpha_k, \hat{d}^{(k)}, \mu_{k+1}, R(k), fails, \mathcal{F}_k, P_{mode})$;
24:        **go to** line 32;
25:    **else**
26:        $x^{(k)} \leftarrow x^{(R(k))}$, $\hat{d}^{(k)} \leftarrow d^{(R(k))}$, $y_d^{(k)} \leftarrow y_d^{(R(k))}$, $B^{(k)} \leftarrow B^{(R(k))}$, $d_{cP}^{(k)} \leftarrow d_{cP}^{(R(k))}$,
           $d_{cf}^{(k)} \leftarrow d_{cf}^{(R(k))}$, $\alpha_k \leftarrow 1$, and $\mu_{k+1} \leftarrow \mu_{R(k)+1}$;
27:        $(P_{mode}, fails, success, type)$ = CHECK $(x^{(k)}, \alpha_k, \hat{d}^{(k)}, \mu_{k+1}, R(k), fails, \mathcal{F}_k, P_{mode})$;
28:        **while** not $success$ **do**
29:            $\alpha_k \leftarrow \tau\alpha_k$;
30:            $(P_{mode}, fails, success, type)$ = CHECK $(x^{(k)}, \alpha_k, \hat{d}^{(k)}, \mu_{k+1}, R(k), fails, \mathcal{F}_k, P_{mode})$;
31:        **end while**
32:    **end if**
33:    **if** $success$ **then**
34:        $R(k+1) \leftarrow k+1$
35:        **if** $type = b$ or $type = h$ **then**
36:            $\mathcal{F}_{k+1} \leftarrow \mathcal{F}_k \cup \{R(k)\}$;
37:        **end if**
38:    **end if**
39:    $x^{(k+1)} \leftarrow x^{(k)} + \alpha_k \hat{d}^{(k)}$, and $k \leftarrow k+1$;
40: **end while**.
```

5 Experimental Results

Here, we present comparative results obtained from the *i*FiSQP method (Algorithm 5), implemented in MATLAB, and the FiSQP method of Gould et al. [25]; the latest corresponding program for FiSQP method in MATLAB was provided by Loh and Robinson, co-authors of [25]. Both methods are tested on a set of small/medium scale test problems from the CUTEst library [27] listed in Tables 1.1 and 1.3 of [23].

In our implementation, we made use of the primal simplex solver of Cplex [33] in Algorithm 1 to compute the steering direction. The symmetric positive-definite matrix $B^{(k)}$ is chosen using a modified Newton strategy as in [25]. In order to compute the search direction in Algorithm 2, we added conditions (30) as termination tests to the source code of the `qpa`, `qpb`, `qpc` and `cqp` solvers in GALAHAD [26] to find a desirable approximate solution of the QP subproblem (28) satisfying conditions (30). When no GALAHAD's QP-solver (`qpa`, `qpb`, `qpc` and `cqp`) can find a proper approximate solution of (28), we use the QP-solver of Cplex to find the search direction; however, there are ill-conditioned problems, for which neither an approximate nor an exact solution of an associated subproblem can be found. In the almost feasible iterations, that is when $\bar{h}(d^{(k)}; x^{(k)}) = 0$, the accelerating direction is computed by the method introduced in [25].

Our algorithm terminates in any one of the following situations:

1. $\|\Psi(x^{(k)}, y^{(k)})\|_\infty \leq \epsilon_1$. In this case, $(x^{(k)}, y^{(k)})$ is known as an approximate first order KKT point of (1).
2. $h^{(k)} \leq \epsilon_1$ and $\Delta\breve{P}(d^{(k)}; x^{(k)}, B^{(k)}, \mu_k) \leq \epsilon_2$. In this case, $(x^{(k)}, y^{(k)})$ is also known as an approximate first order KKT point of (1).
3. $h^{(k)} \geq 100\epsilon_1$ and $\Delta\bar{h}(s^{(k)}; x^{(k)}) \leq \epsilon_2$. In this case, $x^{(k)}$ is detected as an infeasible stationary point.
4. The iteration number exceeds max_{iter}. This case is known as a failure of a method.
5. The CPU time exceeds 10 minutes time limit. This case is also considered as a failure of a method.

The initial trust region radii of the steering subproblem (26) and the accelerating subproblem (15) are chosen as $\rho_0 = \rho_0^A = 10^2$ and they are updated in every iteration as in [13]. In agreement with the code of [25], we set parameters of our method as the ones given in Table 1.

Table 1 Parameters used in the programs

Parameter	Value	Parameter	Value	Parameter	Value
ζ_1	10^{-3}	ζ_2	10^{-6}	η	0.5
κ_1	10^{-3}	κ_2	10^{-4}	κ_3	10^{-4}
τ	0.5	β	0.99	γ	10^{-3}
μ_0	1	$\underline{\rho}$	1	$\overline{\rho}$	10^4
max_{fails}	3	ϵ_1	10^{-5}	ϵ_2	10^{-12}

Table 2 The infinity norm of the KKT residual of QP subproblems at the computed approximate solution for the test problem BT13

iter	FiSQP	*i*FiSQP
1	1.46E–11	3.43E–01
2	3.55E–15	3.43E+01
3	2.78E–16	6.01E+00
4	1.11E–16	1.97E–01
5	6.51E–18	5.49E–02
6	1.78E–15	1.24E–02
7	0.00E+00	2.31E–03
8	1.11E–16	9.11E–04
9	2.78E–17	1.13E–04
10	1.36E–20	5.47E–05
11	6.78E–21	8.57E–06
12	4.34E–19	2.79E–11
13	2.12E–22	9.65E–16
14	1.68E–23	1.21E–18
15	1.32E–23	9.14E–15

Table 2 gives the infinity norm of the KKT residual of QP subproblems at the computed approximate solution during solving the test problem BT13 by FiSQP and *i*FiSQP. Here, looking at the size of the infinity norm of the residual for the two methods, inexact solutions of the QP subproblems for *i*FiSQP are evidently used in the early iterations, while the FiSQP always compute exact solutions, within a considerable tolerance in every iteration. This behavior can be expected in almost all the test problems. This observation shows that solving QP subproblems inexactly, while reducing the computational costs, may have no adverse effect on the performance of the algorithm.

The list of the test problems on which the FiSQP and *i*FiSQP algorithms failed, are reported in Tables 4 and 5, respectively. Table 3 gives the meaning of the values for the flag 'status'.

We note that in every iteration of the FiSQP algorithm the accelerating direction is computed, while our algorithm calculates the accelerating direction only in almost feasible iterations. In our experiments, on the average, in 67.56% of iterations, the *i*FiSQP algorithm computes the accelerating direction.

Figure 1 shows the performance profile [18] for the CPU times consumed by testing *i*FiSQP and FiSQP methods on a set of small and medium test problems listed in Tables 1.1 and 1.3 of [23]. We should note that the profile does not include problems for which the two algorithms converged to different stationary points; the deleted problems are DUAL3, DUALC2, DUALC5, DUALC8, QPNBOEI1, HIMMELBK,

Table 3 The meaning of the values for the flag ‘status’ [23, Table 5.2]

Status	Meaning
–5	Cplex was unable to find a verifiably optimal solution to a steering subproblem
–6	Cplex was unable to find a verifiably optimal solution to a predictor subproblem
–7	Cplex crashed while solving either a steering or predictor subproblem
–8	A steering step was computed that increased the feasibility model
–9	The computed step was too small to make additional progress
–10	A NaN was encountered while evaluating a problem function
1	The maximum number of 10000 iterations was reached before a solution was found
2	The maximum CPU limit of 10 minutes was reached before a solution was found

Table 4 The list of failed test problems by FiSQP algorithm and their statuses

Name	Status	Name	Status	Name	Status	Name	Status
ACOPP118	−6	COOLHANS	−7	KTMODEL	−10	POWELLSQ	1
ACOPP30	−6	CRESC50	1	LEWISPOL	−8	QC	1
ACOPP57	−6	DALLASL	−6	LHAIFAM	1	S365	−10
ACOPR118	−6	EQC	1	MESH	−6	SAWPATH	−5
ACOPR14	−6	ERRINBAR	1	MSS1	−6	SSEBLIN	1
ACOPR30	−6	FLT	−6	NYSTROM5	−6	STEENBRC	−6
ACOPR57	−6	HEART6	−6	PFIT1	−6	STEENBRE	−6
AGG	1	HIMMELBD	1	PFIT2	−6	STEENBRG	−6
ALLINITA	−6	HIMMELBJ	−6	PFIT3	−6	TRO11X3	−5
ANTWERP	1	HS106	1	POLAK2	−5	TRO21X5	−5
ARGAUSS	−8	HS13	−7	POLAK3	−5	TRO3X3	−7
AVION2	1	HYDROELM	2	POLAK6	−5	ZAMB2-8	1

Table 5 The list of failed test problems by *i*FiSQP algorithm and their statuses

Name	Status	Name	Status	Name	Status	Name	Status
ACOPR118	2	HIMMELBJ	−6	POLAK3	−5	STEENBRG	−6
AGG	1	HS13	1	POLAK6	−5	TENBARS1	1
ALLINITA	1	HYDROELM	2	POWELLSQ	−6	TENBARS2	1
ALLINITC	−6	KTMODEL	−10	QC	1	TENBARS3	1
ANTWERP	1	LEWISPOL	−8	QPNBOEI2	1	TRO11X3	−6
ARGAUSS	−8	MESH	−5	SAWPATH	−5	TRO21X5	−6
AVION2	1	NYSTROM5	1	SSEBLIN	1	TRO3X3	−5
CRESC100	−5	PFIT2	−5	STEENBRB	−6	YFITNE	−8
ELATTAR	1	PFIT3	1	STEENBRC	−5	ZAMB2-8	1
EQC	1	PFIT4	−5	STEENBRE	1		
HIMMELBD	−6	POLAK2	−5	STEENBRF	2		

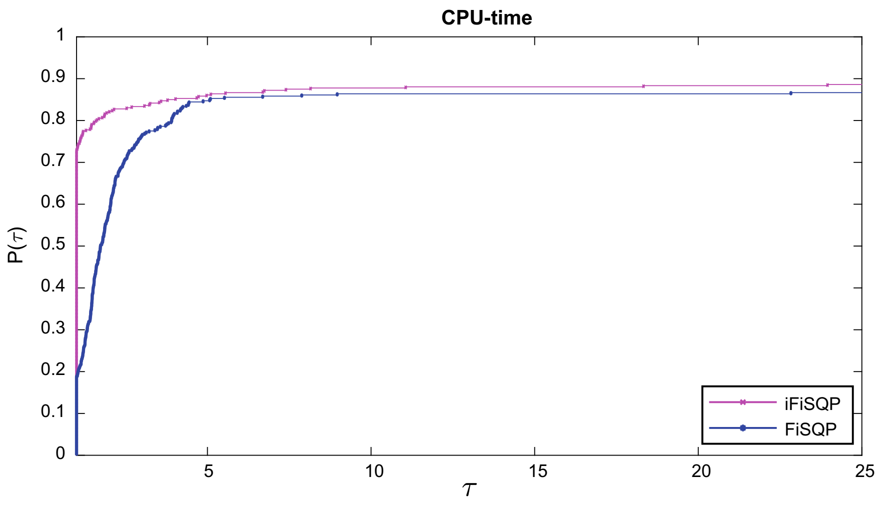


Fig. 1 The performance profile for CPU times of *i*FiSQP and FiSQP methods

HS101. Figure 1 shows that computing inexact solutions of QP subproblems, instead of exact solutions, and computing the accelerating direction only on almost feasible iterations apear to improve the roboustness and efficiency of the SQP algorithm.

6 Concluding Remarks

An inexact nonmonotone sequential quadratic programming algorithm was proposed. Similar to the algorithms of [24, 25], in each step of the algorithm a linear program is solved and the steering direction is obtained. Using the steering direction, either an infeasible stationary point is detected or a strongly convex feasible quadratic programming (QP) subproblem is constructed. It should be noted that we compute an inexact (or truncated) solution of the QP subproblem, as opposed to an exact solution of [24, 25]. In order to accelerate the rate of convergence, in certain iterations an accelerating direction is computed. Using a backtracking line-search procedure and a nonmonotone filter strategy of [25], a step-length is found along the search direction or the accelerating direction. Competitive numerical results attest to the competitiveness of the proposed algorithm.

Acknowledgements The authors thank Sharif University of Technology for supporting this work. They are also grateful to Yueling Loh and Daniel Robinson for providing the source code of the algorithm used in our comparative testing. Thanks are also due to Mehi Al-Baali and the anynomous reviewer for their constructive comments and suggestions.

References

1. Ahmadzadeh, H. and Mahdavi-Amiri, N.: A competitive inexact nonmonotone filter SQP method: convergence analysis and numerical results. Submitted in revised form to the special issue of OMS for the Fifth International Conference on Numerical Analysis and Optimization, 41 pages (2020).
2. Ansari, M.R. and Mahdavi-Amiri, N.: A robust combined trust region-line search exact penalty projected structured scheme for constrained nonlinear least squares. Optim. Methods Softw. 30(1), 162–190 (2015).
3. Barclay, A., Gill, P.E. and Rosen, J.B.: SQP methods and their application to numerical optimal control. In Variational calculus, optimal control and applications, pp. 207–222. Birkhäuser, Basel (1998).
4. Bartholomew-Biggs, M.: Nonlinear Optimization with Financial Applications. Springer Science & Business Media (2006).
5. Bartholomew-Biggs, M.: Nonlinear Optimization with Engineering Applications. Springer Science & Business Media (2008).
6. Bertsekas, D.P.: Constrained Optimization and Lagrange Multiplier Methods. First edn. Athena Scientific (1996).
7. Betts, J.T.: Practical Methods for Optimal Control and Estimation Using Nonlinear Programming. SIAM publications (2010).
8. Biegler, L.T.: Nonlinear Programming: Concepts, Algorithms, and Applications to Chemical Processes. SIAM publications (2010).
9. Biegler, L.T., Ghattas, O., Heinkenschloss, M. and van Bloemen Waanders, B.: Large-scale PDE-constrained optimization: an introduction. In Large-Scale PDE-Constrained Optimization. Springer (2003).
10. Burke, J.V., Curtis, F.E. and Wang, H.: A sequential quadratic optimization algorithm with rapid infeasibility detection. SIAM J. on Optim. 24(2), 839–872 (2014).
11. Boggs, P.T. and Tolle, J.W.: Sequential quadratic programming. Acta Numer. 4(1), 1–51 (1995).
12. Byrd, R.H., Curtis, F.E. and Nocedal, J.: Infeasibility detection and SQP methods for nonlinear optimization. SIAM J. Optim. 20(5), 2281–2299 (2010).
13. Byrd, R.H., Lopez-Calva, G. and Nocedal, J.: A line search exact penalty method using steering rules. Math. Program. 133(1), 39–73 (2012).
14. Curtis, F.E., Johnson, T., Robinson, D.P. and Wächter, A.: An inexact sequential quadratic optimization algorithm for nonlinear optimization. SIAM J. Optim. 24(3), 1041–1074 (2014).
15. Deb, K.: Optimization for Engineering Design: Algorithms and Examples. PHI LearningPvt. Ltd. (2012).
16. Dembo, R.S. and Tulowitzki, U.: Sequential truncated quadratic programming methods. In Numerical Optimization. Proceedings of the SIAM Conference on Numerical Optimization, Boulder, Colorado, June 12-14, 1984, 20, pp. 83–101, SIAM Publications.
17. Dixit, A.K. and Sherrerd, J.J.F.: Optimization in Economic Theory. Oxford University Press on Demand (1990).
18. Dolan, E.D. and Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Program. 91(2), 201–213 (2002).
19. Fletcher, R. and Leyffer, S.: Solving mixed integer nonlinear programs by outer approximation. Math. Program. 66(1), 327–349 (1994).
20. Fletcher, R., Leyffer, S. and Toint, P.L.: On the Global Convergence of a Filter–SQP Algorithm. SIAM J. Optim. 13(1), 44–59 (2002).
21. Gerard, C. and Reha, T.: Optimization Methods in Finance. Mathematics, Finance and Risk, Cambridge University Press (2007).
22. Gill, P.E. and Wong, E.: Sequential quadratic programming methods. In Lee, J., Leyffer, S. (eds.) Mixed Integer Nonlinear Programming, pp. 147–224. Springer, New York, NY, (2012).
23. Gould, N.I.M., Loh, Y. and Robinson, D.P.: A Nonmonotone Filter SQP Method: Local Convergence and Numerical Results. preprint, RAL-P-2014-012R (2014).

24. Gould, N.I.M., Loh, Y. and Robinson, D.P.: A filter method with unified step computation for nonlinear optimization. SIAM J. Optim. 24(1), 175–209 (2014).
25. Gould, N.I.M., Loh, Y. and Robinson, D.P.: A nonmonotone filter SQP method: Local convergence and numerical results. SIAM J. Optim. 25(3), 1885–1911 (2015).
26. Gould, N.I.M., Orban, D. and Toint, P.L.: GALAHAD, a library of thread-safe Fortran 90 packages for large-scale nonlinear optimization. ACM Trans. Math. Softw. (TOMS) 29(4), 353–372 (2003).
27. Gould, N.I.M., Orban, D. and Toint, P.L.: CUTEst: a constrained and unconstrained testing environment with safe threads for mathematical optimization. Comput. Optim. Appl. 60(3), 545–557 (2015).
28. Gould, N.I.M. and Robinson, D.P.: A second derivative SQP method: Global convergence. SIAM J. Optim. 20(4), 2023–2048 (2010).
29. Gould, N.I.M. and Robinson, D.P.: A second derivative SQP method: Local convergence and practical issues. SIAM J. Optim. 25(3), 2049–2079 (2010).
30. Gregory, J.: Constrained Optimization in the Calculus of Variations and Optimal Control Theory. Chapman and Hall/CRC Press (2018).
31. Herzog, R. and Kunisch, K.: Algorithms for PDEconstrained optimization. GAMMMitteilungen 33(2), 163–176 (2010).
32. HSL(2013). A collection of Fortran codes for large scale scientific computation. http://www.hsl.rl.ac.uk
33. IBM, ILOG CPLEX: High-Performance Software for Mathematical Programming and Optimization (2006).
34. Lee, J. and Leyffer, S.: Mixed Integer Nonlinear Programming, Vol. 154. Springer Science & Business Media (2011).
35. Leyffer, S. :Integrating SQP and branch-and-bound for mixed integer nonlinear programming. Comput. Opim. Appl. 18(3), 295–309 (2001).
36. Mahdavi-Amiri, N. and Bartels, R.H.: Constrained nonlinear least squares: An exact penalty approach with projected structured quasi-Newton updates. ACM Trans. Math. Softw. (TOMS) 15(3), 220–242 (1989).
37. Mangasarian, O.L. and Fromovitz, S.: The Fritz John necessary optimality conditions in the presence of equality and inequality constraints. J. Math. Anal. Appl. 17(1), 37–47 (1967).
38. Maratos, N.: Exact penalty function algorithms for finite dimensional and control optimization problems. Ph.D. diss., Imperial College London (University of London) (1978).
39. Murray, W.and Prieto, F.J.: A sequential quadratic programming algorithm using an incomplete solution of the subproblem. SIAM J. Optim. 5(3), 590–640 (1995).
40. Nocedal, J. and Wright, S.J.: Numerical Optimization. 2nd ed., Springer, New York, NY, USA (2006).
41. Rao, A.V.: A survey of numerical methods for optimal control. Adv. Astronaut. Sci. 135(1), 497–528 (2009).
42. Wächter, A.: An interior point algorithm for large-scale nonlinear optimization with applications in process engineering. Ph.D. diss., Carnegie Mellon University (2002).
43. Wächter, A. and Biegler, L.T.: Line search filter methods for nonlinear programming: motivation and global convergence. SIAM J. Optim. 16(1), 1–31 (2005).
44. Zhou, M., Cai, Y., Su, H., Wozny G. and Pan, H.: A survey on applications of optimization-based integrating process design and control for chemical processes. Chem. Eng. Commun. 205(10), 1365–1383 (2018).

Behavior of Limited Memory BFGS When Applied to Nonsmooth Functions and Their Nesterov Smoothings

Azam Asl and Michael L. Overton

Abstract The motivation to study the behavior of limited-memory BFGS (L-BFGS) on nonsmooth optimization problems is based on two empirical observations: the widespread success of L-BFGS in solving large-scale smooth optimization problems, and the remarkable effectiveness of the full BFGS method in solving small to medium-sized nonsmooth optimization problems, based on using a gradient, not a subgradient, oracle paradigm. We first summarize our theoretical results on the behavior of the scaled L-BFGS method with one update applied to a simple convex nonsmooth function that is unbounded below, stating conditions under which the method converges to a non-optimal point regardless of the starting point. We then turn to empirically investigating whether the same phenomenon holds more generally, focusing on a difficult problem of Nesterov, as well as eigenvalue optimization problems arising in semidefinite programming applications. We find that when applied to a nonsmooth function directly, L-BFGS, especially its scaled variant, often breaks down with a poor approximation to an optimal solution, in sharp contrast to full BFGS. Unscaled L-BFGS is less prone to breakdown but conducts far more function evaluations per iteration than scaled L-BFGS does, and thus it is slow. Nonetheless, it is often the case that both variants obtain better results than the provably convergent, but slow, subgradient method. On the other hand, when applied to Nesterov's smooth approximation of a nonsmooth function, scaled L-BFGS is generally much more efficient than unscaled L-BFGS, often obtaining good results even when the problem is quite ill-conditioned. Summarizing, we find that although L-BFGS is often a reliable method for minimizing ill-conditioned smooth problems, when the condition number is so large that the function is effectively nonsmooth, L-BFGS

A. Asl
University of Chicago Booth School of Business, Chicago, IL, USA
e-mail: azam.asl@chicagobooth.edu

M. L. Overton (✉)
Courant Institute of Mathematical Sciences, New York University, New York, USA
e-mail: mo1@nyu.edu

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*, Springer Proceedings in Mathematics & Statistics 354, https://doi.org/10.1007/978-3-030-72040-7_2

frequently fails. This behavior is in sharp contrast to the behavior of full BFGS, which is consistently reliable for nonsmooth optimization problems. We arrive at the conclusion that, for large-scale nonsmooth optimization problems for which full BFGS and other methods for nonsmooth optimization are not practical, it is often better to apply L-BFGS to a smooth approximation of a nonsmooth problem than to apply it directly to the nonsmooth problem.

1 Introduction

We consider the unconstrained optimization problem

$$\min_{x \in \mathbb{R}^n} \quad f(x),$$

where the function f is convex but nonsmooth. By this we mean that it is not differentiable everywhere, and, typically, is not differentiable at minimizers.

Classical approaches to optimization of convex nonsmooth functions generally require the method to have access to an oracle that, given $x \in \mathbb{R}^n$, returns the function value $f(x)$ and a subgradient $g \in \partial f(x)$. The oldest such method, the subgradient method of Shor, which dates to the 1960s, uses the iteration

$$x_{k+1} = x_k - t_k g_k, \text{ for some } g_k \in \partial f(x_k), \tag{1}$$

where $\{t_k\}$ is a pre-determined sequence of positive stepsizes. One well-known result states that, assuming f is convex and bounded below, and provided the steplengths $\{t_k\}$ are square-summable (that is, $\sum_{k=0}^{\infty} t_k^2 < \infty$, and hence the steps are "not too long"), but not summable (that is, $\sum_{k=0}^{\infty} t_k = \infty$, and hence the steps are "not too short"), then convergence of $f(x_k)$ to the minimal value of f must take place [28]. However, despite the strength of this theoretical result, it is well known that convergence to the optimal value is often very slow. This observation led to the development of other algorithms, particularly the bundle methods pioneered by Lemaréchal [23] for nonsmooth convex functions in the mid-1970s, as well as the bundle methods of Kiwiel [22] for nonsmooth, nonconvex problems in the 1980s. As suggested by the name, at each iteration, bundle methods use subgradient information obtained at former iterates as well as at the current iterate x_k to generate the next iterate x_{k+1}. Convergence results are available for these methods, but the computational cost per iteration is significant for large n as computing x_{k+1} from x_k usually requires the solution of a quadratic program in n variables [7, p. 306 and 313]. For more methods for nonsmooth optimization using the subgradient oracle, see [6, 7].

In this paper, we do not use the subgradient oracle paradigm. One reason for this is that, in practice, in the presence of rounding errors, it is often difficult, if not impossible, to determine whether the function f is differentiable at a given point x and hence to return a vector g which can be guaranteed to be a subgradient. A second is that since convex functions (and more generally, locally Lipschitz functions) are differentiable almost everywhere, there is no reason, at least in the absence of exact line searches, to suppose that a method will ever generate a point x_k where f is not differentiable. We therefore use the simpler paradigm that, given x, an oracle returns $f(x)$ and $g = \nabla f(x)$, if f is differentiable at x. Clearly, the subgradient and gradient oracles coincide when f is indeed differentiable at x. What if this is not the case? In theory, we need to assume that the oracle informs the method that the function is not differentiable and therefore a gradient cannot be provided. But in practice, the gradient oracle is simply implemented as if f *is* differentiable at x, breaking ties arbitrarily if necessary. For example, if $f(x)$ is defined as $\max(f_1(x), f_2(x))$, where f_1 and f_2 are smooth functions, the oracle returns $\nabla f_1(x)$ if $f_1(x) > f_2(x)$, $\nabla f_2(x)$ if $f_1(x) < f_2(x)$, and either one if $f_1(x) = f_2(x)$, a property that is difficult to determine anyway in the presence of rounding errors. A subgradient oracle might, in principle, return any convex combination of $\nabla f_1(x)$ and $\nabla f_2(x)$, but there is no reason for it to return anything other than $\nabla f_1(x)$ or $\nabla f_2(x)$, unless, for example, the intent is to return a "steepest descent" subgradient.

Using the gradient oracle instead of the subgradient oracle, we may ask the question: what can be said if we consider the ordinary gradient method,

$$x_{k+1} = x_k - t_k g_k, \text{ where } g_k = \nabla f(x_k), \tag{2}$$

where the steplength t_k is obtained, not from a predetermined sequence, but from a backtracking or Armijo-Wolfe line search? The answer is well known: it is often the case that x_k converges to a point $\bar{x}$ where f is not differentiable and not minimal. See [2] for a historical discussion and for a detailed analysis of an interesting special case. Note that in relevant illustrative examples, it is typical that f *is* differentiable at all iterates $\{x_k\}$, and nondifferentiable only at the limit point $\bar{x}$. This demonstrates that the power of Shor's subgradient method is not so much that it is defined even if f is not differentiable at x_k, as that the acceptable predetermined sequence of stepsizes $\{t_k\}$, unlike stepsizes generated by a line search, ensures convergence to a minimal value.

In the early 2000s, Burke, Lewis and Overton introduced the gradient sampling algorithm [8] for nonsmooth, nonconvex optimization. This method uses the gradient, not the subgradient, oracle paradigm, and a convergence analysis for a rather general class of functions states that, with probability one, the method generates iterates $\{x_k\}$ where f is differentiable and that, if f is convex and bounded below, the function values $\{f(x_k)\}$ converge to the minimal value (more generally, that all cluster points of $\{x_k\}$ are Clarke stationary). However, the cost per iteration is prohibitive when the number of variables is large. See the survey paper [5] for more details, as well as recent work [10] introducing more efficient variants of the gradient sampling method.

As discussed by Lewis and Overton [25], the "full" BFGS quasi-Newton method is a very effective alternative choice for nonsmooth optimization, and its $O(n^2)$ cost per iteration is generally much less than the cost of the bundle or gradient sampling methods, but its convergence results for nonsmooth functions are limited to very special cases. It also uses the gradient, not the subgradient, oracle paradigm.

Since the limited memory variant of BFGS [24] (L-BFGS) costs only $O(n)$ operations per iteration, like the gradient and subgradient methods, it is natural to ask whether L-BFGS could be an effective method for nonsmooth optimization. This is the topic of this paper, which is organized as follows. In Sect. 2, we summarize our theoretical results. Then in Sect. 3, we give extensive experimental results. Although the methods we discuss are applicable to nonconvex problems, we restrict our discussion to the convex case for simplicity, focusing on a difficult nonsmooth problem of Nesterov as well as eigenvalue optimization problems, including those arising from semidefinite programming formulations of the max cut and matrix completion problems. We also consider Nesterov's smooth approximations to these nonsmooth problems. We make some concluding remarks in Sect. 4.

2 Limited Memory BFGS for Nonsmooth Optimization in Theory

We begin this section by defining the concept of an Armijo-Wolfe line search. Then we discuss the full BFGS method on which L-BFGS is based, along with its properties, before discussing our results for the L-BFGS method.

2.1 Armijo-Wolfe Line Search

The BFGS and L-BFGS methods rely on an Armijo-Wolfe line search (also often known as a "weak Wolfe" line search). Let c_1 and c_2 be parameters (the Armijo parameter and the Wolfe parameter respectively) satisfying $0 < c_1 < c_2 < 1$. Let $x_k \in \mathbb{R}^n$ be a given iterate where f is differentiable and let $d_k \in \mathbb{R}^n$ be a descent direction for f from x_k, that is, with $\nabla f(x_k)^T d_k < 0$. We say that a positive steplength t_k satisfies the Armijo condition if

$$f(x_k + t_k d_k) \leq f(x_k) + c_1 t_k \nabla f(x_k)^T d_k, \tag{3}$$

and that t_k satisfies the Wolfe condition if f is differentiable at $x_k + t_k d_k$ and

$$\nabla f(x_k + t_k d_k)^T d_k \geq c_2 \nabla f(x_k)^T d_k. \tag{4}$$

The Armijo condition imposes a "sufficient decrease" in the value of $f(x_k + t_k d_k)$ compared to $f(x_k)$, while the Wolfe condition imposes a "sufficient increase" in the directional derivative $\nabla f(x_k + t_k d_k)^T d_k$ compared to the negative value $\nabla f(x_k)^T d_k$. If f is bounded below on the ray $\{x_k + t d_k : t > 0\}$, then it is known that points t_k satisfying these conditions exist under various assumptions on f, such as f being convex (but not necessarily smooth). For further discussion, including specification of a bracketing line search algorithm based on bisection and doubling to compute t_k, see [25].

While it is often recommended in the context of general smooth nonlinear optimization to set the Armijo parameter to a rather small value such as 10^{-4} [31, p. 62], we note that for satisfactory complexity results for the gradient method applied to smooth, strongly convex functions, the Armijo parameter must *not* be too small [9, pp. 466–468]. An analysis of the gradient method using an Armijo-Wolfe line search applied to a nonsmooth function that we discuss later in this paper (see (9)) was given in [2]. It was shown that, in this case, the success or failure of the gradient method depends critically on the Armijo parameter being sufficiently small, and experiments applying the gradient method to another nonsmooth function (10) confirm the importance of this issue [2, Fig. 6]. On the other hand, as we will see in Sect. 2.3, the choice of Armijo parameter is less critical to the success or failure of L-BFGS when applied to the same function (9).

2.2 Full BFGS

The Broyden-Fletcher-Goldfarb-Shanno (BFGS) method, proposed independently by these four authors in 1970, is a quasi-Newton method that maintains an approximation H_k to the inverse of the Hessian $\nabla^2 f(x_k)$. Let an initial iterate $x_0 \in \mathbb{R}^n$ and an initial positive definite matrix H_0 be given. The BFGS method is defined by the following iteration. For $k = 0, 1, 2, \ldots$, let

$$\begin{aligned}
d_k &= -H_k \nabla f(x_k) \\
x_{k+1} &= x_k + t_k d_k \text{ where } t_k > 0 \text{ satisfies (3), (4)} \\
s_k &= t_k d_k && (5)\\
y_k &= \nabla f(x_{k+1}) - \nabla f(x_k) && (6)\\
\rho_k &= \frac{1}{s_k^T y_k} \\
H_{k+1} &= \left(I - \rho_k s_k y_k^T\right) H_k \left(I - \rho_k y_k s_k^T\right) + \rho_k s_k s_k^T && (7)
\end{aligned}$$

Equation (7) is called the BFGS update. It is easy to check that the Wolfe condition ensures that $s_k^T y_k > 0$, and it is well known that the positive definiteness of H_{k+1} follows as a consequence. It is also clear that H_{k+1} can be computed in $O(n^2)$ operations.

The most important convergence property of BFGS is due to Powell [33]. It states that if f is twice continuously differentiable and strongly convex on the level set $S = \{x : f(x) \le f(x_0)\}$, then the sequence $\{x_k\}$ converges to the global minimizer. Furthermore, the convergence theory of Dennis and Moré states that if we further assume that the Hessian of f, $\nabla^2 f(x)$, is Lipschitz continuous at the global minimizer, then the rate of convergence is superlinear. See [31] for more details.

For many years, BFGS was not considered as a possible stand-alone method for nonsmooth optimization, although Lemaréchal briefly mentioned this possibility in a 1982 technical report. Instead, Lemaréchal and others focused on using the BFGS update to provide second-order information to bundle methods [7, p. 313]. In contrast, Lewis and Overton [25] advocated using the pure BFGS method, with no bundle method component, as a practical method for nonsmooth optimization, using the gradient, not the subgradient, oracle paradigm. The Wolfe condition (4) ensures in theory that f is differentiable at all iterates, although in practice, as already noted, it is difficult to check whether or not f is differentiable at a given point. If f is not differentiable at x, then the gradient oracle may return very different gradients at points that are close to x. The consequence is that, in the nonsmooth case, BFGS builds a very ill-conditioned inverse "Hessian" approximation, with some tiny eigenvalues corresponding to "infinitely large" curvature in the directions defined by the associated eigenvectors. This phenomenon, illustrated in [25, Fig. 4], is apparently what makes BFGS so effective for nonsmooth optimization. Remarkably, the condition number of the inverse Hessian approximation often reaches 10^{16} before the method breaks down numerically, usually because, as a consequence of rounding error, the line search is unable to return t_k satisfying the Armijo condition even though, in principle, an Armijo-Wolfe step exists. Convergence of $f(x_k)$ to the minimal value of f often appears to be linear (not superlinear, as in the smooth case).

Although empirically BFGS works very well when f is nonsmooth, convergence results are limited to a few special cases. The following results hold for all x_0 as long as f is differentiable at x_0 and for all positive definite H_0. We use $x^{(i)}$ to denote the ith entry of the vector x.

- $n = 1$ with $f(x) = |x|$: the sequence generated by BFGS converges linearly to zero and is related to a certain binary expansion of the starting point [25].
- $f(x) = |x^{(1)}| + \sum_{i=2}^{n} x^{(i)}$: eventually a direction is identified on which f is unbounded below [36]; see also [26].
- $f(x) = \|x\|_2$: iterates converge to the origin [17]. This is a special case of a more general result whose proof is based on Powell's theorem mentioned above.

As far as we know, even the case $f(x) = |x^{(1)}| + (x^{(2)})^2$ remains open!

BFGS has been used successfully in many practical applications of nonsmooth optimization including the design of fixed-order controllers for linear dynamical systems with input and output, shape optimization for spectral functions of Dirichlet-Laplacian operators and condition metric optimization. For more details, see [25, pp. 159–160]. Software is available in the HANSO package.[1]

Finally, BFGS has also proved useful for optimization problems with nonsmooth constraints. Consider problems of the form

$$\begin{aligned} &\min \ f(x) \\ &\text{subject to } c_i(x) \le 0, \quad i = 1, \ldots, p \end{aligned}$$

[1] http://www.cs.nyu/overton/software/hanso/.

where f and $c_1, \dots, c_p$ may not be differentiable at local minimizers. A successive quadratic programming (SQP) method based on BFGS was introduced by [11] to solve problems of this form, and applied to problems in static output feedback control design that involve spectral radius and pseudospectral radius constraints. Although there are no theoretical results, it is typically much more efficient in practice than an SQP gradient sampling method [12] which does have convergence results. Software is available in the GRANSO package.[2]

2.3 Limited Memory BFGS

Unlike the full BFGS method, L-BFGS does not store a matrix approximating the inverse of the Hessian $\nabla^2 f(x)$. Instead, it uses an implicit approximation. Let m be a small integer. Instead of using (7) to update a stored matrix H_k, the matrix H_{k+1} is implicitly defined by applying m BFGS updates successively, starting with an initial matrix $H_{k+1}^{(0)}$ and using the updates defined by the pairs (s_j, y_j), $j = k - m + 1, \dots k$, defined in (5) and (6). This method is called L-BFGS-m. We consider two variants. In the *scaled* variant, with

$$H_{k+1}^{(0)} = \frac{s_k^T y_k}{y_k^T y_k} I, \tag{8}$$

a scaling of the identity matrix often known as Barzilai-Borwein scaling [4] is used to initialize the implicit updating at iteration k. In the *unscaled* variant, $H_{k+1}^{(0)}$ is set to the identity matrix. We also refer to these two different variants of L-BFGS-m as *with* and *without scaling*, respectively. Substantial numerical experience with applying L-BFGS-m to minimize smooth functions shows that the use of scaling is strongly preferred. For more details on L-BFGS, including efficient implementation, see [24, 31].

Our theoretical analysis of the behavior of limited memory BFGS uses the scaled variant with $m = 1$: H_{k+1} is defined by applying just one BFGS update to (8). This method, L-BFGS-1, is sometimes called *memoryless BFGS* [31, p. 180]. We focus on the nonsmooth function

$$f(x) = a|x^{(1)}| + \sum_{i=2}^{n} x^{(i)} \tag{9}$$

with $a \geq \sqrt{n-1}$. This function is obviously unbounded below, but it is bounded below along any steepest descent direction $d = -\nabla f(x)$. One advantage of studying this function is its simplicity, but another is that it is easy to determine whether a method succeeds or fails when it is applied to (9): a method is successful only if it generates a sequence of function values $f(x^{(k)}) \to -\infty$ or identifies a direction d_k on which $\{f(x_k + td_k) : t > 0\}$ is unbounded below. Otherwise, since Armijo-Wolfe

[2] http://www.timmitchell.com/software/granso/.

steps always exist along directions on which f is bounded below, the sequence of function values $f(x^{(k)})$ is well defined, bounded below, and must converge to a non-optimal finite value. The following theorem shows that this last scenario occurs when the Armijo parameter is not sufficiently small compared to the parameter a defining the function definition in (9). The proof is given in [1, Sect. 3.2].

Theorem 1 *Suppose f is defined by* (9) *with $a \geq 2\sqrt{n-1}$. Set x_0 to* any *point with $x_0^{(1)} \neq 0$. If the Armijo parameter c_1 is chosen so that*

$$\frac{1-c_1}{c_1}(n-1) < a^2 + a\sqrt{a^2 - 3(n-1)}$$

holds, then the scaled L-BFGS-1 method is well defined, in the sense that Armijo-Wolfe steps always exist, but fails in the sense that $f(x_k)$ is bounded below as $k \to \infty$.

Note that $a \geq 2\sqrt{n-1}$ implies

$$a^2 + a\sqrt{a^2 - 3(n-1)} \geq 4(n-1) + 2\sqrt{n-1}\sqrt{n-1} = 6(n-1).$$

So, if $c_1 > 1/7$, the failure condition holds. It was proved in [2] that, when applied to (9), the gradient method with an Armijo-Wolfe line search fails in the same sense if the stronger condition

$$\frac{1-c_1}{c_1}(n-1) < a^2$$

on the Armijo parameter holds. So, surprisingly, in some cases scaled L-BFGS-1 fails although the gradient method succeeds in generating $f(x_k) \to -\infty$.

If the specific Armijo-Wolfe bracketing line search given in [25] is used, we obtain a failure condition for scaled L-BFGS-1 that is *independent* of the Armijo parameter. The proof of the next result is given in [1, Sect. 3.3].

Theorem 2 *Suppose f is defined by* (9) *with $a \geq 2\sqrt{n-1}$. Set x_0 to* any *point with $x_0^{(1)} \neq 0$. If scaled L-BFGS-1 is implemented using the Armijo-Wolfe bracketing line search of [25], the method is well defined in the sense that the line search always returns an Armijo-Wolfe steplength t_k, but fails in the sense that $f(x_k)$ is bounded below as $k \to \infty$.*

In fact, in the experiments reported in [1] we observe that $a \geq \sqrt{3(n-1)}$ suffices for the method to fail. Furthermore, experiments indicate that, for any number of updates m, there is a threshold for the parameter a beyond which scaled L-BFGS-m always fails in the same sense: $f(x_k)$ is bounded below as $k \to \infty$. This threshold value for a increases with m.

3 Limited Memory BFGS for Nonsmooth Optimization in Practice

In this section we carry out extensive experiments applying L-BFGS-m, with and without scaling, using the Armijo-Wolfe bracketing line search of [25], to several challenging nonsmooth examples. All of the functions we consider from now on are bounded below. In most cases, in assessing whether a method succeeds or fails, we compare the final computed result with the known optimal value. We also compare the results with those obtained by the full BFGS method using the same line search. We used the HANSO package which implements both full and limited-memory BFGS, with the default settings for the Armijo and Wolfe parameters: $c_1 = 10^{-4}$ and $c_2 = 0.5$. In many cases, we also compared the results to those obtained by the subgradient method with predetermined stepsizes $t_k = 1/k$. All methods, including the subgradient method, are implemented using the gradient oracle paradigm discussed in Sect. 1: no attempt is made to determine whether the objective function is differentiable at a given iterate x_k. Instead of using a stopping criterion such as discussed in [25, p. 159], we run each method until it reaches a maximum number of function evaluations or iterations, or the line search fails to find an acceptable step (either because a limit on the number of bisections in the line search is exceeded or because of rounding errors). If the method terminates because of failure in the line search when $f(x_k)$ is not a good approximation to the optimal value, we say that the method "breaks down". We find, in general, that this behavior is typical for L-BFGS, but not for full BFGS. For this reason we also compare these methods on *smoothed* versions of the objective functions.

We begin with a difficult nonsmooth problem devised by Nesterov, and then we go on to consider eigenvalue optimization and semidefinite programming problems.

3.1 Nesterov's Les Houches Problem

We now consider a nonsmooth function introduced by Nesterov. The function is defined by

$$f(x) = \max\{|x^{(1)}|,\ |x^{(i)} - 2x^{(i-1)}|,\ \ i = 2, \ldots, n\}. \tag{10}$$

Let $\hat{x}^{(1)} = 1, \hat{x}^{(i)} = 2\hat{x}^{(i-1)} + 1, i = 2, \ldots, n$. Then $f(\hat{x}) = 1 = f(\mathbf{1})$, where $\mathbf{1} = [1, \ldots, 1]^T$, although $\|\hat{x}\|_\infty \approx 2^n$ and $\|\mathbf{1}\|_\infty = 1$, so the level sets of f are very distorted. The minimizer is $\mathbf{0} = [0, \ldots, 0]^T$ with $f(\mathbf{0}) = 0$. We call this Nesterov's "Les Houches" function [30].

In Fig. 1, we show results for L-BFGS-1 (memoryless BFGS), with and without scaling, as well as full BFGS and the subgradient method, for the Les Houches problem with $n = 500$. We display all the function values that were computed, including those computed in the bracketing line search. We put a limit of 10,000 function evaluations on each method. The starting point x_0 (used by all the methods) was drawn

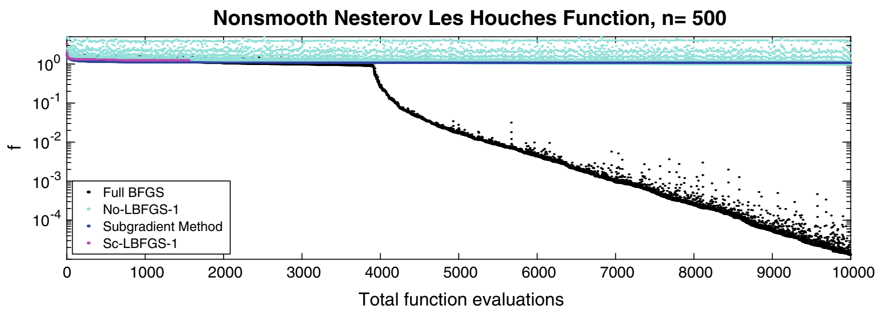


Fig. 1 Comparing full BFGS, L-BFGS-1 with and without scaling and the subgradient method on the nonsmooth Les Houches problem (10) with $n = 500$. Here and below, "No-LBFGS" and "Sc-LBFGS" refer to the methods without scaling and with scaling respectively

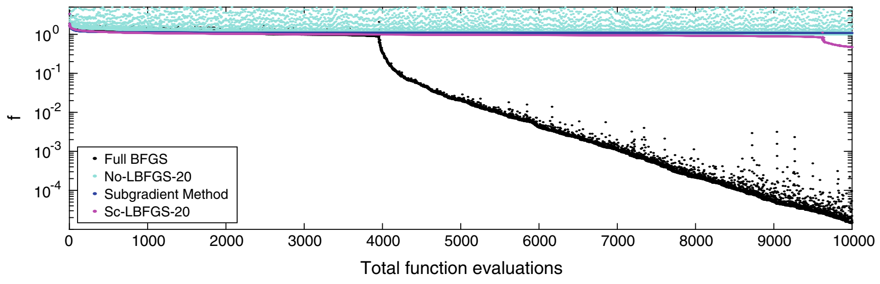


Fig. 2 Comparing full BFGS, L-BFGS-20 with and without scaling and the subgradient method on the nonsmooth Les Houches problem (10) with $n = 500$

randomly from the ball of radius 0.1 centered at the vector of all ones, using the standard normal distribution.

We see from Fig. 1 that scaled L-BFGS-1 (magenta dots) breaks down, with failure in the line search, after fewer than 2000 function evaluations. In contrast, unscaled L-BFGS-1 (cyan) runs for the full 10,000 function evaluations. However, its scattered plot indicates that the method performs many function evaluations per iteration in the line search, indicating that, not surprisingly given its name, the search directions it generates are not well scaled. Despite this, the method obtains a somewhat lower answer than the subgradient method (dark blue). Full BFGS (black) performs much better than any of the other methods, reducing the function value to about 10^{-5} (recall that the optimal value is zero). It is interesting to note that its convergence rate picks up rapidly right after it has lowered the function value down to 1. We do not know the reason for this.

We now increase the number of updates m from 1 to 20 and repeat this experiment: see Fig. 2. Unlike scaled L-BFGS-1, scaled L-BFGS-20 does not quit early, and furthermore it also demonstrates a suddenly faster convergence rate toward the end of the experiment similar to that of full BFGS (although the final answer it obtains is not nearly as accurate as full BFGS). It obtains a function value of size 0.47, whereas unscaled L-BFGS-20 gets a final answer of about 0.998 and the subgradient method obtains 1.083. In this experiment, increasing the number of updates from 1 to 20 enhanced the performance of scaled L-BFGS far more than it did for unscaled L-BFGS.

The conclusion from these experiments is that with a small m, unscaled L-BFGS-m performs better and with a larger m it is the scaled variant which performs better. However, neither method performs nearly as well as full BFGS.

3.2 Smoothed Versions of Nesterov's Les Houches Problem

Since L-BFGS-m performed poorly on (10), we consider instead applying it to a smoothed version. Let

$$A = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ -2 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & 0 & -2 & 1 \end{bmatrix}.$$

Then, (10) is equivalent to

$$f(x) = g(Ax), \tag{11}$$

where $g : \mathbb{R}^n \to \mathbb{R}$ is defined by

$$g(y) = \max\{|y^{(i)}| : i = 1, 2, \dots, n\}. \tag{12}$$

Consider the Nesterov smoothing [29, 35] of the vector-max problem (12):

$$g_\mu(y) = \mu \log \sum_{i=1}^{n} \left(e^{y^{(i)}/\mu} + e^{-y^{(i)}/\mu}\right) - \mu \log(2n). \tag{13}$$

Without the constant term $-\mu \log(2n)$, this function is sometimes known as the softmax function [27, p. 205]. The unique minimizer of $g_\mu(y)$ is $y_\mu^* = \mathbf{0}$ with $g_\mu^* = 0$. Since A is full-rank (although one of its singular values converges to zero as $n \to \infty$), via (11) we know that the unique minimizer of $f_\mu(x) = g_\mu(Ax)$, x_μ^*, is also $\mathbf{0}$, with the same optimal value $f_\mu^* = 0$, and it can be verified that

$$\|\nabla^2 f_\mu(\mathbf{0})\|_2 = \frac{1}{n\mu}\|A\|_2^2. \tag{14}$$

We follow the standard approach [9, Sect 9.1] to defining the condition number of the strongly convex function f_μ as

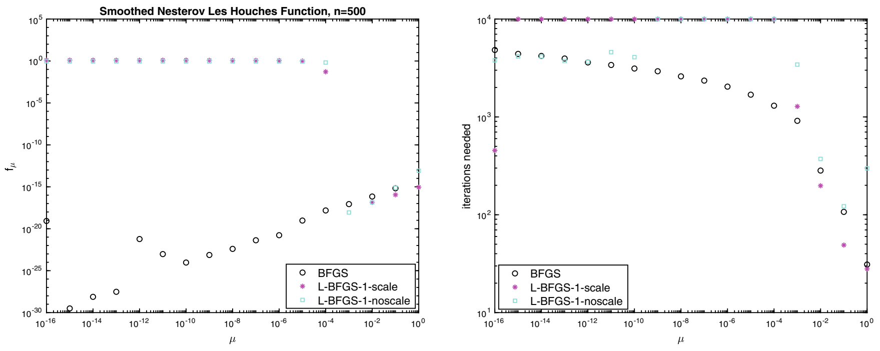


Fig. 3 Comparing BFGS and L-BFGS-1 with and without scaling on the smoothed Les Houches function (13) for $n = 500$. The left panel shows the final function value and the right panel shows the iteration count, both as a function of the smoothing parameter μ. The maximum number of iterations is set to 10^4

$$\kappa(f_\mu) = \left(\max_{x\in S} \|\nabla^2 f_\mu(x)\|_2\right)\left(\max_{x\in S} \|(\nabla^2 f_\mu(x))^{-1}\|_2\right) \tag{15}$$

where $S = \{x : f(x) \le f(x_0)\}$. For small μ, the second factor is enormous as all eigenvalues of $\nabla^2 f_\mu(x_0)$ are tiny. Using (14) as a lower bound for the first factor we conclude that, for small μ,

$$\kappa(f_\mu) \gg \frac{1}{\mu}.$$

We now report on experiments we conducted applying full BFGS and L-BFGS, with and without scaling, to the smooth function f_μ, with $n = 500$ as before. All of the methods start from the same initial point used earlier. The left panel of Fig. 3 shows the final function value computed by full BFGS and L-BFGS-1 with and without scaling as a function of the smoothing parameter μ using a log-log scale. Let us focus first on the results for full BFGS (black circles).

BFGS always finds a solution with magnitude smaller than 10^{-15}, even for a very small μ, when the function is extremely ill conditioned. This is a remarkable property of full BFGS: its accuracy does not deteriorate significantly[3] as the condition number $\kappa(f_\mu)$ of the smoothed problem blows up with $\mu \to 0$. In fact, when μ is sufficiently small, say $\mu = 10^{-16}$ (approximately the rounding unit in IEEE double precision used by MATLAB), the smoothed problem is essentially equivalent to the original nonsmooth problem when rounding errors are taken into account, so the left panel shows the transition of the accuracy of full BFGS from smoothed variants of the problem to the limiting nonsmooth problem. The right panel shows the number of iterations that were required, again as a function of the smoothing parameter μ

[3] Surprisingly, the accuracy increases somewhat as μ decreases, but this is at the level of rounding errors and could perhaps be explained by a rounding error analysis. Certainly the scatter at the bottom left corner of the left panel of Fig. 3 is a consequence of rounding error.

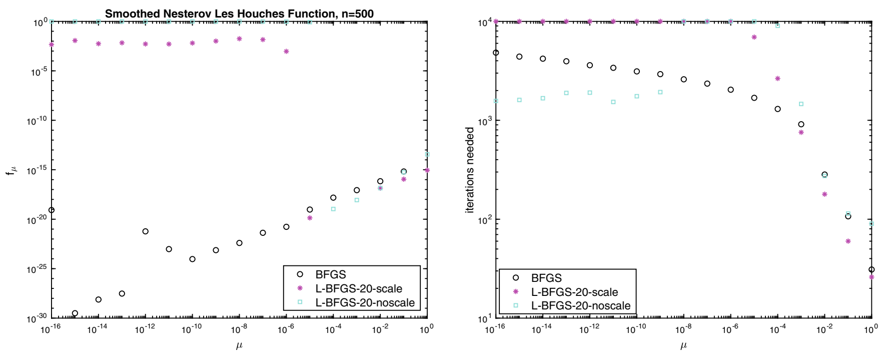


Fig. 4 Comparing BFGS and L-BFGS-20 with and without scaling on the smoothed Les Houches function (13) for $n = 500$. The left panel shows the final function value and the right panel shows the iteration count, both as a function of the smoothing parameter μ. The maximum number of iterations is set to 10^4

and again using a log-log scale. The maximum number of iterations (not function evaluations) was set to 10^4 for each μ. Remarkably, we see that the number of iterations required for full BFGS to accurately minimize f_μ does not significantly increase as $\mu \to 0$, even though the condition number $\kappa(f_\mu)$ blows up as μ decreases to zero, and the number required for the effectively nonsmooth instance $\mu = 10^{-16}$ is not much more than the number required for much better conditioned smoothed problems arising from moderate values of μ.

The results for L-BFGS-1 are very different. Unscaled L-BFGS-1 (cyan squares) finds an accurate answer for $\mu \geq 10^{-3}$, but the number of iterations required increases rapidly as μ is decreased further so the iteration limit is reached for μ ranging from 10^{-4} to 10^{-9}. However, starting with $\mu = 10^{-10}$, unscaled L-BFGS-1 breaks down before reaching the maximum number of iterations. The behavior of scaled L-BFGS-1 (magenta asterisks) is similar except that it breaks down only for $\mu = 10^{-16}$. Note that, since we are displaying the number of iterations, not the number of function evaluations, the performance of unscaled L-BFGS-1 looks better than it really is: the scaled version is computing substantially fewer function evaluations per line search.

When we increase the number of updates to $m = 20$, scaled L-BFGS reacts much better than unscaled L-BFGS; see Fig. 4. Unscaled L-BFGS-20 finds accurate answers for $\mu \geq 10^{-4}$ before hitting the iteration limit, while the scaled version does so for $\mu \geq 10^{-5}$. Furthermore, when the maximum iteration limit is reached, scaled L-BFGS-20 achieves an answer of magnitude $\approx 10^{-2}$, whereas unscaled L-BFGS-20 is still giving an answer of magnitude $\approx 10^0$, similar to unscaled L-BFGS-1. However, overall, both are still doing poorly compared to full BFGS.

Our conclusions from this subsection are consistent with the generally accepted wisdom concerning L-BFGS. For smooth problems, even very ill-conditioned ones, it is best to use the scaled version of L-BFGS, and choosing the number of updates m to be larger rather than smaller gives better performance, although, in contrast to full BFGS, the number of iterations required increases significantly with the conditioning of the problem. Again in contrast with full BFGS, when the ill conditioning increases to the nonsmooth limit implicit in consideration of rounding errors, scaled L-BFGS generally fails to converge to an optimal solution. However, for the smooth but ill-conditioned problems considered in this subsection, unscaled L-BFGS offers no advantage compared to scaled L-BFGS. The most important conclusion is that, while applying full BFGS directly to nonsmooth problems works remarkably well, this is not the case for L-BFGS; at least for the Les Houches problem, it is far preferable to apply scaled L-BFGS to a smooth approximation of the nonsmooth problem.

3.3 Max Eigenvalue Problem

Let S^N denote the space of $N \times N$ real symmetric matrices, and let $\mathcal{A} : S^N \to \mathbb{R}^n$ denote a linear operator acting on X as follows:

$$\mathcal{A}X = \begin{bmatrix} \langle A_1, X\rangle \\ \vdots \\ \langle A_n, X\rangle \end{bmatrix}, \tag{16}$$

with $A_i \in S^N$ for $i = 1., \ldots, n$. Its adjoint operator; $\mathcal{A}^T : \mathbb{R}^n \to S^N$, is defined by

$$\mathcal{A}^T y = \sum_{i=1}^{n} y^{(i)} A_i. \tag{17}$$

The Max Eigenvalue problem is to minimize the function

$$f(y) = \lambda_{\max}(C - \mathcal{A}^T y), \tag{18}$$

where $C \in S^N$ and $\lambda_{\max} : S^N \to \mathbb{R}$ denotes largest eigenvalue of its argument. It is well known that $\lambda_{\max}$ is a convex function on S^N. Early papers on eigenvalue optimization include [32].

Assuming the maximum eigenvalue of $C - \mathcal{A}^T y$ is simple, the gradient of f is

$$\nabla f(y) = -\mathcal{A}(qq^T) = -[q^T A_1 q, \ldots, q^T A_n q]^T.$$

Following the gradient oracle paradigm discussed in Sect. 1, we make no attempt to estimate whether or not the maximum eigenvalue is simple at a given iterate. However, at optimal solutions, we generally expect that $C - \mathcal{A}^T y$ has a multiple largest eigenvalue and hence f is not differentiable. As is well known, eigenvalue optimization problems are instances of semidefinite programs, and hence small problems can be solved using CVX [16].

Using the standard normal distribution, we generated a random instance of this problem, defining $\mathcal{A}$ and C with $N = 50$ and $n = 49$. Figure 5 shows the performance

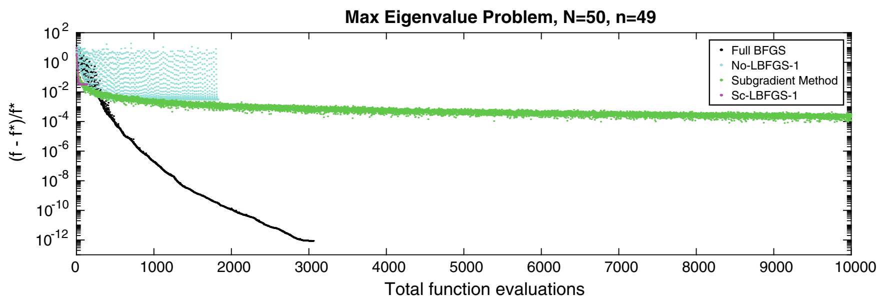


Fig. 5 Comparing BFGS, L-BFGS-1 with and without scaling and the subgradient method on a randomly generated Max Eigenvalue problem (18) with $N = 50$ and $n = 49$

of full BFGS, L-BFGS-1 with and without scaling, and the subgradient method (as before with $t_k = 1/k$) for minimizing (18). As earlier, we display all the function values that were computed, including those computed in the bracketing line search. All methods were terminated after 10^4 function evaluations. Each function evaluation $f(y)$ makes a call to the MATLAB function `eig` to compute all the eigenvalues of $C - \mathcal{A}^T y$.

The vertical axis shows the relative error $|(f - f^*)/f^*|$, where we used the SDPT3 solver in CVX to obtain the optimal solution f^* with accuracy 10^{-14}. As in the experiment on the nonsmooth Les Houches problem reported in Fig. 1, scaled L-BFGS-1 (magenta dots) breaks down early. However, unlike in that experiment, here unscaled L-BFGS-1 (cyan dots) also breaks down early, and as a result the subgradient method (green dots) obtains a better answer, though not nearly as good as full BFGS (black dots).

It's also of interest to examine the multiplicity of the eigenvalues of $C - \mathcal{A}^T y$ at the optimal solution y^* and its computed approximations. From SDPT3, we know that the optimal multiplicity for this problem is 5. Table 1 shows the top 6 eigenvalues of the final answer found by each method. Besides SDPT3, only BFGS and the subgradient method are able to determine the correct optimal multiplicity. BFGS finds a solution with 11 correct digits, while the subgradient method obtains 3 correct digits. Both variants of L-BFGS-1 converge to answers with multiplicity 4. Although the multiplicity is wrong, unscaled L-BFGS-1 gets a better answer than its scaled counterpart, with 2 correct digits, but at the cost of requiring many more function evaluations.

Next, we repeat this experiment on the same problem, increasing the number of L-BFGS updates from $m = 1$ to $m = 20$. See Fig. 6 as well as Table 2 which presents the top 6 eigenvalues for the final answer obtained by scaled and unscaled L-BFGS-20. Both methods find the right multiplicity, with scaled L-BFGS-20 obtaining 3 correct digits and unscaled L-BFGS-20 obtaining 4 correct digits.

In summary, we observe that for the Max Eigenvalue problem, unlike the Les Houches problem, increasing m from 1 to 20 does not result in scaled L-BFGS doing better than unscaled L-BFGS.

Table 1 Top 6 eigenvalues of $C - \mathcal{A}^T y$ for Max Eigenvalue problem (18) where y is computed by SDPT3, full BFGS, scaled/unscaled L-BFGS-1 and the subgradient method for a randomly generated problem with $N = 50$ and $n = 49$. The optimal multiplicity is 5

SDPT3	BFGS	Sc L-BFGS-1	No L-BFGS-1	Subgradient
7.82702970305352	7.82702970306035	8.08455876518360	7.85155000878711	7.82953885641783
7.82702970305349	7.82702970306035	8.08455876518359	7.85155000044960	7.82746561454846
7.82702970305348	7.82702970306034	8.08197715145863	7.85154996050940	7.82673229360627
7.82702970305346	7.82702970306031	8.05541534062475	7.85141043900471	7.82472408900949
7.82702970305334	7.82702970306017	7.84362627205676	7.69075549655739	7.82286893229481
7.70350538019538	7.70350432059448	7.56258926523925	7.48734455288558	7.70188538367848

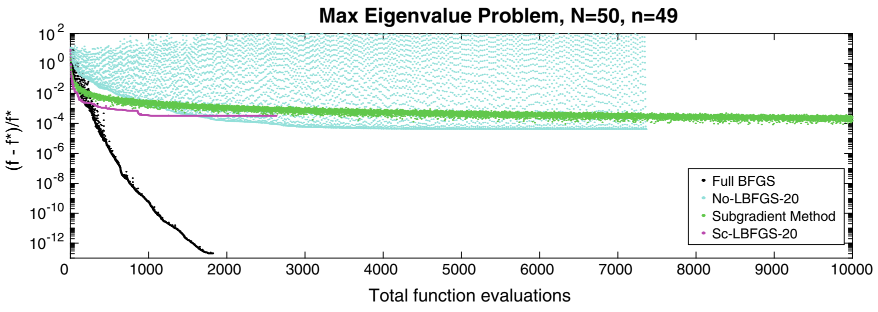


Fig. 6 Comparing BFGS, L-BFGS-20 with and without scaling and subgradient method on a randomly generated Max Eigenvalue problem (18) with $N = 50$ and $n = 49$

Table 2 Top 6 eigenvalues of $C - \mathcal{A}^T y$ for Max Eigenvalue problem (18) where y is computed by scaled and unscaled L-BFGS-20 for the same problem reported in Table 1. The optimal multiplicity is 5

Sc L-BFGS-20	No L-BFGS-20
7.82959659952176	7.82735384547039
7.82959659952176	7.82735380191247
7.82959659952174	7.82735377961470
7.82959659952166	7.82735377236995
7.82959659950328	7.82735372682267
7.62982269438813	7.69181664817458

3.4 Smoothed Max Eigenvalue Problem

Consider now Nesterov smoothing of the Max Eigenvalue function (18) [13]

$$f_\mu(y) = \mu \log \sum_{i=1}^{N} \exp(\lambda_i(C - \mathcal{A}^T y)/\mu) - \mu \log N, \tag{19}$$

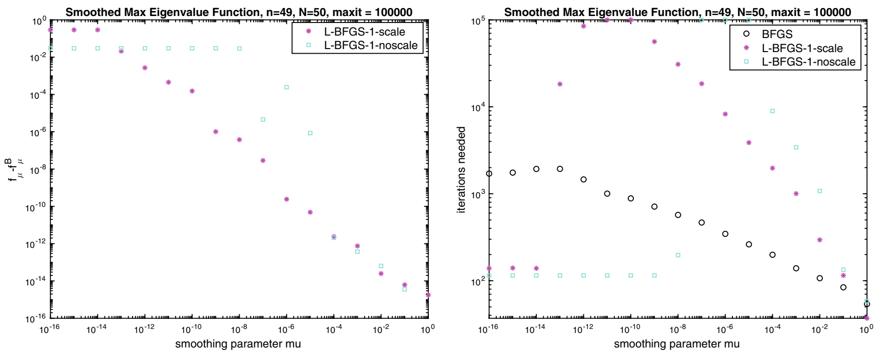


Fig. 7 Comparing L-BFGS-1 with and without scaling on the smoothed Max Eigenvalue problem (19) for $N = 50$ and $n = 49$. The left panel shows the final function value, shifted by f_μ^B, the optimal value computed by BFGS, and the right panel shows the iteration count, both as a function of the smoothing parameter μ. The maximum number of iterations is set to 10^5

where $\lambda_1(W) \geq \lambda_2(W) \geq \cdots \geq \lambda_N(W)$ denote the ordered eigenvalues of a symmetric matrix $W \in S^N$. Thus, λ_1 is equivalent to $\lambda_{\max}$. Unlike the Les Houches problem, where the nonsmooth optimal value is equal to the smoothed optimal value, that is $f^* = f_\mu^* = 0$, for any μ as $\mu \to 0$, the same statement is not true for the Max Eigenvalue problem. The smoothed Max Eigenvalue problem requires a complete eigendecomposition in order to obtain every eigenvalue for a given matrix, and since CVX does not allow such functions but only those that it knows to be convex such as the maximum eigenvalue function, we could not compute the optimal value f_μ^* from CVX. Instead, we use full BFGS with the max number of iterations set to 10^5 to minimize (19) to high accuracy: we denote this computed value by f_μ^B.

In Fig. 7 we report on an experiment using the smoothed version of the same instance of the randomly generated Max Eigenvalue problem as earlier with $N = 50$ and $n = 49$, using L-BFGS-1 to minimize (19). The left panel shows the final value computed by scaled (magenta asterisks) and unscaled (cyan squares) L-BFGS-1 shifted by f_μ^B (the answer found by full BFGS), as a function of the smoothing parameter μ, in log-log scale. The right panel shows the number of iterations as a function of μ, also in log-log scale. The maximum number of iterations is 10^5.

In the left panel we see that for $\mu = 1$ down to $\mu = 10^{-4}$ both methods yield about the same accuracy as each other, but that this deteriorates as μ decreases. Scaled L-BFGS-1 continues to obtain a reasonable approximation to the presumed accurate solution f_μ^B for μ down to 10^{-9}, although this accuracy continues to decrease as μ is reduced. Looking at the right panel, we see that starting with $\mu = 10^{-10}$ scaled L-BFGS-1 hits the maximum iteration limit and starting with 10^{-12} it breaks down before reaching the maximum iteration limit. In contrast, unscaled L-BFGS-1 hits the maximum iteration limit for $\mu = 10^{-5}$ and breaks down for $\mu \leq 10^{-8}$.

Table 3 shows the top 6 eigenvalues of the final answer found by BGFS and L-BFGS-1 for the smoothed Max Eigenvalue problem with $\mu = 10^{-7}$. We also repeat

Table 3 Top 6 eigenvalues of $C - \mathcal{A}^T y$ for Max Eigenvalue problem where y is computed by applying BFGS, scaled L-BFGS-1 and unscaled L-BFGS-1 to f_μ with $\mu = 10^{-7}$, for the same instance of the randomly generated Max Eigenvalue problem as in Table 1. The optimal multiplicity is 5. The first column gives the top 6 eigenvalues of the solution to the original nonsmooth problem

SDPT3	BFGS	Sc L-BFGS-1	No L-BFGS-1
7.82702970305352	7.82702976093363	7.82702978971152	7.82703432112405
7.82702970305349	7.82702968960443	7.82702971836333	7.82703424950540
7.82702970305348	7.82702966768912	7.82702969644540	7.82703422776180
7.82702970305346	7.82702963829977	7.82702966707913	7.82703419808905
7.82702970305334	7.82702954704101	7.82702957585856	7.82703410710019
7.70350538019538	7.70350539089203	7.70374015675041	7.69886385782543

Table 4 Top 6 eigenvalues of $C - \mathcal{A}^T y$ for Max Eigenvalue problem where y is computed by applying BFGS, scaled L-BFGS-20 and unscaled L-BFGS-20 to f_μ with $\mu = 10^{-7}$, for the same instance of the randomly generated Max Eigenvalue problem as in Table 1. The optimal multiplicity is 5

Sc L-BFGS-20	No L-BFGS-20
7.82702976201688	7.82702976326908
7.82702969067290	7.82702969273000
7.82702966877706	7.82702966880138
7.82702963938214	7.82702964120678
7.82702954813253	7.82702954355970
7.70348252862186	7.70344821498698

the optimal top 6 eigenvalues of the minimizer of the original nonsmooth function f found by SDPT3 for the sake of comparison. Note that the result computed by applying scaled L-BFGS-1 to the smoothed problem agrees with the nonsmooth optimal value f^* to 8 digits, compared to 0 digits when applied directly to the nonsmooth problem (Table 1).

We repeated this experiment with $m = 20$, reported in Fig. 8. In the left panel, we observe that the loss of accuracy in L-BFGS as a function of μ is less pronounced with 20 updates. Roughly speaking, overall the error decreases by a factor of 10^{-2}. In the right panel we see that neither scaled nor unscaled L-BFGS-20 reaches the maximum iteration limit, but that both methods break down for sufficiently small μ. The top eigenvalues produced by L-BFGS-20 for $\mu = 10^{-7}$ are shown in Table 4.

Comparing the final computed maximum eigenvalue in Tables 1, 2, 3, and 4, note that full BFGS obtains a more accurate solution when applied to the original nonsmooth problem than it does when applied to the smoothed approximation using $\mu = 10^{-7}$, while the opposite is true for scaled L-BFGS-1 and scaled L-BFGS-20. In summary, as with the Les Houches problem, it is much more effective to apply L-BFGS to the smoothed max eigenvalue problem than directly to the nonsmooth problem. As earlier, this is in sharp contrast to the behavior of full BFGS.

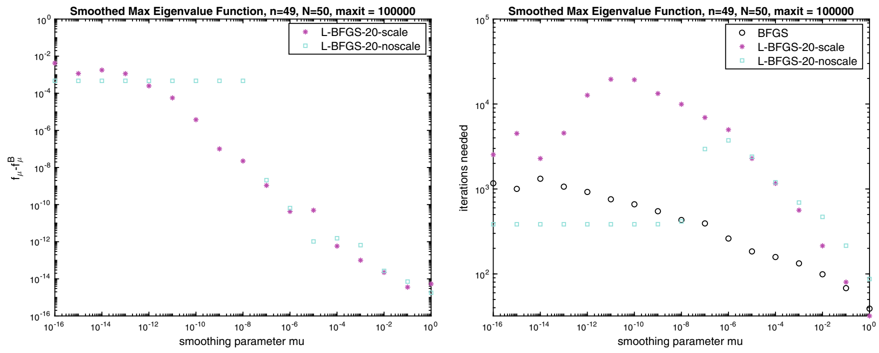


Fig. 8 Comparing L-BFGS-20 with and without scaling on the smoothed Max Eigenvalue problem (19) for $N = 50$ and $n = 49$. The left panel shows the final function value, shifted by f_μ^B, the optimal value computed by BFGS, and the right panel shows the iteration count, both as a function of the smoothing parameter μ. The maximum number of iterations is set to 10^5

3.5 Semidefinite Programming

Consider the following primal and dual semidefinite programs (SDP) in standard form [20, 21]

$$\max_{X \in S^N} \quad \langle C, X\rangle \tag{20}$$
$$\text{subject to} \quad \mathcal{A}X = b \quad \text{and} \quad X \in S_+^N,$$
$$\min_{y \in \mathbb{R}^n} \quad b^T y \tag{21}$$
$$\text{subject to} \quad Z = \mathcal{A}^T y - C \quad \text{and} \quad Z \in S_+^N,$$

where $b \in \mathbb{R}^n$, $C \in S^N$ and $\mathcal{A} : S^N \to \mathbb{R}^n$ is a linear operator as defined in (16) and (17). Here $S_+^N \subseteq S^N$ denotes the cone of positive semidefinite $N \times N$ matrices. Let us assume that strong duality holds, so that the optimal primal and dual values are the same, and that the optimal values are attained. It follows that if X^* is an optimal solution to the primal problem (20) and Z^* is an optimal solution to the dual problem (21), we have $X^* Z^* = 0$. Further assume that X^* is nonzero, and consequently, Z^* has at least one eigenvalue equal to zero. Then the dual problem (21) is equivalent to the following unconstrained eigenvalue optimization problem

$$\min_{y \in \mathbb{R}^n} \quad f(y), \tag{22}$$

with the exact penalty dual function [14]

$$f(y) = b^T y + \alpha \max\{\lambda_{\max}(C - \mathcal{A}^T y),\ 0\}, \tag{23}$$

for sufficiently large α, where $\lambda_{\max}$ denotes maximum eigenvalue as earlier. Note that this exact penalty function differs from the eigenvalue optimization formulation in [21], namely

$$b^T y + \alpha\lambda_{\max}(C - \mathcal{A}^T y)$$

which does not include the $\max\{\cdot\,,\ 0\}$ operator. In that formulation, to give a correct equivalence α must be exactly equal to a critical value, as opposed to greater than or equal to this value. For the SDP problems we consider in the following subsections we already know valid lower bounds for α. Note that at an optimal solution y^* the maximum eigenvalue of $-Z^* = C - \mathcal{A}^T y^*$ is zero, often with multiplicity greater than one, and hence f is nonsmooth at y^*.

3.6 Max Cut Problem

Our first example of semidefinite programming (SDP) arises from the Max Cut problem. This subsection and a subsequent one on the Matrix Completion problem were motivated by the recent paper [14] and the observation made there that the first-order algorithms they used to minimize the penalized dual function (23) arising from Max Cut and Matrix Completion SDP relaxations were slow. Here we compare full BFGS, scaled and unscaled L-BFGS and the subgradient method (again with $t_k = 1/k$) on penalized dual functions arising from Goemans-Williamson SDP relaxations of the Max Cut problem. We note that one of the key ideas in [14] is that, when the primal SDP optimal solution X^* has rank much less than N, an accurate estimate of the optimal value of the SDP obtained from minimizing the penalized dual function allows the use of a novel method for obtaining efficient low-rank solutions to the *primal* SDP even when N is large.

The primal Max Cut SDP relaxation and its dual are [19]:

$$\begin{aligned} &\max_{X} \quad \frac{1}{4}\langle L, X\rangle && (24)\\ &\text{subject to} \quad \operatorname{diag}(X) = \mathbf{1} \quad \text{and} \quad X \in S_+^N, \\ &\min_{y\in\mathbb{R}^n} \quad \mathbf{1}^T y && (25)\\ &\text{subject to} \quad Z = \operatorname{Diag}(y) - \frac{1}{4}L \quad \text{and} \quad Z \in S_+^N, \end{aligned}$$

where L is the Laplacian matrix of a given undirected graph, diag () maps the diagonal of a matrix to a vector, and Diag () maps a vector to a diagonal matrix. Note that these are instances of the primal and dual SDP introduced in (20) and (21), respectively. By definition for the SDP Max Cut problem we have $n = N$. The exact penalty dual function (23) for the Max Cut SDP relaxation is

$$f(y) = \mathbf{1}^T y + \alpha\ \max\{\lambda_{\max}(L - \operatorname{Diag}(y)),\ 0\}. \qquad (26)$$

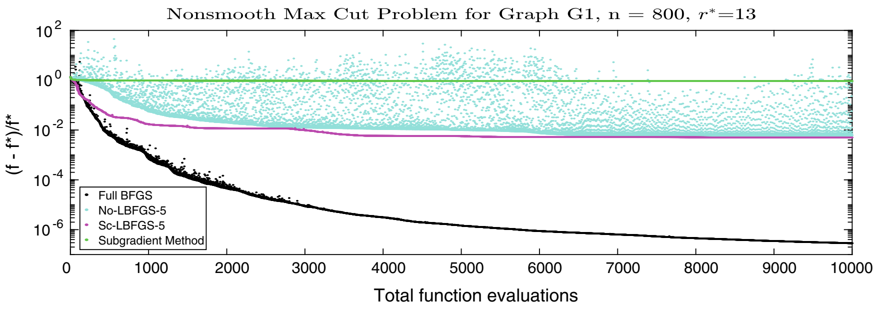


Fig. 9 Comparing BFGS, L-BFGS-5 with and without scaling and the subgradient method on the penalized dual Max Cut problem (26)

Due to the constant trace property of the primal Max Cut SDP (24), the trace (nuclear) norm of the primal optimal solution is known to be N, and hence any solution y^* to the penalized dual max cut problem (26) with $\alpha \geq N$ is also a solution to the dual SDP (25) and vice versa [14, Lemma 6.1].

We picked graph $G1$ from the Gset group in the sparse matrix collection [18] for the following experiment. $G1$ is an unweighted graph with $N = 800$ vertices and the adjacency matrix is a sparse symmetric matrix with 38352 nonzero entries (all equal to 1). Since N is relatively small we can apply the SDPT3 solver via CVX [15] to the primal SDP (24), obtaining the optimal primal and dual value $f^* = 12083.19765$. The rank r^* of the optimal primal solution X^* is 13, and strict complementarity holds, so the nullity of the dual solution Z^* is also 13.

In Fig. 9, we compare the performance of full BFGS, L-BFGS-5 with and without scaling, and the subgradient method (with $t_k = 1/k$) to minimize the penalized dual function (26) with $\alpha = 2N = 1600$. We display all the function values that were computed, with the maximum number of function evaluations set to 10^4. The vertical axis shows the relative error $(f - f^*)/f^*$. In contrast to the two experiments presented in Figs. 5 and 6 for the nonsmooth Max Eigenvalue problem, the results of this experiment are in favor of L-BFGS when compared to the subgradient method. Both scaled L-BFGS-5 (magenta dots) and unscaled L-BFGS-5 (cyan dots) reduce the relative error down to below 10^{-2} while the subgradient method (blue dots) reduces the relative error to about 10^0. Full BFGS (black dots) reduces the relative error to below 10^{-6}.

In Fig. 10, we show the *negative* of the top 20 eigenvalues of the final negative dual slack matrix $-Z$ (equivalently, the smallest 20 eigenvalues of Z) obtained by the four methods, along with values obtained by SDPT3. It is interesting to note that full BFGS approximates the eigenvalues well, in the sense that it clearly separates the first 13 approximately zero eigenvalues from the approximations to the nonzero eigenvalues, although not as decisively as SDPT3. However, the final eigenvalues obtained by L-BFGS-5 are not clearly separated.

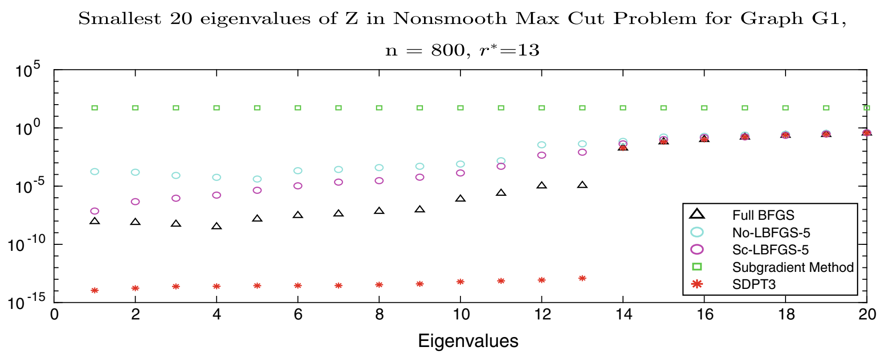


Fig. 10 Comparing smallest 20 eigenvalues of the dual slack matrix Z obtained by BFGS, L-BFGS-5 with and without scaling and the subgradient method on the penalized dual Max Cut problem (26) for the $G1$ graph with $n = 800$. The nullity of the optimal dual slack matrix Z^* is 13. The smallest 20 eigenvalues obtained from SDPT3 are shown as well. The lack of monotonicity at the left end of some of the plots occurs because we actually plotted the absolute values of the ordered largest eigenvalues of $-Z$, and some of these eigenvalues are positive, either because of rounding errors or insufficient accuracy in the optimization

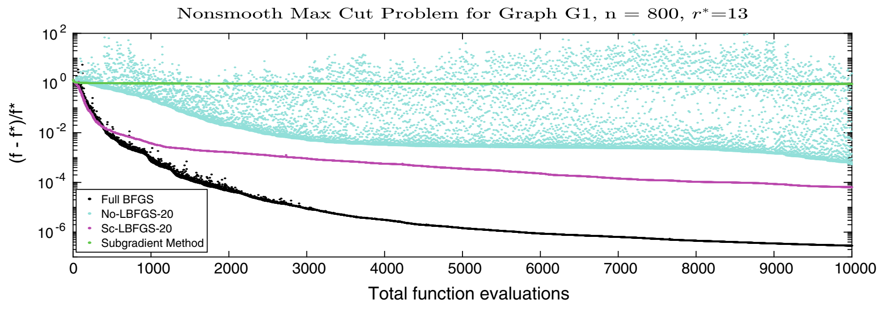


Fig. 11 Comparing BFGS, L-BFGS-20 with and without scaling and the subgradient method on the penalized dual Max Cut problem (26)

Next we increase the number of L-BFGS updates from $m = 5$ to $m = 20$, showing the results in Figs. 11 and 12. The results for BFGS and the subgradient method are shown again for comparison. We see that scaled L-BFGS-20 now reduces the relative error down to 10^{-4} and unscaled to about 10^{-3}, compared to about 10^{-2} for scaled and unscaled L-BFGS-5. However, the eigenvalue plot is similar to the corresponding plot for L-BFGS-5: neither variant is able to discover that the nullity of the optimal dual slack matrix Z^* is 13.

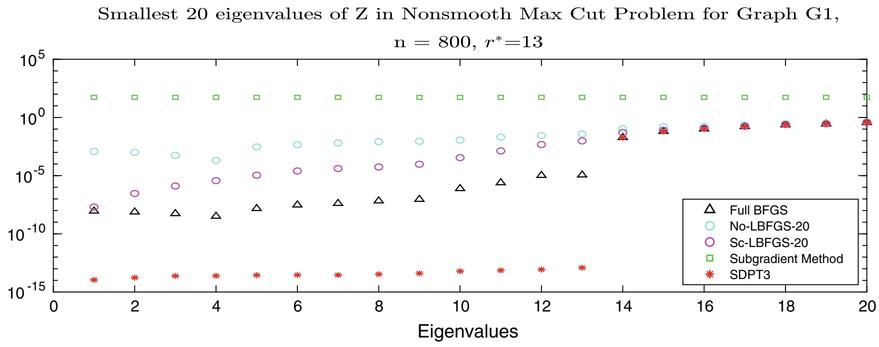


Fig. 12 Comparing smallest 20 eigenvalues of the dual slack matrix Z obtained by BFGS, L-BFGS-5 with and without scaling and the subgradient method on the penalized dual Max Cut problem (26) for $G1$ graph with $n = 800$. The nullity of the optimal dual slack matrix Z^* is 13. The smallest 20 eigenvalues obtained from SDPT3 are shown as well. See legend of Fig. 10 regarding eigenvalue monotonicity

3.7 Smoothed Max Cut Problem

Consider now Nesterov smoothing of the penalized dual Max Cut problem (26):

$$f_\mu(y) = \mathbf{1}^T y + \alpha\mu \log\left(1 + \sum_{i=1}^{n} \exp\left(\lambda_i(L - \mathrm{Diag}\,(y))/\mu\right)\right) - \alpha\mu \log(n+1). \tag{27}$$

Note the presence of the term "1" which does not appear in (19): this reflects the presence of the max$\{\cdot\,,\ 0\}$ operator in the penalty function (23).

Figure 13 shows the results of applying BFGS, scaled L-BFGS-5 and scaled L-BFGS-20 to minimize (27) with $\mu = 10^{-7}$. We do not include the unscaled L-BFGS variants because it seems clear that they offer no advantage on smooth problems. Interestingly, and in contrast to our other experiments with smoothed nonsmooth functions, scaled L-BFGS-20 does better than full BFGS during its first several thousand function evaluations, but eventually it is overtaken by full BFGS. Scaled L-BFGS-5 does relatively poorly.

Table 5 shows the final answers we obtained from full BFGS, scaled L-BFGS-20 and scaled L-BFGS-5 on the smoothed and nonsmooth Max Cut problem. The optimal SDP value f^* is also shown. All three of full BFGS, scaled L-BFGS-5 and scaled L-BFGS-20 obtain better approximations to f^* when applied directly to the nonsmooth problem than when applied to the smoothed problem with $\mu = 10^{-7}$. This is in contrast to what we observed for the Les Houches problem and the Max Eigenvalue problem, where this was true only for full BFGS.

It would be interesting to investigate whether L-BFGS might be useful in the solution of large-scale Max Cut problems or other SDPs. A first step in this direction appears in [3, Sect. 4.2.5], utilizing a variant of the smoothed exact penalty dual

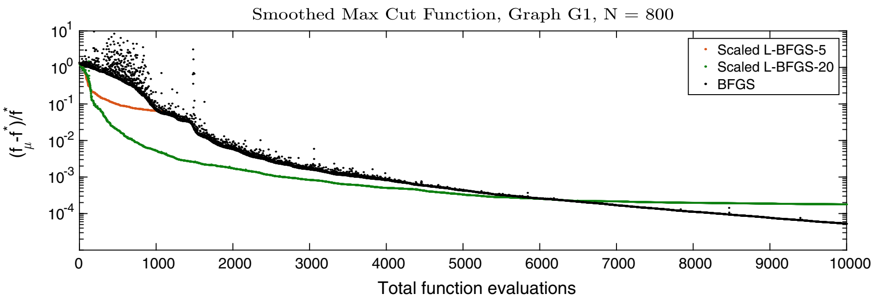


Fig. 13 Comparing scaled L-BFGS-5, scaled L-BFGS-20 and BFGS on Smoothed Max Cut problem (27) for the same problem as in the Nonsmooth Max Cut problem. The smoothing parameter is $\mu = 10^{-7}$

Table 5 Final objective value we obtained from full BFGS, scaled L-BFGS-20 and scaled L-BFGS-5 on the smoothed and nonsmooth Max Cut problem presented in Figs. 13, 11 and 9 respectively. The optimal SDP value f^* is also shown for comparison

SDP-optimal	12083.19765454945
BFGS-smooth	12083.83341694415
BFGS-nonsmooth	12083.20108505506
L-BFGS-20-smooth	12085.35533081401
L-BFGS-20-nonsmooth	12083.97779371002
L-BFGS-5-smooth	12848.47893036591
L-BFGS-5-nonsmooth	12143.81352524515

function (27) that includes only the largest eigenvalues in the smoothing, since these are the ones that dominate (27). This allows the use of MATLAB's `eigs` to compute only the largest few eigenvalues of $C - \mathcal{A}^T y$ via the Lanczos method.

3.8 Matrix Completion Problem

The Matrix Completion problem is as follows. Suppose $\mathcal{X} \in \mathbb{R}^{N_1 \times N_2}$ denotes a low-rank matrix for which we only have access to some of its entries and would like to recover entirely by minimizing the rank over all matrices whose entries agree with the known values. This rank minimization problem is NP-hard, so we relax it by minimizing a well-known convex surrogate for the rank: the nuclear norm (sum of all the singular values). Let Ω be the set of pairs (i, j) for which $\mathcal{X}_{ij}$ is known. Then the nuclear norm minimization problem can be expressed as the following SDP [34]

$$\begin{aligned} \max_{X \in S^{(N_1+N_2)}} \quad & -\operatorname{Tr}(W_1) - \operatorname{Tr}(W_2) \\ \text{subject to} \quad & U_{ij} = \mathcal{X}_{ij}, \ (i,j) \in \Omega, \\ X = & \begin{bmatrix} W_1 & U \\ U^T & W_2 \end{bmatrix} \in S_+^{(N_1+N_2)}. \end{aligned} \tag{28}$$

We write the primal problem in the *max* form in order to be consistent with the SDP form (20), with $N = N_1 + N_2$. Define the constraint $U_{ij} = \mathcal{X}_{ij}$ for $(i, j) \in \Omega$ in linear operator form $\mathcal{B}(U) = b$, where $b \in \mathbb{R}^n$ with $n = |\Omega|$ is the vector consisting of the known entries of $\mathcal{X}$ in some prescribed order and $\mathcal{B} : \mathbb{R}^{N_1 \times N_2} \to \mathbb{R}^n$, with $\mathcal{B}^T$ its adjoint operator. The dual SDP is

$$\begin{aligned} \min_{y \in \mathbb{R}^n} \quad & b^T y \\ \text{subject to} \quad & Z = \begin{bmatrix} I_{N_1} & \mathcal{B}^T(y) \\ \left(\mathcal{B}^T(y)\right)^T & I_{N_2} \end{bmatrix}, \quad Z \in S_+^{(N_1+N_2)}. \end{aligned} \tag{29}$$

The exact penalty dual function (23) for the Matrix Completion problem is then

$$f(y) = b^T y + \alpha \max \left\{ \lambda_{\max} \left(- \begin{bmatrix} I_{N_1} & \mathcal{B}^T(y) \\ \left(\mathcal{B}^T(y)\right)^T & I_{N_2} \end{bmatrix} \right), \ 0 \right\}. \tag{30}$$

For the experiment in this part, we generated a low-rank random matrix $\mathcal{X}$ of size $N_1 = 20$ by $N_2 = 160$ with rank 3. We then selected the ordered pairs in Ω randomly with the probability of each (i, j) being included set to 0.2. For this problem instance we got $|\Omega| = n = 587$. We then applied the various methods to minimize (30) with $\alpha = 2\|X^*\|_* = 2\operatorname{Tr}(X^*) = -2f^* = 3.0796$, where $f^* = -1.5398$ and $X^* \in S_+^{N_1+N_2}$ were obtained from solving the SDP (29) via SDPT3. Note that the optimal value is negative because of the minus sign in the *max* formulation of the primal SDP.

Figure 14 shows the performance of full BFGS, L-BFGS-5 with and without scaling, and the subgradient method (with $t_k = 1/k$). The vertical axis shows the relative error $(f - f^*)/|f^*|$. The maximum number of function evaluations is set to 10^4. As is evident from the plot, both variants of L-BFGS-5 outperform the subgradient method, even though scaled L-BFGS-5 quits early before 5000 evaluations and unscaled L-BFGS-5 just before 10000 evaluations.

Figure 15 presents the negative of the top 20 eigenvalues of the final negative dual slack matrix $-Z$, or equivalently, the 20 smallest eigenvalues of Z, obtained by the four methods, along with values obtained by SDPT3. As before, BFGS is able to separate the zero and nonzero eigenvalues of Z^*, agreeing with SDPT3 that the nullity of Z^* is effectively 11. Note that this is larger than 3, the rank of the original

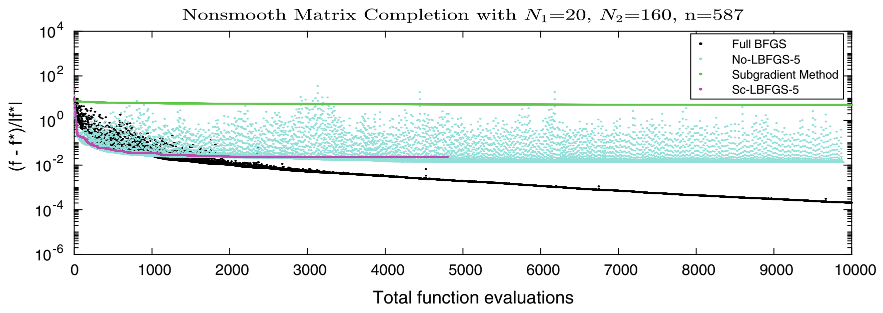


Fig. 14 Comparing BFGS, LBFGS-5 with and without scaling and the subgradient method on the penalized dual Matrix Completion problem (30)

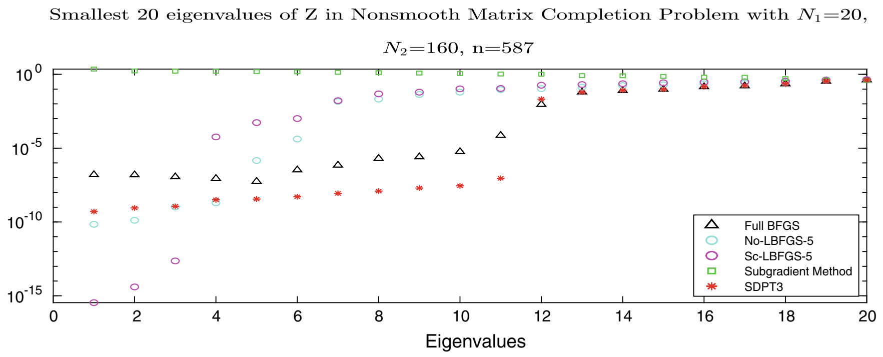


Fig. 15 Comparing smallest 20 eigenvalues of the dual slack matrix Z obtained by BFGS, L-BFGS-5 with and without scaling and the subgradient method on the penalized dual Matrix Completion problem (30). See legend of Fig. 10 regarding eigenvalue monotonicity

matrix $\mathcal{X}$, implying that 20% was not enough observations to reconstruct $\mathcal{X}$. It is interesting that the eigenvalues of the solution found by scaled L-BFGS-5 do suggest a nullity of 3, but this may just be a coincidence.

In the experiment reported in Figs. 16 and 17, we increase m to 20 and again we compare the relative error and the smallest 20 eigenvalues of the dual slack matrix, respectively. In both plots the result from BFGS and the subgradient method are repeated for comparison. In Fig. 16, neither L-BFGS-20 method quits early this time; the unscaled variant gets a slightly lower answer. The eigenvalues shown for L-BFGS-20 in Fig. 17 do not suggest any conclusion about the nullity of Z^*.

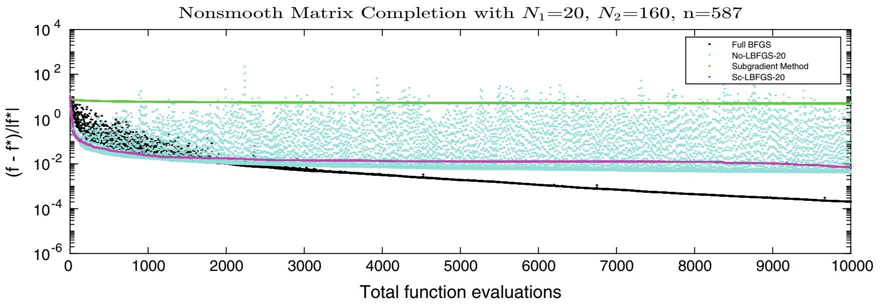


Fig. 16 Comparing BFGS, LBFGS-20 with and without scaling and subgradient method on the penalized dual Matrix Completion problem (30)

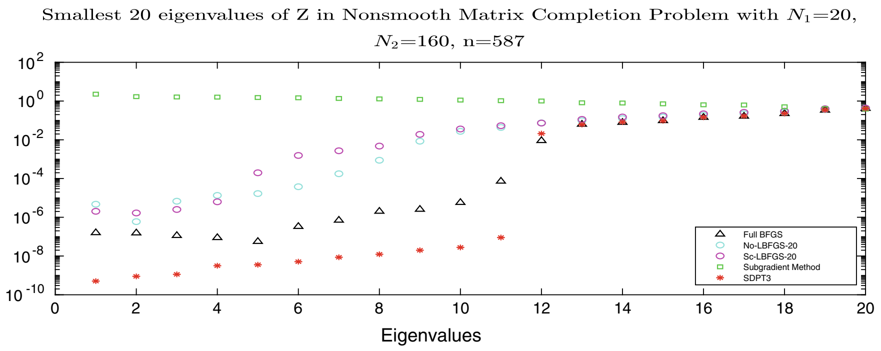


Fig. 17 Comparing smallest 20 eigenvalues of the dual slack matrix Z obtained by BFGS, L-BFGS-20 with and without scaling and the subgradient method on the penalized dual Matrix Completion problem (30). See legend of Fig. 10 regarding eigenvalue monotonicity

3.9 Smoothed Matrix Completion Problem

Nesterov smoothing of the penalized dual Matrix Completion problem (30) gives

$$\begin{aligned} f_\mu(y) &= b^T y \\ &+ \alpha\mu \log\left(1 + \sum_{i=1}^{N_1+N_2} \exp\left(\lambda_i\left(-\begin{bmatrix} I_{N_1} & \mathcal{B}^T(y) \\ \left(\mathcal{B}^T(y)\right)^T & I_{N_2}\end{bmatrix}\right)/\mu\right)\right) \\ &- \alpha\mu \log(N_1 + N_2 + 1). \end{aligned} \tag{31}$$

In Fig. 18, we show the result of applying full BFGS, scaled L-BFGS-5 and scaled L-BFGS-20 to (31) with $\mu = 10^{-7}$. The underlying reference matrix, $\mathcal{X}$, is the same matrix as in the nonsmooth experiment in Fig. 14, with $N_1 = 20$, $N_2 = 160$, $f^* = -1.5398$. We set $\alpha = 3.0796$ as before. The maximum number of function evaluations is set to 10^4.

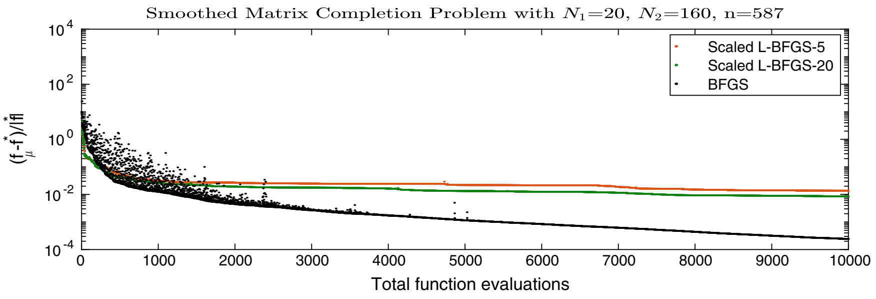


Fig. 18 Comparing scaled L-BFGS-5, scaled L-BFGS-20 and BFGS on Smoothed Matrix Completion problem (31) for the same problem as in the Nonsmooth Matrix Completion problem. The smoothing parameter is $\mu = 10^{-7}$

Table 6 Final objective value we obtained from full BFGS, scaled L-BFGS-20 and scaled L-BFGS-5 on the smoothed and nonsmooth Matrix Completion problem presented in Figs. 18, 16 and 14 respectively. The optimal SDP value f^* is also shown for comparison

SDP-optimal	−1.53978555575175
BFGS-smooth	−1.53941270679104
BFGS-nonsmooth	−1.53946718487809
L-BFGS-20-smooth	−1.52686081626082
L-BFGS-20-nonsmooth	−1.52864965852708
L-BFGS-5-smooth	−1.51868588317043
L-BFGS-5-nonsmooth	−1.50422184883968

We see in Fig. 18 that although the relative error obtained by scaled L-BFGS-20 and full BFGS are about the same as when applied to the nonsmooth function, scaled L-BFGS-5 gets a lower error when it is applied to the smoothed function, and more importantly, it does not break down early.

Table 6 shows the final answers we obtained from full BFGS, scaled L-BFGS-20 and scaled L-BFGS-5 on the smoothed and nonsmooth Matrix Completion problem. The optimal SDP value f^* is also shown for comparison.

4 Concluding Remarks

In Sect. 2, we presented theoretical results showing that L-BFGS may converge to non-optimal points when applied to a simple class of nonsmooth functions. In Sect. 3, we investigated whether the same phenomenon holds in practice. We found that when applied to a nonsmooth function directly, L-BFGS, especially its scaled variant, often breaks down with a poor approximation to an optimal solution, in sharp contrast to full BFGS. Unscaled L-BFGS is less prone to breakdown but conducts far more function evaluations per iteration than scaled L-BFGS does, and thus it is slow. Nonetheless, it is often the case that both variants obtain better results than the provably convergent, but slow, subgradient method.

On the other hand, when applied to a smooth approximation of a nonsmooth function, scaled L-BFGS invariably performs better than unscaled L-BFGS, often obtaining good results even when the problem is quite ill-conditioned. In particular, scaled L-BFGS may be a reasonable approach to finding approximate minimizers of smoothed exact penalty dual functions arising in large-scale semidefinite programs, although further investigation is needed to investigate the practicality of this approach. Minimization of the SDP exact penalty dual function is a key component of a recently proposed method for solving large-scale SDPs with low-rank primal solutions [14].

Most importantly, we find that although L-BFGS is often a reliable method for minimizing ill-conditioned smooth problems, it frequently fails when the condition number is so large that the function is effectively nonsmooth. This behavior is in sharp contrast to the behavior of full BFGS, which is consistently reliable for nonsmooth optimization problems. We arrive at the conclusion that, for large-scale nonsmooth optimization problems for which full BFGS and other methods are not practical, it is often better to apply L-BFGS to a smoothed variant of a nonsmooth problem than to apply it directly to the nonsmooth problem.

Acknowledgements The work of the first author was conducted as part of her Ph.D. studies at the Courant Institute of Mathematical Studies, New York University. Many thanks to Margaret H. Wright for providing her financial support through a grant from the Simons Foundation (417314,MHW).

References

1. Azam Asl and Michael L. Overton. Analysis of limited-memory BFGS on a class of nonsmooth convex functions. *IMA Journal of Numerical Analysis*, 01 2020. drz052.
2. Azam Asl and Michael L. Overton. Analysis of the gradient method with an Armijo-Wolfe line search on a class of non-smooth convex functions. *Optimization Methods and Software*, 35(2):223–242, 2020.
3. Azam Asl. *Behavior of the Limited Memory BFGS Method on Nonsmooth Optimization Problems in Theory and Practice*. PhD thesis, New York University, 2020. https://cs.nyu.edu/media/publications/asl_thesis_final_UtpoLsu.pdf.
4. Jonathan Barzilai and Jonathan M. Borwein. Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, 8(1):141–148, 1988.
5. James V. Burke, Frank E. Curtis, Adrian S. Lewis, Michael L. Overton, and Lucas E. A. Simões. Gradient sampling methods for nonsmooth optimization. In Adil M. Bagirov, Manlio Gaudioso, Napsu Karmitsa, Marko M. Mäkelä, and Sona Taheri, editors, *Numerical Nonsmooth Optimization: State of the Art Algorithms*, pages 201–225. Springer International Publishing, Cham, 2020.
6. Adil M. Bagirov, Manlio Gaudioso, Napsu Karmitsa, Marko M. Mäkelä, and Sona Taheri. *Numerical Nonsmooth Optimization: State of the Art Algorithms*. Springer International Publishing, Cham, 2020.

7. Adil Bagirov, Napsu Karmitsa, and Marko M. Mäkelä. *Introduction to Nonsmooth Optimization*. Springer, Cham, 2014. Theory, Practice and Software.
8. James V. Burke, Adrian S. Lewis, and Michael L. Overton. A robust gradient sampling algorithm for nonsmooth, nonconvex optimization. *SIAM J. Optim.*, 15(3):751–779, 2005.
9. Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, March 2004.
10. Frank E. Curtis and Minhan Li. Gradient sampling methods with inexact subproblem solutions and gradient aggregation, 2020. arXiv:2005.07822.
11. Frank E. Curtis, Tim Mitchell, and Michael L. Overton. A BFGS-SQP method for nonsmooth, nonconvex, constrained optimization and its evaluation using relative minimization profiles. *Optimization Methods and Software*, 32(1):148–181, 2017.
12. Frank E. Curtis and Michael L. Overton. A sequential quadratic programming algorithm for nonconvex, nonsmooth constrained optimization. *SIAM J. Optim.*, 22(2):474–500, 2012.
13. Alexandre d'Aspremont. Smooth optimization with approximate gradient. *SIAM J. Optim.*, 19(3):1171–1183, 2008.
14. Lijun Ding, Alp Yurtsever, Volkan Cevher, Joel A. Tropp, and Madeleine Udell. An optimal-storage approach to semidefinite programming using approximate complementarity, February 2019. arXiv:1902.03373.
15. Michael Grant and Stephen Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. https://web.stanford.edu/~boyd/papers/pdf/graph_dcp.pdf.
16. Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. http://cvxr.com/cvx, March 2014.
17. J. Guo and A. Lewis. Nonsmooth variants of Powell's BFGS convergence theorem. *SIAM Journal on Optimization*, 28(2):1301–1311, 2018.
18. The University of Florida sparse matrix collection: Gset group. http://www.cise.ufl.edu/research/sparse/matrices/Gset/index.html, March 2014.
19. Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 42(6):1115–1145, November 1995.
20. C. Helmberg, M.L. Overton, and F. Rendl. The spectral bundle method with second-order information. *Optimization Methods and Software*, 29(4):855–876, 2014.
21. C. Helmberg and F. Rendl. A spectral bundle method for semidefinite programming. *SIAM Journal on Optimization*, 10(3):673–696, 2000.
22. Krzysztof C. Kiwiel. *Methods of descent for nondifferentiable optimization*, volume 1133 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin, 1985.
23. C. Lemaréchal. An extension of Davidon methods to non differentiable problems. *Math. Programming Stud.*, (3):95–109, 1975.
24. Dong C. Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Math. Programming*, 45(3, (Ser. B)):503–528, 1989.
25. Adrian S. Lewis and Michael L. Overton. Nonsmooth optimization via quasi-Newton methods. *Math. Program.*, 141(1-2, Ser. A):135–163, 2013.
26. A. S. Lewis and S. Zhang. Nonsmoothness and a variable metric method. *J. Optim. Theory Appl.*, 165(1):151–171, 2015.
27. Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, 2012.
28. Angelia Nedić and Dimitri P. Bertsekas. Incremental subgradient methods for nondifferentiable optimization. *SIAM J. Optim.*, 12(1):109–138, 2001.
29. Yu. Nesterov. Smooth minimization of non-smooth functions. *Math. Program.*, 103(1, Ser. A):127–152, 2005.
30. Yu. Nesterov. Private communication, 2016. Les Houches, France.
31. J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, New York, 2nd edition, 2006.

32. M.L. Overton. On minimizing the maximum eigenvalue of a symmetric matrix. *SIAM J. Matrix Anal. Appl.*, 9:256–268, 1988.
33. M. J. D. Powell. Some global convergence properties of a variable metric algorithm for minimization without exact line searches. In *Nonlinear Programming*, pages 53–72, Providence, 1976. Amer. Math. Soc. SIAM-AMS Proc., Vol. IX.
34. Benjamin Recht, Maryam Fazel, and Pablo A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501, 2010.
35. Lieven Vandenberghe. Optimization methods for large-scale systems, 2019. http://www.seas.ucla.edu/~vandenbe/236C/lectures/smoothing.pdf, Lecture Notes for ECE236C.
36. Yuchen Xie and Andreas Waechter. On the convergence of BFGS on a class of piecewise linear non-smooth functions, December 2017. arXiv:1712.08571.

Subgradient Smoothing Method for Nonsmooth Nonconvex Optimization

A. M. Bagirov, N. Sultanova, S. Taheri, and G. Ozturk

Abstract In this chapter an unconstrained nonsmooth nonconvex optimization problem is considered and a method for solving this problem is developed. In this method the subproblem for finding search directions is reduced to the unconstrained minimization of a smooth function. This is achieved by using subgradients computed in some neighborhood of a current iteration point and by formulating the search direction finding problem to the minimization of the convex piecewise linear function over the unit ball. The hyperbolic smoothing technique is applied to approximate the minimization problem by a sequence of smooth problems. The convergence of the proposed method is studied and its performance is evaluated using a set of nonsmooth optimization academic test problems. In addition, the method is implemented in GAMS and numerical results using different solvers from GAMS are reported. The proposed method is compared with a number of nonsmooth optimization methods.

Keywords Nonsmooth optimization · Nonconvex optimization · Subdifferential mapping · Smoothing techniques

1 Introduction

Consider the following unconstrained optimization problem.

$$\begin{cases} \text{minimize} & f(x) \\ \text{subject to} & x \in \mathbb{R}^n, \end{cases} \tag{1}$$

A. M. Bagirov (✉) · N. Sultanova
School of Engineering, Information Technology and Physical Sciences,
Federation University Australia, Ballarat, Australia
e-mail: a.bagirov@federation.edu.au

S. Taheri
School of Science, RMIT University, Melbourne, Australia

G. Ozturk
Department of Industrial Engineering, Eskisehir Technical University,
Eskisehir, Turkey

where the objective function f is, in general, locally Lipschitz continuous (LLC). There are no further assumptions on the structure of this function.

Problem (1) arises in variety of applications such as economics, mechanics, engineering, control theory, optimal shape design and machine learning [1–10], to name a few. Various algorithms have been developed to solve Problem (1). The proximal bundle [11, 12], bundle-Newton [13], variable metric [14], gradient set splitting [15], subgradient [16], gradient sampling [17] methods and hybrid methods, such as the discrete gradient [18], the quasisecant [19] and limited memory bundle [20] methods are among them.

One interesting approach for solving Problem (1) is the use of smooth optimization methods. Such an approach has been studied in [21–23]. Although this approach is not supported by strong convergence results, it enables us to apply powerful smooth optimization methods for solving nonsmooth optimization problems.

In this chapter we propose a rather different approach to apply smooth optimization methods for solving nonsmooth optimization problems. Using subgradients computed in some neighborhood of the current iteration point we formulate the search direction finding subproblem as a minimization of a convex piecewise linear function over the unit ball. Then by applying the hyperbolic smoothing technique this subproblem is replaced by the sequence of unconstrained smooth optimization problems. A new method is designed based on this approach to solve Problem (1).

The proposed method is tested using nonsmooth optimization academic test problems and compared with the number of nonsmooth optimization solvers. We also demonstrate the implementation of the proposed method using several smooth optimization solvers from the GAMS software.

The rest of the chapter is organized as follows. Section 2 provides some necessary preliminaries. Section 3 presents definitions and results used for designing the new method. In Sect. 4 we propose an algorithm for computing the search directions and describe the minimization algorithm to solve Problem (1). We present the results of numerical experiments in Sect. 5. Section 6 concludes the chapter.

2 Preliminaries

In what follows we denote by $\mathbb{R}^n$ the n-dimensional Euclidean space, by $\langle u, v\rangle = \sum_{i=1}^n u_i v_i$ the inner product of vectors $u, v \in \mathbb{R}^n$ and by $\|\cdot\|$ the associated Euclidean norm. Furthermore, $S_1 = \{u \in \mathbb{R}^n : \|u\| = 1\}$ is the unit sphere, $B_\varepsilon(x) = \{u \in \mathbb{R}^n : \|u - x\| < \varepsilon\}$ is the open ball centered at the point x with the radius $\varepsilon > 0$ and $B_\varepsilon = B_\varepsilon(0_n)$.

Let the function $f : \mathbb{R}^n \to \mathbb{R}$ be LLC at $x \in \mathbb{R}^n$. The *generalized directional derivative* $f^\circ(x, d)$ of f at x in the direction $d \in \mathbb{R}^n$ is defined as [24, 25]

$$f^\circ(x, d) = \limsup_{y\to x, \alpha\downarrow 0} \frac{f(y + \alpha d) - f(y)}{\alpha}.$$

Note that the generalized directional derivative always exists for LLC functions. The *subdifferential* of the function f at x is [25]

$$\partial f(x) = \Big\{ \xi \in \mathbb{R}^n : \ f^\circ(x,d) \geq \langle \xi, d \rangle \ \forall d \in \mathbb{R}^n \Big\}.$$

According to Rademacher's theorem the LLC function f is differentiable almost everywhere and its subdifferential $\partial f(x)$ at a point $x \in \mathbb{R}^n$ can also be defined as

$$\partial f(x) = \text{conv} \left\{ \lim_{i \to \infty} \nabla f(x_i) : \ x_i \to x \text{ and } \nabla f(x_i) \text{ exists} \right\},$$

where "conv" denotes the convex hull of a set. Each vector $v \in \partial f(x)$ is called a *subgradient*. Note that the subdifferential $\partial f(x)$ is a compact and convex set at any $x \in \mathbb{R}^n$. The point $x^* \in \mathbb{R}^n$ is called *stationary* if $0_n \in \partial f(x^*)$. Stationarity is a necessary condition for local optimality and, in the convex case, it is also sufficient for global optimality.

Hyperbolic smoothing technique.

For a given $\tau > 0$ the hyperbolic smoothing function ϕ_τ to approximate the function

$$\theta(x) = \max\{0, x\}, \ x \in \mathbb{R}$$

is defined as [26–29]

$$\phi_\tau(x) = \frac{x + \sqrt{x^2 + \tau^2}}{2}. \tag{2}$$

Here $\tau > 0$ is called a precision or smoothing parameter. The hyperbolic smoothing function for the general finite maximum function was studied in [26]. Consider the function

$$h(x) = \max_{i \in I} \ h_i(x), \ I = \{1, \ldots, k\},$$

where the functions $h_i, \ i \in I$ are continuously differentiable. Using an additional auxiliary variable $t \in \mathbb{R}$ we define the following function:

$$F(x,t) = t + \sum_{i \in I} \ \max\{0, h_i(x) - t\}.$$

It is obvious that $h(x) = F(x, h(x))$. Applying (2) we can write the hyperbolic smoothing of the function F for a given $\tau \geq 0$ as

$$\Psi_\tau(x,t) = t + \sum_{i \in I} \frac{h_i(x) - t + \sqrt{(h_i(x) - t)^2 + \tau^2}}{2}. \tag{3}$$

The function $\Psi_\tau(x,t)$ is differentiable and for any $x \in \mathbb{R}^n,\ t \in \mathbb{R}$ and $\tau > 0$ we have [26]

$$0 < \Psi_\tau(x,t) - F(x,t) \le \frac{k\tau}{2}.$$

3 Theoretical Background

In this section we introduce some definitions and results which will be used to design a new method for solving, in general, nonsmooth nonconvex optimization problems and to study its convergence.

Let $f : \mathbb{R}^n \to \mathbb{R}$ be LLC and $\varepsilon > 0$ be a given number. The Goldstein ε-subdifferential of the function f at a point $x \in \mathbb{R}^n$ is defined as [24]:

$$\partial^G_\varepsilon f(x) = \mathrm{cl}\,\mathrm{conv}\left\{\xi \in \mathbb{R}^n : \xi \in \partial f(y),\ y \in B_\varepsilon(x)\right\}.$$

The set $\partial^G_\varepsilon f(x)$ is compact and convex for any $x \in \mathbb{R}^n$.

Definition 1 The Goldstein ε-directional derivative $f^\circ_\varepsilon(x,d)$ of the function f at $x \in \mathbb{R}^n$ in a direction $d \in \mathbb{R}^n$ is:

$$f^\circ_\varepsilon(x,d) = \max_{\xi \in \partial^G_\varepsilon f(x)} \langle \xi, d\rangle.$$

Proposition 1 *Let $f : \mathbb{R}^n \to \mathbb{R}$ be LLC and $\varepsilon > 0$ be a given number. Then for any $d \in S_1$*

$$f(x+\varepsilon d) - f(x) \le \varepsilon f^\circ_\varepsilon(x,d).$$

Proof By Lebourg's mean value theorem [25] for any $x, y \in \mathbb{R}^n$ there exists $\alpha \in (0,1)$ such that

$$f(y) - f(x) = \langle \xi, y - x\rangle$$

for some $\xi \in \partial f(\alpha x + (1-\alpha)y)$. Then we have that for $\varepsilon > 0$ and $d \in S_1$ there exist $\varepsilon' \in (0,\varepsilon)$ and $\xi' \in \partial f(x+\varepsilon' d)$ such that

$$f(x+\varepsilon d) - f(x) = \varepsilon\langle \xi', d\rangle.$$

It follows that $\xi \in \partial^G_\varepsilon f(x)$ and therefore,

$$f(x+\varepsilon d) - f(x) \le \varepsilon \max_{\xi \in \partial^G_\varepsilon f(x)} \langle \xi, d\rangle = \varepsilon f^\circ_\varepsilon(x,d).$$

This completes the proof. □

Following Proposition 1 we will assume that a function $f : \mathbb{R}^n \to \mathbb{R}$ satisfies the following assumption.

Assumption 1 *Let $f : \mathbb{R}^n \to \mathbb{R}$ be an LLC function. There exists $\varepsilon_0 > 0$ such that*

$$f(x + \varepsilon d) - f(x) \le \varepsilon \langle \xi, d\rangle \tag{4}$$

for all $\varepsilon \in (0, \varepsilon_0],\ x \in \mathbb{R}^n,\ \xi \in \partial f(x + \varepsilon d),\ d \in S_1$.

Convex functions and functions represented as a difference of two polyhedral functions as well as a difference of general convex functions and polyhedral functions satisfy this assumption.

Definition 2 A point $x^* \in \mathbb{R}^n$ is called the ε-stationary point of the function f iff

$$0_n \in \partial^G_\varepsilon f(x^*).$$

Proposition 2 *Assume that $x \in \mathbb{R}^n$ is not an ε-stationary point of the function f. Let*

$$\xi^0 = \operatorname*{argmin}_{\xi \in \partial^G_\varepsilon f(x)} \|\xi\|$$

and $d^0 = -\|\xi^0\|^{-1}\xi^0$. Then for any $d \in S_1$

$$f(x + \varepsilon d^0) - f(x) \le -\varepsilon\|\xi^0\|.$$

Proof Since x is not an ε-stationary point $\xi^0 \ne 0_n$. The set $\partial^G_\varepsilon f(x)$ is convex and compact, therefore the optimality condition implies that

$$\langle \xi^0, \xi - \xi^0\rangle \ge 0 \quad \forall \xi \in \partial^G_\varepsilon f(x).$$

This means that $\langle \xi, d^0\rangle \le -\|\xi^0\|$ for all $\xi \in \partial^G_\varepsilon f(x)$. Therefore,

$$f^\circ_\varepsilon(x, d) \le -\|\xi^0\|.$$

Then the proof follows from Proposition 1. □

It follows from Proposition 2 that if the point x is not the ε-stationary then the set $\partial^G_\varepsilon f(x)$ can be used to find descent directions of the function f at x. However, it is not always easy to calculate the set $\partial^G_\varepsilon f(x)$. Below we introduce an algorithm which uses only several elements from this set to compute descent directions.

Finally, we give the definition of the so-called (ε, δ)-stationary points where $\varepsilon, \delta > 0$. Similar points are defined in [19, 30].

Definition 3 A point x is called the (ε, δ)-stationary point of Problem (1) if

$$0_n \in \partial^G_\varepsilon f(x) + B_\delta(0_n). \tag{5}$$

4 Minimization Algorithm

First we design an algorithm to find descent directions for the objective function f in Problem (1) and then, we introduce the minimization algorithm to solve this problem.

4.1 Computation of Descent Directions

The algorithm for computing descent directions of the function f is as follows.

Algorithm 1 Computation of descent directions.

Data. Let the numbers $\varepsilon > 0,\ c \in (0,1)$ and the tolerance $\delta > 0$ be given. Select any $d^1 \in S_1$.

Step 1. Compute a subgradient $\xi^1 \in \partial f(x + \varepsilon d^1)$. Set $V_1(x) = \{\xi^1\}$ and $k = 1$.

Step 2. Compute $\bar{d}$ as a solution to the minimization problem

$$\begin{cases} \text{minimize} & \max\limits_{i \in I_k} \langle \xi^i, d \rangle \\ \text{subject to} & d \in S_1, \end{cases} \tag{6}$$

where $I_k = \{1, \ldots, k\}$. If $D_k \equiv \max\limits_{i \in I_k} \langle \xi^i, \bar{d} \rangle > -\delta$, then **stop**.

Step 3. If

$$f(x + \varepsilon \bar{d}) - f(x) \le c\varepsilon D_k, \tag{7}$$

then **stop**. Otherwise set $d^{k+1} = \bar{d}$.

Step 4. Compute a subgradient $\xi^{k+1} \in \partial f(x + \varepsilon d^{k+1})$ and construct the set

$$V_{k+1}(x) = \text{conv}\left\{ V_k(x) \cup \{\xi^{k+1}\} \right\}.$$

Set $k = k + 1$ and go to Step 2.

Next we show that Algorithm 1 terminates after a finite number of iterations. First, we prove the following propositions and corollaries.

Proposition 3 *If $D_k > -\delta$, then*

$$\min_{\xi \in V_k(x)} \|\xi\| < \delta. \tag{8}$$

Proof Let $\tilde{\xi}$ be a solution to the problem

$$\begin{cases} \text{minimize} & \frac{1}{2}\|\xi\|^2 \\ \text{subject to} & \xi \in V_k(x). \end{cases} \tag{9}$$

If $\tilde{\xi} = 0$, then (8) is true. Now assume that $\tilde{\xi} \neq 0$. Since $\tilde{\xi}$ is the solution to Problem (9) it follows from the necessary condition for a minimum that

$$\langle \tilde{\xi}, \xi - \tilde{\xi} \rangle \geq 0 \ \ \forall \xi \in V_k(x),$$

or

$$\|\tilde{\xi}\|^2 \leq \langle \tilde{\xi}, \xi \rangle \ \ \forall \xi \in V_k(x). \tag{10}$$

Since $D_k > -\delta$ we have

$$\max_{i=1,\dots,k} \langle \xi^i, d \rangle > -\delta \ \ \forall d \in S_1. \tag{11}$$

Consider $\tilde{d} = -\|\tilde{\xi}\|^{-1}\tilde{\xi}$. Then it follows from (11) that there exists $i \in I_k$ such that

$$\langle \tilde{\xi}, \xi^i \rangle < \delta \|\tilde{\xi}\|.$$

This together with (10) completes the proof. □

Corollary 1 *If $D_k > -\delta$, then the point x is an (ε, δ)-stationary point.*

Proof Since $V_k(x) \subset \partial^G_\varepsilon f(x)$ it follows from Proposition 3 that

$$\min_{\xi \in \partial^G_\varepsilon f(x)} \|\xi\| < \delta,$$

which means that x is an (ε, δ)-stationary point. □

Remark 1 It follows from Proposition 3 that if $D_k > -\delta$ in Step 2 of Algorithm 1, then the point $x \in \mathbb{R}^n$ is an approximate stationary point satisfying (5) for given $\varepsilon > 0$ and $\delta > 0$.

Corollary 2 *If $D_k \geq 0$, then $0_n \in V_k(x)$.*

Proof Assume the contrary, that is $0_n \notin V_k(x)$. Then $\tilde{\xi} \neq 0$, where $\tilde{\xi}$ is the solution to Problem (9). Since $D_k \geq 0$ we have

$$\max_{i \in I_k} \langle \xi^i, d \rangle \geq 0 \ \ \forall d \in S_1. \tag{12}$$

Consider $\tilde{d} = -\|\tilde{\xi}\|^{-1}\tilde{\xi}$. Then there exists $i \in I_k$ such that

$$\langle \tilde{\xi}, \xi^i \rangle \leq 0.$$

However, it follows from (10) that $0 \geq \langle \tilde{\xi}, \xi^i \rangle \geq \|\tilde{\xi}\|^2 > 0 \, \forall i \in I_k$. This is a contradiction, and therefore, $0_n \in V_k(x)$. □

Proposition 4 *If* $\min_{\xi \in V_k(x)} \|\xi\| < \delta$, *then* $D_k > -\delta$.

Proof Assume the contrary, that is $\min_{\xi \in V_k(x)} \|\xi\| < \delta$ but $D_k \leq -\delta$. This means that

$$\langle \xi^i, \bar{d} \rangle \leq -\delta, \; i \in I_k,$$

where $\bar{d} \in S_1$ is the solution to Problem (6). Let

$$\|\tilde{\xi}\| = \min_{\xi \in V_k(x)} \|\xi\| < \delta.$$

Since $\tilde{\xi} \in V_k(x)$ we get

$$\tilde{\xi} = \sum_{i \in \tilde{I}} \alpha_i \xi^i, \quad \sum_{i \in \tilde{I}} \alpha_i = 1, \;\; \alpha_i \in (0, 1], \; i \in \tilde{I} \subseteq I_k,$$

and therefore,

$$\langle \tilde{\xi}, \bar{d} \rangle \leq -\delta. \tag{13}$$

On the other hand, we have

$$\left| \langle \tilde{\xi}, \bar{d} \rangle \right| \leq \|\tilde{\xi}\| \|\bar{d}\| = \|\tilde{\xi}\| < \delta,$$

which contradicts (13). The proof is complete. □

Proposition 5 *Let the function* f *be LLC at* $x \in \mathbb{R}^n$. *Then Algorithm 1 terminates after a finite number of iterations.*

Proof Algorithm 1 has two stopping criteria. The first criterion is in Step 2, which means that the point x is (ε, δ)-stationary and the second criterion is in Step 3, which means that $\bar{d}$ is the descent direction satisfying the condition (7). If both conditions for the termination of the algorithm are not satisfied, then $\xi^{k+1} \notin V_k(x)$. Indeed, in this case

$$f(x + \varepsilon \bar{d}) - f(x) > c\varepsilon D_k.$$

It follows from (4) that

$$f(x + \varepsilon \bar{d}) - f(x) \leq \varepsilon \langle \xi^{k+1}, \bar{d} \rangle,$$

and therefore, $\langle \xi^{k+1}, \bar{d} \rangle > cD_k$. It then follows that $-\langle \xi^{k+1}, \bar{d} \rangle < -cD_k \leq -c(-\delta)$ and consequently

$$\langle \xi^{k+1}, \bar{d} \rangle > -c\delta. \tag{14}$$

Assume the contrary, that is $\xi^{k+1} \in V_k(x)$. Since $D_k \le -\delta$, we have

$$\langle \xi^i, \bar{d} \rangle \le -\delta, \ i \in I_k.$$

Assuming that $\xi^{k+1} \in V_k(x) = \text{conv}\,\{\xi^1, \dots, \xi^k\}$ we can write

$$\xi^{k+1} = \sum_{i \in I_k} \alpha_i \xi^i, \ \ \sum_{i \in I_k} \alpha_i = 1, \ \ \alpha_i \ge 0, \ i \in I_k.$$

Then we get

$$\langle \xi^{k+1}, \bar{d} \rangle = \Big\langle \sum_{i \in I_k} \alpha_i \xi^i, \bar{d} \Big\rangle = \sum_{i \in I_k} \Big\langle \alpha_i \xi^i, \bar{d} \Big\rangle = \sum_{i \in I_k} \alpha_i \Big\langle \xi^i, \bar{d} \Big\rangle \le -\delta,$$

which contradicts (14). This means that $\xi^{k+1} \notin V_k(x)$.

Now we show that Algorithm 1 is terminating. Assume the contrary. Then Algorithm 1 generates an infinite sequence $\{d^k\}$ where $d^k \in S_1$. From (14) we have

$$\langle \xi^k, d^k \rangle > -c\delta \ \ \forall\, k = 2, 3, \dots. \tag{15}$$

Let $L > 0$ be the Lipschitz constant of f on the bounded set cl $B_\varepsilon(x)$. It follows from Theorem 3.1.4 [12] that $\|\xi\| \le L$ for all $\xi \in \partial f(y), \ y \in \text{cl}\, B_\varepsilon(x)$. In addition, since the direction d^{k+1} is the solution to Problem (6) and Algorithm 1 is not terminating we get

$$\max_{i \in I_k} \ \langle \xi^i, d^{k+1} \rangle \le -\delta.$$

Therefore, d^{k+1} is the solution to the system

$$\langle \xi^i, d \rangle + \delta \le 0, \ i \in I_k.$$

Then we obtain

$$\|d^{k+1} - d^j\| > \frac{(1-c)\delta}{L} \ \ \forall j = 2, \dots, k. \tag{16}$$

Indeed, if there exists $j \in \{2, \dots, k\}$ such that

$$\|d^{k+1} - d^j\| \le \frac{(1-c)\delta}{L},$$

then we have

$$\big| \langle \xi^j, d^{k+1} \rangle - \langle \xi^j, d^j \rangle \big| \le (1-c)\delta.$$

This implies that

$$\langle \xi^j, d^j \rangle \le \langle \xi^j, d^{k+1} \rangle + (1-c)\delta \le -c\delta,$$

which contradicts (15). Therefore, the inequality (16) holds and can be rewritten as

$$\min_{j=2,\dots,k} \|d^{k+1} - d^j\| > \frac{(1-c)\delta}{L}.$$

This means that Algorithm 1 generates a sequence $\{d^k\}$ of directions $d^k \in S_1$ such that the distance between d^k and the set of all previous directions is bounded below. Since the set S_1 is compact the number of such directions is finite. This completes the proof. □

4.2 Solving Subproblem in Finding Search Directions

The most important step in Algorithm 1 is Step 2, where Problem (6) is solved to find search directions. In order to solve this problem, we first reduce it to the minimization of the convex piecewise linear function over the unit ball. We then replace the constrained problem by the unconstrained one using a distance function. We use the hyperbolic smoothing technique to replace the problem by the sequence of smooth problems and apply smooth optimization solvers to solve it.

Note that the search direction finding problem in most bundle-type methods usually solved by dualization which gives rise to a structured quadratic programming problem. There exists in literature several efficient algorithms for solving this family of problems (see, for example, [31–33]).

Denote the objective function in Problem (6) by

$$\varphi_k(d) = \max_{i \in I_k} \langle \xi^i, d \rangle.$$

Then Problem (6) can be rewritten as

$$\begin{cases} \text{minimize} & \varphi_k(d) \\ \text{subject to} & d \in S_1. \end{cases} \tag{17}$$

In addition to Problem (17), we consider the convex programming problem

$$\begin{cases} \text{minimize} & \varphi_k(d) \\ \text{subject to} & d \in B_1, \end{cases} \tag{18}$$

where $B_1 = \{y \in \mathbb{R}^n : \|y\|^2 \le 1\}$. Denote by G_{1k} and G_{2k} sets of solutions and by D_{1k} and D_{2k} the optimal values of Problems (17) and (18), respectively.

Proposition 6 *The following hold for Problems* (17) *and* (18)*:*

(1) if $D_{2k} = 0$, then $D_{1k} \ge 0$. If in this case $D_{1k} = 0$, then $G_{1k} \subseteq G_{2k}$, otherwise $G_{1k} \not\subseteq G_{2k}$.

(2) if $D_{2k} < 0$*, then* $G_{1k} = G_{2k}$ *and* $D_{1k} = D_{2k}$*.*

Proof Note that the function φ_k is positively homogeneous, that is

$$\varphi_k(\lambda d) = \lambda \varphi_k(d) \ \ \forall d \in \mathbb{R}^n \text{ and } \lambda \geq 0. \tag{19}$$

Since $0_n \in B_1$ we always have $D_{2k} \leq 0$. It follows from the inclusion $S_1 \subset B_1$ that $D_{1k} \geq D_{2k}$. Therefore, we consider only two cases: (1) $D_{2k} = 0$ and (2) $D_{2k} < 0$.
Case (1) If $D_{2k} = 0$, then $D_{1k} \geq 0$, which triggers the stopping criterion in Algorithm 1. According to Corollary 2 in this case $0_n \in V_k(x)$. Take any $\bar{d} \in G_{1k}$.

- If $D_{1k} = 0$, then $\varphi_k(\bar{d}) = 0$ and $\varphi_k(d) \geq \varphi_k(\bar{d})$ for all $d \in S_1$. Since for any $d \in B_1$ there exist $d_0 \in S_1$ and $\lambda \in [0, 1]$ such that $d = \lambda d_0$ it follows from (19) that $\varphi_k(d) \geq 0$ for all $d \in B_1$. This means that $\bar{d} \in G_{2k}$ and therefore, $G_{1k} \subseteq G_{2k}$.
- If $D_{1k} > 0$, then (19) implies that $\varphi_k(d) > 0$ for all $d \in B_1, \ d \neq 0_n$. This means that in this case $G_{2k} = \{0_n\}$, and therefore, $G_{1k} \not\subset G_{2k}$.

Case (2) Consider the case when $D_{2k} < 0$. We prove that in this case $G_{1k} = G_{2k}$, which means that $D_{1k} = D_{2k}$. Let $\tilde{d}$ be the solution to Problem (18). It is clear that $\tilde{d} \neq 0_n$. Then $\tilde{d} \in S_1$. Indeed, assume that $\tilde{d} \notin S_1$. Then we can construct $\bar{d} = \lambda \tilde{d}$ with $\lambda > 1$ such that $\bar{d} \in S_1 \subset B_1$. Applying (19) we have

$$\varphi_k(\bar{d}) = \lambda \varphi_k(\tilde{d}) = \lambda D_{2k} < D_{2k},$$

which contradicts that $\tilde{d}$ is the solution to Problem (18). Then we get that all solutions of Problem (18) lie on the unit sphere S_1. This means that $G_{1k} = G_{2k}$, and therefore $D_{1k} = D_{2k}$. The proof is completed. □

Results from this proposition show that we can replace the nonconvex minimization problem (17) by the convex programming problem (18). Note that by solving Problem (18) we get either the same stopping criterion or the same descent direction that can be obtained by solving (17).

We apply the hyperbolic smoothing technique to replace Problem (18) by the sequence of smooth problems. Consider the function

$$F_k(d, t) = t + \sum_{i \in I_k} \max \left\{0, \langle \xi^i, d \rangle - t \right\}.$$

It is clear that $F_k(d, \varphi_k(d)) = \varphi_k(d)$. It follows from results obtained in [26] that the set of minimizers of functions F_k and φ_k coincide when $t = \varphi_k(d)$. Define a function

$$\Phi_k(d) \equiv F_k(d, \varphi_k(d)), \ d \in \mathbb{R}^n.$$

Then Problem (18) can be reformulated as

$$\begin{cases} \text{minimize} & \Phi_k(d) \\ \text{subject to} & d \in B_1. \end{cases} \tag{20}$$

For a given $(d, \varphi_k(d))$, the index set I_k can be represented as $I_k = I_{1k} \cup I_{2k}$, where

$$I_{1k} = \left\{ i \in I_k : \langle \xi^i, d \rangle < \varphi_k(d) \right\} \text{ and } I_{2k} = \left\{ i \in I_k : \langle \xi^i, d \rangle = \varphi_k(d) \right\}.$$

The subdifferential of the function Φ_k at $d \in \mathbb{R}^n$ can be written as

$$\partial \Phi_k(d) = \sum_{i \in I_{2k}} \text{conv} \left\{ 0_n, \xi^i \right\}.$$

For a given $\tau > 0$ applying (3) to the function Φ_k we get

$$\Psi_{k\tau}(d) = \varphi_k(d) + \frac{1}{2} \sum_{i \in I_k} \left[\langle \xi^i, d \rangle - \varphi_k(d) + \left((\langle \xi^i, d \rangle - \varphi_k(d))^2 + \tau^2 \right)^{1/2} \right].$$

Applying Proposition 5 from [26] we have $0 < \Psi_{k\tau}(d) - \Phi_k(d) \le k\tau/2$. The gradient of the function Ψ_τ is

$$\nabla \Psi_{k\tau}(d) = \frac{1}{2} \sum_{i \in I_k} \left(1 + \beta_{i\tau}(d) \right) \xi^i,$$

where

$$\beta_{i\tau}(d) = \frac{\langle \xi^i, d \rangle - \varphi_k(d)}{\left((\langle \xi^i, d \rangle - \varphi_k(d))^2 + \tau^2 \right)^{1/2}}, \quad i \in I_k.$$

From Proposition 6 in [26] it follows that if $z = \lim_{\tau \downarrow 0} \nabla \Psi_{k\tau}(d)$, then $z \in \partial \Phi_k(d)$. In addition, we have the following proposition (see Proposition 7 in [26]).

Proposition 7 *Assume that sequences $\{d_l\}$ and $\{\tau_l\}$ are given such that $d_l \in \mathbb{R}^n$ and $\tau_l > 0,\ l = 1, 2, \ldots$. Moreover, $d_l \to d,\ \tau_l \to 0$ as $l \to \infty$ and*

$$z = \lim_{l \to \infty} \nabla \Psi_{\tau_l}(d_l).$$

Then $z \in \partial \Phi_k(d)$.

We replace a nonsmooth optimization problem (20) by the sequence of the following smooth optimization problems:

$$\begin{cases} \text{minimize} & \Psi_{k\tau}(d) \\ \text{subject to} & d \in B_1. \end{cases} \tag{21}$$

Next using the penalty function method we replace the constrained problem (21) by the unconstrained problem:

$$\begin{cases} \text{minimize} & \Psi_{k\tau}(d) + \widehat{L} g_{B_1}(d) \\ \text{subject to} & d \in \mathbb{R}^n. \end{cases} \tag{22}$$

Here $\widehat{L} \geq L$, and $L > 0$ is the Lipschitz constant of the function φ_k which can be computed explicitly as

$$L = \max_{i \in I_k} \|\xi^i\|$$

and g_{B_1} is the distance function of the set B_1:

$$g_{B_1}(d) = \max \Big\{0, \|d\|^2 - 1\Big\}.$$

It is well known that Problems (21) and (22) have the same set of solutions (see, for example, Lemma 5.1.5 from [12]).

Applying hyperbolic smoothing (3) to the function g_{B_1} we get its following smooth approximation:

$$H_\tau(d) = \frac{1}{2}\Big[\|d\|^2 - 1 + \Big((\|d\|^2 - 1)^2 + \tau^2\Big)^{1/2}\Big].$$

Then Problem (22) can be reformulated as a smooth unconstrained optimization problem

$$\begin{cases} \text{minimize} & \Psi_{k\tau}(d) + \widehat{L} H_\tau(d) \\ \text{subject to} & d \in \mathbb{R}^n. \end{cases} \tag{23}$$

To solve Problem (18) we take a sequence $\{\tau_l\}$ of a precision parameter τ where $\tau_l \downarrow 0$ as $l \to \infty$ and solve Problem (23). It is shown in [26] that the sequence of solutions to this problem converges to the set of solutions of Problem (18).

4.3 Minimization Algorithms

First we introduce an algorithm to find (ε, δ)-stationary points of Problem (1) for given $\varepsilon > 0$ and $\delta > 0$. Then we develop an algorithm to compute Clarke stationary points of Problem (1). An algorithm for finding (ε, δ)-stationary points utilizes Algorithm 1 to find descent directions and proceeds as follows.

Algorithm 2 Finding (ε, δ)-stationary points.

Data. Numbers $\varepsilon > 0,\ \delta > 0,\ c_1 \in (0, 1)$ and $c_2 \in (0, c_1]$.

Step 1. Select any starting point $x_1 \in \mathbb{R}^n$ and set $l = 1$.
Step 2. Apply Algorithm 1 to compute a search direction $d_l \in S_1$ at the point $x = x_l$ for given $\varepsilon, \delta > 0$ and $c = c_1 \in (0, 1)$ such that for some $k > 1$

$$D_l = \min_{d \in S_1} \max_{i \in I_k} \langle \xi^i, d \rangle.$$

This algorithm terminates after a finite number k of iterations with either $D_l > -\delta$ or for the search direction $d_l \in S_1$ we get

$$f(x_l + \varepsilon d_l) - f(x_l) \le c_1 \varepsilon D_l. \tag{24}$$

Step 3. If $D_l > -\delta$, then **stop**. The point x_l is (ε, δ)-stationary.
Step 4. If (24) holds, then find the step length $\alpha_l > 0$ as follows:

$$\alpha_l = \operatorname{argmax} \left\{ \alpha \ge 0 : \; f(x_l + \alpha d_l) - f(x_l) \le c_2 \alpha D_l \right\}. \tag{25}$$

Step 5. Set $x_{l+1} = x_l + \alpha_l d_l, \; l = l + 1$ and go to Step 2.

In the next proposition we prove that Algorithm 2 is finite convergent to the set of (ε, δ)-stationary points of Problem (1).

Proposition 8 *Assume that the function f is bounded below, that is*

$$f^* = \inf \left\{ f(x) : x \in \mathbf{R}^n \right\} > -\infty. \tag{26}$$

Then Algorithm 2 finds (ε, δ)-stationary points in at most M iterations, where

$$M \equiv \left\lceil \frac{f(x_1) - f^*}{c_2 \varepsilon \delta} \right\rceil. \tag{27}$$

Proof Assume the contrary, that is the sequence $\{x_l\}$ generated by Algorithm 2 is infinite and the points x_l are not (ε, δ)-stationary points for any $l = 1, 2, \ldots$. This means that

$$D_l \le -\delta \; \forall l = 1, 2, \ldots$$

Then the descent direction d_l will be found at x_l so that the sufficient decrease condition (24) is satisfied

$$f(x_l + \varepsilon d_l) - f(x_l) \le c_1 \varepsilon D_l \le c_2 \varepsilon D_l.$$

Furthermore, it follows from (25) that $\alpha_l \ge \varepsilon$. Therefore, we have

$$\begin{aligned} f(x_{l+1}) - f(x_l) &= f(x_l + \alpha_l d_l) - f(x_l) \\ &< c_2 \alpha_l D_l \\ &\le c_2 \varepsilon D_l. \end{aligned}$$

This together with the condition $D_l \le -\delta$ implies that

$$f(x_{l+1}) \le f(x_1) - l c_2 \varepsilon \delta.$$

Therefore, $f(x_l) \to -\infty$ as $l \to \infty$ which contradicts (26). Therefore, the algorithm is finite convergent. Since $f^* \le f(x_{l+1})$ it is clear that the maximum number of iterations M is given by (27).

Remark 2 One can design a very simple procedure for the estimation of the step-length $\alpha_l,\ l \ge 1$ in Step 4 of Algorithm 2. Since $c_2 \le c_1$ we always have $\alpha_k \ge \varepsilon$. In order to estimate α_l we define a sequence $\theta_s = s\varepsilon,\ s \ge 1$. Then α_l is defined as the largest θ_s satisfying the inequality (25) in Step 4 of Algorithm 2.

Now we design an algorithm for finding Clarke stationary points of Problem (1).

Algorithm 3 Finding Clarke stationary points.

```
Data.   A tolerance $\sigma > 0$, sequences $\{\varepsilon_j\}, \{\delta_j\}$ such that $\varepsilon_j \downarrow 0,\ \delta_j \downarrow 0$ as $j \to \infty$.
Step 1. Choose any starting point $x_1 \in \mathbb{R}^n$. Set $j = 1$.
Step 2. Apply Algorithm 2 with $\varepsilon = \varepsilon_j$ and $\delta = \delta_j$ starting from the point $x_j$ to
    find an $(\varepsilon_j, \delta_j)$-stationary point $x_{j+1}$.
Step 3. Set $j = j + 1$. If $\varepsilon_j \le \sigma$ and $\delta_j \le \sigma$, then **stop**. Otherwise go to Step 2.
```

Next we prove the convergence of Algorithm 3.

Proposition 9 *Assume that the function f satisfies Assumption 1, the set $\mathcal{L}(x_1) = \{x \in \mathbb{R}^n : f(x) \le f(x_1)\}$ is bounded for the starting point x_1 and $\sigma = 0$. Then any accumulation point of the sequence $\{x_j\}$ generated by Algorithm 3 is the Clarke stationary point of Problem (1).*

Proof Since the function f is LLC and the set $\mathcal{L}(x_1)$ is bounded we have $f^* > -\infty$. Then, according to Proposition 8 the $(\varepsilon_j, \delta_j)$-stationary point will be generated in Step 2 of Algorithm 2 after a finite number of steps $l_j > 0$ for all iterations $j > 0$. Since the point x_{j+1} is the $(\varepsilon_j, \delta_j)$-stationary point for any $j > 0$ it follows from the definition of such points that

$$\min \left\{ \|\xi\| : \xi \in \partial^G_{\varepsilon_j} f(x_{j+1}) \right\} \le \delta_j. \tag{28}$$

Furthermore, Algorithm 3 is a descent algorithm and we have $\{x_j\} \subset \mathcal{L}(x_1)$. The boundedness of the set $\mathcal{L}(x_1)$ implies that the sequence $\{x_j\}$ has at least one accumulation point. Let $\bar{x}$ be an accumulation point of $\{x_j\}$. Then there exists the subsequence $\{x_{j_i}\}$ such that $x_{j_i} \to \bar{x}$ as $i \to \infty$. Replacing j by $j_i - 1$ in (28) we get

$$\min\left\{\|\xi\| : \xi \in \partial^G_{\varepsilon_{j_i-1}} f(x_{j_i})\right\} \le \delta_{j_i-1}. \tag{29}$$

It follows from upper semicontinuity of the subdifferential mapping that at the point $\bar{x}$ for any $\gamma > 0$ there exists $\eta > 0$ such that

$$\partial^G_\varepsilon f(y) \subset \partial f(\bar{x}) + B_\gamma(0_n)$$

for all $y \in B_\eta(\bar{x})$ and $\varepsilon \in (0, \eta)$. Since the sequence $\{x_{j_i}\}$ converges to $\bar{x}$ there exists i_0 such that $x_{j_i} \in B_\eta(\bar{x})$ for all $i > i_0$. On the other hand, since $\varepsilon_j, \delta_j \downarrow 0$ as $j \to \infty$ there exists $j_0 > 0$ such that $\varepsilon_j < \eta$ and $\delta_j < \gamma$ and for all $j > j_0$. This means that there exists i_1 such that $j_i > j_0 + 1$ for all $i > i_1$. Let $i_2 = \max\{i_0, i_1\}$, then we have

$$\partial^G_{\varepsilon_{j_i}} f(x_{j_i}) \subset \partial f(\bar{x}) + B_\gamma(0_n) \tag{30}$$

for all $i > i_2$. It follows from (29) and (30) that for any $i > i_2$ we get

$$\min\left\{\|\xi\| : \xi \in \partial f(\bar{x})\right\} \le 2\gamma.$$

Since γ is arbitrary we have $0_n \in \partial f(\bar{x})$. That is the point $\bar{x}$ is Clarke stationary. $\square$

5 Numerical Experiments

To test the efficiency of the proposed method and to compare it with various nonsmooth optimization methods we carry out numerical experiments using some academic test problems available from [24, 34]. We call the proposed method SSM—Subgradient Smoothing Method. We use the following methods for comparison:

- Proximal bundle method (PBUN) [12];
- Newton-bundle method (PNEW) [13];
- Variable metric bundle method (PVAR) [14];
- Discrete gradient method (DGM) [18]; and
- Quasi-Newton method for nonsmooth optimization (QN-NSO) [35].

Algorithms PBUN, PNEW, PVAR, DGM and SSM are implemented in Fortran 95 and compiled using the gfortran compiler. We use the MATLAB implementation of QN-NSO available from "https://cs.nyu.edu/overton/software/hanso/". The experiments are performed on an Intel(R)Core(TM)2 i5 with CPU 1.86 GHz and 1.97GB of RAM.

Parameters in the SSM are chosen as follows: $c_1 = 0.2,\ c_2 = 0.05,\ \varepsilon_{j+1} = 0.5\varepsilon_j,\ j \ge 1,\ \varepsilon_1 = 1$ and $\delta_j \equiv 10^{-7}$ for all $j \ge 1$. Parameters for the implementation of the PBUN, PNEW and PVAR are described in [36] and the parameters for the DGM are given in [18].

In the experiments, we use the academic test problems: CB2, WF, SPIRAL, EVD52, Polak6, Davidon2, OET5, OET6, Wong1, Wong2, Wong3, Polk2, Polak3, Watson, Rosenbrock, Crescent, CB3, DEM, QL, LQ, Mifflin1, Mifflin2, Wolfe, Shor, El-Attar, Maxquad, Gill, Steiner2, Maxq, Maxl, Goffin, MAXHILB, L1HILB and Sheel Dual. Detailed description of these problems can be found in [24, 34]. For each problem, 20 random starting points are used and they are the same for all algorithms.

We do not use all test problems from [24, 34] since the objective functions in some problems are unbounded from below. Furthermore, since we use many starting points for each test problem those with many local solutions are excluded to make easier comparison of algorithms. Nevertheless, we use test problems with very few local solutions.

All algorithms are local search algorithms and do not intend to always find the global minimum of a nonconvex objective function. Starting from the same point algorithms may converge to different stationary points. We say that an algorithm solves a nonconvex optimization problem if it finds its local minimizer with a given tolerance even if this local minimizer is different from the global one. An algorithm finds a solution to a problem with a tolerance $\sigma > 0$ if

$$\frac{|\bar{f} - f_{local}|}{1 + |f_{local}|} \le \sigma.$$

Here $\bar{f}$ is the best value of the objective function found by the algorithm and f_{local} is the closest to $\bar{f}$ among values of the objective function at its known local minimizers. We choose $\sigma = 10^{-4}$.

First, we use performance profiles to demonstrate the pairwise comparison of the proposed method with the others. Then we present results obtained using smooth optimization solvers from the optimization package GAMS [37].

Results with performance profiles. The performance profiles were introduced in [38]. They are defined as the function $\rho_s(\tau)$ for some parameter $\tau \ge 1$ using CPU time, the number of function and (sub)gradient evaluations. The ratio of the number of function and subgradient evaluations are usually too large, therefore it is scaled using the base two logarithm.

The value $\rho_s(0)$ gives the percentage of test problems for which the corresponding algorithm is the best. The value $\rho_s(\tau)$ for a given $\tau \ge 0$ shows the percentage of problems solved by a solver s using τ times more computational effort (CPU time, the number of function and (sub)gradient evaluations) comparing with the best solver. The value of $\rho_s(\tau)$ at the rightmost abscissa gives the percentage of test problems that the corresponding algorithm can solve. This number characterizes the robustness of the algorithm, that is its ability to solve most problems irrespective of the choice of a starting point. Moreover, the relative efficiency of each algorithm can be directly seen from the performance profiles: the higher the particular curve, the better the corresponding algorithm. This means that the algorithm can solve more problems using less computational effort than other algorithms.

We demonstrate the efficiency and robustness of the algorithms using the number of function and subgradient evaluations. We do not include the CPU time since it is almost 0 for all algorithms in most of the test problems. Performance profiles with

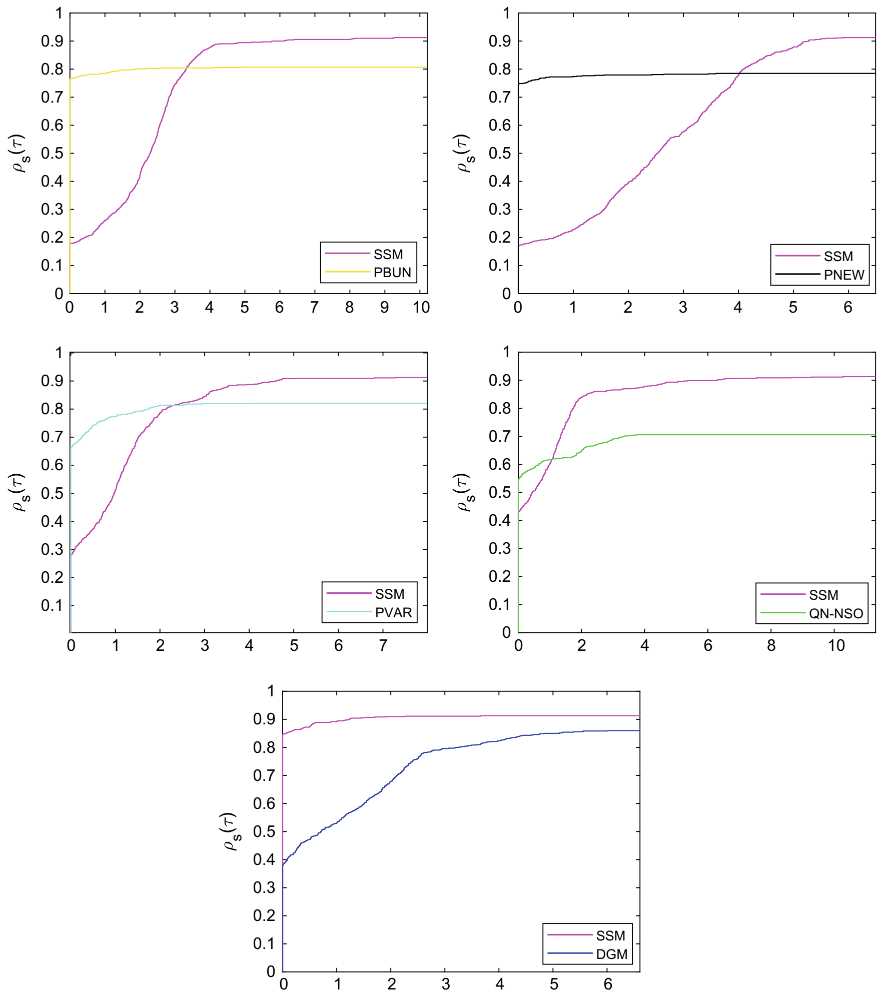


Fig. 1 Performance profiles using the number of function evaluations

the academic test problems mentioned above are presented in Figs. 1 and 2. Since the DGM is a semi-derivative free method we do not include it in Fig. 2.

The results presented in Figs. 1 and 2 demonstrate that the SSM is the most robust method in comparison with other five methods used in our experiments. It outperforms the other methods in terms of the accuracy and solves more than 90% of problems. This method requires more function and subgradient evaluations than PBUN, PNEW, PVAR and QN-NSO, and thus it is not as efficient as the other four algorithms. Nevertheless, it can be observed from Fig. 1 that the SSM is more efficient than DGM.

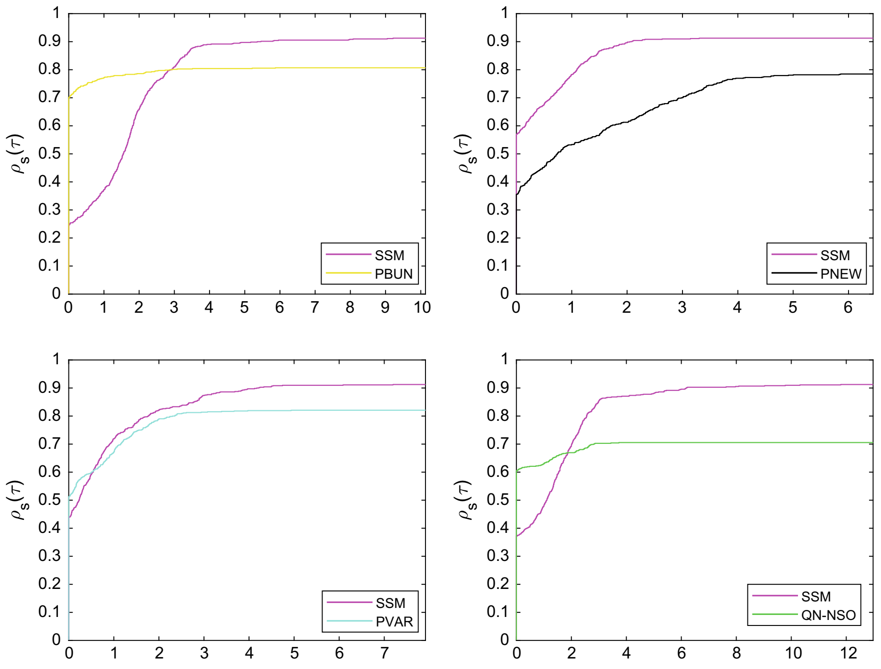


Fig. 2 Performance profiles using the number of subgradient evaluations

Experiments with GAMS. Here we present results obtained using the GAMS implementation of the SSM where CONOPT, MINOS and SNOPT solvers are applied to solve subproblems for finding search directions (for details of these solvers, see [37]). We compare the SSM with the same solvers using the DNLP, which is a GAMS option to handle nonsmoothness. Ten test problems are selected with the different number of variables. Starting points from [24, 34] are used in numerical experiments. Results are given in Table 1, where the following notations are used:

- n: number of variables;
- f_{opt}: optimum value of the objective function;
- f_v: function value obtained by an algorithm;
- n_t: number of iterations;
- n_f: number of function evaluations;
- n_s: number of subgradient evaluations.

Results presented in Table 1 demonstrate that the SSM in combination with all three solvers is much more robust than the DNLP option of the same solvers. The latter solver fails to solve most of test problems with a required accuracy, however the SSM is able to find solutions with the given accuracy for all problems except two of them: Maxq and Goffin problems with the solver MINOS.

Table 1 Results obtained using GAMS solvers

Prob	n	f_{opt}	SSM			DNLP		
			f_v	n_f	n_s	f_v	n_f	n_t
			CONOPT					
CB2	2	1.9522	1.9522	175	47	1.9523	41	41
WF	2	0.0000	0.0000	219	52	0.0000	41	41
Spiral	2	0.0000	0.0000	85810	10873	0.0480	91	91
Crescent	2	0.0000	0.0000	130	19	0.0000	16	16
EVD52	3	3.5997	3.6000	221	34	3.7595	32	32
Wong1	7	680.6301	680.7588	1074	202	700.7282	32	32
Polak2	10	54.5982	54.6037	297	91	54.6036	43	43
Polak3	11	3.7035	3.7039	301	69	3.9047	73	73
Maxq	20	0.0000	0.0000	853	227	0.0000	224	224
Goffin	50	0.0000	0.0001	10845	3365	0.0041	39435	39435
			MINOS					
CB2	2	1.9522	1.9522	201	57	1.9523	22	173
WF	2	0.0000	0.0000	219	52	0.0000	7	96
Spiral	2	0.0000	0.0000	9294	1306	0.1250	8	77
Crescent	2	0.0000	0.0001	130	19	0.0000	15	125
EVD52	3	3.5997	3.6000	216	35	4.2126	14	145
Wong1	7	680.6301	680.6972	455	130	681.4484	319	1947
Polak2	10	54.5982	54.6036	511	109	60.9528	4	52
Polak3	11	3.7035	3.7037	288	52	4.0400	48	278
Maxq	20	0.0000	62.7313	207	42	0.0000	72	375
Goffin	50	0.0000	52.7149	4577	1307	152.4616	105	704
			SNOPT					
CB2	2	1.9522	1.9522	385	111	1.9522	300	1251
WF	2	0.0000	0.0000	219	52	0.0000	18	112
Spiral	2	0.0000	0.0000	6316	851	0.1250	9	68
Crescent	2	0.0000	0.0001	130	19	0.0004	26	178
EVD52	3	3.5997	3.6000	216	35	3.5997	194	470
Wong1	7	680.6301	680.6974	361	114	686.0448	100	417
Polak2	10	54.5982	54.6036	265	89	54.6036	38	134
Polak3	11	3.7035	3.7037	778	105	4.0006	54	184
Maxq	20	0.0000	0.0000	2373	713	0.0000	130	611
Goffin	50	0.0000	0.0001	12321	3859	111.4632	303	1321

6 Conclusions

In this chapter, we introduced a new method, the subgradient smoothing method, for solving nonsmooth optimization problems. The main feature of this method, which makes it different from other existing nonsmooth optimization methods, is that one can use any smooth optimization method to find descent directions. This allows us to use powerful smooth optimization methods to solve general nonsmooth optimization problems.

We presented results of numerical experiments using well-known nonsmooth optimization test problems. The subgradient smoothing method was implemented both in Fortran and GAMS to compare it with other nonsmooth optimization methods as well as with nonsmooth optimization solvers in GAMS. Results demonstrated that the proposed method is the most robust in comparison with the other five methods used in numerical experiments. The subgradient smoothing method considerably outperforms GAMS solvers with the DNLP option.

Acknowledgements The work is financially supported by the Australian Government through the Australian Research Council's Discovery Projects funding scheme (Project No. DP19000580) and partially by the Academy of Finland (Project No. 319274).

References

1. Bradley, P.S., Fayyad, U.M., Mangasarian, O.L.: Mathematical programming for data mining: Formulations and challenges. INFORMS J. on Comput. **11**, 217–238 (1999)
2. Carrizosa, E., Romero Morales, D.: Supervised classification and mathematical optimization. Comput. and Oper. Res. **40**(1), 150–165 (2013)
3. Clarke, F.H., Ledyaev, Y.S., Stern, R.J., Wolenski, P.R.: Nonsmooth Analysis and Control Theory. Springer, New York (1998)
4. Haslinger, J., Neittaanmäki, P.: Finite Element Approximation for Optimal Shape, Material and Topology Design, 2nd edition. John Wiley & Sons, Chichester (1996)
5. Kärkkäinen, T., Heikkola, E.: Robust formulations for training multilayer perceptrons. Neural Comput. **16**, 837–862 (2004)
6. Karmitsa, N., Bagirov, A., Taheri, S.: New diagonal bundle method for clustering problems in large data sets. Eur. J. of Oper. Res. **263**(2), 367–379 (2017)
7. Karmitsa, N., Bagirov, A., Taheri, S.: Clustering in large data sets with the limited memory bundle method. Pattern Recognit. **83**, 245–259 (2018)
8. Mistakidis, E.S., Stavroulakis, G.E.: Nonconvex Optimization in Mechanics. Smooth and Nonsmooth Algorithms, Heuristics and Engineering Applications by F.E.M. Kluwert Academic Publishers, Dordrecht (1998)
9. Moreau, J., Panagiotopoulos, P.D., Strang, G. (eds.): Topics in Nonsmooth Mechanics. Birkhäuser Verlag, Basel (1988)
10. Outrata, J., Kocvara, M., Zowe, J.: Nonsmooth approach to optimization problems with equilibrium constraints: theory, applications and numerical results, vol. 28. Springer Science & Business Media (2013)
11. Kiwiel, K.C.: Proximity control in bundle methods for convex nondifferentiable minimization. Math. Program. **46**(1), 1059–1066 (1990)
12. Mäkelä, M.M., Neittaanmaki, P.: Nonsmooth Optimization. World Scientific, Singapore (1993)

13. Lukśan, L., Vlćek, J.: A bundle-Newton method for nonsmooth unconstrained minimization. Math. Program.: Series A **83**(1), 373–391 (1998)
14. Lukśan, L., Vlćek, J.: Globally convergent variable metric method for convex nonsmooth unconstrained minimization. J. of Optim. Theory and Appl. **102**(3), 593–613 (1999)
15. Gaudioso, M., Gorgone, E.: Gradient set splitting in nonconvex nonsmooth numerical optimization. Optim. Methods and Softw. **25**(1), 59–74 (2010)
16. Shor, N.Z.: Minimization Methods for Nondifferentiable Functions. Berlin, New York, Springer-Verlag (1985)
17. Burke, J.V., Lewis, A.S., Overton, M.L.: A robust gradient sampling algorithm for nonsmooth, nonconvex optimization. SIAM J. on Optim. **15**(3), 751–779 (2005)
18. Bagirov, A.M., Karasozen, B., Sezer, M.: Discrete gradient method: Derivative-free method for nonsmooth optimization. J. of Optim. Theory and Appl. **137**, 317–334 (2008)
19. Bagirov, A.M., Ganjehlou, A.N.: Quasisecant method for minimizing nonsmooth functions. Optim. Methods and Softw. **25**(1), 3–18 (2010)
20. Haarala, N., Miettinen, K., Mäkelä, M.M.: Globally convergent limited memory bundle method for large-scale nonsmooth optimization. Math. Program. **109**(1), 181–205 (2007)
21. Asl, A., Overton, M.L.: Analysis of limited-memory BFGS on a class of nonsmooth convex functions, arxiv (2018). Preprint
22. Asl, A., Overton, M.L.: Analysis of the gradient method with an Armijo-Wolfe line search on a class of non-smooth convex functions. Optim. Methods and Softw. **35**(2), 223–242 (2020)
23. Curtis, F.E., Mitchell, T., Overton, M.L.: A BFGS-SQP method for nonsmooth nonconvex, constrained optimization and its evaluation using relative minimization profiles. Optim. Methods and Softw. **32**(1), 148–181 (2017)
24. Bagirov, A.M., Karmitsa, N., Mäkelä, M.M.: Introduction to Nonsmooth Optimization: Theory, Practice and Software. Springer (2014)
25. Clarke, F.H.: Optimization and Nonsmooth Analysis. New York, John Wiley (1983)
26. Bagirov, A.M., Al Nuaimat, A., Sultanova, N.: Hyperbolic smoothing function method for minimax problems. Optim. **62**(6), 759–782 (2013)
27. Bagirov, A.M., Al Nuaimat, A., Sultanova, N., Taheri, S.: Solving minimax problems: Local smoothing versus global smoothing. In: M. Al-Baali, L. Grandinetti, A. Purnama (eds.) Numerical Analysis and Optimization, Springer Proceedings in Mathematics & Statistics, pp. 23–43. Springer, Cham (2018)
28. Xavier, A.E.: The hyperbolic smoothing clustering method. Pattern Recognit. **43**, 731–737 (2010)
29. Xavier, A.E., Oliveira, A.A.F.D.: Optimal covering of plane domains by circles via hyperbolic smoothing. J. of Glob. Optim. **31**(3), 493–504 (2005)
30. Bagirov, A.M., Jin, L., Karmitsa, N., Al Nuaimat, A., Sultanova, N.: Subgradient method for nonconvex nonsmooth optimization. J. of Optim. Theory and Appl. **157**(2), 416–435 (2013)
31. Frangioni, A.: Solving semidefinite quadratic problems within nonsmooth optimization algorithms. Comput. and Oper. Res. **23**, 1099–1118 (1996).
32. Kiwiel, K.C.: A dual method for certain positive semidefinite quadratic programming problems. SIAM J. on Sci. Stat. Comput. **10**, 175–186 (1989)
33. Lukśan, L.: Dual method for solving a special problem of quadratic programming as a subproblem at linearly constrained nonlinear minmax approximation. Kybernetika: **20**(6), 445–457 (1984)
34. Lukśan, L., Vlćek, J.: Test problems for nonsmooth unconstrained and linearly constrained optimization. Technical report, No. 78, Institute of Computer Science, Academy of Sciences of the Czech Republic (2000)
35. Lewis, A.S., Overton, M.L.: Nonsmooth optimization via quasi-Newton methods. Math. Program. **141**, 135–163 (2013)
36. Lukśan, L., Vlćek, J.: Algorithm 811: NDA: Algorithms for nondifferentiable optimization. ACM Trans. on Math. Softw. **27**(2), 193–213 (2001)

37. Rosenthal, R.E.: GAMS: a user's manual. GAMS Development Corporation: Washington, DC (2008)
38. Dolan, E.D., More, J.J.: Benchmarking optimization software with performance profiles. Math. Program. **91**, 201–213 (2002)

On Some Optimization Problems that Can Be Solved in $O(n)$ Time

Yanqin Bai and Kees Roos

Abstract We consider nine elementary problems in optimization. We simply explore the conditions for optimality as known from the duality theory for convex optimization. This yields a quite straightforward solution method for each of these problems. The main contribution of this paper is that we show that even in the harder cases the solution needs only $O(n)$ time.

Keywords Optimization problems · Linear time methods · Optimality conditions

1 Introduction

This paper was inspired by a result in [2]. In that paper we needed the optimal objective value of the minimization problem

$$\min_{y,z,\beta}\left\{\|z\| \;:\; y \geq 0,\ \mathbf{1}^T y = 1,\ y = z + \beta v,\ z^T v = 0\right\},$$

where v is a given vector and $\mathbf{1}$ the all-one vector in $\mathbf{R}^n$; the variables are the scalar β and the vectors y and z in $\mathbf{R}^n$. It is a so-called second-order cone problem [1]. It turned out that the problem can be solved analytically in $O(n \log n)$ time. To obtain this result the entries of v must be ordered; this explains the factor $\log n$. The approach

The first author was supported by the National Natural Science Foundations of China, Grant #11771275.

Y. Bai
Department of Mathematics, Shanghai University, Shanghai 200444, China
e-mail: yqbai@shu.edu.cn

K. Roos (✉)
Department of Electrical Engineering, Mathematics and Computer Science,
Technical University Delft, 2628 CD Delft, The Netherlands
e-mail: c.roos@tudelft.nl

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*, Springer Proceedings in Mathematics & Statistics 354, https://doi.org/10.1007/978-3-030-72040-7_4

that led us to this surprising result is quite straightforward. It simply explores the conditions for optimality as known from the duality theory for convex optimization.

It is a natural question whether there are more nontrivial problems that can be solved analytically in a similar way. In this paper we show this true for problems of the following form:

$$\min_x \left\{ \|a - x\|_{p_1} \ : \ \|x\|_{p_2} \leq 1 \right\},$$

where a denotes a given vector in $\mathbf{R}^n$, and p_1 and p_2 are 1, 2 or ∞. In words, given a point $a \in \mathbf{R}^n$, we look for a point x in the unit sphere—with respect to the p_2-norm—that has minimal distance to a—with respect to the p_1-norm. Figure 1 provides a graphical illustration of the solution of each of the nine problems considered in this paper when $n = 2$, and $a = [1.3;\ 0.8]$.

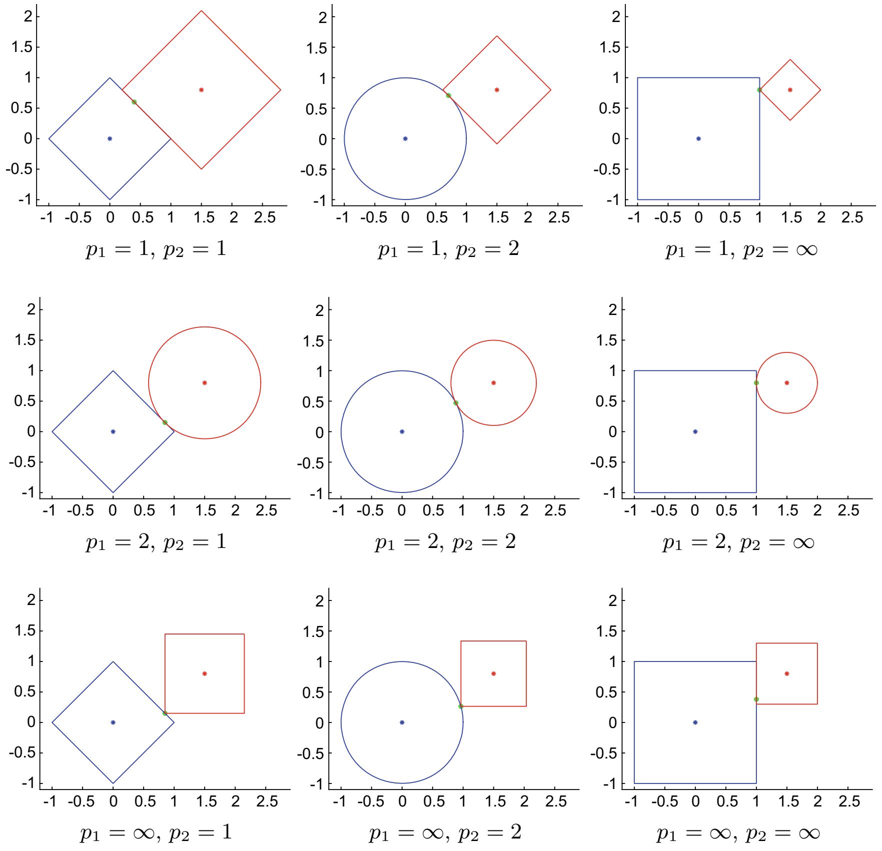


Fig. 1 Illustration of the optimal solutions of the nine problems considered in this paper, for $n = 2$ and $a = [1.3; 0.8]$. The blue dot represents the origin, the red dot a and the green dot the (or sometimes 'an') optimal solution x. The blue curve surrounds the region where the p_2-norm is less than 1, whereas the red curve depicts the p_1-neighborhood of a that just touches the blue region

Obviously, there are nine different (ordered) pairs (p_1, p_2). For each of these nine pairs we show that the above problem can be solved in linear time. In doing so, we always assume without saying that the vector a is ordered nonincreasingly:

$$a_1 \geq a_2 \geq \cdots \geq a_n.$$

It turns out that in some cases (specifically, if $p_1 = p_2$ or $p_2 = \infty$) the solution is trivial, or almost trivial; in other cases this is certainly not obviously the case. But as we show, in each case the problem can be solved in linear time. As far as the authors know, the method leading to this result is new; at least we are not aware of any such result in the existing literature.

In our analysis duality plays a crucial role. As a consequence we also need the so-called dual norm of $\|.\|_p$, for $p \in \{1, 2, \infty\}$, which is defined by

$$\|y\|_{p^*} = \max_x \left\{ x^T y \ : \ \|x\|_p = 1 \right\},$$

where x and y are vectors in $\mathbf{R}^n$. For future use we also recall an important consequence of this definition, namely the so-called Hölder inequality:

$$\|x\|_p \, \|y\|_{p^*} \geq x^T y, \qquad \forall x, y \in \mathbf{R}^n.$$

The outline of the paper is as follows.

Section 2 is preliminary. It consists of four subsections. Section 2.1 describes the fundamental role of duality in our approach. It recalls the so-called *vanishing gap condition* for optimality. For the problems that we consider in this paper this condition implies the primal and dual feasibility conditions, which is quite exceptional. Section 2.2 contains three lemmas dealing with the question of when the Hölder inequality holds with equality, for each of the three values of p considered in this paper. Section 2.3 serves to show that we may restrict our investigations to the case where the given vector a is nonnegative (cf. Lemma 4), and in Sect. 2.4 we distinquish easy types from harder types (p_1, p_2).

Section 3 contains the analysis of the nine problems, each in a separate subsection. Finally, Sect. 4 contains some recommendations for further research.

2 Preliminaries

2.1 Duality

As announced in the previous section, we consider problems of the following form:

$$\min_x \left\{ \|a - x\|_{p_1} \ : \ \|x\|_{p_2} \leq 1 \right\}, \tag{1}$$

where a denotes a given vector in $\mathbf{R}^n$, and p_1 and p_2 are $1, 2$ or ∞. The dual problem of (1) is given by

$$\max_y \left\{ a^T y - \|y\|_{p_2^*} \ : \ \|y\|_{p_1^*} \leq 1 \right\}, \tag{2}$$

where $\|.\|_{p_1^*}$ refers to the dual norm of $\|.\|_{p_1}$, and similarly for p_2.

In one case the solutions of problem (1) and problem (2) are immediate, namely if a is feasible for the primal problem, i.e., $\|a\|_{p_2} \leq 1$. Then $x = a$ solves the primal problem, because then the objective value equals zero, which is minimal. On the other hand, $y = 0$ is feasible for the dual problem, yielding zero as dual objective value. Hence, if we take $x = a$ and $y = 0$ then the feasibility conditions are satisfied and the primal and dual objective values are equal. This means that we have solved the problem in case $\|a\|_{p_2} \leq 1$. We call this the trivial case of the problem.

In the sequel we only consider the nontrivial case, i.e., $\|a\|_{p_2} > 1$. In that case any optimal solution x will satisfy $x \neq a$. Since then $\|a - x\|_{p_1} > 0$, the optimal value of the primal problem will be positive. As a consequence, $y = 0$ does not close the duality gap. Therefore, at optimality we also have $y \neq 0$.

Now let x and y be primal and dual feasible, respectively. Then the duality gap can be reduced as follows:

$$\begin{aligned} \|a - x\|_{p_1} - \left(a^T y - \|y\|_{p_2^*}\right) &= \|a - x\|_{p_1} - a^T y + \|y\|_{p_2^*} \\ &\geq \|a - x\|_{p_1} \|y\|_{p_1^*} - a^T y + \|y\|_{p_2^*} \|x\|_{p_2} \\ &\geq (a - x)^T y - a^T y + y^T x \\ &= 0. \end{aligned}$$

where the second inequality follows by using the Hölder inequality twice. Thus we see that the duality gap vanishes if and only if

$$\|a - x\|_{p_1} = \|a - x\|_{p_1} \|y\|_{p_1^*} = (a - x)^T y \tag{3}$$

and

$$\|y\|_{p_2^*} = \|y\|_{p_2^*} \|x\|_{p_2} = y^T x. \tag{4}$$

Since $x \neq a$, (3) implies $\|y\|_{p_1^*} = 1$, whence $y \neq 0$. The latter implies $\|y\|_{p_2^*} > 0$. But then (4) implies $\|x\|_{p_2} = 1$. We conclude that in the nontrivial case the duality gap vanishes if and only if

$$\|x\|_{p_2} = 1 = \|y\|_{p_1^*} \tag{5}$$

$$\|y\|_{p_2^*} = y^T x \tag{6}$$

$$\|a - x\|_{p_1} = y^T (a - x). \tag{7}$$

Obviously (5) implies that the feasibility conditions in (1) and (2) are satisfied. Therefore, it suffices to solve the above system, under the assumption that $x \neq a$.

As stated before, we assume $p_1, p_2 \in \{1, 2, \infty\}$. For the sake of convenience we call the problems (1) and problem (2) of type (p_1, p_2).

Next we include a section with some lemmas that enable us to restate the conditions (6) and (7) in a way that is more tractable.

2.2 Basic Lemmas

For future use we deal in this section with three elementary lemmas; they deal with the question when Hölder's inequality holds with equality. The first lemma concerns the well-known lemma of Cauchy-Schwartz, where $p^* = p = 2$.

Lemma 1 *The inequality $\|x\|_2 \, \|y\|_2 \geq x^T y$ holds with equality if and only if $x = \lambda y$ or $y = \lambda x$ for some $\lambda \geq 0$.*

Proof We omit the proof, because the result is well-known. □

Less well-known are the next two lemmas that deal with the cases $p = 1$ and $p = \infty$.

Lemma 2 *The inequality $\|x\|_1 \, \|y\|_\infty \geq x^T y$ holds with equality if and only if $x_i y_i \geq 0$ for each i and $x_i \neq 0$ implies $|y_i| = \|y\|_\infty$.*

Proof We may write

$$\|x\|_1 \, \|y\|_\infty = \sum_{i=1}^{n} |x_i| \, \|y\|_\infty \geq \sum_{i=1}^{n} |x_i| \, |y_i| \geq \sum_{i=1}^{n} x_i y_i = x^T y.$$

For each i, the ith terms in the three subsequent summations are not increasing. Hence it follows that $\|x\|_1 \, \|y\|_\infty = x^T y$ holds if and only if these terms are mutually equal. In other words,

$$|x_i| \, \|y\|_\infty = |x_i| \, |y_i| = x_i y_i, \quad 1 \leq i \leq n.$$

The first equality holds if and only if $x_i \neq 0$ implies $|y_i| = \|y\|_\infty$. The second equality holds if and only if $|x_i y_i| = x_i y_i$, which is equivalent to $x_i y_i \geq 0$. □

Lemma 3 *The inequality $\|x\|_\infty \, \|y\|_1 \geq x^T y$ holds with equality if and only if $x_i y_i \geq 0$ for each i and $y_i \neq 0$ implies $|x_i| = \|x\|_\infty$.*

Proof This lemma follows from the previous lemma by interchanging x and y. □

2.3 Simplifying Observations

In this section we mention some properties of optimal solutions x and y of respectively (1) and (2) that are easy to understand. They lead us to the conclusion that in the following nine sections we only need to consider the case where a is a nonnegative vector, and also that we may safely assume that the optimal solutions x and y are nonnegative.

First we note that the contribution of x_i to $\|x\|_{p_2}$, with $p_2 = 1,\ 2$ or ∞, is determined completely by the absolute value $|x_i|$ of x_i. As a consequence, if x is feasible for (1) this will remain so if we change the sign of one or more of the entries in x.

Now consider the expression that we want to minimize: $\|a - x\|_{p_1}$. The contribution of x_i to this expression depends monotonically on $|a_i - x_i|$. If $x_i a_i \geq 0$ then $|a_i + x_i| \geq |a_i - x_i|$. Therefore, we may safely assume that each x_i has the same sign as a_i. A similar argument makes clear that we may assume that each entry y_i has the same sign as a_i, because changing the sign of y_i leaves $\|y\|_{p_1^*}$ and $\|y\|_{p_2^*}$ invariant in (2). On the other hand, the contribution of the product $a_i y_i$ to the dual objective value is maximal if the sign of y_i is the same as that of a_i. Therefore, if y is optimal then $a_i y_i \geq 0$.

We use the above observations as a preparation for the following lemma that makes clear that in the analysis of the system (5)–(7) we may safely assume $a \geq 0$. In this lemma we use a map f_S, where S is a subset of the indices 1 to n, which is defined as follows: for each vector $z \in \mathbf{R}^n$, $f_S(z)$ is the vector that arises from z by changing the signs of the entries z_i, $i \in S$. Obviously, when S is fixed, f_S is one-to-one, and idempotent, i.e., $f_S^2 = f_S$.

Lemma 4 *Let x and y denote solutions of the system (5)–(7) and $S \subseteq \{1,\ 2\ldots,\ n\}$. Then $f_S(x)$ and $f_S(y)$ solve the system when a is replaced by $f_S(a)$.*

Proof Let x, y and S be as in the lemma. It is obvious that $\|x\|_{p_2}$ does not change if x is replaced by $f_S(x)$, because the norm of a vector does only depend on the absolute values of its entries. So, the same holds for the other norms in the system, in particularly also for $\|a - x\|_{p_1}$, since if $i \in S$ then also $a_i - x_i$ changes sign, because $(-a_i) - (-x_i) = -(a_i - x_i)$. Also the inner products do not change, because, e.g., $(-x_i)(-y_i) = x_i y_i$ for each $i \in S$. Hence the lemma follows. □

We apply this lemma as follows. If the vector a has negative entries we define the index set $S = \{i\ :\ a_i < 0\}$. Then $f_S(a) \geq 0$. We then solve the system (5)–(7) with a replaced by $f_S(a)$. Let the solution be denoted as x' and y'. Then it follows from Lemma 4 that $x = f_S(x')$ and $y = f_S(y')$ are the solutions of the original system. As a consequence, below we may always assume that the vector a is nonnegative.

Table 1 A specific solutions of (1) for each of the nine cases

$p_1 \backslash p_2$	1	2	∞
1	$\frac{a}{\|a\|_1}$	$\min(a, \alpha\mathbf{1})$	$\min(a, \mathbf{1})$
2	$(a - \alpha\mathbf{1})^+$	$\frac{a}{\|a\|_2}$	$\min(a, \mathbf{1})$
∞	$(a - \alpha\mathbf{1})^+$	$(a - \alpha\mathbf{1})^+$	$\frac{a}{\|a\|_\infty}$

2.4 Easy and Harder Cases

In the following sections we deal with each of the nine types separately. It will turn out that for five of the nine problem-types a specific solution of (1) can be expressed nicely in a, as shown in Table 1. These are the types with $p_1 = p_2$ or $p_2 = \infty$. We call these types for the moment *easy*. It maybe worth pointing out that $x = \min(a, \mathbf{1})$ solves the primal problem in all cases with $p_2 = \infty$, also if $p_1 = \infty$. From Fig. 1 one easily understands that—at least in some cases—multiple solutions exist. In general, we are not satisfied with the specific solution in Table 1 alone, but we intend to describe the whole set of optimal solutions.

For the remaining four cases Table 1 also shows a specific solution of (1), but their descriptions need besides the vector a also a parameter α. Below we describe in more detail how α can be obtained, for each of the four *hard* cases. The notation x^+ is used to denote the vector that arises from a vector x by replacing its negative entries by zero. In other words, $x^+ = \max(x, 0)$.

Table 2 shows that in all cases one specific dual optimal solution can be expressed in a alone or in a and an arbitrary primal optimal solution x; this will become apparent in the related sections below. In this table $x > 0$ is used to denote the set of indices i for which x_i is positive. In a similar way $a \geq \mathbf{1}$ denotes the index set $\{i \ : \ a_i \geq 1\}$ and $a = \max(a)$ the index set $\{i \ : \ a_i = \max(a)\}$. For any index set I, we use a_I to denote the vector that arises from a by putting $a_i = 0$ if $i \notin I$. This explains the meaning of the notations $\mathbf{1}_{a \geq \mathbf{1}}$ and $\mathbf{1}_{a = \max(a)}$ in Table 2. It may be verified that if $p_1 = 2$ and $p_2 = \infty$ the dual optimal solution can be expressed in a alone; this follows by substitution of the primal optimal solution in Table 1 into $a - x$, which yields the vector $(a - \mathbf{1})_{a > \mathbf{1}}$.

As far as the authors know, up till now problems that are not ‘easy’ in the above sense, can be solved only algorithmically. The main motivation of this paper, however, is to show that these problems are also easy in the sense that they can be solved analytically in $O(n)$ time. So, formally, in terms of computational complexity all nine types belong to the same class. Nevertheless, we will refer to the four types that are not ‘easy’ in the above sense as the harder-types, just to separate them from the ‘easy’ types.

The $O(n)$ time solution method for each of the four harder-type problems is achieved by introducing the parameter α that was mentioned before. It divides the index set $\{1, \ldots, n\}$ into two classes I and J, according to

$$I = \{i \;:\; a_i > \alpha\}, \qquad J = \{i \;:\; a_i \leq \alpha\}.$$

The number α is uniquely determined by a linear or quadratic equation $f(\alpha) = 0$, with $f(\alpha)$ as in Table 3. We use $|I|$ to denote the cardinality of the index set I. The number α and hence also I can be computed in linear time. After this the solution of the problem at hand needs $O(n)$ additional time. For the details we refer to the related sections below.

3 Analysis of the Nine Problems

3.1 Problems of Type (1, 1)

With $\|a\|_1 > 1$, the primal problem is given by

$$\min_u \{\|a - x\|_1 \;:\; \|x\|_1 \leq 1\}, \tag{8}$$

and the dual problem by

$$\max_{y,z} \left\{a^T y - \|y\|_\infty \;:\; \|y\|_\infty \leq 1\right\}. \tag{9}$$

We recall from (5)–(7) the optimality conditions for x and y:

Table 2 Solutions of (2) for the five easy cases

$p_1 \backslash p_2$	1	2	∞
1	$\mathbf{1}$	$\dfrac{x}{\|x\|_\infty}$	$\mathbf{1}_{a\geq\mathbf{1}}$
2	$\dfrac{a-x}{\|a-x\|_2}$	$\dfrac{a}{\|a\|_2}$	$\dfrac{a-x}{\|a-x\|_2}$
∞	$\dfrac{\mathbf{1}_{x>0}}{\|\mathbf{1}_{x>0}\|_1}$	$\dfrac{x}{\|x\|_1}$	$\dfrac{\mathbf{1}_{a=\max(a)}}{\left\|\mathbf{1}_{a=\max(a)}\right\|_1}$

Table 3 Definition of the number α

type	$f(\alpha)$	α
(1, 2)	$1 - \|a_J\|_2^2 - \|I\|\,\alpha^2$	$\|x\|_\infty$
(2, 1)	$1 - \|a_I\|_1 + \|I\|\,\alpha$	$\|a - x\|_\infty$
$(\infty, 1)$	$1 - \|a_I\|_1 + \|I\|\,\alpha$	$\|a - x\|_\infty$
$(\infty, 2)$	$1 - \|a_I\|_2^2 + 2\alpha\,\|a_I\|_1 - \|I\|\,\alpha^2$	$\|a - x\|_\infty$

Table 4 Optimal solutions for type (1, 1)

a_i	x_i	y_i
> 0	$\leq a_i$	1
0	0	$0 \leq y_i \leq 1$

$$\|x\|_1 = 1 = \|y\|_\infty \tag{10}$$
$$\|y\|_\infty = y^T x. \tag{11}$$
$$\|a - x\|_1 = (a - x)^T y \tag{12}$$

As explained in Sect. 2.3 we may assume that a, and also x and y are nonnegative. According to Lemma 2, if (10) holds, then (11) is equivalent to

(i) for each i: $x_i \neq 0$ implies $y_i = \|y\|_\infty = 1$.

Similarly, by Lemma 3, if (10) holds, then (12) is equivalent to

(ii) for each i: $(a_i - x_i)\, y_i \geq 0$ and $a_i - x_i \neq 0$ implies $y_i = \|y\|_\infty = 1$,

Next we derive properties from the above conditions. Suppose that $a_i > 0$ for some i. Then either $x_i \neq 0$ or $a_i - x_i \neq 0$. Hence, by (i), (ii) and (10), $y_i = \|y\|_\infty = 1$. But then (ii) also implies $x_i \leq a_i$. This justifies the first line in Table 4.

The second line deals with the case where $a_i = 0$. If $x_i > 0$, we get from (i) that $y_i = 1$. As in the previous case, then (ii) gives $x_i \leq a_i$, whence $x_i = 0$. Since $\|y\|_\infty = 1$, this justifies the second line in Table 4.

This is all the information we can extract from the system (10)–(12). It means that the two (lower) lines in Table 4 represent all the possibilities for the triples (a_i, x_i, y_i), provided that $\|x\|_1 = 1$. In general multiple optimal solutions for problem (8) exist, because every vector x satisfying

$$0 \leq x \leq a, \qquad \|x\|_1 = 1$$

is optimal. Since $\|a\|_1 > 1$, one of these vectors is $x = a/\|a\|_1$, as given in Table 1.

If a has only positive entries then (9) has only one optimal solution, namely $y = \mathbf{1}$. If a has zero entries, then also other solutions exist. Then any vector y satisfying

$$\mathbf{1}_{a>0} \leq y \leq \mathbf{1},$$

is optimal, where $\mathbf{1}_{a>0}$ denotes the vector whose entries are 1 where a is positive and zero elsewhere.

3.2 Problems of Type (1, 2)

In this section the primal problem is

$$\min_x \left\{ \|a - x\|_1 \ : \ \|x\|_2 \leq 1 \right\}, \tag{13}$$

where $\|a\|_2 > 1$. Its dual problem is

$$\max_y \left\{ a^T y - \|y\|_2 \ : \ \|y\|_\infty \leq 1 \right\}, \tag{14}$$

According to (5)–(7), x is optimal for (13) and y for (14) if and only if

$$\|x\|_2 = 1 = \|y\|_\infty \tag{15}$$

$$\|y\|_2 = y^T x \tag{16}$$

$$\|a - x\|_1 = (a - x)^T y. \tag{17}$$

As established in Sect. 2.3, we may take for granted that $a \geq 0$, $x \geq 0$ and $y \geq 0$.

We have $\|y\|_\infty = 1$, by (15). So $y \neq 0$. Also, $\|x\|_2 = 1$. As a consequence, (16) holds if and only if $\|x\|_2 \, \|y\|_2 = y^T x$. This in turn is equivalent with

(i) $x = \dfrac{y}{\|y\|_2}$,

by Lemma 1. Moreover, by Lemma 3 (17) holds if and only if

(ii) for each i: $(a_i - x_i) y_i \geq 0$ and if $a_i - x_i \neq 0$ then $y_i = \|y\|_\infty = 1$.

Since $y = \|y\|_2 \, x$, by (i), and $y \neq 0$, we may conclude that x_i and y_i have the same sign, for each i, and they vanish at the same time. Therefore, (ii) implies $x_i \leq a_i$, for each i. We define

$$I := \{ i \ : \ x_i < a_i \}, \qquad J := \{ i \ : \ x_i = a_i \}, \tag{18}$$

Now let $i \in I$ and $j \in J$. Since $a_i > x_i$, (ii) implies $y_i = \|y\|_\infty$. Since $y = \|y\|_2 \, x$, we also have $x_i = \|x\|_\infty$. It follows that

$$a_i > x_i = \|x\|_\infty \geq x_j = a_j, \quad i \in I, j \in J. \tag{19}$$

This shows that the entries in a_I are strictly larger than those in a_J.

Recall that we always assume that the entries of a are ordered nonincreasingly. Therefore, (19) implies the existence of an index q such that

$$I = \{ i \ : \ i \leq q \}, \qquad J = \{ i \ : \ i > q \}. \tag{20}$$

Putting $\alpha = \|x\|_\infty$, we see that (19) holds if and only if

$$a_q > \alpha \geq a_{q+1}, \tag{21}$$

Moreover, when knowing q and α, x uniquely follows from (18) and (19), according to

$$x_i = \begin{cases} \alpha, & \text{if } i \le q \\ a_i, & \text{if } i > q. \end{cases} \tag{22}$$

Since x is nonzero and $\|y\|_\infty = 1$, we deduce from $y = \|y\|_2\, x$ that

$$y = \frac{x}{\|x\|_\infty}. \tag{23}$$

Next we arrive at the main objective of this paper, namely to show that in the current case q and also α can be found in $O(n)$ time. Because of (22) and (23) we may therefore conclude that (13) and (14) can be solved in $O(n)$ time.

From (15) we get $\|x\|_2 = 1$. Also using (22) we may write

$$1 = \|x\|_2^2 = \sum_{i=1}^{n} x_i^2 = \sum_{i \le q} {x_i}^2 + \sum_{i>q} {x_i}^2 = q\alpha^2 + \sum_{i>q} a_i^2.$$

Since $q = |I|$ and $\sum_{i>q} a_i^2 = \|a_J\|_2^2$, we recognize at this stage that α satisfies $f(\alpha) = 0$, with $f(\alpha)$ as defined in Table 3 for type (1, 2). Since α is nonnegative, α uniquely follows from q, because $f(\alpha) = 0$ holds if and only if

$$\alpha^2 = \frac{1 - \sum_{i>q} a_i^2}{q}.$$

As the next lemma reveals, q uniquely follows from (21). In order to prove this we define the vector τ as follows:

$$\tau_k = \frac{1 - \sum_{i>k} a_i^2}{k}, \quad 1 \le k \le n. \tag{24}$$

We then must find q such that $\alpha^2 = \tau_q$, with τ_q satisfying

$$a_q^2 > \tau_q \ge a_{q+1}^2. \tag{25}$$

Lemma 5 *q is the first index such that*

$$\tau_q = \max_k \{\tau_k \;:\; 1 \le k \le n\}. \tag{26}$$

Proof For $k < n$ the definitions of τ_k and τ_{k+1} imply

$$(k+1)\tau_{k+1} = 1 - \sum_{i>k+1} a_i^2 = a_{k+1}^2 + 1 - \sum_{i>k} a_i^2 = a_{k+1}^2 + k\tau_k. \tag{27}$$

This can be rewritten in the following two ways:

$$\begin{aligned}(k+1)\,(\tau_{k+1}-\tau_k) &= a_{k+1}^2-\tau_k\\ k\,(\tau_{k+1}-\tau_k) &= a_{k+1}^2-\tau_{k+1}.\end{aligned}$$

From this we deduce

$$\tau_{k+1} > \tau_k \quad \Leftrightarrow \quad a_{k+1}^2 > \tau_k \quad \Leftrightarrow \quad a_{k+1}^2 > \tau_{k+1}. \tag{28}$$

So, τ is (strictly!) increasing at k if and only if $a_{k+1}^2 > \tau_k$ and this holds if and only if $a_{k+1}^2 > \tau_{k+1}$, for each $k < n$. From this we draw two conclusions. First that (25) holds if and only if τ is increasing at $k = q-1$ and nonincreasing at $k = q$. Second, if τ is nonincreasing at some $k < n$ it remains nonincreasing if k increases. This can be understood as follows. Suppose that τ is nonincreasing at some $k < n$, i.e., $\tau_{k+1} \le \tau_k$. Then $a_{k+1}^2 \le \tau_{k+1}$. Since $0 \le a_{k+2} \le a_{k+1}$, it follows that also $a_{k+2}^2 \le \tau_{k+1}$. This in turn implies $\tau_{k+2} \le \tau_{k+1}$, which proves the claim. The above two properties imply the statement in the lemma. □

The vector τ can be computed in $O(n)$ time by first computing τ_1 and then using (27), which gives[1]:

$$\tau_1 = 1 + a_1^2 - \|a\|^2, \qquad \tau_{k+1} = \frac{a_{k+1}^2 + k\tau_k}{k+1}, \quad 1 \le k < n. \tag{29}$$

Then (26) yields the value of q, still in $O(n)$ time. As mentioned before, this means that the current approach solves problem (13) and problem (14) in $O(n)$ time. Obviously, both solutions are unique.

Example 1 Table 5 shows the outcome of our analysis for a randomly generated vector a. It shows that τ is maximal at $k = 5$. So $I = \{1, \ldots, 5\}$, and $\alpha = \sqrt{\tau_5} = 0.3554$. So $x_i = 0.3554$ for $i \in I$ and $x_i = a_i$ for $i > 5$.

3.3 *Problems of Type* $(1, \infty)$

With $\|a\|_\infty > 1$, we consider the problem

$$\min_x \{\|a - x\|_1 \;:\; \|x\|_\infty \le 1\}. \tag{30}$$

The dual of this problem is

$$\max_y \left\{a^T y - \|y\|_1 \;:\; \|y\|_\infty \le 1\right\}. \tag{31}$$

The conditions for optimality are

[1] It may be worth mentioning that (29) reveals that τ_{i+1} is a convex combination of a_{i+1}^2 and τ_i.

Table 5 Numerical illustration type (1, 2)

i	a_i	τ_i	x_i	$a - x$	y_i
1	0.9293	−1.0048	0.3554	0.5739	1.0000
2	0.8308	−0.1573	0.3554	0.4754	1.0000
3	0.6160	0.0216	0.3554	0.2606	1.0000
4	0.5853	0.1019	0.3554	0.2299	1.0000
$5 = q$	0.4733	0.1263	0.3554	0.1179	1.0000
6	0.3517	0.1259	0.3517	0.0000	0.9897
7	0.3500	0.1254	0.3500	0.0000	0.9849
8	0.2511	0.1176	0.2511	0.0000	0.7066
9	0.2435	0.1111	0.2435	0.0000	0.6852
10	0.0000	0.1000	0.0000	0.0000	0.0000

Table 6 Optimal solutions for type $(1, \infty)$

a_i	x_i	y_i
> 1	1	1
$= 1$	1	$\in [0, 1]$
< 1	a_i	0

$$\|x\|_\infty = 1 = \|y\|_\infty \tag{32}$$
$$\|y\|_1 = y^T x \tag{33}$$
$$\|a - x\|_1 = (a - x)^T y. \tag{34}$$

As always we assume that $a \geq 0$, $x \geq 0$ and $y \geq 0$. According to Lemma 3, if (32) holds, then (33) holds if and only if

(i) for each i: $y_i \neq 0$ implies $x_i = \|x\|_\infty$;

and, by the same lemma, if (32) holds, then (34) holds if and only if

(ii) for each i: $(a_i - x_i)\, y_i \geq 0$ and $a_i - x_i \neq 0$ implies $y_i = \|y\|_\infty$.

We consider three cases, according to the value of a_i.

Let $a_i > 1$. Since $x_i \leq \|x\|_\infty = 1$, we then have $a_i - x_i > 0$. Then (ii) implies $y_i = \|y\|_\infty = 1$, and because of this (i) implies $x_i = \|x\|_\infty = 1$, where we also used (32). So, if $a_i > 1$, then $x_i = 1$ and $y_i = 1$.

If $a_i < 1$, we must have $y_i = 0$. Because otherwise $y_i > 0$, and then (i) would give $x_i = 1$ again. But then $a_i - x_i < 0$. This would imply $(a_i - x_i)\, y_i < 0$, contradicting (ii). So $y_i = 0$. But then we have $y_i < \|y\|_\infty$, which implies $x_i = a_i$, by (ii).

Finally, let $a_i = 1$. Suppose $x_i \neq a_i$. Then (ii) implies $y_i = 1$. Then, as before, (i) implies $x_i = 1$, whence $x_i = a_i$. Note that in that case (i) and (ii) are satisfied.

We conclude that at optimality x and y are as given in Table 6.

The primal solution is unique, and as given in Table 1, namely $x = \min(a, \mathbf{1})$. On the other hand, if all entries of a differ from 1, y is also unique. More precisely, then $y = \mathbf{1}_{a>\mathbf{1}}$. Otherwise there are multiple optimal solution. Every vector y such that

$$\mathbf{1}_{a>\mathbf{1}} \leq y \leq \mathbf{1}_{a\geq\mathbf{1}}$$

is dual optimal.

3.4 *Problems of Type* (2, 1)

The problem that we consider in this section is

$$\min_x \left\{ \|a - x\|_2 \ : \ \|x\|_1 \leq 1 \right\}, \tag{35}$$

where $\|a\|_1 > 1$. Its dual problem is

$$\max_{y,z} \left\{ a^T y - \|y\|_\infty \ : \ \|y\|_2 \leq 1 \right\}. \tag{36}$$

According to (5)–(7), x is optimal for (35) and y for (36) if and only if

$$\|x\|_1 = 1 = \|y\|_2 \tag{37}$$
$$\|y\|_\infty = y^T x \tag{38}$$
$$\|a - x\|_2 = (a - x)^T y. \tag{39}$$

As before, under reference to Sect. 2.3, we assume that a, x and y are nonnegative. Then Lemma 2, (37) and (38) imply

(i) for each i: $x_i \neq 0$ implies $y_i = \|y\|_\infty$,

whereas, by Lemma 1, (37) and (39) imply

(ii) $y = \dfrac{a - x}{\|a - x\|_2}$.

We define

$$I := \{i \ : \ x_i > 0\}, \qquad J := \{i \ : \ x_i = 0\}. \tag{40}$$

Let $i \in I$. Then (i) implies $y_i = \|y\|_\infty$. Due to (37), $y \neq 0$. Hence $y_i > 0$. Because of (ii) we thus obtain $a_i > x_i$. From $y_i = \|y\|_\infty$ and (ii) we deduce that $a_i - x_i = \|a - x\|_\infty$. Now defining

$$\alpha = \|a - x\|_\infty, \tag{41}$$

we get $a_i - x_i = \alpha > 0$, whence

$$x_i = a_i - \alpha, \quad i \in I. \tag{42}$$

Hence

$$\|x\|_1 = \sum_{i \in I} (a_i - \alpha) = \|a_I\|_1 - |I|\,\alpha.$$

Since $\|x\|_1 = 1$, we obtain $f(\alpha) = 0$, where $f(\alpha) = 1 - \|a_I\|_1 + |I|\,\alpha$, as announced in Table 3 for type (2, 1). This gives

$$\alpha = \frac{\|a_I\|_1 - 1}{|I|}. \tag{43}$$

Thus we find that if the index set I is known, then we can compute x and y: first one computes α from (43), and then x_I from (42). Since $x_J = 0$, we then know x, and y follows from (ii).

The question remains how we can find I. For that purpose we first observe that if $i \in I$ and $j \in J$ then

$$a_i = x_i + \alpha > \alpha = \|a - x\|_\infty \geq a_j - x_j = a_j. \tag{44}$$

This shows that the entries in a_I are strictly larger than those in a_J. Since the entries of a are ordered nonincreasingly, there must exist an index q such that

$$I = \{i \ : \ i \leq q\}, \quad J = \{i \ : \ i > q\}.$$

Then (44) holds if and only if

$$a_q > \alpha \geq a_{q+1}, \tag{45}$$

with α as in (43). We define the vector τ according to

$$\tau_k = \frac{\sum_{i \leq k} a_i - 1}{k}, \quad 1 \leq k \leq n. \tag{46}$$

Then (45) holds if and only if

$$a_q > \tau_q \geq a_{q+1}, \tag{47}$$

and then we necessarily have $\alpha = \tau_q$. We are now in a similar situation as in Sect. 3.2, and we proceed accordingly with the next lemma.

Lemma 6 *q is the first index such that*

$$\tau_q = \max_k \{\tau_k \ : \ 1 \leq k \leq n\}. \tag{48}$$

Proof For $k < n$ the definition of τ_k implies

$$(k+1)\tau_{k+1} = \sum_{j\le k+1} a_j - 1 = a_{k+1} + \sum_{j\le k} a_j - 1 = a_{k+1} + k\tau_k. \tag{49}$$

This can be rewritten in the following two ways:

$$\begin{aligned}(k+1)\,(\tau_{k+1} - \tau_k) &= a_{k+1} - \tau_k\\ k\,(\tau_{k+1} - \tau_k) &= a_{k+1} - \tau_{k+1}.\end{aligned}$$

From this we deduce

$$\tau_{k+1} > \tau_k \quad \Leftrightarrow \quad a_{k+1} > \tau_k \quad \Leftrightarrow \quad a_{k+1} > \tau_{k+1}, \tag{50}$$

which proves that τ is increasing at k if and only if $a_{k+1} > \tau_k$ and this holds if and only if $a_{k+1} > \tau_{k+1}$, for each $k < n$. From here on we can use the same arguments as in the proof of Lemma 5. From (47) we conclude that τ is increasing at $k = q - 1$ and nonincreasing at $k = q$. Next, if τ is nonincreasing at some $k < n$ it remains nonincreasing if k increases, because if $\tau_{k+1} \le \tau_k$ then $a_{k+1} \le \tau_{k+1}$. Since $0 \le a_{k+2} \le a_{k+1}$, it follows that also $a_{k+2} \le \tau_{k+1}$. This in turn implies $\tau_{k+2} \le \tau_{k+1}$, proving the claim. From this the lemma follows. □

As in Sect. 3.4, the vector τ can be computed in $O(n)$ recursively from[2]

$$\tau_1 = a_1 - 1, \qquad \tau_{k+1} = \frac{a_{k+1} + k\tau_k}{k+1}, \quad 1 \le k < n. \tag{51}$$

Then (48) yields the value of q, still in $O(n)$ time. Due to (42) this means that problem (35) and it dual problem can be solved in $O(n)$ time. Obviously, the solutions of (35) and (36) are unique.

Example 2 Table 7 demonstrates our analysis for a randomly generated vector a. It shows that τ is maximal at $q = 5$. So $I = \{1, \ldots, 5\}$, and $\tau = 1.2799$.

3.5 Problems of Type (2, 2)

The primal problem is

$$\min_x \{\|a - x\|_2 \ : \ \|x\|_2 \le 1\}, \tag{52}$$

with $\|a\|_2 > 1$, and its dual problem

$$\max_y \left\{a^T y - \|y\|_2 \ : \ \|y\|_2 \le 1\right\}. \tag{53}$$

[2] It may be worth mentioning that (51) reveals that τ_{k+1} is a convex combination of a_{k+1} and τ_k.

Table 7 Numerical illustration type (2, 1)

i	a_i	τ_i	x_i	$a_i - x_i$	y_i
1	1.6363	0.6363	0.3564	1.2799	0.4181
2	1.6351	1.1357	0.3552	1.2799	0.4181
3	1.4449	1.2388	0.1650	1.2799	0.4181
4	1.3639	1.2701	0.0841	1.2799	0.4181
$5 = q$	1.3192	1.2799	0.0393	1.2799	0.4181
6	1.0433	1.2405	0.0000	1.0433	0.3409
7	0.2997	1.1061	0.0000	0.2997	0.0979
8	0.0000	0.9678	0.0000	0.0000	0.0000
9	0.0000	0.8603	0.0000	0.0000	0.0000
10	0.0000	0.7742	0.0000	0.0000	0.0000

According to (5)–(7) the optimality conditions are

$$\|x\|_2 = 1 = \|y\|_2 \tag{54}$$
$$\|y\|_2 = y^T x \tag{55}$$
$$\|a - x\|_2 = (a - x)^T y. \tag{56}$$

According to Lemma 1, (54) and (55) hold if and only if

(i) $x = \frac{y}{\|y\|_2}$,

and by the same lemma, (54) and (56) hold if and only if

(ii) $y = \frac{a-x}{\|a-x\|_2}$.

From (i) we derive that x and y have the same direction. Since x and y are both unit vectors, we must have $y = x$. By (ii), the vectors y and $a - x$ have the same direction. Since $y \neq 0$ this implies $a - x = \alpha y$ for some $\alpha > 0$. Thus we obtain $(1 + \alpha)x = a$. This proves that x has the same direction as a. Since x is a unit vector, it follows that $x = \frac{a}{\|a\|_2}$, as in Table 1. Since $y = x$, we have solved (52) and (53). In this case both the primal and the dual solution are unique.

3.6 Problems of Type $(2, \infty)$

The problem can then be stated as

$$\min_u \{\|a - x\|_2 \ : \ \|x\|_\infty \leq 1\}. \tag{57}$$

The dual problem is

Table 8 Optimal solutions for type $(2, \infty)$

a_i	x_i	y_i
> 1	1	$(a_i - x_i)/\|a - x\|_2$
≤ 1	a_i	0

$$\max_{y,z} \left\{ a^T y - \|y\|_1 \ : \ \|y\|_2 \leq 1 \right\}. \tag{58}$$

As in previous sections, we assume $\|a\|_\infty > 1$ and that x, y and a are nonnegative. According to (5)–(7), x is optimal for (57) and y for (58) if and only if

$$\|x\|_\infty = 1 = \|y\|_2 \tag{59}$$

$$\|y\|_1 \quad = \quad y^T x \tag{60}$$

$$\|a - x\|_2 \quad = \quad (a - x)^T y. \tag{61}$$

Let us assume (59). Then Lemma 3 states that (60) holds if and only if

(i) for each i: $y_i \neq 0$ implies $x_i = \|x\|_\infty = 1$,

whereas Lemma 1 states that (61) holds if and only if

(ii) $y = \dfrac{a - x}{\|a - x\|_2}$.

At optimality $\|a - x\|_2 > 0$, whence $x \neq a$. Let i be such that $y_i > 0$. Then (i) implies $x_i = 1$. Since y_i and $a_i - x_i$ have the same sign, we get $a_i > x_i = 1$.

We just showed that $y_i > 0$ implies $a_i > 1$. As a consequence we have $y_i = 0$ if $a_i \leq 1$. By (ii) we then have $x_i = a_i$. We conclude that at optimality x and y are as given in Table 8. It follows that both x and y are unique, with x as in Table 1: $x = \min(a, \mathbf{1})$.

3.7 Problems of Type $(\infty, 1)$

With $\|a\|_1 > 1$, we consider the problem

$$\min_x \{\|a - x\|_\infty \ : \ \|x\|_1 \leq 1\}. \tag{62}$$

The dual of this problem is

$$\max_y \left\{ a^T y - \|y\|_\infty \ : \ \|y\|_1 \leq 1 \right\}. \tag{63}$$

As before, we only consider the case where a, x and y are nonnegative. The optimality conditions are

$$\|x\|_1 = 1 = \|y\|_1 \tag{64}$$
$$\|y\|_\infty = y^T x \tag{65}$$
$$\|a-x\|_\infty = (a-x)^T y. \tag{66}$$

According to Lemma 2, if (64) holds, then (65) is equivalent to

(i) for each i: $x_i \neq 0$ implies $y_i = \|y\|_\infty$;

and, for the same reason, then (66) is equivalent to

(ii) for each i: $y_i \neq 0$ implies $a_i - x_i = \|a-x\|_\infty$.

We partition the index set in the same way as in Sect. 3.4. So

$$I = \{i \ : \ x_i > 0\}, \quad J = \{i \ : \ x_i = 0\}.$$

Then (i) implies

$$y_i = \|y\|_\infty, \quad i \in I. \tag{67}$$

Since $y \neq 0$, by (64), we get $y_i > 0$. So (ii) applies, which implies $a_i - x_i = \|a-x\|_\infty$. Defining

$$\alpha = \|a-x\|_\infty, \tag{68}$$

it follows that

$$x_i = a_i - \alpha, \quad i \in I, \tag{69}$$

and hence we may write

$$\|x\|_1 = \sum_{i\in I}(a_i - \alpha) = \|a_I\|_1 - |I|\,\alpha.$$

Since $\|x\|_1 = 1$ we obtain

$$\alpha = \frac{\|a_I\|_1 - 1}{|I|}. \tag{70}$$

So, when we know I we can compute α from (70), and then the nonzero entries of x follows from (68). An interesting observation is that the formula for α is the same as (43) in Sect. 3.4. Like there, we also may write

$$a_i = x_i + \alpha > \alpha = \|a-x\|_\infty \geq a_j - x_j = a_j, \quad i \in I,\ j \in J, \tag{71}$$

Hence we have, for some index q,

$$I = \{i \ : \ i \leq q\}, \quad J = \{i \ : \ i > q\}.$$

Then (71) holds if and only if

$$a_q > \alpha \geq a_{q+1}, \tag{72}$$

with α as in (69). Thus the problem of finding q is the exactly the same as in Sect. 3.4. So we may state without further proof the following lemma.

Lemma 7 *One has $\alpha = \tau_q$, where q is the first index such that*

$$\tau_q = \max_k \{\tau_k \ : \ 1 \leq k \leq n\}, \tag{73}$$

and where the vector τ is defined recursively by

$$\tau_1 = a_1 - 1, \qquad \tau_{k+1} = \frac{a_{k+1} + k\tau_k}{k+1}, \quad 1 \leq k < n. \tag{74}$$

This means that problem (62) can be solved in $O(n)$ time, and the solution is unique.

In Sect. 3.4 the dual vector y was uniquely determined by x. This is now different, as becomes clear below. We derived from (i) that for indices $i \in I$, where x is positive, the entries y_i are positive and equal to $\|y\|_\infty$. If $i \in J$, where x is zero, (ii) requires that if $a_i \neq \|a - x\|_\infty$ then $y_i = 0$. So, if $a_i = \alpha$ then condition (ii) is void, and hence the only condition on y_i becomes $0 \leq y_i \leq \|y\|_\infty$. This can happen only if $a_{q+1} = \alpha$. Since $\alpha = \tau_q$ this is equivalent to $\tau_{q+1} = \tau_q$, by (74). Stated otherwise, we can have $0 \leq y_{q+1} \leq \|y\|_\infty$ if and only if τ is not decreasing at q. More generally, if q' is the highest index at which τ is maximal, with $q' \geq q$, i.e., if

$$\tau_q = \tau_{q+1} = \cdots = \tau_{q'} = \alpha,$$

which happens if and only if

$$a_q = a_{q+1} = \cdots = a_{q'} = \alpha. \tag{75}$$

then for any i such that $q \leq i \leq q'$ we can have $0 \leq y_i \leq \|y\|_\infty$. Any such vector y is obtained by first defining a vector z as follows:

$$z_i = \begin{cases} 1 & \text{if } i \leq q, \\ \in [0, 1] & \text{if } q < i \leq q', \\ 0 & \text{if } i > q', \end{cases} \tag{76}$$

and then taking

$$y = \frac{z}{\|z\|_1}. \tag{77}$$

We then have $\|y\|_1 = 1$ and, because of (69) and (75), for each positive y_i that $a_i - x_i = \alpha = \|a - x\|_\infty$. This implies that y is dual feasible and also optimal.

Example 3 Table 9 demonstrates our analysis for a given vector a. It shows that τ is maximal at $q = 5$. So $I = \{1, \ldots, 5\}$, and $\alpha = \tau_5 = \tau_6 = 0.5362$.

Table 9 Numerical illustration type $(\infty, 1)$

i	a_i	τ_i	x_i	$a_i - x_i$	y_i
1	0.9174	−0.0826	0.3812	0.5362	0.1772
2	0.7655	0.3415	0.2293	0.5362	0.1772
3	0.7384	0.4738	0.2022	0.5362	0.1772
4	0.6834	0.5262	0.1472	0.5362	0.1772
$5 = q$	0.5762	0.5362	0.0400	0.5362	0.1772
$6 = q'$	0.5362	0.5362	0.0000	0.5362	0.1142
7	0.2691	0.4980	0.0000	0.2691	0.0000
8	0.2428	0.4661	0.0000	0.2428	0.0000
9	0.1526	0.4313	0.0000	0.1526	0.0000
10	0.0000	0.3882	0.0000	0.0000	0.0000

3.8 Problems of Type $(\infty, 2)$

With $\|a\|_2 > 1$, we consider the problem

$$\min_x \{\|a - x\|_\infty \; : \; \|x\|_2 \le 1\} . \tag{78}$$

The dual of this problem is

$$\max_y \left\{a^T y - \|y\|_2 \; : \; \|y\|_1 \le 1\right\} . \tag{79}$$

As always, $a \ge 0$, $x \ge 0$ and $y \ge 0$. The conditions for optimality are

$$\|x\|_2 = 1 = \|y\|_1 \tag{80}$$
$$\|y\|_2 = y^T x \tag{81}$$
$$\|a - x\|_\infty = (a - x)^T y \tag{82}$$

According to Lemma 1, if (80) holds, then (81) is equivalent to

(i) $x = \dfrac{y}{\|y\|_2}$;

and, by Lemma 2, (82) is equivalent to

(ii) for each i: $y_i \ne 0$ implies $a_i - x_i = \|a - x\|_\infty$.

With

$$I = \{i \; : \; x_i > 0\} , \quad J = \{i \; : \; x_i = 0\} ,$$

the pair (I, J) is a partition of the index set. Let $i \in I$. So, $x_i > 0$. Now (i) implies $y_i > 0$. Therefore, (ii) implies $a_i - x_i = \|a - x\|_\infty$. Since $\|a - x\|_\infty > 0$, we get $x_i < a_i$. To simplify the presentation we define

$$\alpha = \|a - x\|_\infty \,. \tag{83}$$

Then we have

$$x_i = a_i - \alpha, \quad i \in I. \tag{84}$$

Now let $j \in J$. Then using $x_j = 0$, we may write

$$a_i = x_i + \alpha > \alpha = \|a - x\|_\infty \geq a_j - x_j = a_j. \tag{85}$$

This proves that the entries in a_I are strictly larger than those in a_J. Since the entries of a are ordered nonincreasingly, we get, for some q,

$$I = \{1, 2, \ldots, q\}, \qquad J = \{q + 1, \ldots, n\}.$$

Assuming that J is not empty, (85) implies

$$a_q > \alpha \geq a_{q+1}. \tag{86}$$

Otherwise, i.e., when $q = n$, we define $a_{n+1} = 0$; so we can always work as if (86) holds. Because of (80) we may write

$$1 = \|x\|_2^2 = \sum_{i \in I} x_i^2 = \sum_{i \in I} (a_i - \alpha)^2 = \|a_I\|_2^2 - 2\alpha \, \|a_I\|_1 + |I| \, \alpha^2.$$

Thus we obtain that α is one of the two roots of the equation $f(\alpha) = 0$, where

$$f(\alpha) := 1 - \|a_I\|_2^2 + 2\alpha \, \|a_I\|_1 - |I| \, \alpha^2.$$

Before proceeding, it will be convenient to introduce the notation

$$\sigma_{jk} := \sum_{i=1}^{k} a_i{}^j, \quad j \in \{1, 2\}, \, k \in \{1, \ldots, n\}. \tag{87}$$

Then $\|a_I\|_1 = \sigma_{1q}$ and $\|a_I\|_2^2 = \sigma_{2q}$, and hence $f(\alpha)$ can be rewritten as

$$f(\alpha) = 1 - \sigma_{2q} + 2\alpha\sigma_{1q} - q\alpha^2. \tag{88}$$

Now the sum of the two roots equals $2\sigma_{1q}/q$. So their average value is σ_{1q}/q. By (86) we have $a_q > \alpha$. Combining this with $a_1 \geq a_2 \geq \cdots \geq a_q$, we conclude that $\sigma_{1q} > q\alpha$, whence $\sigma_{1q}/q > \alpha$. It thus follows that the root α is smaller than the other root. This means that the discriminant of the equation $f(\alpha) = 0$ is positive. In other words

$$\sigma_{1q}^2 - q\left(\sigma_{2q} - 1\right) > 0. \tag{89}$$

Motivated by the solution technique developed in some of the preceding sections, we define

$$f_k(\xi) = 1 - \sigma_{2k} + 2\xi\sigma_{1k} - k\xi^2, \qquad 1 \le k \le n. \tag{90}$$

and

$$\omega_k := \sigma_{1k}^2 - k\left(\sigma_{2k} - 1\right), \qquad \tau_k := \frac{1}{k}\left(\sigma_{1k} - \sqrt{\omega_k}\right), \qquad 1 \le k \le n. \tag{91}$$

Obviously, ω_k is just the discriminant of the equation $f_k(\xi) = 0$ and if $\omega_k \ge 0$ then τ_k is its smallest root. In particular, $\tau_q = \alpha$. Hence, according to (86) we need to find q such that

$$a_q > \tau_q \ge a_{q+1}. \tag{92}$$

When knowing q, α follows from $\alpha = \tau_q$, and then x follows from (84).

The question remains how much time it takes to solve q from (92) and similarly for x and y. We claim that all this can be done in $O(n)$ time. This can be understood as follows.

Clearly, $\omega_1 = 1$ and $\tau_1 = a_1 - 1$. For $j = 1, 2$, the recursive computation of $\sigma_{j1}, \ldots, \sigma_{jq}$ requires $O(q)$ time, and so does the computation of ω_q and τ_q. If we have found q such that τ_q satisfies (92), then we also know α, because $\alpha = \tau_q$. Then x follows from $x_i = a_i - \alpha$ if $i \le q$ and $x_i = 0$ otherwise. Finally, from (i) we derive that

$$y = \frac{x}{\|x\|_1}.$$

Thus we have shown that problem (78) and its dual problem can be solved in $O(n)$ time.

Example 4 Table 10 shows the outcome of our analysis for a randomly generated vector a. Because of (92) the optimal value of q is 6 in this example. Note that the sequence τ_k is increasing, until it becomes undefined due to $\omega_k < 0$.

In Table 10 one may observe that the vector ω shows behaviour that we recognize from the vector τ in preceding sections: (i) when ω_k is nonincreasing at some k it remains nonincreasing when k grows, and (ii) the optimal index q occurs at the moment when ω attains its maximal value. It turns out that this surprising observation can be turned into the next lemma. A consequence of this lemma is that the index q can be obtained without computing τ. One only needs to compute the first $q + 1$ entries of the vector ω.

Table 10 Numerical illustration for a problem of type $(\infty, 2)$

k	a_k	$f_k(a_k)$	ω_k	τ_k	x_k	$a_k - x_k$	y_k
1	2.9667	1.0000	1.0000	1.9667	0.6736	2.2932	0.3023
2	2.7888	0.9683	1.9683	2.1763	0.4957	2.2932	0.2224
3	2.6370	0.8683	2.8366	2.2361	0.3439	2.2932	0.1543
4	2.5963	0.8241	3.6607	2.2689	0.3031	2.2932	0.1360
5	2.5521	0.7629	4.4236	2.2876	0.2590	2.2932	0.1162
$6 = q$	2.4462	0.5415	4.9651	2.2932	0.1530	2.2932	0.0687
7	2.0900	−1.1531	3.8120	2.3035	0.0000	2.0900	0.0000
8	1.7484	−4.3254	−0.5134	–	0.0000	1.7484	0.0000
9	1.6817	−5.1398	−5.6532	–	0.0000	1.6817	0.0000
10	0.0000	−52.0241	−57.6773	–	0.0000	0.0000	0.0000

Lemma 8 *q is the first index such that*

$$\omega_q = \max_k \{\omega_k \ : \ 1 \leq k \leq n\}. \tag{93}$$

Proof We first derive from (92) that the index q satisfies

$$f_q(a_q) > 0 \geq f_q(a_{q+1}). \tag{94}$$

Recall that τ_q is the smallest roots of the quadratic equation $f_q(\xi) = 0$. For the moment, let τ'_q denote the second (i.e., largest) root. By its definition (90), $f_q(\xi)$ is concave. We therefore have

$$f_q(\xi) > 0 \quad \Leftrightarrow \quad \tau_q < \xi < \tau'_q, \tag{95}$$

where τ_q and τ'_q are such that

$$q\tau_q = \sigma_{1q} - \sqrt{\omega_q}$$
$$q\tau'_q = \sigma_{1q} + \sqrt{\omega_q}.$$

By (92), $a_{q+1} \leq \tau_q < a_q$. The first inequality makes clear that a_{q+1} does not belong to (τ_q, τ'_q). Therefore, we immediately get the second inequality in (94): $f_q(a_{q+1}) \leq 0$. According to (95), the first inequality in (94) holds if and only if $\tau_q < a_q < \tau'_q$. We already have $a_q > \tau_q$. So it remains to prove $a_q < \tau'_q$. Since the entries of a are ordered nonincreasingly, we have $qa_q \leq \sigma_{1q}$. Since $\sigma_{1q} < \sigma_{1q} + \sqrt{\omega_q} = q\tau'_q$, we obtain $a_q < \tau'_q$, as desired. Thus (94) has now been proven.

We proceed by showing that the sequence $f_k(a_k)$ is nonincreasing for $1 \leq k \leq n$. By the definition (90) of $f_k(\xi)$ we may write

$$f_k(\xi) = 1 - \sum_{i \le k} a_i^2 + 2\xi \sum_{i \le k} a_i - k\xi^2 = 1 - \sum_{i \le k} (a_i - \xi)^2. \tag{96}$$

Since $a_k \ge a_{k+1}$, we get $a_i - a_{k+1} \ge a_i - a_k$ for each i. So one also has

$$\sum_{i \le k+1} (a_i - a_{k+1})^2 = \sum_{i \le k} (a_i - a_{k+1})^2 \ge \sum_{i \le k} (a_i - a_k)^2 .$$

Thus it follows from (96) that, for any $k < n$,

$$f_{k+1}(a_{k+1}) = f_k(a_{k+1}) \le f_k(a_k). \tag{97}$$

This proves that the sequence $f_k(a_k)$ is nonincreasing when k increases. Because of this, (94) implies that $f_k(a_k)$ is positive if and only if $k \le q$. This has important consequences for the sequence ω_k, $1 \le k \le n$. This becomes clear by considering $\omega_{k+1} - \omega_k$. This expression can be reduced as follows.

$$\begin{aligned}
\omega_{k+1} - \omega_k &= \sigma_{1,k+1}^2 - (k+1)\left(\sigma_{2,k+1} - 1\right) - \left(\sigma_{1k}^2 - k\left(\sigma_{2k} - 1\right)\right) \\
&= \sigma_{1,k+1}^2 - (k+1)\sigma_{2,k+1} + (k+1) - \sigma_{1k}^2 + k\sigma_{2k} - k \\
&= 1 + \sigma_{1,k+1}^2 - \sigma_{1k}^2 - k\left(\sigma_{2,k+1} - \sigma_{2k}\right) - \sigma_{2,k+1} \\
&= 1 + \left(\sigma_{1,k+1} - \sigma_{1k}\right)\left(\sigma_{1,k+1} + \sigma_{1k}\right) - k\, a_{k+1}^2 - \sigma_{2,k+1} \\
&= 1 + a_{k+1}\left(a_{k+1} + 2\sigma_{1k}\right) - k\, a_{k+1}^2 - \sigma_{2,k+1} \\
&= 1 - \sigma_{2k} + 2a_{k+1}\sigma_{1k} - k a_{k+1}^2 \\
&= f_k(a_{k+1}) = f_{k+1}(a_{k+1}).
\end{aligned}$$

We may conclude that $\omega_{k+1} > \omega_k$ holds if and only if $f_{k+1}(a_{k+1}) > 0$. Since we have $f_k(a_k) \ge f_{k+1}(a_{k+1})$ for each k and because of (94) it follows that $f_{k+1}(a_{k+1}) > 0$ holds if and only if $k + 1 \le q$. So, when k runs from 1 to n then ω increases at k if and only if $k \le q - 1$, and from $k = q$ on ω is nonincreasing. Hence the lemma follows. □

3.9 *Problems of Type* (∞, ∞)

While assuming $\|a\|_\infty > 1$ we consider the problem

$$\min_x \{\|a - x\|_\infty \ : \ \|x\|_\infty \le 1\}. \tag{98}$$

The dual of this problem is

$$\max_y \left\{a^T y - \|y\|_1 \ : \ \|y\|_1 \le 1\right\}. \tag{99}$$

As in the previous eight sections, we assume $a \geq 0$, $x \geq 0$ and $y \geq 0$. The conditions for optimality are given by

$$\|x\|_\infty = 1 = \|y\|_1 \tag{100}$$
$$\|y\|_1 = y^T x \tag{101}$$
$$\|a - x\|_\infty = (a-x)^T y. \tag{102}$$

According to Lemma 3, (100) and (101) imply

(i) for each i: $y_i \neq 0$ implies $x_i = \|x\|_\infty$;

and, by Lemma 2, (100) and (102) imply

(ii) for each i: $y_i \neq 0$ implies $a_i - x_i = \|a - x\|_\infty$.

Let $y_i > 0$. Then (i) and (ii) imply $x_i = \|x\|_\infty$ and $a_i - x_i = \|a - x\|_\infty$. By adding these two equalities we obtain

$$a_i = \|x\|_\infty + \|a - x\|_\infty .$$

Hence, for any other $j \neq i$, since $x_j \leq \|x\|_\infty$ and $a_j - x_j \leq \|a - x\|_\infty$, we get

$$a_i \geq x_j + (a_j - x_j) = a_j .$$

Hence, since the entries in a are ordered nonincreasingly, $a_i = a_1$. So, y_i is zero for each i with $a_i < a_1$ and maybe also for one or more indices i with $a_i = a_1$. Therefore, if I denotes the set of indices with $y_i > 0$ and J its complement, then

$$I \subseteq \{i \ : \ a_i = a_1\}, \qquad J = \{i \ : \ i \notin I\} \supseteq \{i \ : \ a_i < a_1\},$$

with I nonempty, whereas $y_I > 0$ with $\|y_I\|_1 = 1$ and $y_J = 0$. The dual objective value at y equals $a^T y - \|y\|_1 = a_1 - 1$. Since the optimal primal objective value has the same value, this implies $\|a - x\|_\infty = a_1 - 1$. This value is positive, because $\|a\|_\infty = a_1 > 1$.

For x we are left with the following conditions. By (i), $x_i = 1$ for $i \in I$; then (ii) also holds because $a_i - x_i = a_1 - 1 = \|a - x\|_\infty$. For the remaining indices i ($i \in J$) there is a lot of freedom. The only condition for each $i \in J$ is that the value of x_i does not change the given values of $\|x\|_\infty$ $(= 1)$ and $\|a - x\|_\infty$ $(= a_1 - 1)$. So, with $\alpha = a_1 - 1$, we must have

$$0 \leq x_J \leq \mathbf{1}_J$$
$$-\alpha \mathbf{1}_J \leq a_J - x_J \leq \alpha \mathbf{1}_J .$$

Summarizing, a vector x is optimal for problem (98) if and only if

$$x_I = \mathbf{1}_I, \quad \max(0, a_J - \alpha \mathbf{1}_J) \leq x_J \leq \min(\mathbf{1}_J, a_J + \alpha \mathbf{1}_J). \tag{103}$$

Table 11 Two numerical solutions for a problem of type (∞, ∞)

i	a_i	$x_i^{(1)}$	$a_i - x_i^{(1)}$	$x_i^{(2)}$	$a_i - x_i^{(2)}$	y_i
1	1.9154	1.0000	0.9154	1.0000	0.9154	0.5354
2	1.9154	1.0000	0.9154	1.0000	0.9154	0.0000
3	1.9154	1.0000	0.9154	1.0000	0.9154	0.4646
4	1.2754	0.3600	0.9154	1.0000	0.2754	0
5	1.0543	0.1389	0.9154	1.0000	0.0543	0
6	1.0361	0.1207	0.9154	1.0000	0.0361	0
7	0.9148	0	0.9148	1.0000	−0.0852	0
8	0.8802	0	0.8802	1.0000	−0.1198	0
9	0.5620	0	0.5620	1.0000	−0.4380	0
10	0	0	0	0.9154	−0.9154	0

Example 5 For a randomly generated vector a, Table 11 shows two solutions, $x^{(1)}$ and $x^{(2)}$. In $x^{(1)}$ we took for each entry the smallest possible value, and in $x^{(2)}$ the largest possible value, according to (103). All other optimal vectors x are obtained by taking for each x_i a value between these two extreme values. One of these solutions is $x = a/\|a\|_\infty$, as mentioned in Table 1, and as easily can be verified. Another 'nice' solution is $x = \min(\mathbf{1}, a)$.

4 Concluding Remarks

This paper was inspired by a result in [2], where a nontrivial second-order cone optimization problem was solved analytically in linear time. This raised the question whether there exist other (classes of) problems that can be solved in linear time, by a variant of the same method. In this paper we consider a class of nine potentially important, easily stated and fundamental problems that form such a class. It is worth noting that in the four harder cases an important characteristic of the new method is that it first yields an 'optimal partition' of the variables in the problem. After this the values of the variables can be easily found. Though the problems considered in this paper are quite basic, hopefully it will inspire further research in this direction.

Figure 1 and Tables 5, 7, 9, 10, 11 were generated by using Matlab. The related Matlab files can be used to solve each of the problems that we considered in this paper for any vector a; they can be provided by writing to the second author.

Acknowledgements The authors want to express their thanks to an anonymous referee who carefully read the first draft of this paper. His critical remarks and questions were very stimulating and helpful during the preparation of the final version.

References

1. A. Ben-Tal and A. Nemirovski. *Lectures on modern convex optimization*. MPS/SIAM Series on Optimization. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2001.
2. Zhang Wei and Kees Roos. Using Nemirovski's Mirror-Prox method as Basic Procedure in Chubanov's method for solving homogeneous feasibility problems, 2019. Optimization Online.

Iteration Complexity of a Fixed-Stepsize SQP Method for Nonconvex Optimization with Convex Constraints

Francisco Facchinei, Vyacheslav Kungurtsev, Lorenzo Lampariello, and Gesualdo Scutari

Abstract We consider an SQP method for solving nonconvex optimization problems whose feasible set is convex and with an objective function that is the sum of a smooth nonconvex term and a nonsmooth, convex one. In the proposed method, at each iteration, a direction is generated by solving a strongly convex approximation to the original problem and then a fixed-stepsize is taken in that direction. The complexity result we establish is, as far as we are aware, the best available for the rather general setting we consider.

Keywords SQP methods · Iteration complexity · Fixed-stepsize

1 Introduction

We consider the nonconvex optimization problem with convex constraints

$$\begin{array}{lll} \underset{x}{\text{minimize}} & f(x) + q(x) & \\ \text{s.t.} & g(x) \leq 0 & \\ & Ax + b = 0 & \\ & x \in K, & \end{array} \tag{P}$$

F. Facchinei (✉)
Department of Computer, Control, and Management Engineering Antonio Ruberti, Sapienza University of Rome, Rome, Italy
e-mail: francisco.facchinei@uniroma1.it

V. Kungurtsev
Department of Computer Science, Faculty of Electrical Engineering, Czech Technical University in Prague, Prague, Czech Republic
e-mail: vyacheslav.kungurtsev@fel.cvut.cz

L. Lampariello
Department of Business Studies, Roma Tre University, Rome, Italy
e-mail: lorenzo.lampariello@uniroma3.it

G. Scutari
School of Industrial Engineering, Purdue University, West-Lafayette, IN, USA
e-mail: gscutari@purdue.edu

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*, Springer Proceedings in Mathematics & Statistics 354, https://doi.org/10.1007/978-3-030-72040-7_5

where

- $K \subseteq \mathbb{R}^n$ is a nonempty, convex, and compact set;
- $f : \mathbb{R}^n \to \mathbb{R}$ is $C^{1,1}$ (i.e., continuously differentiable with locally Lipschitz continuous gradients) on an open set containing K, with $L_{\nabla f}$ being the Lipschitz modulus of ∇f on K;
- $q : \mathbb{R}^n \to \mathbb{R}$ is convex on K and locally Lipschitz continuous on an open set containing K;
- $g_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \dots, m$, is convex on K and $C^{1,1}$ on an open set containing K, with $L_{\nabla g_i}$ being the Lipschitz modulus of ∇g_i on K;
- $A \in \mathbb{R}^{p \times n}$ and $b \in \mathbb{R}^p$;

and we denote the feasible set of (P) by

$$\mathcal{X} \triangleq \left\{ x \in \mathbb{R}^n \,:\, g(x) \le 0,\ Ax + b = 0,\ x \in K \right\}.$$

We assume that the feasible set $\mathcal{X}$ is nonempty. The algorithm we present is a fixed-stepsize method. More specifically, the search direction is iteratively computed by solving the strongly convex SQP-type approximating (at the base point x) subproblem

$$\begin{array}{ll} \underset{d}{\text{minimize}} & f(x) + \nabla f(x)^T d + \frac{1}{2} d^T H(x) d + q(x+d) \\ \text{s.t.} & g(x) + \nabla g(x)^T d \le 0 \\ & A(x+d) + b = 0, \\ & x + d \in K, \end{array} \tag{P$_x$}$$

where $\nabla g(x)$ is the transposed Jacobian of g evaluated at x and $H : \mathbb{R}^n \to \mathbb{R}^{n \times n}$ is assumed to be continuous over K with $H(x)$ a positive definite symmetric matrix for every $x \in K$ (for example, we could take $H(x) = I$ for all x). By compactness of K and continuity of H, there exist constants c and L_H such that

$$\min_{x \in K} \lambda_{\min}(H(x)) \ge c > 0 \quad \text{and} \quad \max_x \{ \|H(x)\| \mid x \in K \} \le L_H < \infty, \tag{1}$$

where $\lambda_{\min}(H(x))$ denotes the minimum eigenvalue of $H(x)$.

We denote by $\widetilde{\mathcal{X}}(x)$ and $d(x)$ the feasible set (which is shown to be nonempty in Proposition 1) and the unique solution of strongly convex problem (P$_x$), respectively.

Algorithm 1 below can be summarized as follows. Given the current iterate x^ν, a fixed step $\gamma > 0$ is taken along the search direction $d(x)$ so that

$$x^{\nu+1} = x^\nu + \gamma d(x^\nu). \tag{2}$$

Note that since x^0 satisfies the linear equality constraints in the original problem (P) (see Data in Algorithm 1), also all points x^ν generated by the algorithm are such that $Ax^\nu + b = 0$, and the equality constraints in (P$_x$) can therefore be rewritten as $Ad = 0$. In Sect. 2.3 we show that the stopping criterion used in step S.2 is sensible and closely related to the violation of the KKT conditions.

Algorithm 1: SQP method for nonconvex problems with convex constraints

```
Data: $\gamma \in (0, 1]$, $K \ni x^0 : Ax^0 + b = 0$, $\delta > 0$, $\nu \longleftarrow 0$;
repeat
(S.1)    compute the solution $d(x^\nu)$ of problem $(\mathrm{P}_{x^\nu})$;
(S.2)    if $\|d(x^\nu)\| \leq \delta$ then
             stop and return $x^\nu$;
         end
(S.3)    set $x^{\nu+1} = x^\nu + \gamma d(x^\nu)$, $\nu \longleftarrow \nu + 1$;
end
```

The setting of Problem (P) and the corresponding direction-defining subproblems have been analyzed in [3, Sect. 5]. This setting covers a wide array of problems arising, for example, in contemporary applications in statistical and machine learning, where typical examples for q are $q(x) = \|x\|_1$ or $q(x) = \|x\|_2^2$. In [3], convergence to KKT points is established for a diminishing stepsize method. We remark that the algorithm presented in [3] is, in turn, a simplification of a more general method first analyzed in [2]. In [2] some complexity results are obtained also for a fixed-stepsize policy using SQP-like subproblems; however, in the case of the setting described above and under the same assumptions relied upon in this paper, only some weaker *global convergence rates* are obtained there (see [2] for the distinction between the stronger notion of iteration complexity and global convergence rate). The main contribution of this paper is to show that Algorithm 1 exhibits an actual iteration complexity of $O(\delta^{-2})$ (recall that δ is the stopping tolerance used in step S.2 of the algorithm; see Theorem 2 for a clarification on the stopping criterion used in the algorithm). To prove this result, as main technical tool, we resort to the analysis of the decrease, at each iteration, of the classical penalty function

$$W(x; \varepsilon) \triangleq f(x) + q(x) + \frac{1}{\varepsilon}\theta(x),$$

with ε a positive penalty parameter, and

$$\theta(x) \triangleq \max_{i,j}\{g_i(x)_+,\ |a_j^T x + b_j|\},$$

where $\alpha_+ \triangleq \max\{0, \alpha\}$ and a_j^T denotes the jth row in matrix A. Note that, in the setting of this paper, W acts as a Lyapunov function for our algorithm; however we do not need to know, let alone use, an appropriate value for ε. For this and other reasons that are explained in [2], we refer to this penalty function as a *ghost* penalty function.

The paper is organized as follows. In the next section we discuss some preliminary issues: optimality conditions, measures of optimality and a suitable constraint qualification. The main complexity result is given in Sect. 3.

2 Preliminaries

2.1 Stationarity and Constraint Qualifications

We recall the KKT conditions for problem (P):

$$\begin{array}{c} 0 \in \nabla f(x) + \partial q(x) + \nabla g(x)\xi + A^T\pi + N_K(x) \\ 0 \leq \xi \perp g(x) \leq 0 \\ Ax + b = 0 \\ x \in K, \end{array} \tag{3}$$

where $\partial q(x)$ is the set of subgradients of q at x, $N_K(x)$ is the normal cone to K at x, $\perp$ denotes orthogonality, i.e., $a \perp b$ means $a^T b = 0$, and finally, $\xi \in \mathbb{R}^m$ and $\pi \in \mathbb{R}^p$ denote the multipliers for the inequality and equality constraints, respectively. In order to introduce our constraint qualification, it is expedient to define also the set of “abnormal” multipliers

$$M_0(x) \triangleq \Big\{(\xi, \pi) \mid \xi \in N_{\mathbb{R}^m_-}(g(x)),\ \pi \in \mathbb{R}^p,\ 0 \in \nabla g(x)\xi + A^T\pi + N_K(x)\Big\}.$$

If $\hat{x}$ is a local minimum point of (P), it is well-known that either $\hat{x}$ satisfies the KKT conditions, or $M_0(\hat{x}) \neq \{0\}$ (i.e., $\hat{x}$ is a Fritz-John point). The constraint qualification (CQ) we use is the Mangasarian-Fromovitz one that, in our setting including the geometric constraint $x \in K$, can be stated as follows.

Definition 1 The Mangasarian-Fromovitz Constraint Qualification (MFCQ) holds at $\hat{x} \in K$ if $M_0(\hat{x}) = \{0\}$.

Under the convexity assumption on the constraints, this condition is automatically satisfied at any infeasible point in K: if $\hat{x} \in K$ is not feasible, then $\hat{x}$ is easily seen not to be stationary for the violation-of-the-constraints measure θ on K, i.e.,

$$0 \notin \partial\theta(\hat{x}) + N_K(\hat{x}),$$

since θ is a convex function on K whose zero minimal value is attained only at feasible points. In turn, it is immediate to check that $0 \notin \partial\theta(\hat{x}) + N_K(\hat{x})$ is equivalent to $M_0(\hat{x}) = \{0\}$. On the other hand, by [4, Exercise 6.39], this condition is equivalent to the Slater-like CQ requiring a point $\bar{x} \in \operatorname{rel\,int} K$ to exist such that $g(\bar{x}) < 0$, $A\bar{x} + b = 0$ and $\{0\} = \{\pi \mid \pi^T A w = 0,\ \forall w \in T_K(x)\}$ for every feasible x, where $T_K(x)$ denotes the tangent cone to K at x. In more standard settings, if $K = \mathbb{R}^n$, $M_0(\hat{x}) = \{0\}$ reduces to the classical Mangasarian-Fromovitz condition, which in turn, whenever the equality constraints are not present, is equivalent to the standard Slater constraint qualification, i.e., to the existence of a point $\bar{x}$ such that $g(\bar{x}) < 0$.

In the light of these considerations, we make the following standing assumption for our analysis.

Assumption. *The MFCQ holds on $\mathcal{X}$.*

A direct consequence of the assumed MFCQ for (P) is that the MFCQ holds also for (P$_x$) for any $x \in K$ and that $\widetilde{\mathcal{X}}(x)$ is nonempty. The following proposition is proven in [3, Proposition 5.2].

Proposition 1 *For any $x \in K$ there exists $d \in \operatorname{rel\,int}(K - x)$ such that $g(x) + \nabla g(x)^T d < 0$, $A(x + d) + b = 0$ and $\{0\} = \{\pi \mid (A^T\pi)^T w = 0, \forall w \in T_{K-x}(v)\}$ for every $v \in \widetilde{\mathcal{X}}(x)$. A fortiori, the feasible set of (P$_x$) is nonempty.*

Of course, the nonemptiness of the feasible set of (P$_x$) in turn implies that this problem has a unique solution because the objective function is strongly convex.

2.2 Giving an Explicit Bound on the Multipliers

It is rather classical to show that, under the MFCQ, the set of KKT multipliers for the inequality constraints in subproblems (P$_x$) is bounded. Leveraging the convexity of the feasible set of (P) and the MFCQ, here we prove a stronger result, in that we can give an *explicit* uniform bound (over K) on the set of KKT multipliers for the inequality constraints in subproblems (P$_x$); in fact, we show that this bound can actually be expressed by means of problem-related constants that are conceptually known a priori.

Theorem 1 *The KKT multipliers ξ for the inequality constraints in subproblems (P$_x$) are bounded from above on K by a uniform constant M: for any multiplier ξ, and for all $x \in K$,*

$$\|\xi\|_1 \leq M \triangleq \frac{D(L_f + L_q + L_H D)}{|\sigma|}, \tag{4}$$

with $L_f \triangleq \max_{x \in K} \|\nabla f(x)\|$, $L_q \triangleq \max_{x,d}\{\|\rho\| \mid x + d \in K, \rho \in \partial q(x + d)\}$, D is the diameter of the set K, i.e., $D \triangleq \max_{v,\,w}\{\|v - w\|\| \mid v,\ w \in K\}$, L_H is defined in (1), and σ is a negative constant such that $0 > \sigma \geq g_i(\bar{x})$ for every i, where $\bar{x}$ is a Slater point for problem (P).

Proof First we bound the KKT multipliers for inequality constraints in a general convex problem under the Slater CQ; then, we specify the result for subproblems (P$_x$). Thus, consider the convex problem

$$\begin{aligned} \underset{d}{\text{minimize}}\ & \phi(d) \\ \text{s.t.}\quad & \beta(d) \leq 0 \\ & Ad + c = 0 \\ & d \in S, \end{aligned} \tag{5}$$

where

- $S \subseteq \mathbb{R}^n$ is a nonempty, convex, and compact set;
- $\phi : \mathbb{R}^n \to \mathbb{R}$ is convex on S and locally Lipschitz continuous on an open set containing S;
- $\beta_i : \mathbb{R}^n \to \mathbb{R}$, for every $i = 1, \dots, m$, are convex on S and continuously differentiable on an open set containing S;
- $c \in \mathbb{R}^p$.

Furthermore, the following problem-related constants are involved in the forthcoming analysis:

- D is the diameter of S, i.e., $D \triangleq \max_{v,\, w}\{\|v - w\| \mid v,\ w \in S\}$;
- L is an upper bound on the norm of the subgradients of ϕ on S: $L \triangleq \sup_{d,\omega}\{\|\omega\| \mid d \in S, \omega \in \partial\phi(d)\}$, which is finite because the set of subgradients is locally bounded.

Under the Slater CQ, let $d \in S$ be a KKT point for problem (5):

$$\omega + \nabla\beta(d)\xi + A^T\pi + \eta = 0, \tag{6}$$

$\beta(d) \le 0$, $Ad + c = 0$, for some $\omega \in \partial\phi(d)$, and KKT multipliers $\xi \in \mathbb{R}^m$, $\pi \in \mathbb{R}^p$ and $\eta \in N_S(d)$. Setting $I \triangleq \{i \in \{1, \dots, m\} : \beta_i(d) = 0\}$, we have, for some $\bar{d} \in \operatorname{rel\,int} S$, $A\bar{d} + c = 0$ and, for every $i \in I$,

$$0 > \sigma \ge \beta_i(\bar{d}) \ge \beta_i(d) + \nabla\beta_i(d)^T(\bar{d} - d) = \nabla\beta_i(d)^T(\bar{d} - d), \tag{7}$$

where the third inequality is due to the convexity of β_i. We can write

$$\begin{aligned} -DL &\le -\|\bar{d} - d\|\|\omega\| \le -(\bar{d} - d)^T\omega \le -(\bar{d} - d)^T\omega - (\bar{d} - d)^T\eta \\ &= (\bar{d} - d)^T\nabla\beta_I(d)\xi_I + (\bar{d} - d)^T A^T\pi \le \sigma\sum_{i\in I}\xi_i = \sigma\|\xi\|_1, \end{aligned}$$

where the third inequality is due to $\eta \in N_S(d)$, the first equality follows from (6), the second equality holds because $(\bar{d} - d)^T A = 0$ since both d and $\bar{d}$ satisfy the equality constraints, and the last inequality is a consequence of (7). Therefore, for any multiplier ξ satisfying (6),

$$\|\xi\|_1 \le \frac{DL}{|\sigma|}.$$

We now apply the previous general result to the convex subproblems (P$_x$). Since the MFCQ holds on $\mathcal{X}$, let $\bar{x} \in \mathcal{X}$ be a Slater point for (P): we observe that, for every $x \in K$,

$$0 > \sigma \ge g_i(\bar{x}) \ge g_i(x) + \nabla g_i(x)^T\bar{d}, \tag{8}$$

and $A(x + \bar{d}) + b = 0$, where $\bar{d} \triangleq \bar{x} - x$. We remark that σ provides a *uniform* bound in (8), not depending on x. Therefore, following the same line of reasoning as for the general case with

$$\phi(d) = \nabla f(x)^T d + \tfrac{1}{2} d^T H(x) d + q(x+d), \;\; \beta_i(d) = g_i(x) + \nabla g_i(x)^T d,$$
$$c = b + Ax,$$

and noting that

$$\begin{aligned} L &= \max_{x,d,\rho}\{\|\nabla f(x) + H(x)d + \rho\| \mid x \in K, x+d \in K, \rho \in \partial q(x+d)\} \\ &\le L_f + L_H D + L_q, \end{aligned}$$

we get the sought bound on the norm of the KKT multipliers ξ, not depending on x, for the subproblems (P$_x$) over K. □

2.3 Detecting Stationarity

In this section we first recall two results from [3] showing that $\|d(x)\|$ is a valid stationarity measure. Then, we give an actual quantitative relation between $\|d(x)\|$ and the violation of the KKT conditions. The following two results correspond to Propositions 5.3 and 5.4 in [3].

Proposition 2 *The function $d(\bullet)$ is continuous on K.*

Proposition 3 *We have $\|d(x)\| = 0$ at $x \in K$ if and only if x is a KKT point for (P).*

The results above indicate that $\|d(x)\|$, being a continuous measure that is 0 if and only if x is a KKT point for (P), can be employed in order to detect stationarity. In fact, the stopping criterion in Algorithm 1 requires $\|d(x)\|$ to be smaller than a prescribed accuracy δ. Even more, here we show that, apart from some problem-dependent constants, $\|d(x)\|$ provides a bound on the violation of the KKT conditions for (P). For the sake of completeness, we recall that the stationarity conditions (3) for problem (P) can be equivalently rewritten as

$$\begin{aligned} &\left\|\operatorname{prox}_{\frac{I_K+q}{1+\|(\xi,\pi)\|}}\left(x - \tfrac{\nabla f(x)+\nabla g(x)\xi + A^T\pi}{1+\|(\xi,\pi)\|}\right) - x\right\| = 0, \\ &\max_i \left|g_i(x)\tfrac{\xi_i}{1+\|(\xi,\pi)\|}\right| = 0, \\ &\max_i\{g_i(x)_+\} = 0, \end{aligned} \tag{9}$$

with $x \in K$ such that $Ax + b = 0$, and where I_K denotes the indicator function for K, $\xi \in \mathbb{R}^m_+$ and $\pi \in \mathbb{R}^p$, and prox is the proximal operator: $\operatorname{prox}_h(y) \triangleq \operatorname{argmin}_x(h(x) + \frac{1}{2}\|x-y\|_2^2)$.

Theorem 2 *The quantity $\|d(x)\|$, with $x \in K$, provides a bound on the violation of conditions (9):*

$$\begin{aligned} &\left\|\operatorname{prox}_{\frac{I_K+q}{1+\|(\xi,\pi)\|}}\left(x - \tfrac{\nabla f(x)+\nabla g(x)\xi + A^T\pi}{1+\|(\xi,\pi)\|}\right) - x\right\| \le (2+L_H)\|d(x)\|, \\ &\max_i |g_i(x)\tfrac{\xi_i}{1+\|(\xi,\pi)\|}| \le (\max_i L_{g_i})\|d(x)\|, \\ &\max_i\{g_i(x)_+\} \le (\max_i L_{g_i})\|d(x)\|, \end{aligned} \tag{10}$$

for some $\xi \in \mathbb{R}^m_+$, $\pi \in \mathbb{R}^p$, *where* $L_{g_i} \triangleq \max_{x \in K} \|\nabla g_i(x)\|$ *and* L_H *is defined in* (1).

Proof Recalling that $d(x)$ satisfies the KKT conditions for subproblem (P_x), we have

$$0 \in \nabla f(x) + H(x)d(x) + \partial q(x + d(x)) + \nabla g(x)\xi + A^T\pi + N_{K-x}(d(x)),$$

for some $\xi \in N_{\mathbb{R}^m_-}(g(x) + \nabla g(x)^T d(x))$ and $\pi \in \mathbb{R}^p$, which is equivalent to the following relation:

$$x + d(x) = \operatorname{prox}_{\frac{I_K + q}{1+\|(\xi,\pi)\|}}\left(x + d(x) - \frac{\nabla f(x) + H(x)d(x) + \nabla g(x)\xi + A^T\pi}{1 + \|(\xi,\pi)\|}\right). \tag{11}$$

In order to bound the measure of KKT conditions violation, focus first on the Lagrangian-related term in (9), i.e.,

$$\begin{aligned}
\left\|\operatorname{prox}_{\frac{I_K + q}{1+\|(\xi,\pi)\|}}\left(x - \frac{\nabla f(x) + \nabla g(x)\xi + A^T\pi}{1 + \|(\xi,\pi)\|}\right) - x\right\| &= \Bigg\| d(x) \\
&+ \operatorname{prox}_{\frac{I_K + q}{1+\|(\xi,\pi)\|}}\left(x - \frac{\nabla f(x) + \nabla g(x)\xi + A^T\pi}{1 + \|(\xi,\pi)\|}\right) \\
&- \operatorname{prox}_{\frac{I_K + q}{1+\|(\xi,\pi)\|}}\left(x + d(x) - \frac{\nabla f(x) + H(x)d(x) + \nabla g(x)\xi + A^T\pi}{1 + \|(\xi,\pi)\|}\right)\Bigg\| \\
&\le \|d(x)\| + \|-d(x) + H(x)d(x)\| \le (2 + L_H)\|d(x)\|,
\end{aligned}$$

where the first equality follows, in view of (11), by adding and subtracting $d(x)$ in the left-hand side term, and the inequality is due to the nonexpansive property of the proximal mapping.

As for the feasibility violation measure, since $g_i(x) \le -\nabla g_i(x)^T d(x)$, we have

$$\max_i\{g_i(x)_+\} \le (\max_i L_{g_i})\|d(x)\|.$$

Finally, concerning complementarity conditions, consider $\bar{\imath} \in \{1, \dots, m\}$ such that $|g_{\bar{\imath}}(x)\xi_{\bar{\imath}}| = \max_i |g_{\bar{\imath}}(x)_+\xi_{\bar{\imath}}||g_{\bar{\imath}}(x)\xi_{\bar{\imath}}|$ with $g_{\bar{\imath}}(x) + \nabla g_{\bar{\imath}}(x)^T d(x) = 0$, otherwise $\xi_{\bar{\imath}} = 0$. Hence,

$$\max_i |g_i(x)\xi_i| \le (\max_i L_{g_i})\|d(x)\|(1 + \|(\xi,\pi)\|).$$

□

Theorem 2 justifies the choice of condition $\|d(x)\| \le \delta$ as stopping criterion in step S.3 of Algorithm 1: if $\|d(x)\| \le \delta$, then the KKT conditions, rewritten as in (9), are satisfied at x within an accuracy δ, up to some given, problem-dependent constants.

3 Complexity Analysis

We are finally ready to study the iteration complexity of Algorithm 1: we give an upper bound, that depends only on some problem-dependent constants, on the maximal number of iterations needed to satisfy the stopping criterion $\|d(x^\nu)\| \leq \delta$ at step S.2, and, thus, to reach a point satisfying the KKT conditions with an accuracy δ, up to some given, problem-dependent constants. In the following theorem, we establish that, if the stepsize γ is sufficiently small, this can be done in at most $O(\delta^{-2})$ iterations. For reader's convenience, we recall that, in Theorem 3, M is the constant defined in Theorem 1.

Theorem 3 *In the setting described in the introduction and supposing the MFCQ assumption holds, consider the sequence $\{x^\nu\}$ generated by Algorithm 1 with a positive stepsize $\gamma \leq \min\left\{\frac{c}{L_{\nabla f}+M\max_i L_{\nabla g_i}},\ 1\right\}$ and fix an accuracy $\delta \leq 1$, with c defined in (1). Then, Algorithm 1 stops at step S.2 in at most*

$$\left\lceil \frac{2}{\delta^2}\frac{f(x^0)+q(x^0)+\frac{1}{M}\max_i\{g_i(x^0)_+\}-(f^m+q^m)}{c\gamma}\right\rceil \tag{12}$$

iterations, where $f^m+q^m \triangleq \min_x\{f(x)+q(x)|x\in K\}$.

Proof Since the starting point x^0 belongs to the convex set K, the stepsize satisfies $\gamma \leq 1$ by construction and, by the last constraint in (P_{x^ν}), $x^\nu + d(x^\nu) \in K$ for all ν, it is easily seen that all points x^ν generated by the algorithm belong to K. As a consequence, the sequence $\{x^\nu\}$ is bounded. Also, since x^0 is such that $Ax^0+b=0$, we have $Ad(x^\nu)=0$ and $Ax^\nu+b=0$, for every ν. Preliminarily, observe that, at each step, in view of Proposition 1, multipliers $\{\xi^\nu\}$ and $\{\pi^\nu\}$ exist with $\xi^\nu \in N_{\mathbb{R}^m_-}(g(x^\nu)+\nabla g(x^\nu)^T d(x^\nu))$ such that

$$0\in\nabla f(x^\nu)+H(x^\nu)d(x^\nu)+\partial q(x^\nu+d(x^\nu))+\nabla g(x^\nu)\xi^\nu+A^T\pi^\nu+N_{K-x^\nu}(d(x^\nu)). \tag{13}$$

We also recall that, by (1), $H(x)$ satisfies

$$d(x^\nu)^T H(x^\nu)d(x^\nu) \geq c\|d(x^\nu)\|^2. \tag{14}$$

Also, by the convexity of q, for every $\rho^\nu \in \partial q(x^\nu+d(x^\nu))$,

$$\rho^{\nu T} d(x^\nu) \geq q(x^\nu+d(x^\nu))-q(x^\nu). \tag{15}$$

Moreover, since $d(x^\nu)$ is feasible for (P_{x^ν}), on the one hand,

$$-\xi_i^\nu\nabla g_i(x^\nu)^T d(x^\nu)=\xi_i^\nu g_i(x^\nu), \tag{16}$$

due to $\xi^\nu \in N_{\mathbb{R}^m_-}(g(x^\nu)+\nabla g(x^\nu)^T d(x^\nu))$. Therefore, by (13)–(16), we have for some $\rho^\nu \in \partial q(x^\nu+d(x^\nu))$ and $\zeta^\nu \in N_{(K-x^\nu)}(d(x^\nu))$,

$$\begin{aligned}
&\nabla f(x^\nu)^T d(x^\nu) + c\|d(x^\nu)\|^2 + q(x^\nu + d(x^\nu)) - q(x^\nu) \\
&\quad \le \nabla f(x^\nu)^T d(x^\nu) + d(x^\nu)^T H(x^\nu) d(x^\nu) + \rho^{\nu T} d(x^\nu) \\
&\quad = -\xi^{\nu T} \nabla g(x^\nu)^T d(x^\nu) - \pi^{\nu T} A d(x^\nu) - \zeta^{\nu T} d(x^\nu) \le \xi^{\nu T} g(x^\nu) \\
&\quad \le \xi^{\nu T} [\max_i \{g_i(x^\nu)_+\}] e \le \theta(x^\nu) \|\xi^\nu\|_1,
\end{aligned}$$

where the second inequality is due to $Ad(x^\nu) = 0$ and $0 \in \beta\mathbb{B}^n_\infty \cap (K - x^\nu)$. Hence

$$\nabla f(x^\nu)^T d(x^\nu) \le -c\|d(x^\nu)\|^2 + \theta(x^\nu)\, \|\xi^\nu\|_1 + q(x^\nu) - q(x^\nu + d(x^\nu)). \tag{17}$$

Let us now consider the nonsmooth (ghost) penalty function already described in the introduction:

$$W(x; \varepsilon) = f(x) + q(x) + \frac{1}{\varepsilon} \max_{i,j} \{g_i(x)_+, |a_j^T x + b_j|\}.$$

W is used as a Lyapunov function (but does not appear anywhere in the algorithm itself).

Recalling $a_j^T x^\nu + b_j = 0$ and $a_j^T d(x^\nu) = 0$ for every ν, we have

$$\begin{aligned}
&W(x^{\nu+1}; \varepsilon) - W(x^\nu; \varepsilon) = f(x^\nu + \gamma d(x^\nu)) - f(x^\nu) + q(x^\nu + \gamma d(x^\nu)) - q(x^\nu) \\
&\quad + \frac{1}{\varepsilon}\Big[\max_i \{g_i(x^\nu + \gamma d(x^\nu))_+\} - \max_i \{g_i(x^\nu)_+\}\Big] \\
&\overset{(a)}{\le} \gamma \nabla f(x^\nu)^T d(x^\nu) + \frac{(\gamma)^2 L_{\nabla f}}{2} \|d(x^\nu)\|^2 + q(x^\nu + \gamma d(x^\nu)) - q(x^\nu) \\
&\quad + \frac{1}{\varepsilon}\Big[\max_i \{(g_i(x^\nu) + \gamma \nabla g_i(x^\nu)^T d(x^\nu))_+\} - \max_{i,j} \{g_i(x^\nu)_+\} \\
&\quad + \frac{(\gamma)^2 \max_i \{L_{\nabla g_i}\}}{2} \|d(x^\nu)\|^2\Big] \\
&\overset{(b)}{\le} \gamma \nabla f(x^\nu)^T d(x^\nu) + \frac{(\gamma)^2}{2}(L_{\nabla f} + \frac{\max_i \{L_{\nabla g_i}\}}{\varepsilon}) \|d(x^\nu)\|^2 \\
&\quad + q(x^\nu + \gamma d(x^\nu)) - q(x^\nu) + \frac{1}{\varepsilon}\Big[\max_i \{(1-\gamma) g_i(x^\nu)_+\} - \max_i \{g_i(x^\nu)_+\}\Big] \\
&= \gamma \nabla f(x^\nu)^T d(x^\nu) + \frac{(\gamma)^2}{2}(L_{\nabla f} + \frac{\max_i \{L_{\nabla g_i}\}}{\varepsilon}) \|d(x^\nu)\|^2 \\
&\quad + q(x^\nu + \gamma d(x^\nu)) - q(x^\nu) - \frac{\gamma}{\varepsilon}\Big[\max_i \{g_i(x^\nu)_+\}\Big] \\
&= \gamma \nabla f(x^\nu)^T d(x^\nu) + \frac{(\gamma)^2}{2}(L_{\nabla f} + \frac{\max_i \{L_{\nabla g_i}\}}{\varepsilon}) \|d(x^\nu)\|^2 - \frac{\gamma}{\varepsilon}\theta(x^\nu) \\
&\quad + q(x^\nu + \gamma d(x^\nu)) - q(x^\nu),
\end{aligned} \tag{18}$$

where (a) follows from the descent lemma applied to f and g_i, for every $i = 1, \ldots, m$; and (b) holds for any positive $\gamma \le 1$, since $\nabla g_i(x^\nu)^T d(x^\nu) \le -g_i(x^\nu)$. Furthermore, we observe that

$$\gamma[\nabla f(x^\nu)^T d(x^\nu) - \frac{1}{\varepsilon}\theta(x^\nu)] + q(x^\nu + \gamma d(x^\nu)) - q(x^\nu) \le \gamma[-c\|d(x^\nu)\|^2$$
$$+ (\|\xi^\nu\|_1 - \frac{1}{\varepsilon})\,\theta(x^\nu)] + \gamma[q(x^\nu) - q(x^\nu + d(x^\nu))] + q(x^\nu + \gamma d(x^\nu)) - q(x^\nu)$$
$$\le \gamma[-c\|d(x^\nu)\|^2 + (\|\xi^\nu\|_1 - \frac{1}{\varepsilon})\theta(x^\nu)], \tag{19}$$

where the first inequality is due to (17) and the second relation is a consequence of the convexity of q. Since, in view of Theorem 1, for any multiplier ξ^ν, $\|\xi^\nu\| \le M$, where M is defined in (4), we obtain from (19), for every ν,

$$\nabla f(x^\nu)^T d(x^\nu) - \frac{1}{\varepsilon}\theta(x^\nu) + \frac{1}{\gamma}[q(x^\nu + \gamma d(x^\nu)) - q(x^\nu)] \le -c\|d(x^\nu)\|^2, \quad \forall \varepsilon \le \frac{1}{M}. \tag{20}$$

Combining (18) and (20), we get

$$\begin{aligned} W(x^{\nu+1}; \tfrac{1}{M}) - W(x^\nu; \tfrac{1}{M}) &\le -\gamma c\|d(x^\nu)\|^2 + \tfrac{(\gamma)^2}{2}(L_{\nabla f} + \tfrac{\max_i\{L_{\nabla g_i}\}}{\frac{1}{M}})\|d(x^\nu)\|^2 \\ &= -\gamma\left[c - \tfrac{\gamma}{2}(L_{\nabla f} + M\max_i\{L_{\nabla g_i}\})\right]\|d(x^\nu)\|^2. \end{aligned}$$

Thus,

$$W(x^{\nu+1}; \frac{1}{M}) - W(x^\nu; \frac{1}{M}) \le -\frac{c}{2}\gamma\|d(x^\nu)\|^2. \tag{21}$$

Suppose now that the procedure did not stop until iteration N, i.e., $\|d(x^\nu)\| > \delta$ for all iterates up to $N-1$. We get from (21)

$$\delta^2 N < \sum_{\nu=0}^{N-1}\|d(x^\nu)\|^2 \le 2\frac{f(x^0) + q(x^0) + \frac{1}{M}\max_i\{g_i(x^0)_+\} - (f^m + q^m)}{c\gamma},$$

Thus the algorithm stops in at most a number of iterations equal to (12), giving a complexity of $\mathcal{O}(\delta^{-2})$. □

Theorem 3, together with the boundedness of K and Propositions 2 and 3, easily yields the following asymptotic convergence result for Algorithm 1.

Corollary 1 *Under the assumptions of Theorem 3, if we set $\delta = 0$ at step S.2 of Algorithm 1, the sequence generated by Algorithm 1 is bounded and each of its limit points is a KKT solution.*

Remark 1 In the introduction we mentioned that Algorithm 1 is a simplified version of the algorithm analyzed in [2]. The analysis in this paper can be extended to the algorithm considered in [2] under the assumptions we make here.

Remark 2 The bound of order $O(\delta^{-2})$ given in Theorem 3 is the best available for the class of problems we considered here when only first-order information is used, see [1] and references therein.

4 Conclusions

We have presented the first complexity analysis for an SQP-type method for nonconvex optimization problems with convex constraints and an objective function that is the sum of a smooth term and of a (possibly nonsmooth) convex one. These results are of obvious theoretical interest and hopefully will pave the way for the complexity analysis of more standard line-search-type SQP methods. Furthermore, while we do not expect a fixed-stepsize method to be practically more efficient than a traditional line-search one in general, a fixed-stepsize strategy could still be of practical interest, for example, whenever the computation of the objective function is very expensive.

Acknowledgements The authors are very grateful to a Referee whose very accurate and insightful comments helped improve the paper. Francisco Facchinei was partially supported by Progetto di Ateneo Distributed optimization algorithms for Big Data. Vyacheslav Kungurtsev was supported by the OP VVV project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics". Lorenzo Lampariello was partially supported by MIUR PRIN 2017 grant 20177WC4KE. Gesualdo Scutari was partially supported by NSF Grants CIF 1564044, CIF 1719205, and CMMI 1832688; and ARO Grant W911NF1810238.

References

1. Cartis, C., Gould, N. I., Toint, P. L. (2018) Second-order optimality and beyond: Characterization and evaluation complexity in convexly constrained nonlinear optimization. *Foundations of Computational Mathematics,* 18(5), 1073-1107.
2. Facchinei, F., Kungurtsev, V., Lampariello, L., Scutari, G. (2021) Ghost penalties in nonconvex constrained optimization: Diminishing stepsizes and iteration complexity. *Mathematics of Operations Research.* Available on-line at https://doi.org/10.1287/moor.2020.1079
3. Facchinei, F., Kungurtsev, V., Lampariello, L., Scutari, G. (2020) Diminishing stepsize methods for nonconvex composite problems via ghost penalties: from the general to the convex regular constrained case. *Optimization Methods and Software*. Available on-line at https://doi.org/10.1080/10556788.2020.1854253
4. Rockafellar, R. T., Wets, R. J. B. (2009) *Variational Analysis.* Springer Science & Business Media.

Modelling and Inferring the Triggering Function in a Self-Exciting Point Process

Craig Gilmour and Desmond J. Higham

Abstract Self-exciting spatio-temporal point processes offer a flexible class of models that have found success in a range of applications. They involve a triggering effect that accounts for the clustering patterns observed in many natural and sociological applications. In this work, we focus on the key step of inferring or designing the form of the triggering function. In the inference setting, we use a nonparametric approach to fit a process to a range of datasets arising in criminology. By analysing this public domain data we find that the inferred trigger shape varies across different categories of crime. Motivated by these observations, and also by hypotheses from the criminology literature, we then propose a variation on the classical Epidemic-Type Aftershock Sequences trigger, which we call the Delayed Response trigger. After calibrating both parametric models, we show that Delayed Response is comparable with Epidemic-Type Aftershock Sequences in terms of predicting future events, and additionally provides an estimate of the time lag before the risk of triggering is maximised.

Keywords Inference · Trigger · Criminology

1 Introduction

Self-exciting point processes allow the occurrence of an event to increase the probability of a subsequent event, close by in space and time. Such follow-on events are said to have been "triggered" by that previous event. Originally used to model earthquakes, in recent years this class of models has been shown to be relevant in the context of social interactions, disease propagation and crime [21, 24].

In this work, we focus on the key problem of how to specify a suitable form for the trigger function. We make four main contributions.

C. Gilmour
Department of Mathematics and Statistics, University of Strathclyde, Glasgow G1 1XH, UK

D. J. Higham (✉)
School of Mathematics, University of Edinburgh, Edinburgh EH9 3FD, UK
e-mail: d.j.higham@ed.ac.uk

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*, Springer Proceedings in Mathematics & Statistics 354, https://doi.org/10.1007/978-3-030-72040-7_6

1. To infer the trigger function directly, and nonparametrically, from the data in the case of public domain data sets recording acts of crime. This gives, for the first time, a quantified indication of how the overall shape of the triggering depends on the type of crime involved.
2. For each crime type, to display both the temporal and spatial pattern of the triggering effect.
3. To propose a new parametrised form for the trigger, which we call Delayed Response, differing from the widely-used Epidemic-Type Aftershock Sequences version, motivated by the inference results and by empirical observations from the criminology literature.
4. To show that the new Delayed Response version is comparable with the Epidemic-Type Aftershock Sequences model in terms of accuracy in predicting future events.

The manuscript is organized as follows. Section 2 summarizes the ideas behind self-exciting point processes. In Sect. 3 we infer the shape of the triggering function from real data sets. Section 4 looks at parametrised trigger models and introduces the Delayed Response trigger. In Sect. 5 the trigger functions are fitted to real data and compared. Conclusions are given in Sect. 6.

2 Background

A spatial-temporal point process, $N(\cdot, \cdot, \cdot)$, is a random measure on a region of $\mathbb{R} \times \mathbb{R} \times \mathbb{R}$ taking non-negative integer values [24]. Let x and y denote spatial coordinates in two dimensions, with t denoting time. The process can then be characterized by its conditional intensity function

$$\lambda(x, y, t) = \lim_{\Delta x, \Delta y, \Delta t \to 0} \frac{\mathbb{E}(N(x + \Delta x, y + \Delta y, t + \Delta t) - N(x, y, t))}{\Delta x \Delta y \Delta t}. \qquad (1)$$

Here, $N(x + \Delta x, y + \Delta y, t + \Delta t) - N(x, y, t)$ records how many events have occurred with the first spatial coordinate in $(x, x + \Delta x)$, the second in $(y, y + \Delta y)$ and time in $(t, t + \Delta t)$.

In our context, events are recorded acts of crime, and the notation $\{t_i, x_i, y_i\}$ will be used for an event that took place at time t_i in location (x_i, y_i). Here, it has been proposed that the conditional intensity function should take the Epidemic-Type Aftershock Sequences (ETAS) form [19]

$$\lambda(x, y, t) = \mu(x, y, t) + \sum_{t > t_i} g(t - t_i, x - x_i, y - y_i). \qquad (2)$$

In (2), μ represents the *background rate*, and g is the *triggering function* that quantifies how an event creates a "knock-on" effect of increased likelihood of further events nearby in space and future time. This type of self-exciting model is often referred to

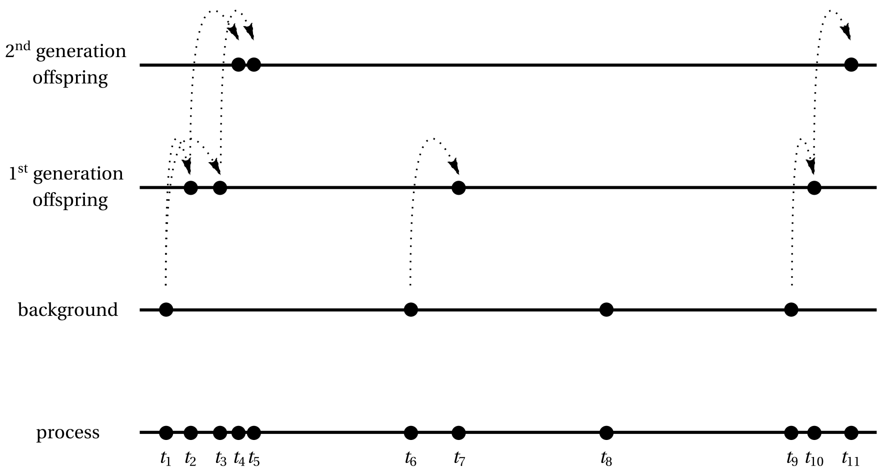

2nd generation offspring
1st generation offspring
background
process
t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9 t_{10} t_{11}

Fig. 1 Illustration of a Hawkes process. Here we have suppressed the spatial components, and the horizontal axis represents time. We show first and second generation effects. A background event can trigger a *first-generation offspring* event, which in turn can trigger a *second-generation offspring* event, and so on. In Hawkes processes we typically see a stronger clustering of events than we do for a simple Poisson process. In practice, only the event times (bottom row of picture) $t_1, t_2, t_3, \ldots, t_{11}$ would be available, and recovering the background/triggering narrative is an inference problem

as a Hawkes process. Specifying the form of the trigger g in (2) is a key modelling step. Some authors have proposed specific, parametrized forms; notably Gaussian in space and exponentially decaying in time[1] [18, 24]

$$g(t, x, y) = f(t)h(x, y) = \alpha\omega e^{-\omega t} \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}. \tag{3}$$

In Sect. 4 we discuss the form of (3), and its motivation in the criminology setting.

A Hawkes process can be viewed as a branching process [14], which is helpful from the perspective of simulation. Events occur according to the background rate μ. An event which happens at time t_i produces offspring events at a temporal rate proportional to $f(t - t_i)$ for times $t > t_i$ independently of each other. The direct offspring of the background events are known as *first-generation offspring*. These individual events can then trigger further events independently of each other, which are known as *second-generation offspring*, and so on. A simple overview of such a process is shown in Fig. 1.

[1] In some literature the temporal aspect of this triggering function is written in the equivalent form $\alpha e^{-\beta t}$, with the interpretation that each arrival increases the intensity by α, with this influence decaying at rate β [14]. We use the form $\alpha\omega e^{-\omega t}$ because it has the useful property that each event is expected to trigger α events.

3 Non-parametric Estimation

Marsan and Lengliné proposed a method for matching a Hawkes processes to earthquake data without making any prior parametric assumptions [16]. The method, named Model Independent Stochastic Declustering, is similar to expectation-maximization (EM), but uses a probability weighted histogram when estimating the triggering function and background rate. Essentially, this approach involves discretising the background observation window into cells and/or discretising the triggering components, and building probability weighted histograms in the maximization step based on the sum of the probabilities from the expectation step which fall in each discretised region.

In this section, we use the method to infer the triggering function for various categories of crime data from [2]. This allows us to (a) judge whether the ETAS-style trigger (3) is appropriate, and (b) look for different triggering patterns between the crime types. Full details of the implementation can be found in [10].

We applied the method to public domain data concerning five different crime types in Chicago: homicide, motor vehicle theft, theft with a reported value of over 500 dollars, aggravated assault, and burglary. The homicide data consists of 4465 events which took place between 2005 and 2014, the motor vehicle theft data consists of 9860 events in 2014, the data for theft of a value over 500 dollars consists of 14909 events in 2014, the aggravated assault data consists of 13251 events which took place from 2012 until the end of 2014, and the burglary data consists of 13071 events in 2014.

We focus here on two aspects, both of which are of interest from a criminology perspective: namely, the temporal and spatial shape of the triggering effect.

Figure 2 shows the temporal decay of the time lag between an event and a triggered follow-on event, in units of days. We see that each plot produces a profile that decays over time with an exponential-type behaviour. However, there is a significant difference in the rates. The half-life is around 10 days for motor vehicle theft and burglary, around 20 days for theft over 500 dollars, and around 40 days for aggravated assault. Although homicide produces a qualitatively similar picture, the the decay rate is much larger, with a half-life of around 400 days.

Figure 3 concerns the spatial aspect of triggering. In each plot, with the triggering event located at the origin, the markers indicate the location of a follow-on triggered event. Here, the horizontal and vertical axes represent east-west and north-south directions, with distance measured in kilometers. We note that streets in Chicago typically follow an east-west and north-south orientation, with block sizes of around 100–200 m [1]. We see that theft over 500 dollars, aggravated assault and burglary show spatial concentration—a triggered event is typically within one or two blocks of the parent event. For homicide and motor vehicle theft there is less evidence of spatial concentration.

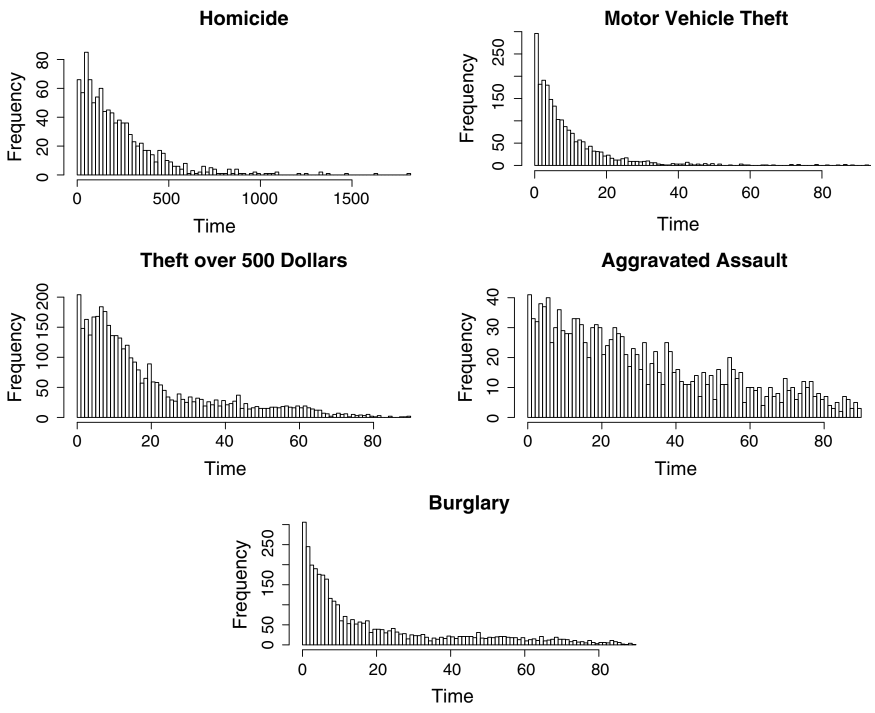


Fig. 2 Histograms show the time, in days, after an initial event when crimes are assumed to have been triggered for the non-parametric point process fitted to five different crime types

4 Parametric Trigger Models

Several authors have motivated the use of self-exciting point processes in the context of crime by observing that certain categories, including burglary and gang violence, produce highly clustered event sequences that can be accounted for by a triggering mechanism. For example, a gang shooting may lead to retaliatory acts of violence against rival gangs, and burglars often target houses which have recently been burgled and nearby houses [19]. Hence, self-exciting point processes have been used to model crime in a number of studies, including to model burglary and other crimes in Kent [20], Los Angeles [19] and Pittsburgh [25], to model gang rivalries in Los Angeles [9], to predict gun crime and homicides in Chicago [18], to model the use of improvised explosive devices during “The Troubles” [26], and to model civilian deaths in Iraq [15].

The parametrized triggering function in (3) captures the broad requirement that likelihood decays in time and space, and is generally consistent with the inferred profiles in Figs. 2 and 3. The function is also sufficiently simple that the model parameters have natural interpretations and can be inferred from data. Of course, isotropic trigger models, where the triggering effect depends on distance but not direction, do

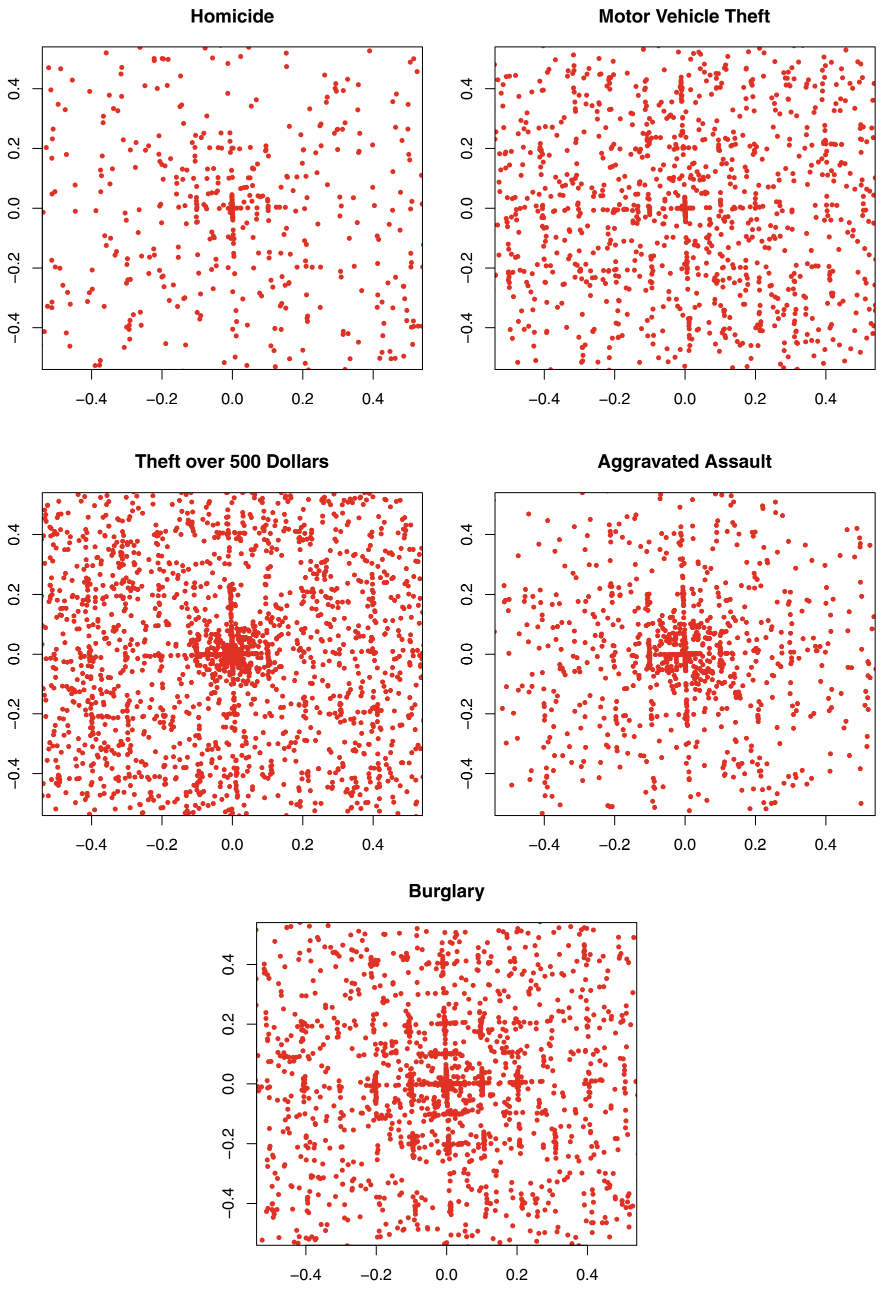


Fig. 3 Locations of assumed triggered events in relation to the event that triggered them. Euclidean distance is measured in kilometers and the direction north is vertical

not account for geographical heterogeneities (lakes, hills, one-way systems, types of urban zone), and hence more detailed information could be incorporated at the expense of model complexity. Another unnatural feature of the model (3) is that the trigger takes its largest value, temporally, immediately after the event has taken place: we have $f(0) = \alpha\omega$ and exponential decay thereafter. To remedy this potential drawback, we propose the following alternative trigger, which we will refer to as the delayed response (DR) model:

$$f(t)h(x, y) = \bar{\alpha}\bar{\omega}^2 t e^{-\bar{\omega}t} \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}. \tag{4}$$

As t varies, this function has maximum intensity at time $t = 1/\bar{\omega}$, with $\bar{\alpha}$ giving the expected number of events triggered.

We argue that the DR trigger function (4) is more readily justified than the ETAS version (3), on the grounds that the risk of follow-on crime increases over a certain period following the initial crime, before then starting to decay. For example, it has been reported in the criminology literature that perpetrators of homicide often have a ‘cooling-off’ period between victims [8]. In terms of burglary, several theories have been posited to suggest that repeat victimisation may be more likely to occur after a period of time has passed from the initial incident, including the notion that a burglar will want to wait until the previous victims have replaced their stolen belongings, or that information about a ‘successful’ crime being committed would take time to pass between criminals [22].

Figure 4 shows how the two triggering functions (3) and (4) compare.

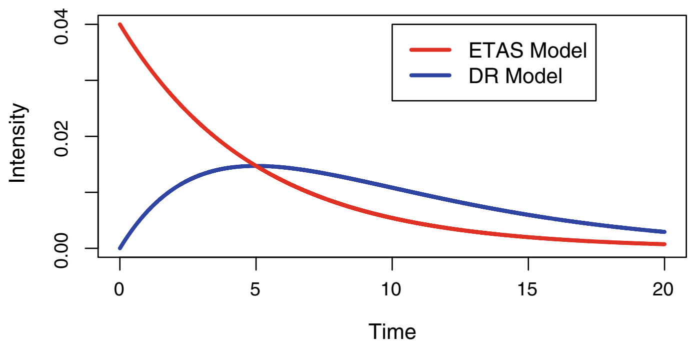


Fig. 4 Example of the ETAS model (3) in blue, with $\alpha = 0.2$ and $\omega = 0.2$, and the DR model (4) in red, with $\bar{\alpha} = 0.2$ and $\bar{\omega} = 0.2$

5 Parametrised Trigger on Real Data

In this section we fit the two models (3) and (4) to real data and compare their predictive power.

5.1 EM-Algorithm

Following work in this context by [18, 20, 27, 29], the models are fitted using an EM approach, We note that local convergence of the algorithm has been studied [7]. Because the underlying problem is not convex, the algorithm suffers from a drawback that is shared by many numerical methods, namely the tendency to converge to local maxima [28].

Full details of our approach may be found in [10]. We estimate the background rate $\mu(x, y, t)$ through kernel density estimation. The EM-algorithm begins with an initial guess $(\alpha^{(0)}, \mu^{(0)}, \omega^{(0)})$. A lower triangular matrix of probabilities, $P^{(0)}$, is also initialized, with p_{ii} representing the probability that event i was a background event, and $p_{ij}^{(k+1)}$ representing the probability that it was caused by event j for $i > j$. The algorithm then iterates between expectation and maximisation steps [11].

The expectation step at the $(k + 1)$th iteration for a self-exciting point process with triggering function (3) takes the form

$$p_{ii}^{(k+1)} = \frac{\mu^{(k)}(t_i, x_i, y_i)}{\mu^{(k)}(t_i, x_i, y_i) + \sum_{j=1}^{i-1} g(t_i - t_j, x_i - x_j, y_i - y_j)}, \tag{5}$$

$$p_{ij}^{(k+1)} = \frac{g(t_i - t_j; \alpha^{(k)}, \omega^{(k)})}{\mu^{(k)}(t_j, x_j, y_j) + \sum_{j=1}^{i-1} g(t_i - t_j, x_i - x_j, y_i - y_j)}. \tag{6}$$

The corresponding maximization step is then

$$\alpha^{(k+1)} = \frac{\sum_{i=1}^{n} \sum_{j=1}^{i-1} p_{ij}^{(k+1)}}{n}, \tag{7}$$

$$\omega^{(k+1)} = \frac{\sum_{i=1}^{n} \sum_{j=1}^{i-1} p_{ij}^{(k+1)}}{\sum_{i=1}^{n} \sum_{j=1}^{i-1} p_{ij}^{(k+1)}(t_i - t_j)}, \tag{8}$$

$$\sigma^{2(k+1)} = \frac{\sum_{i=1}^{n} \sum_{j=1}^{i-1} p_{ij}((x_i - x_j)^2 + (y_i - y_j)^2)}{\sum_{i=1}^{n} \sum_{j=1}^{i-1} 2p_{ij}(t_i - t_j)}. \tag{9}$$

With the DR model (4), the steps in the EM-algorithm are the same, with the exception that (8) changes to

$$\bar{\omega}^{(k+1)} = \frac{\sum_{i=1}^{n} \sum_{j=1}^{i-1} 2 p_{ij}^{(k+1)}}{\sum_{i=1}^{n} \sum_{j=1}^{i-1} p_{i,j}^{(k+1)} (t_i - t_j)}. \tag{10}$$

5.2 Results on Real Data

We fitted the ETAS and DR models with Gaussian spatial components to the five different Chicago crime data sets mentioned in Sect. 3. When fitting the models we assumed that no crime was triggered more than a kilometer from the original event or after 90 days of an event occurring, with the exception of the homicide data where we did not limit the length of time over which an event could trigger another. These results are summarised in Tables 1 and 2. The corresponding temporal factors in the trigger functions are displayed on the left of Figs. 5 and 6.

5.3 Prediction Results

Having fitted both models to the crime data, we now test their performance in terms of predicting future behaviour. To do this, following the type of approach used in [18, 19], we separated Chicago into 75 $\times$ 75 m gridsquares and used hourly time points. Having fitted the two models to data from previous time points, at each new

Table 1 Parameters found when fitting the ETAS model with Gaussian spatial components, (3), to the five selected different crime types

Crime type	α	ω	σ
Homicide	0.1782	0.004	0.3681
Motor vehicle theft	0.2432	0.1688	0.2897
Theft over 500 dollars	0.2507	0.1610	0.2868
Aggravated assault	0.1756	0.1205	0.2528
Burglary	0.2442	0.1692	0.2369

Table 2 Parameters found when fitting the DR model with Gaussian spatial components, (4), to the five selected different crime types

Crime type	$\bar{\alpha}$	$\bar{\omega}$	σ
Homicide	0.2438	0.006	0.3775
Motor vehicle theft	0.3661	0.2023	0.3182
Theft over 500 dollars	0.3702	0.2093	0.3206
Aggravated assault	0.1915	0.1063	0.3150
Burglary	0.3431	0.2764	0.2771

hourly interval we applied each model. We then ranked the gridsquares in terms of the predicted intensity. This quantifies the relative likelihood of future events occurring in each gridsquare, according to each model. To judge the result, we then checked what proportion of the total number of actual (unseen) events in the next hour were accounted for by the first gridsquare, the first two gridsquares, first three gridsquares, and so on.

The right-hand pictures in Figs. 5 and 6 show results that compare (3) and (4). Here, for example, 0.01 on the x-axis corresponds to using the top 1% of the gridsquares and 0.1 on the y-axis corresponds to accounting for 10% of the total activity. In general, as is the case for a classic Receiver Operating Characteristic (ROC) curve, a higher curve indicates better performance. In the figures, a red line is used for the ETAS model (3) and a blue line for the DR model (4). We see that the two models have comparable performance on the four categories in Fig. 5. For the burglary data in Fig. 6, ETAS starts to outperform DR as more gridsquares are considered.

We note that for the DR model, the maximum trigger effect for motor vehicle theft, theft over 500 dollars, aggravated assault and burglary arises at around 5-10 days, whereas for homicide this value increases to around 150 days. These results are consistent with, and add further insight to, previous findings; in particular, a study of repeat victimization in [22] found that the highest risk of a follow-on burglary arose during the first week after an initial event.

6 Discussion

Mathematical models are used in a vast range of disciplines within science and engineering as a means to investigate hypotheses, understand patterns and make predictions. Their use in the field of criminology has undergone a recent dramatic growth [4, 5], although it is imperative to acknowledge that any practical deployment of automated data-driven algorithms must address important, and possibly insurmountable, challenges around ethics, privacy and fairness [3, 6, 12, 13, 17, 23].

Our work focused on the development and evaluation of high-level mathematical models that allow us to incorporate and test hypotheses from the field of criminology about the overall mechanisms at play—what are the “laws of motion” for criminal events and how do they differ between crime types? We concentrated on one specific issue: the use of a triggering effect in self-exciting spatio-temporal point processes. Using a nonparametric approach, we quantified and visualised the characteristic triggering patterns for five distinct categories of crime. We then compared two parametrised trigger functions: the widely used Exponential Type After Shock model (3) and a new Delayed Response model (4). Both models have the same number of parameters and the same overall exponential decay in space and time. The Exponential Type After Shock model is based on the hypothesis that the trigger effect is greatest at the exact instant that an event takes place, whereas the Delayed Response model incorporates a tuneable “lag time” before the likelihood of a triggered event is maximum. The two models were seen to perform similarly with respect to predictive

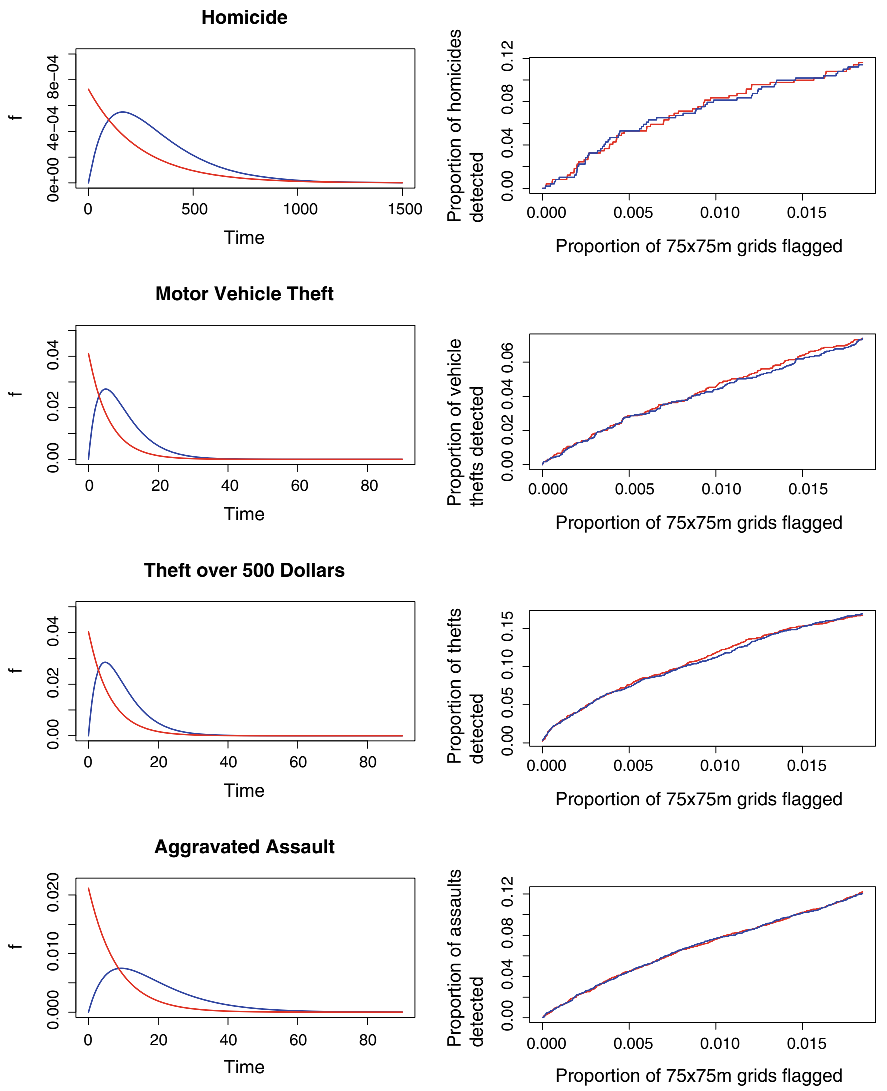


Fig. 5 Left: red and blue lines show the temporal component of the ETAS and DR trigger functions, respectively, with the parameters from Tables 1 and 2. Right: corresponding prediction results on unseen data; here, higher is better

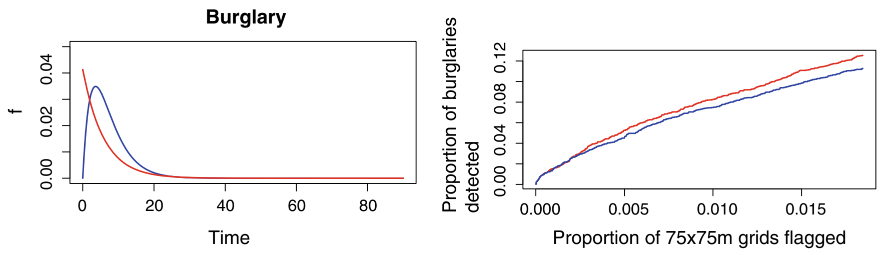


Fig. 6 As for Fig. 5, with burglary data

power. The Delayed Response model also delivers lag times that are consistent with empirical studies in repeat victimization, and in this sense we believe that it adds value to current modelling activities in this area.

Acknowledgements CG was supported by EPSRC Programme Grant EP/P020720/1. DJH was supported by grant EP/M00158X/1 from the EPSRC/RCUK Digital Economy Programme and by EPSRC Programme Grant EP/P020720/1. Illustrative R code may be downloaded from https://www.maths.ed.ac.uk/~dhigham/algfiles.html

References

1. Street and Site Plan Design Standards, City of Chicago. https://www.cityofchicago.org/dam/city/depts/cdot/StreetandSitePlanDesignStandards407.pdf, 2007. [Online; accessed 13-November-2018].
2. City of Chicago Data Portal. https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-present/ijzp-q8t2/data, 2016. [Online; accessed 20-November-2016].
3. L. Bennett Moses and J. Chan. Algorithmic prediction in policing: assumptions, evaluation, and accountability. *Policing and Society*, 28(7):806–822, 2018.
4. H. Berestycki, S. Johnson, J. Ockendon, and M. Primicerio. Criminality. *European Journal of Applied Mathematics*, 21(4-5), 2010.
5. A. Bertozzi, S. Johnson, and M. Ward. Mathematical modelling of crime and security: Special issue of EJAM. *European Journal of Applied Mathematics*, 27(3):311–316, 2016.
6. A. A. Braga. The effects of hot spots policing on crime. *The ANNALS of the American Academy of Political and Social Science*, 578(1):104–125, 2001.
7. A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (methodological)*, pages 1–38, 1977.
8. J. E. Douglas, R. K. Ressler, A. W. Burgess, and C. R. Hartman. Criminal profiling from crime scene analysis. *Behavioral Sciences & the Law*, 4(4):401–421, 1986.
9. M. Egesdal, C. Fathauer, K. Louie, J. Neuman, G. Mohler, and E. Lewis. Statistical and stochastic modeling of gang rivalries in Los Angeles. *SIAM Undergraduate Research Online*, 3:72–94, 2010.
10. C. Gilmour. *Self-exciting Point Processes and their Applications to Crime Data*. PhD thesis, University of Strathclyde, 2019.
11. P. Grindrod. *Mathematical Underpinnings of Analytics*. Oxford University Press, 2015.
12. P. Hunt, J. Saunders, and J. S. Hollywood. *Evaluation of the Shreveport predictive policing experiment*. Rand Corporation, 2014.

13. T. Karppi. “The computer said so”: On the ethics, effectiveness, and cultural techniques of predictive policing. https://doi.org/10.1177.2056305118768296, May 2018. Social Media+ Society.
14. P. J. Laub, T. Taimre, and P. K. Pollett. Hawkes processes. *arXiv:1507.02822*, 2015.
15. E. Lewis, G. Mohler, P. J. Brantingham, and A. L. Bertozzi. Self-exciting point process models of civilian deaths in Iraq. *Security Journal*, 25(3):244–264, 2012.
16. D. Marsan and O. Lengliné. Extending earthquakes’ reach through cascading. *Science*, 319(5866):1076–1079, 2008.
17. A. Meijer and M. Wessels. Predictive policing: Review of benefits and drawbacks. *International Journal of Public Administration*, 42(12):1031–1039, 2019.
18. G. Mohler. Marked point process hotspot maps for homicide and gun crime prediction in Chicago. *International Journal of Forecasting*, 30(3):491–497, 2014.
19. G. O. Mohler, M. B. Short, P. J. Brantingham, F. P. Schoenberg, and G. E. Tita. Self-exciting point process modeling of crime. *Journal of the American Statistical Association*, 106(493):100–108, 2011.
20. G. O. Mohler, M. B. Short, S. Malinowski, M. Johnson, G. E. Tita, A. L. Bertozzi, and P. J. Brantingham. Randomized controlled field trials of predictive policing. *Journal of the American Statistical Association*, 110(512):1399–1411, 2015.
21. Y. Ogata. Statistical models for earthquake occurrences and residual analysis for point processes. *Journal of the American Statistical Association*, 83(401):9–27, 1988.
22. N. Polvi, T. Looman, C. Humphries, and K. Pease. The time course of repeat burglary victimization. *The British Journal of Criminology*, 31(4):411–414, 1991.
23. J. H. Ratcliffe, R. B. Taylor, A. P. Askey, K. Thomas, J. Grasso, K. J. Bethel, R. Fisher, and J. Koehnlein. The Philadelphia predictive policing experiment. *Journal of Experimental Criminology*, in press, 2020.
24. A. Reinhart. A review of self-exciting spatio-temporal point processes and their applications. *Statistical Science*, 33(3):299–318, 2018.
25. A. Reinhart and J. Greenhouse. Self-exciting point processes with spatial covariates: modelling the dynamics of crime. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 67(5):1305–1329, 2018.
26. S. Tench, H. Fry, and P. Gill. Spatio-temporal patterns of ied usage by the provisional Irish republican army. *European Journal of Applied Mathematics*, 27(3):377–402, 2016.
27. A. Veen and F. P. Schoenberg. Estimation of space–time branching process models in seismology using an em–type algorithm. *Journal of the American Statistical Association*, 103(482):614–624, 2008.
28. C. J. Wu. On the convergence properties of the EM algorithm. *The Annals of Statistics*, pages 95–103, 1983.
29. S. F. C. K. Zipkin, J. and A. Bertozzi. Point-process models of social network interactions: Parameter estimation and missing data recovery. *European Journal of Applied Mathematics*, 27, 2016.

A New Multi-point Stepsize Gradient Method for Optimization

Yakui Huang, Yu-Hong Dai, and Xin-Wei Liu

Abstract The Barzilai-Borwein (BB) gradient method, which employs two-point stepsizes computed by the information of two consecutive iterations, is efficient in solving large-scale unconstrained optimization. In this paper, motivated by the success of BB method and the multi-point stepsize proposed by Dai and Fletcher (Mathematical Programming, 2006, 106: 403–421), we develop a new efficient gradient method, called MPSG, which adaptively uses the Dai-Fletcher long stepsize and Dai-Fletcher short stepsize. The R-linear convergence of MPSG for general n-dimensional strictly convex quadratic functions is established. By making use of two modified multi-point stepsizes and nonmonotone line searches, MPSG is extended to solve general unconstrained optimization. Moreover, the proposed MPSG method is further generalized for solving extreme eigenvalue problems. Numerical experiments on quadratic optimization, general unconstrained optimization and extreme eigenvalue problems demonstrate the efficiency of our method.

Keywords Gradient methods · Multi-point stepsize · Quadratic optimization · Unconstrained optimization · Extreme eigenvalue problems

Y. Huang · X.-W. Liu
Institute of Mathematics, Hebei University of Technology, Tianjin 300401, China
e-mail: huangyakui2006@gmail.com

X.-W. Liu
e-mail: mathlxw@hebut.edu.cn

Y.-H. Dai (✉)
State Key Laboratory of Scientific and Engineering Computing, Institute of Computational Mathematics and Scientific/Engineering Computing, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China
e-mail: dyh@lsec.cc.ac.cn

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*, Springer Proceedings in Mathematics & Statistics 354, https://doi.org/10.1007/978-3-030-72040-7_7

1 Introduction

The Barzilai-Borwein (BB) gradient method [4] is efficient for solving large-scale unconstrained optimization problem

$$\min_{x\in\mathbb{R}^n} f(x), \tag{1}$$

where $f:\mathbb{R}^n\to\mathbb{R}$ is continuously differentiable. In particular, the BB method updates iterates by

$$x_{k+1} = x_k - \alpha_k g_k, \tag{2}$$

where $g_k = \nabla f(x_k)$ and $\alpha_k > 0$ is calculated by

$$\alpha_k^{BB1} = \arg\min_{\alpha>0} \|\alpha^{-1}s_{k-1} - y_{k-1}\| = \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T y_{k-1}} \tag{3}$$

or

$$\alpha_k^{BB2} = \arg\min_{\alpha>0} \|s_{k-1} - \alpha y_{k-1}\| = \frac{s_{k-1}^T y_{k-1}}{y_{k-1}^T y_{k-1}}, \tag{4}$$

where $s_{k-1} = x_k - x_{k-1},\ y_{k-1} = g_k - g_{k-1}$ and $\|\cdot\|$ means the Euclidean norm. Clearly, when $s_{k-1}^T y_{k-1} > 0$, there holds $\alpha_k^{BB1} \geq \alpha_k^{BB2}$. Unlike the traditional steepest descent (SD) method [8], the BB method generally generates nonmonotone objective values. However, it often performs much better than the SD method in practice, see [22, 33, 40] for example. Theoretically, when $f(x)$ is a strictly quadratic function, i.e.,

$$f(x) = \frac{1}{2}x^T Ax - b^T x, \tag{5}$$

where $b \in \mathbb{R}^n$ and $A \in \mathbb{R}^{n\times n}$ is symmetric and positive definite, the BB method converges R-superlinearly for the two-dimensional case [4], which is better than the Q-linear convergence rate of the SD method [2]. For general n-dimensional case, the BB method is globally convergent [32] and the convergence rate is R-linear [15].

For general unconstrained optimization, to ensure global convergence of the BB method, line searches are often necessary. Raydan [33] suggested to incorporate the Grippo-Lampariello-Lucidi (GLL) nonmonotone line search in [24], which requires the new objective value is smaller than the maximal function value in recent M iterations, i.e.,

$$f(x_k - \lambda_k g_k) \leq f_r - \sigma\lambda_k g_k^T g_k, \tag{6}$$

where $f_r = \max_{0\leq j\leq \min\{k,M-1\}} f(x_{k-j})$ with M being a positive integer and λ_k is the step length. By combining the GLL nonmonotone line search, a global BB (GBB) method was proposed. With carefully investigation of performances of the BB method, Fletcher [22] argued that when a not high accuracy is required, the BB method could be even competitive with nonlinear conjugate gradient (CG) methods. Extensive studies support the view of Fletcher, especially for the case of smooth unconstrained optimization, see [6, 22, 33] for example.

A great advantage of the BB method is its easy extension to solve general unconstrained and constrained optimization. For unconstrained optimization, Zhou et al. [42] proposed the adaptive BB (ABB) method, which employs the stepsize

$$\alpha_k = \begin{cases} \alpha_k^{BB2}, & \text{if } \alpha_k^{BB2}/\alpha_k^{BB1} < \tau; \\ \alpha_k^{BB1}, & \text{otherwise}, \end{cases} \tag{7}$$

where $\tau \in (0, 1)$. Although ABB is originally developed for quadratics, it performs well for general functions [42]. Dai et al. [13] developed an adaptive cyclic BB (CBB) method which reuses the same BB stepsize for m consecutive iterations. They observed that when $m > n/2 > 3$, where n is the problem dimension, CBB is locally superlinearly convergent. Motivated by the Broyden class of quasi-Newton methods [7], Dai et al. [14] derived a family of BB-like methods whose stepsize is given by the convex combination of the two BB stepsizes, i.e.,

$$\alpha_k = \gamma\alpha_k^{BB1} + (1-\gamma)\alpha_k^{BB2}, \tag{8}$$

where $\gamma \in [0, 1]$. Clearly, α_k^{BB1} and α_k^{BB2} correspond to $\gamma = 1$ and 0, respectively. It is shown that each method in the family converges R-superlinearly when minimizing two-dimensional strictly convex quadratics and R-linearly for the n-dimensional case. As for constrained optimization, by combining gradient projection techniques, Birgin et al. [5] generalized the GBB method to minimize differentiable functions on closed convex sets. Dai and Fletcher [11] suggested to solve the bound constrained optimization by the so-called projected alternating BB method which uses the stepsize

$$\alpha_k = \begin{cases} \alpha_k^{BB1}, & \text{for odd } k; \\ \alpha_k^{BB2}, & \text{for even } k. \end{cases} \tag{9}$$

By incorporating smoothing techniques, Huang and Liu [27] generalized the modified projected alternating BB method for solving non-Lipschitz constrained optimization. BB-like methods have been successfully applied in many important fields including image restoration [37], signal processing [31], eigenvalue problems [30], nonnegative matrix factorization [28], sparse reconstruction [38], machine learning [36], etc. See [6, 14, 20, 22, 40] and references therein for more BB-like methods and applications.

In [12], Dai and Fletcher considered singly linearly constrained quadratic programs subjected to upper and lower bounds. They introduced a new multi-point stepsize calculated by using information of recent m iterations:

$$\alpha_{k+1}^{DF1} = \arg\min_{\alpha} \|\alpha^{-1}\tilde{s}_k - \tilde{y}_k\| = \frac{\tilde{s}_k^T\tilde{s}_k}{\tilde{s}_k^T\tilde{y}_k} = \frac{\sum_{i=0}^{m-1} s_{k-i}^T s_{k-i}}{\sum_{i=0}^{m-1} s_{k-i}^T y_{k-i}}, \tag{10}$$

where $m \geq 1$ and

$$\tilde{s}_k = (s_k^T, s_{k-1}^T, \ldots, s_{k-m+1}^T)^T, \quad \tilde{y}_k = (y_k^T, y_{k-1}^T, \ldots, y_{k-m+1}^T)^T. \tag{11}$$

Obviously, α_{k+1}^{DF1} reduces to α_{k+1}^{BB1} when $m = 1$. The stepsize α_{k+1}^{DF1} performs very well in training support vector machines.

In this paper, motivated by the success of the Dai-Fletcher stepsize, we develop an efficient gradient method using multi-point stepsizes for solving general unconstrained optimization. Particularly, as (4), we can derive another multi-point stepsize as follows

$$\alpha_{k+1}^{DF2} = \arg\min_{\alpha} \|\tilde{s}_k - \alpha \tilde{y}_k\| = \frac{\tilde{s}_k^T \tilde{y}_k}{\tilde{y}_k^T \tilde{y}_k} = \frac{\sum_{i=0}^{m-1} s_{k-i}^T y_{k-i}}{\sum_{i=0}^{m-1} y_{k-i}^T y_{k-i}}. \tag{12}$$

Clearly, when $\tilde{s}_k^T \tilde{y}_k > 0$, there holds $\alpha_{k+1}^{DF1} \geq \alpha_{k+1}^{DF2}$. So, we will refer to α_{k+1}^{DF1} and α_{k+1}^{DF2} as Dai-Fletcher long stepsize and Dai-Fletcher short stepsize, respectively. Based on the two stepsizes α_{k+1}^{DF1} and α_{k+1}^{DF2}, we first propose an adaptive multi-point stepsize gradient (MPSG) method for quadratics. The R-linear convergence of MPSG is established for n-dimensional strictly convex quadratics. Then we extend the MPSG method to solve general unconstrained optimization by slightly modifying the two stepsizes and incorporating the adaptive nonmonotone line search proposed by Dai and Zhang [17]. Furthermore, we extend the method for solving extreme eigenvalues problems. Numerical results on quadratic optimization, general unconstrained optimization and extreme eigenvalue problems show that our proposed MPSG method is very efficient.

The paper is organized as follows. In Sect. 2, we first propose the MPSG method for quadratics and show its R-linear convergence. Then we generalize the method to solve general unconstrained optimization. In Sect. 3, we apply the method to extreme eigenvalue problems. Section 4 present our extensive numerical experiments on solving quadratic optimization, general unconstrained and eigenvalue problems. Finally, some conclusions and discussions are given in the last section.

2 MPSG Method

In this section, based on the Dai-Fletcher long stepsize α_{k+1}^{DF1} and Dai-Fletcher short stepsize α_{k+1}^{DF2}, we propose an adaptive multi-point stepsize gradient method MPSG for quadratics. Then we extend it to solve general unconstrained optimization.

2.1 Quadratic Case

Now we consider to minimize a strictly quadratic function (5). Without loss of generality, we assume that the matrix A is diagonal, i.e.,

$$A = \mathrm{diag}\{\lambda_1, \ldots, \lambda_n\}, \tag{13}$$

where $0 < \lambda_1 \leq \ldots \leq \lambda_n$.

It follows from the definitions of $\tilde{s}_k$ and $\tilde{y}_k$ in (11) that

$$\tilde{s}_k^T \tilde{y}_k = \tilde{s}_k^T B \tilde{s}_k \quad \text{and} \quad \tilde{y}_k^T \tilde{y}_k = \tilde{s}_k^T B^2 \tilde{s}_k,$$

where $B = I \otimes A$ with $\otimes$ being the Kronecker product. By Theorem 4.2.12 in [25], we know that the matrix B has same eigenvalues as A. Thus, by the Cauchy-Schwarz inequality, we obtain

$$\frac{1}{\lambda_n} \leq \alpha_{k+1}^{DF2} \leq \alpha_{k+1}^{DF1} \leq \frac{1}{\lambda_1}. \tag{14}$$

So, the two Dai-Fletcher stepsizes have similar spectral property as the BB stepsizes.

Recent studies show that a gradient method using the adaptive scheme with a long and some short stepsizes performs generally better than that with only one stepsize, see [13, 42] for example. So, our method incorporates an adaptive scheme similar as the one in [42]. In order to get a short stepsize, we would like to use the smaller one of recent two Dai-Fletcher short stepsizes. More precisely, our method uses the following stepsize

$$\alpha_{k+1} = \begin{cases} \min\{\alpha_k^{DF2}, \alpha_{k+1}^{DF2}\}, & \text{if } \alpha_{k+1}^{DF2}/\alpha_{k+1}^{DF1} < \tau; \\ \alpha_{k+1}^{DF1}, & \text{otherwise,} \end{cases} \tag{15}$$

where $\tau \in (0, 1)$. Recall that both α_{k+1}^{DF1} and α_{k+1}^{DF2} need information of recent m iterations. When $k \leq m$, we simply set $m = k$. In what follows, the method (15) will be referred to as MPSG.

To establish the R-linear convergence of MPSG, we need the following notation

$$G(k, l) = \sum_{i=1}^{l} (g_k^{(i)})^2.$$

In [10], Dai proved that if a gradient method employs some stepsize satisfies the following Property (A) then it converges R-linearly.

Property (A) [10]. Suppose that there exist an integer M_1 and positive constants $c_1 \geq \lambda_1$ and c_2 such that

(i) $\lambda_1 \le \alpha_k^{-1} \le c_1$;

(ii) for any integer $l \in [1, n-1]$ and $\epsilon > 0$, if $G(k-j, l) \le \epsilon$ and $(g_{k-j}^{(l+1)})^2 \ge c_2\epsilon$ hold for $j \in [0, \min\{k, M_1\} - 1]$, then $\alpha_k^{-1} \ge \frac{2}{3}\lambda_{l+1}$.

Based on Dai's result, we show R-linear convergence of MPSG in the following theorem.

Theorem 1 *Suppose that the sequence $\{\|g_k\|\}$ is generated by applying MPSG to n-dimensional quadratics with the matrix A has the form* (13) *and* $1 = \lambda_1 \le \lambda_2 \le \cdots \le \lambda_n$. *Then either $g_k = 0$ for some finite k or the sequence $\{\|g_k\|\}$ converges to zero R-linearly.*

Proof We are suffice to prove MPSG satisfies Property (A). Let $c_1 = \lambda_n$, $c_2 = 2$ and $M_1 = m + 1$. By (14), we conclude that (i) of Property (A) holds.

If $G(k-i, l) \le \epsilon$ and $(g_{k-i}^{(l+1)})^2 \ge c_2\epsilon$ hold for $i \in [0, \min\{k, M_1\} - 1]$, we have

$$
\begin{aligned}
(\alpha_{k+1}^{DF1})^{-1} &= \frac{\sum_{i=0}^{m-1} s_{k-i}^T y_{k-i}}{\sum_{i=0}^{m-1} s_{k-i}^T s_{k-i}} = \frac{\sum_{i=0}^{m-1} (\alpha_{k-i}^{DF1})^2 \sum_{j=1}^{n} \lambda_j (g_{k-i}^{(j)})^2}{\sum_{i=0}^{m-1} (\alpha_{k-i}^{DF1})^2 \sum_{j=1}^{n} (g_{k-i}^{(j)})^2} \\
&\ge \frac{\lambda_{l+1} \sum_{i=0}^{m-1} (\alpha_{k-i}^{DF1})^2 \sum_{j=l+1}^{n} (g_{k-i}^{(j)})^2}{\sum_{i=0}^{m-1} (\alpha_{k-i}^{DF1})^2 (\epsilon + \sum_{j=l+1}^{n} (g_{k-i}^{(j)})^2)} \\
&= \frac{\lambda_{l+1}}{\frac{\sum_{i=0}^{m-1} (\alpha_{k-i}^{DF1})^2 \epsilon}{\sum_{i=0}^{m-1} (\alpha_{k-i}^{DF1})^2 \sum_{j=l+1}^{n} (g_{k-i}^{(j)})^2} + 1} \\
&\ge \frac{c_2}{c_2 + 1} \lambda_{l+1} = \frac{2}{3} \lambda_{l+1}.
\end{aligned}
$$

When $\alpha_{k+1}^{DF2} / \alpha_{k+1}^{DF1} < \tau$, by (14), one has

$$
(\alpha_{k+1})^{-1} \ge (\alpha_{k+1}^{DF2})^{-1} \ge (\alpha_{k+1}^{DF1})^{-1}.
$$

So, (ii) of Property (A) holds. This completes the proof. □

2.2 General Unconstrained Optimization

In this subsection, we extend MPSG for solving general unconstrained optimization.

When applied the MPSG method to general functions, it is possible that $s_k^T y_k \le 0$. To address this issue, for BB-like methods, a frequently used strategy is adding safeguards

$$
\alpha_{k+1} = \max\{\alpha_{\min}, \min\{\alpha_{k+1}, \alpha_{\max}\}\}, \tag{16}
$$

where $\alpha_{\max}$ and $\alpha_{\min}$ are preassigned positive constants satisfying $\alpha_{\max} \gg \alpha_{\min} > 0$. However, for multi-point stepsizes α_{k+1}^{DF1} and α_{k+1}^{DF2}, $s_{k-i}^T y_{k-i}$ may be negative for any

$i \in \{0, \ldots, m-1\}$, which may further lead to $\alpha_{k+1}^{DF1} < 0$ and $\alpha_{k+1}^{DF2} < 0$. In this case, Dai and Fletcher [12] suggested to compute the multi-point stepsize by replacing m by $\min\{m, \bar{m}\}$, where $\bar{m}$ is the maximal integer such that $s_{k-i}^T y_{k-i} > 0$ for all $0 \le i \le \bar{m}$. And when $\bar{m} = 0$, setting $\alpha_{k+1} = \alpha_{\max}$. Taking $m = 2$ for example, if $s_k^T y_k \le 0$ then α_{k+1} will be set to $\alpha_{\max}$, which is not a good stepsize in general since it is very large. However, $s_{k-1}^T y_{k-1}$ may be positive which is abandoned by the above strategy. So, to avoid negative stepsizes and exploit the information carried by recent iterations, we would like to use the following modified stepsizes

$$\alpha_{k+1}^{MDF1} = \frac{\sum_{i=0}^{m-1} s_{k-i}^T s_{k-i}}{\sum_{i=0}^{m-1} |s_{k-i}^T y_{k-i}|} \tag{17}$$

and

$$\alpha_{k+1}^{MDF2} = \frac{\sum_{i=0}^{m-1} |s_{k-i}^T y_{k-i}|}{\sum_{i=0}^{m-1} y_{k-i}^T y_{k-i}}. \tag{18}$$

Similarly as the quadratic case, our MPSG method for general unconstrained optimization uses the above two modified stepsizes adaptively, i.e.,

$$\alpha_{k+1} = \begin{cases} \min\{\alpha_k^{MDF2}, \alpha_{k+1}^{MDF2}\}, & \text{if } \alpha_{k+1}^{MDF2}/\alpha_{k+1}^{MDF1} < \tau; \\ \alpha_{k+1}^{MDF1}, & \text{otherwise,} \end{cases} \tag{19}$$

where $\tau \in (0, 1)$. We also enforce safeguards (16) to chop any extreme values of the stepsize.

To ensure global convergence, a nonmonotone line search is often employed by BB-like methods. Although the GLL nonmonotone line search [24] performs well in many cases, Dai [9] presented an example to show that the GLL condition (6) may fail for any fixed M. To address this issue, Zhang and Hager [41] developed a new nonmonotone line search that uses a reference value defined by the convex combination of the former function values. A nonmonotone line search specially suitable for BB-like methods is due to Dai and Zhang [17], which accepts the step length $\lambda_k = 1$ if

$$f(x_k - g_k) \le f_r - \sigma g_k^T g_k. \tag{20}$$

Otherwise, an Armijo-type back tracking line search will be performed to find a step length λ_k such that

$$f(x_k - \lambda_k \alpha_k g_k) \le \min\{f_{\max}, f_r\} - \sigma \lambda_k \alpha_k g_k^T g_k, \tag{21}$$

where $f_{\max}$ is the maximal function value in recent M iterations. See [17] for the update of the reference value f_r.

Using the same arguments as the one in Theorem 3.2 of [17], we can show that, when $f(x)$ is twice-continuously differentiable and is bounded from below, our MPSG method is global convergent in the sense $\liminf_{k\to\infty} \|g_k\| = 0$.

3 Extension to Extreme Eigenvalue Problems

In this section, we extend our MPSG method to solve extreme eigenvalue problems of large-scale symmetric positive definite matrices.

For a given $n \times n$ real symmetric matrix A, its eigenvalue decomposition has the form

$$A = Q\Lambda Q^T,$$

where $Q \in \mathbb{R}^{n\times n}$ is an orthogonal matrix whose columns are eigenvectors of A and $\Lambda = \mathrm{diag}\{\lambda_1, \dots, \lambda_n\}$ with $\lambda_1 \le \dots \le \lambda_n$.

The calculation of eigenvalue and eigenvector is a fundamental problem with important applications in scientific computing and engineering such as principal component analysis [18] and electronic structure calculation [26]. In practice, it is usually realistic to compute the first $r \ll n$ largest/smallest eigenvalues and their corresponding eigenvectors of matrix A.

The problem of computing the first $r \ll n$ largest/smallest eigenvalues and their corresponding eigenvectors of matrix A can be written as Rayleigh quotient minimization [1]

$$\min_{X\in\mathbb{R}^{n\times r}} \quad tr(X^TAX(X^TX)^{-1}) \tag{22}$$

or trace minimization [34, 35]

$$\begin{aligned} \min_{X\in\mathbb{R}^{n\times r}} \quad & tr(X^TAX) \\ \text{s.t.} \quad & X^TX = I_r, \end{aligned} \tag{23}$$

where I_r denotes the $r \times r$ identity matrix. A drawback of the above two models lies in computing inverse or orthogonalization of a matrix, which may be expensive for large matrices.

Recently, Jiang et al. [29] suggested the following unconstrained model

$$\min_{X\in\mathbb{R}^{n\times r}} P_\mu(X) = \frac{1}{4}tr(X^TXX^TX) + \frac{1}{2}tr(X^T(A-\mu I_n)X), \tag{24}$$

where $\mu > 0$ is a scaling parameter. They proved that, with proper μ, the global minimizers of (24) lie in the eigenspace corresponding to the r smallest eigenvalues of A. Furthermore, they suggested an alternate BB method, called EigUncABB, for solving (24), which uses the following modified BB stepsize alternately

$$\alpha_k^{MBB1} = \frac{tr(S_{k-1}^TS_{k-1})}{|tr(S_{k-1}^TY_{k-1})|} \quad\text{and}\quad \alpha_k^{MBB2} = \frac{|tr(S_{k-1}^TY_{k-1})|}{tr(Y_{k-1}^TY_{k-1})},$$

i.e.,

$$\alpha_k = \begin{cases} \alpha_k^{MBB2}, & \text{for odd } k; \\ \alpha_k^{MBB1}, & \text{for even } k. \end{cases} \tag{25}$$

Here, $S_{k-1} = X_k - X_{k-1}$ and $Y_{k-1} = \nabla P_\mu(X_k) - \nabla P_\mu(X_{k-1})$. To ensure global convergence, the EigUncABB method incorporates the nonmonotone line search used in [12], which can be viewed as a simplified version of the Dai-Zhang nonmonotone line search. In particular, the nonmonotone line search requires the step length λ_k to satisfy

$$f(x_k - \lambda_k \alpha_k g_k) \le f_r - \sigma \lambda_k \alpha_k g_k^T g_k, \tag{26}$$

where the reference value f_r is updated as follows:

```
if $f_k < f_{best}$,
    $f_{best} = f_k,\ f_c = f_k,\ l = 0$,
else
    $f_c = \max\{f_c, f_k\},\ l = l + 1$,
    if $l = L$,
        $f_r = f_k,\ f_c = f_k,\ l = 0$,
    end
end
```

Here, f_{best} is the current best function value, i.e.,

$$f_{best} = \min_{1 \le j \le k} f(x_j),$$

l is the number of iterations since the value of f_{best} was obtained, f_c is the maximum value of the objective function since the value of f_{best} was found, and L is a preassigned number.

The EigUncABB method appears very competitive with the Matlab build-in function EIGS and other recent methods. Motivated by EigUncABB, we would like to extend our MPSG method to solve extreme eigenvalue problem (24). In fact, we only need to adopt the two multi-point stepsizes (17) and (18) to matrix cases. Particularly, we can compute them by

$$\alpha_{k+1}^{MDF1} = \frac{\sum_{i=0}^{m-1} tr(S_{k-i}^T S_{k-i})}{\sum_{i=0}^{m-1} |tr(S_{k-i}^T Y_{k-i})|} \tag{27}$$

and

$$\alpha_{k+1}^{MDF2} = \frac{\sum_{i=0}^{m-1} |tr(S_{k-i}^T Y_{k-i})|}{\sum_{i=0}^{m-1} tr(Y_{k-i}^T Y_{k-i})}. \tag{28}$$

Our method for (24) will be referred to as EigUncMPSG, which employs the adaptive stepsize (19) with the above two stepsizes and the above simplified nonmonotone line search. That is, the only difference between EigUncMPSG and EigUncABB is the stepsize. In the next section, we will see that our new EigUncMPSG method often performs better the EigUncABB method.

4 Numerical Results

In this section, we present numerical comparisons of our proposed MPSG method with other recent successful gradient methods on quadratic optimization, general unconstrained optimization and extreme eigenvalue problems. All the algorithms were implemented in Matlab (v.9.0-R2016a) and run on a laptop with an Intel Core i7, 2.9 GHz processor and 8 GB of RAM running Windows 10 system.

4.1 Quadratic Optimization Problems

In this subsection, we compare our MPSG method with the BB1 [4], DY [16], ABBmin2 [23], and SDC [19] methods on solving quadratic optimization problems.

We tested the methods on randomly generated quadratic problems in [39], i.e.,

$$\min_{x \in \mathbb{R}^n} f(x) = (x - x^*)^T V (x - x^*). \tag{29}$$

Here, x^* is a randomly generated solution whose components lie in $[-10, 10]$, and $V = \mathrm{diag}\{v_1, \ldots, v_n\}$ is a diagonal matrix whose diagonal components are $v_1 = 1$, $v_n = \kappa$ and $v_j \in (1, \kappa)$, $j = 2, \ldots, n-1$, are generated by the *rand* function.

As we know, if the Hessian V has different spectral distributions the performances of a gradient method may be quite different. So, we tested problem (29) using five sets of different spectral distributions of the Hessian listed in Table 1 with $n = 1,000$. We stopped the method if the number of iteration exceeds 20,000 or $\|g_k\| \le \epsilon \|g_0\|$, where $\epsilon > 0$ is a given tolerance. To investigate the performances of the compared methods on different tolerances and condition numbers, we set $\epsilon = 10^{-6}, 10^{-9}, 10^{-12}$ and $\kappa = 10^4, 10^5, 10^6$.

Table 1 Distributions of v_j

Problem	Spectrum
*1	$\{v_2, \ldots, v_{n-1}\} \subset (1, \kappa)$
*2	$\{v_2, \ldots, v_{n/5}\} \subset (1, 100)$
	$\{v_{n/5+1}, \ldots, v_{n-1}\} \subset (\frac{\kappa}{2}, \kappa)$
*3	$\{v_2, \ldots, v_{n/2}\} \subset (1, 100)$
	$\{v_{n/2+1}, \ldots, v_{n-1}\} \subset (\frac{\kappa}{2}, \kappa)$
*4	$\{v_2, \ldots, v_{4n/5}\} \subset (1, 100)$
	$\{v_{4n/5+1}, \ldots, v_{n-1}\} \subset (\frac{\kappa}{2}, \kappa)$
*5	$\{v_2, \ldots, v_{n/5}\} \subset (1, 100)$
	$\{v_{n/5+1}, \ldots, v_{4n/5}\} \subset (100, \frac{\kappa}{2})$
	$\{v_{4n/5+1}, \ldots, v_{n-1}\} \subset (\frac{\kappa}{2}, \kappa)$

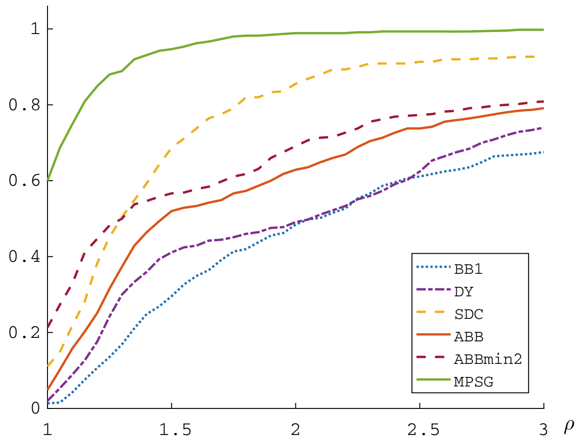


Fig. 1 Performance profiles of compared methods on solving random quadratic problems (29) with spectral distributions in Table 1, iteration metric

The parameter τ was set to 0.9 for the ABBmin2 method as suggested in [23] whereas the pair (h, s) of the SDC method was set to (8, 6) which is more efficient than other choices for this test. The parameters m and τ for our MPSG method were set to 2 and 0.1, respectively. For each value of κ or ϵ, averaged results of 10 instances using the starting point $x_0 = (0, \ldots, 0)^T$ are presented.

The performance profiles of Dolan and Moré [21] on iteration metric was employed to compare the algorithms, where the vertical axis shows the percentage of the problems the method solves within the factor ρ of the metric used by the most effective method in this comparison. It can be seen from Fig. 1 that MPSG is much better than other methods in terms of number of iterations.

Table 2 gives the numbers of averaged iterations of the compared methods. Apparently, MPSG outperforms the BB1, DY, ABB and SDC methods on the first problem set though ABBmin2 is faster than MPSG. For the second to fifth problem sets, MPSG is much better than other methods. In addition, for each tolerance, MPSG is the fastest method in terms of total number of iterations.

4.2 *Unconstrained Optimization Problems*

Now we compare our MPSG method with the GBB method of Raydan [33] on 59 unconstrained problems from [3] listed in Table 3 with $n = 1,000$.

For the Dai-Zhang nonmonotone line search, the parameter values were set as follows:

$$\alpha_{\min} = 10^{-20},\ \alpha_{\max} = 10^{20},\ M = 8,\ \sigma = 10^{-4},\ \alpha_0 = 1/\|g_0\|_\infty.$$

Table 2 The numbers of averaged iterations of the BB1, DY, ABB, ABBmin2, SDC and MPSG methods on solving quadratic problems (29) with spectral distributions in Table 1

Set	ϵ	BB1	DY	SDC	ABB	ABBmin2	MPSG
1	1e–06	266.8	227.4	222.5	214.4	197.4	224.1
	1e–09	2381.3	2657.4	1758.4	1039.2	501.2	699.5
	1e–12	6128.6	6108.0	4262.8	1294.9	661.4	1018.4
2	1e–06	315.5	275.0	170.5	277.2	272.4	143.4
	1e–09	1776.5	1462.8	727.1	1567.0	1329.9	532.4
	1e–12	3151.6	2539.9	1316.8	2688.5	2135.8	908.3
3	1e–06	379.7	321.4	219.1	347.2	422.1	183.4
	1e–09	1685.4	1522.1	852.7	1655.7	1460.1	592.0
	1e–12	3055.0	2608.9	1340.5	2874.9	2364.3	943.8
4	1e–06	523.0	458.4	253.5	449.4	528.7	211.0
	1e–09	1956.4	1737.7	859.8	1802.1	1540.2	612.6
	1e–12	3080.8	2842.8	1310.4	2996.9	2481.6	995.3
5	1e–06	846.1	655.7	679.3	685.7	867.7	616.4
	1e–09	4237.9	3763.3	3361.1	2943.4	3146.0	2518.9
	1e–12	7174.5	7567.8	6070.2	4878.0	4936.7	4239.0
Total	1e–06	2331.1	1937.9	1544.9	1973.9	2288.3	1378.3
	1e–09	12037.5	11143.3	7559.1	9007.4	7977.4	4955.4
	1e–12	22590.5	21667.4	14300.7	14733.2	12579.8	8104.8

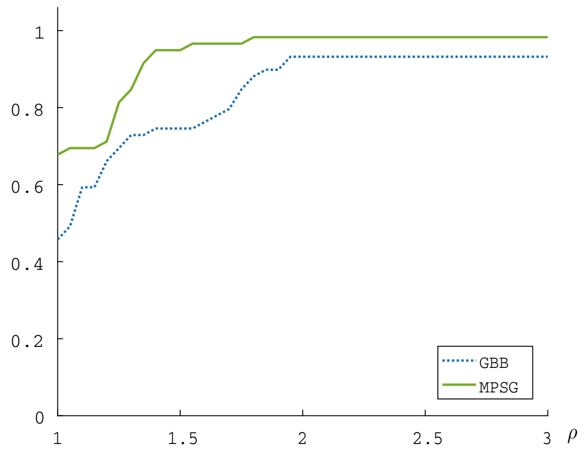


Fig. 2 Performance profile of the MPSG and GBB methods, iteration metric on unconstrained problems listed in Table 3

We used $m = 2$ and $\tau = 0.8$ for MPSG which performs better than other settings. Default parameters were used for GBB. Each method was stopped if the number of iteration exceeds 200,000 or $\|g_k\|_\infty \leq 10^{-6}$.

From Fig. 2 we can see that MPSG performs better than GBB on the test problems.

Table 3 Test problems from [3]

Problem	Name	Problem	Name
1	Extended Freudenstein & Roth function	31	NONDIA
2	Extended Trigonometric	32	DQDRTIC
3	Extended White & Holst	33	Partial Perturbed Quadratic
4	Extended Beale	34	Broyden Tridiagonal
5	Extended Penalty	35	Almost Perturbed Quadratic
6	Perturbed Quadratic	36	Perturbed Tridiagonal Quadratic
7	Raydan 1	37	Staircase 1
8	Raydan 2	38	Staircase 2
9	Diagonal 1	39	LIARWHD
10	Diagonal 2	40	POWER
11	Diagonal 3	41	ENGVAL1
12	Hager	42	EDENSCH
13	Generalized Tridiagonal 1	43	BDEXP
14	Extended Tridiagonal 1	44	GENHUMPS
15	Extended TET	45	NONSCOMP
16	Generalized Tridiagonal 2	46	VARDIM
17	Diagonal 4	47	QUARTC
18	Diagonal 5	48	Extended DENSCHNB
19	Extended Himmelblau	49	Extended DENSCHNF
20	Extended PSC1	50	LIARWHD
21	Generalized PSC1	51	BIGGSB1
22	Extended Powell	52	Generalized Quartic
23	Extended Cliff	53	Diagonal 7
24	Perturbed quadratic diagonal	54	Diagonal 8
25	Quadratic QF1	55	Full Hessian FH3
26	Extended quadratic exponential EP1	56	SINCOS
27	Extended Tridiagonal 2	57	Diagonal 9
28	BDQRTIC	58	HIMMELBG
29	TRIDIA	59	HIMMELH
30	ARWHEAD		

4.3 Extreme Eigenvalue Problems

In this subsection, we compare our EigUncMPSG method with the EigUncABB method in [29] on solving extreme eigenvalue problems.

The methods were compared on the example in [29], where a $16,000 \times 16,000$ matrix was generated by the following Matlab command

$$\text{[lambda,Uex,A]} = \text{laplacian([20, 20, 40], 'DD' 'NN' 'P');}$$

Here, "lambda" is the exact eigenvalues of A while "Uex" is the corresponding eigenvectors. This matrix is symmetric positive definite and can be viewed as the 3D negative Laplacian on a rectangular finite difference grid.

Let $\bar{u}_i$ and $\bar{\lambda}_i$ be the ith approximate eigenvector and eigenvalue obtained by the compared algorithms. Further denote the relative eigenvalue error and residual error of the ith eigenpair by

$$\text{err}_i = \frac{|\bar{\lambda}_i - \lambda_i|}{\max\{1, |\lambda_i|\}} \quad \text{and} \quad \text{resi}_i = \frac{\|A\bar{u}_i - \bar{\lambda}_i\bar{u}_i\|}{\max\{1, |\bar{\lambda}_i|\}}.$$

The mean values of err_i and resi_i, $i = 1, \ldots, r$, are used to measure the quality of the computed solution.

The parameters for our EigUncMPSG are same as the former subsection except $L = 10$ and $\tau = 0.3$. Default parameters were set for EigUncABB. We stopped the iteration when $\|\nabla P_\mu(X_k)\|_F \leq 10^{-4}$. For fair comparison, the same initial point generated by the following Matlab codes was used

$$\text{seed} = 100;\ \text{rng(seed,'twister')};\ X_0 = \text{randn}(n, r);\ X_0 = \text{orth}(X_0).$$

And the initial trial stepsize was set to $\alpha_0 = \|\nabla P_\mu(X_0)\|_F^{-1}$. The initial value of μ was set to $1.01 \times \lambda_{\bar{r}}(X_0^T A X_0)$, where $\bar{r} = \max\{\lfloor 1.1r \rfloor, 10\}$ with $\lfloor\cdot\rfloor$ denoting the nearest integer less than or equal to the corresponding element. In addition, μ was updated as follows:

```
if $\|\nabla P_\mu(X_k)\|_F \leq 0.1^{j_0}\|\nabla P_\mu(X_0)\|_F$ and $j_0 \leq j_{\max}$
  $\mu = 1.01\lambda_{\bar{r}}(X_k^T A X_k)$
  $j_0 = j_0 + 1$
end
```

In our test, as [29], j_0 and $j_{\max}$ were set to 1 and 3, respectively. See [29] for more details on the updating of μ.

Table 4 presents results obtained by the two compared methods where "nfe" denotes the total number of function evaluations and "time" denotes the CPU time in seconds. It can be seen that, for a given r, our EigUncMPSG method often outperforms EigUncABB in terms of function evaluations and CPU time. In addition, the values of relative eigenvalue error obtained by EigUncMPSG are better than or

Table 4 Comparison of EigUncMPSG and EigUncABB on extreme eigenvalue problems

	EigUncABB				EigUncMPSG			
r	resi	err	nfe	Time	resi	err	nfe	Time
20	1.37e–05	9.78e–09	160	2.7	8.48e–05	1.34e–08	144	2.2
50	1.62e–05	1.76e–08	170	5.3	1.66e–05	5.39e–08	151	4.7
100	7.09e–05	2.02e–09	183	10.5	7.51e–05	3.87e–09	193	11.9
200	3.75e–06	5.23e–10	216	29.9	5.71e–05	1.20e–09	195	27.5
300	4.11e–06	8.12e–11	188	44.5	2.10e–06	6.43e–11	158	40.9
400	6.86e–07	6.72e–13	168	69.8	5.59e–07	1.43e–12	172	66.7
500	3.17e–06	2.87e–11	227	124.8	5.79e–07	2.56e–13	221	127.0
600	3.16e–06	1.59e–12	238	175.4	1.10e–06	1.26e–12	199	153.2
700	5.99e–07	1.05e–12	212	215.0	2.47e–07	1.04e–13	239	243.5
800	2.36e–06	1.93e–11	234	306.7	3.27e–06	2.68e–12	193	242.3
900	3.51e–06	7.33e–12	194	312.4	8.84e–08	8.98e–15	235	342.6
1000	1.28e–06	2.06e–12	240	434.3	2.37e–07	3.46e–13	230	429.8

comparable to those by EigUncABB for most of the values of r. This suggests the potential benefits of our new multi-point stepsizes for solving matrix optimization.

Notes and Comments.
Based on the Dai-Fletcher stepsizes, we have proposed an efficient adaptive multi-point stepsize gradient method MPSG for minimizing quadratics and extended it to solve unconstrained optimization by slightly modifying the stepsizes. The R-linear convergence of MPSG for strictly quadratics has been established whereas its global convergence for general unconstrained optimization is ensured by the nonmonotone line search. The proposed MPSG method was further generalized for solving an unconstrained model of extreme eigenvalue problems. Numerical comparisons of our proposed MPSG method and other successful gradient methods were presented.

Notice that both the two Dai-Fletcher stepsizes α_k^{DF1} and α_k^{DF2} treat the recent m iterations equally. However, the information carried by those iterations is updated along the iterative process. So, it is useful to treat the iterations differently. To this end, we may use the following weighted Dai-Fletcher long stepsize

$$\alpha_{k+1}^{WDF1} = \arg\min_{\alpha} \|W_k(\alpha^{-1}\tilde{s}_k - \tilde{y}_k)\| = \frac{\tilde{s}_k^T W_k \tilde{s}_k}{\tilde{s}_k^T W_k \tilde{y}_k} = \frac{\sum_{i=0}^{m-1} w_k^{(i+1)} s_{k-i}^T s_{k-i}}{\sum_{i=0}^{m-1} w_k^{(i+1)} s_{k-i}^T y_{k-i}}, \quad (30)$$

where W_k is a diagonal matrix given by

$$W_k = \mathrm{diag}\{w_k^{(1)}, w_k^{(2)}, \ldots, w_k^{(m)}\} \otimes I_n$$

with $w_k^{(1)}, w_k^{(2)}, \ldots, w_k^{(m)} \geq 0$, which can be viewed as weights for s_{k-i}, $i = 0, 1, \ldots m-1$. Obviously, α_{k+1}^{WDF1} reduces to α_{k+1}^{DF1} when $W_k = I$. The weighted Dai-

Fletcher short stepsize which generalizes α_{k+1}^{DF2} can be obtained by

$$\alpha_{k+1}^{WDF2} = \arg\min_{\alpha} \| W_k(\tilde{s}_k - \alpha \tilde{y}_k) \| = \frac{\tilde{s}_k^T W_k \tilde{y}_k}{\tilde{y}_k^T W_k \tilde{y}_k} = \frac{\sum_{i=0}^{m-1} w_k^{(i+1)} s_{k-i}^T y_{k-i}}{\sum_{i=0}^{m-1} w_k^{(i+1)} y_{k-i}^T y_{k-i}}. \quad (31)$$

Our preliminary results show the above weighted stepsizes lead to an efficient gradient method that comparable to MPSG. However, one remain question is how to choose weights for the m iterations. We will investigate this issue in our future work.

Acknowledgements This work was supported by the Natural Science Foundation of China (No. 11701137, 11991020, 11631013, 11971372, 11991021, 11671116) and Beijing Academy of Artificial Intelligence (BAAI). The authors are grateful to Associate Professor Bo Jiang of Nanjing Normal University for providing the codes of EigUncABB. They also thank the anonymous referees for their useful suggestions and comments.

References

1. Absil, P., Mahony, R., Sepulchre, R., Van Dooren, P.: A Grassmann–Rayleigh quotient iteration for computing invariant subspaces. SIAM Rev. 44(1), 57–73 (2002)
2. Akaike, H.: On a successive transformation of probability distribution and its application to the analysis of the optimum gradient method. Ann. Inst. Stat. Math. 11(1), 1–16 (1959)
3. Andrei, N.: An unconstrained optimization test functions collection. Adv. Model. Optim. 10(1), 147–161 (2008)
4. Barzilai, J., Borwein, J.M.: Two-point step size gradient methods. IMA J. Numer. Anal. 8(1), 141–148 (1988)
5. Birgin, E.G., Martínez, J.M., Raydan, M.: Nonmonotone spectral projected gradient methods on convex sets. SIAM J. Optim. 10(4), 1196–1211 (2000)
6. Birgin, E.G., Martínez, J.M., Raydan, M., et al.: Spectral projected gradient methods: review and perspectives. J. Stat. Softw. 60(3), 539–559 (2014)
7. Broyden, C.G.: A class of methods for solving nonlinear simultaneous equations. Math. Comp. 19(92), 577–593 (1965)
8. Cauchy, A.: Méthode générale pour la résolution des systemes déquations simultanées. Comp. Rend. Sci. Paris 25, 536–538 (1847)
9. Dai, Y.: On the nonmonotone line search. J. Optimz. Theory App. 112(2), 315–330 (2002)
10. Dai, Y.H.: Alternate step gradient method. Optimization 52(4-5), 395–415 (2003)
11. Dai, Y.H., Fletcher, R.: Projected Barzilai–Borwein methods for large-scale box-constrained quadratic programming. Numer. Math. 100(1), 21–47 (2005)
12. Dai, Y.H., Fletcher, R.: New algorithms for singly linearly constrained quadratic programs subject to lower and upper bounds. Math. Program. 106(3), 403–421 (2006)
13. Dai, Y.H., Hager, W.W., Schittkowski, K., Zhang, H.: The cyclic Barzilai–Borwein method for unconstrained optimization. IMA J. Numer. Anal. 26(3), 604–627 (2006)
14. Dai, Y.H., Huang, Y., Liu, X.W.: A family of spectral gradient methods for optimization. Comput. Optim. Appl. 74, 43–65 (2019)
15. Dai, Y.H., Liao, L.Z.: R-linear convergence of the Barzilai and Borwein gradient method. IMA J. Numer. Anal. 22(1), 1–10 (2002)
16. Dai, Y.H., Yuan, Y.X.: Analysis of monotone gradient methods. J. Ind. Mang. Optim. 1(2), 181–192 (2005)

17. Dai, Y.H., Zhang, H.: Adaptive two-point stepsize gradient algorithm. Numer. Alg. 27(4), 377–385 (2001)
18. Daspremont, A., Ghaoui, L.E., Jordan, M.I., Lanckriet, G.R.G.: A direct formulation for sparse PCA using semidefinite programming. SIAM Rev. 49(3), 434–448 (2007)
19. De Asmundis, R., Di Serafino, D., Hager, W.W., Toraldo, G., Zhang, H.: An efficient gradient method using the Yuan steplength. Comp. Optim. Appl. 59(3), 541–563 (2014)
20. Di Serafino, D., Ruggiero, V., Toraldo, G., Zanni, L.: On the steplength selection in gradient methods for unconstrained optimization. Appl. Math. Comput. 318, 176–195 (2018)
21. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Program. 91(2), 201–213 (2002)
22. Fletcher, R.: On the Barzilai–Borwein method. Optimization and Control with Applications pp. 235–256 (2005)
23. Frassoldati, G., Zanni, L., Zanghirati, G.: New adaptive stepsize selections in gradient methods. J. Ind. Mang. Optim. 4(2), 299–312 (2008)
24. Grippo, L., Lampariello, F., Lucidi, S.: A nonmonotone line search technique for Newton's method. SIAM Journal on Numerical Analysis 23(4), 707–716 (1986)
25. Horn, R.A., Johnson., C.: Topics in Matrix Analysis. Cambridge University Press, Cambridge (1991)
26. Hu, J., Jiang, B., Lin, L., Wen, Z., Yuan, Y.: Structured quasi-Newton methods for optimization with orthogonality constraints. SIAM J. Sci. Comput. 41(4), A2239–A2269 (2019)
27. Huang, Y., Liu, H.: Smoothing projected Barzilai–Borwein method for constrained non-Lipschitz optimization. Comp. Optim. Appl. 65(3), 671–698 (2016)
28. Huang, Y., Liu, H., Zhou, S.: Quadratic regularization projected Barzilai–Borwein method for nonnegative matrix factorization. Data Min. Knowl. Disc. 29(6), 1665–1684 (2015)
29. Jiang, B., Cui, C., Dai, Y.H.: Unconstrained optimization models for computing several extreme eigenpairs of real symmetric matrices. Pac. J. Optim. 10(1), 53–71 (2014)
30. Jiang, B., Dai, Y.H.: Feasible Barzilai–Borwein-like methods for extreme symmetric eigenvalue problems. Optim. Method Softw. 28(4), 756–784 (2013)
31. Liu, Y.F., Dai, Y.H., Luo, Z.Q.: Coordinated beamforming for miso interference channel: Complexity analysis and efficient algorithms. IEEE Trans. Signal Process. 59(3), 1142–1157 (2011)
32. Raydan, M.: On the Barzilai and Borwein choice of steplength for the gradient method. IMA J. Numer. Anal. 13(3), 321–326 (1993)
33. Raydan, M.: The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem. SIAM J. Optim. 7(1), 26–33 (1997)
34. Sameh, A., Tong, Z.: The trace minimization method for the symmetric generalized eigenvalue problem. J. Comput. Appl. Math. 123(1), 155–175 (2000)
35. Sameh, A., Wisniewski, J.A.: A trace minimization algorithm for the generalized eigenvalue problem. SIAM J. Numer. Anal. 19(6), 1243–1259 (1982)
36. Tan, C., Ma, S., Dai, Y.H., Qian, Y.: Barzilai–Borwein step size for stochastic gradient descent. In: Advances in Neural Information Processing Systems, pp. 685–693 (2016)
37. Wang, Y., Ma, S.: Projected Barzilai–Borwein method for large-scale nonnegative image restoration. Inverse Probl. Sci. En. 15(6), 559–583 (2007)
38. Wright, S.J., Nowak, R.D., Figueiredo, M.A.: Sparse reconstruction by separable approximation. IEEE Trans. Signal Process. 57(7), 2479–2493 (2009)
39. Yuan, Y.X.: A new stepsize for the steepest descent method. J. Comput. Math. 24(2), 149–156 (2006)
40. Yuan, Y.X.: Step-sizes for the gradient method. AMS IP Studies in Advanced Mathematics 42(2), 785–796 (2008)
41. Zhang, H., Hager, W.W.: A nonmonotone line search technique and its application to unconstrained optimization. SIAM J. Optim. 14(4), 1043–1056 (2004)
42. Zhou, B., Gao, L., Dai, Y.H.: Gradient methods with adaptive step-sizes. Comp. Optim. Appl. 35(1), 69–86 (2006)

A Julia Implementation of Algorithm NCL for Constrained Optimization

Ding Ma, Dominique Orban, and Michael A. Saunders

Abstract Algorithm NCL is designed for general smooth optimization problems where first and second derivatives are available, including problems whose constraints may not be linearly independent at a solution (i.e., do not satisfy the LICQ). It is equivalent to the LANCELOT augmented Lagrangian method, reformulated as a short sequence of nonlinearly constrained subproblems that can be solved efficiently by IPOPT and KNITRO, with warm starts on each subproblem. We give numerical results from a Julia implementation of Algorithm NCL on tax policy models that do not satisfy the LICQ, and on nonlinear least-squares problems and general problems from the CUTEst test set.

Keywords Constrained optimization · Second derivatives · Algorithm NCL · Julia

1 Introduction

Algorithm NCL (nonlinearly constrained augmented Lagrangian [15]) is designed for smooth, constrained optimization problems for which first and second derivatives are available. Without loss of generality, we take the problem to be

D. Ma
Department of Management Science and Department of Marketing, College of Business, City University of Hong Kong, Kowloon, Hong Kong
e-mail: dingma@cityu.edu.hk
URL: https://www.cb.cityu.edu.hk/staff/dingma

D. Orban (✉)
GERAD and Department of Mathematics and Industrial Engineering, Ecole Polytechnique de Montréal, Montreal, QC, Canada
e-mail: dominique.orban@gerad.ca
URL: https://dpo.github.io

M. A. Saunders
Systems Optimization Laboratory, Department of Management Science and Engineering, Stanford University, Stanford, CA, USA
e-mail: saunders@stanford.edu
URL: http://stanford.edu/~saunders

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*, Springer Proceedings in Mathematics & Statistics 354, https://doi.org/10.1007/978-3-030-72040-7_8

$$\text{NCO} \qquad \underset{x\in\mathbb{R}^n}{\text{minimize}} \;\; \phi(x) \qquad \text{subject to} \;\; c(x)=0, \quad \ell \le x \le u,$$

where $\phi(x)$ is a scalar objective function and $c(x) \in \mathbb{R}^m$ is a vector of linear or nonlinear constraints. Inequality constraints are accommodated by including slack variables within x. We take the primal and dual solutions to be (x^*, y^*, z^*). We denote the objective gradient by $g(x) = \nabla\phi(x) \in \mathbb{R}^n$, and the constraint Jacobian by $J(x) \in \mathbb{R}^{m\times n}$. The objective and constraint Hessians are $H_i(x) \in \mathbb{R}^{n\times n}$, $i = 0, 1, \dots, m$.

If $J(x^*)$ has full row rank m, problem NCO satisfies the linear independence constraint qualification (LICQ) at x^*. Most constrained optimization solvers have difficulty if NCO does *not* satisfy the LICQ. An exception is LANCELOT [4, 5, 13]. Algorithm NCL inherits this desirable property by being *equivalent* to the LANCELOT algorithm. Assuming first and second derivatives are available, Algorithm NCL may be viewed as an *efficient implementation* of the LANCELOT algorithm. Previously we have implemented Algorithm NCL in AMPL [1, 6, 15] for tax policy problems [11, 15] that could not otherwise be solved.[1] Here we describe our implementation in Julia [3] and give results on the tax problems and on a set of nonlinear least-squares and general problems from the CUTEst test set [9].

2 LANCELOT and NCL

For problem NCO, LANCELOT implements what we call a *BCL algorithm* (bound-constrained augmented Lagrangian algorithm), which solves a sequence of about 10 bound-constrained subproblems

$$\text{BC}_k \qquad \underset{x}{\text{minimize}} \;\; L(x, y_k, \rho_k) = \phi(x) - y_k^T c(x) + \tfrac{1}{2}\rho_k \|c(x)\|^2 \qquad \text{subject to} \;\; \ell \le x \le u$$

for $k = 0, 1, 2, \dots$, where y_k is an estimate of the dual variable associated with $c(x) = 0$, and $\rho_k > 0$ is a penalty parameter. Each subproblem BC$_k$ is solved (approximately) with a decreasing optimality tolerance ω_k, giving an iterate (x_k^*, z_k^*). If $\|c(x_k^*)\|$ is no larger than a decreasing feasibility tolerance η_k, the dual variable is updated to $y_{k+1} = y_k - \rho_k c(x_k^*)$. Otherwise, the penalty parameter is increased to $\rho_{k+1} > \rho_k$.

Optimality is declared if $c(x_k^*) \le \eta_k$ and η_k, ω_k have already been decreased to specified minimum values η^*, ω^*. Infeasibility is declared if $c(x_k^*) > \eta_k$ and ρ_k has already been increased to a specified maximum value ρ^*.

[1] Available from https://github.com/optimizers/ncl.

If n is large and not many bounds are active at x^*, the BC_k subproblems have *many degrees of freedom*, and LANCELOT must optimize in high-dimension subspaces. The subproblems are therefore computationally expensive. The algorithm in MINOS [16] (we call it an LCL algorithm) reduces this expense by including linearizations of the constraints within its subproblems:

$$\boxed{\begin{array}{lll} \mathrm{LC}_k & \underset{x}{\text{minimize}} & L(x, y_k, \rho_k) = \phi(x) - y_k^T c(x) + \frac{1}{2}\rho_k \|c(x)\|^2 \\ & \text{subject to} & c(x_{k-1}^*) + J(x_{k-1}^*)(x - x_{k-1}^*) = 0, \quad \ell \le x \le u. \end{array}}$$

The SQP algorithm in SNOPT [8] solves subproblems with the same linearized constraints and a quadratic approximation to the LC_k objective. Complications arise for both MINOS and SNOPT if the linearized constraints are infeasible.

Algorithm NCL proceeds in the opposite way by introducing additional variable $r \equiv -c(x)$ into subproblems LC_k to obtain the NCL subproblems

$$\boxed{\begin{array}{lll} \mathrm{NC}_k & \underset{x,\,r}{\text{minimize}} & \phi(x) + y_k^T r + \frac{1}{2}\rho_k \|r\|^2 \\ & \text{subject to} & c(x) + r = 0, \quad \ell \le x \le u. \end{array}}$$

These subproblems have *nonlinear constraints* and far more degrees of freedom than the original NCO! Indeed, the extra variables r make the subproblems more difficult if they are solved by MINOS and SNOPT. However, the subproblems satisfy the LICQ because of r. Also, interior solvers such as IPOPT [10] and KNITRO [12] find r helpful because at each *interior iteration* p they update the current primal-dual point (x_p, r_p, λ_p) by computing a search direction $(\Delta x, \Delta r, \Delta\lambda)$ from a linear system of the form

$$\begin{pmatrix} (H_p + D_p) & & J_p^T \\ & \rho_k I & I \\ J_p & I & \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta r \\ \Delta\lambda \end{pmatrix} = -\begin{pmatrix} g(x_p) + J_p^T\lambda_p - z_p + w_p \\ y_k + \rho_k r_p + \lambda_p \\ c(x_p) + r_p \end{pmatrix}, \tag{1}$$

where D_p, z_p and w_p are an ill-conditioned positive-definite diagonal matrix and two vectors arising from the interior method, and each Lagrangian Hessian $H_p = H_0(x_p) - \sum_i (y_k)_i H_i(x_p)$ may be altered to be more positive definite. Direct methods for solving each sparse system (1) are affected very little by the higher dimension caused by r, and they benefit significantly from $\begin{pmatrix} J_p & I \end{pmatrix}$ always having full row rank.

If an optimal solution for NC_k is $(x_k^*, r_k^*, y_k^*, z_k^*)$ and the feasibility and optimality tolerances have decreased to their minimum values η^* and ω^*, a natural stopping condition for Algorithm NCL is $\|r_k^*\|_\infty \le \eta^*$, because the major iterations drive r toward zero and we see that if $r = 0$, subproblem NC_k is equivalent to the original problem NCO.

We have found that Algorithm NCL is successful in practice because

– there are only about 10 major iterations ($k = 1, 2, \dots, 10$);
– the search-direction computation (1) for interior solvers is more stable than if the solvers are applied to NCO directly;
– IPOPT and KNITRO have run-time options that facilitate *warm starts* for each subproblem NC$_k$, $k > 1$.

3 Optimal Tax Policy Problems

The above observations were confirmed by our AMPL implementation of Algorithm NCL in solving some large problems modeling taxation policy [11, 15, 17]. The problems have very many nonlinear inequality constraints $c(x) \ge 0$ in relatively few variables. They have the form

$$\boxed{\begin{array}{llrl} \text{TAX} & \underset{c,\,y}{\text{maximize}} & & \sum_i \lambda_i U^i(c_i, y_i) \\ & \text{subject to} & U^i(c_i, y_i) - U^i(c_j, y_j) \ge 0 & \text{for all } i, j \\ & & \lambda^T(y - c) \ge 0 & \\ & & c,\ y \ge 0, & \end{array}}$$

where c_i and y_i are the consumption and income of taxpayer i, and λ is a vector of positive weights.[2] The utility functions $U^i(c_i, y_i)$ are each of the form

$$U(c, y) = \frac{(c - \alpha)^{1-1/\gamma}}{1 - 1/\gamma} - \psi \frac{(y/w)^{1/\eta+1}}{1/\eta + 1},$$

where w is the wage rate and α, γ, ψ and η are taxpayer heterogeneities. More precisely, the utility functions are of the form

$$U^{i,j,k,g,h}(c_{p,q,r,s,t}, y_{p,q,r,s,t}) = \frac{(c_{p,q,r,s,t} - \alpha_k)^{1-1/\gamma_h}}{1 - 1/\gamma_h} - \psi_g \frac{(y_{p,q,r,s,t}/w_i)^{1/\eta_j+1}}{1/\eta_j + 1},$$

where (i, j, k, g, h) and (p, q, r, s, t) run over na wage types, nb elasticities of labor supply, nc basic need types, nd levels of distaste for work, and ne elasticities of demand for consumption, with na, nb, nc, nd, ne determining the size of the problem, namely $m = T(T-1)$ nonlinear constraints, $n = 2T$ variables, with $T := na \times nb \times nc \times nd \times ne$.

To achieve reliability, we found it necessary to extend the AMPL model's definition of $U(c, y)$ to be a piecewise-continuous function that accommodates negative values of $(c - \alpha)$.

[2] In this section, (c, y, λ) refer to problem TAX, not the variables in Algorithm NCL.

Table 1 Run-time options for warm-starting IPOPT and KNITRO on subproblem NC_k

	IPOPT	KNITRO
$k = 1$		`algorithm=1`
$k \geq 2$	`warm_start_init_point=yes`	`bar_directinterval=0`
		`bar_initpt=2`
		`bar_murule=1`
$k = 2, 3$	`mu_init=1e-4`	`bar_initmu=1e-4`
		`bar_slackboundpush=1e-4`
$k = 4, 5$	`mu_init=1e-5`	`bar_initmu=1e-5`
		`bar_slackboundpush=1e-5`
$k = 6, 7$	`mu_init=1e-6`	`bar_initmu=1e-6`
		`bar_slackboundpush=1e-6`
$k = 8, 9$	`mu_init=1e-7`	`bar_initmu=1e-7`
		`bar_slackboundpush=1e-7`
$k \geq 10$	`mu_init=1e-8`	`bar_initmu=1e-8`
		`bar_slackboundpush=1e-8`

At a solution, a large proportion of the constraints are essentially active. The failure of LICQ causes numerical difficulties for MINOS, SNOPT, and IPOPT. LANCELOT is more able to find a solution, except it is very slow on each subproblem NC_k. For example, on the smallest problem of Table 2 with 32220 constraints and 360 variables, LANCELOT running on NEOS [18] timed-out at a near-optimal point on the 11th major iteration after 8 hours of CPU.

Note that when the constraints of NCO are inequalities $c(x) \geq 0$ as in problem TAX, the constraints of subproblem NC_k become inequalities $c(x) + r \geq 0$ (and similarly for mixtures of equalities and inequalities). The inequalities mean "more free variables" (more variables that are not on a bound). This increases the problem difficulty for MINOS and SNOPT, but has only a positive effect on the interior solvers.

In wishing to improve the efficiency of Algorithm NCL on larger tax problems, we found it possible to warm-start IPOPT and KNITRO on each NC_k subproblem ($k > 1$) by setting the run-time options shown in Table 1. These options were used by NCL/IPOPT and NCL/KNITRO to obtain the results in Table 2. We see that NCL/IPOPT performed significantly better than IPOPT itself, and similarly for NCL/KNITRO compared to KNITRO. The feasibility and optimality tolerances η_k, ω_k were fixed at $\eta^* = \omega^* = 1\mathrm{e}-6$ for all k. Our Julia implementation saves computation by starting with larger η_k, ω_k and reducing them toward η^*, ω^* as in LANCELOT.

Table 2 Solution of tax problems of increasing dimension using IPOPT and KNITRO on the original problem (cold starts) and the AMPL implementation of Algorithm NCL with IPOPT or KNITRO as subproblem solvers (warm-starting with the options in Table 1). The problem size increases with a problem parameter na. Other problem parameters are fixed at $nb = nc = 3$, $nd = ne = 2$. There are m nonlinear inequality constraints and n variables. For IPOPT, $>$ indicates optimality was not achieved

			IPOPT		KNITRO		NCL/IPOPT		NCL/KNITRO	
na	m	n	itns	Time	itns	Time	itns	Time	itns	Time
5	32220	360	449	217	168	53	322	146	339	63
9	104652	648	>98	>360	928	825	655	1023	307	239
11	156420	792	>87	>600	2769	4117	727	1679	383	420
17	373933	1224			2598	11447	1021	6347	486	1200
21	570780	1512					1761	17218	712	2880

4 Julia Implementation

Modeling languages such as AMPL and GAMS are domain-specific languages, as opposed to full-fledged, general-purpose programming languages like C or Java. In the terminology of Bentley [2], they are *little languages*. As such, they have understandable, yet very real, limitations that make it difficult, inefficient, and perhaps even impossible, to implement an algorithm such as Algorithm NCL in a sufficiently generic manner so that it may be applied to arbitrary problems. Indeed, our AMPL implementation of Algorithm NCL is specific to the optimal tax policy problems, and it would be difficult to generalize it to other problems. One of the main motivations for implementing Algorithm NCL in a language such as Julia is to be able to solve a greater variety of optimization problems.

We now describe the key features of our Julia implementation of Algorithm NCL and show that it solves examples of the same tax problems more efficiently. We then give results on a set of nonlinear least-squares problems from the CUTEst test set to indicate that Algorithm NCL is a reliable solver for such problems where first and second derivatives are available for the interior solvers used at each major iteration. To date, this means that Algorithm NCL is effective for optimization problems modeled in AMPL, GAMS, and CUTEst. (We have not made an implementation in GAMS [7], but it would be possible to build a major-iteration loop around calls to IPOPT or KNITRO in the way that we did for AMPL [15].)

4.1 Key Features

The main advantage of a Julia implementation over our original AMPL implementation is that we may take full advantage of our Julia software suite for optimization, hosted under the *JuliaSmoothOptimizers* (JSO) organization [22]. Our suite provides a general consistent API for solvers to interact with models by providing flexible data types to represent the objective and constraint functions, to evaluate their derivatives, to examine bounds on the variables, to add slack variables transparently, and to provide essentially any information that a solver might request from a model. Thanks to interfaces to modeling languages such as AMPL, CUTEst and JuMP [14], solvers in JSO may be written without regard for the language in which the model was written.

The modules from our suite that are particularly useful in the context of our implementation of Algorithm NCL are the following.

– NLPModels [24] is the main modeling package that defines the API on which solvers can rely to interact with models. Models are represented as instances of a data type deriving from the base type `AbstractNLPModel`, and solvers can evaluate the objective value by calling the `obj()` method, the gradient vector by calling the `grad()` method, and so forth. The main advantage of the consistent API provided by NLPModels is that solvers need not worry about the provenance of models. Other modules ensure communication between modeling languages such as AMPL, CUTEst or JuMP, and NLPModels.
– AmplNLReader [20] is one such module, and, as the name indicates, allows a solver written in Julia to interact with a model written in AMPL. The communication is made possible by the AMPL Solver Library (ASL),[3] which requires that the model be decoded as an `nl` file.
– NLPModelsIpopt [23] is a thin translation layer between the low-level Julia interface to IPOPT provided by the IPOPT.jl package[4] and NLPModels, and lets users solve any problem conforming to the NLPModels API with IPOPT.
– NLPModelsKnitro [25] is similar to NLPModelsIpopt, but lets users solve problems with KNITRO via the low-level interface provided by KNITRO.jl.[5]

Julia is a convenient language built on top of state-of-the-art infrastructure underlying modern compilers such as Clang. Julia may be used as an interactive language for exploratory work in a read-eval-print loop similar to Matlab. However, Julia functions are transparently translated to low-level code and compiled the first time they are called. The net result is compiled code whose efficiency rivals that of binaries generated from standard compiled languages such as C and Fortran. Though this last feature is not particularly important in the context of Algorithm NCL because the compiled solvers IPOPT and KNITRO perform all the work, it is paramount when implementing pure Julia optimization solvers.

[3] http://www.netlib.org/ampl/solvers.

[4] https://github.com/jump-dev/Ipopt.jl.

[5] https://github.com/jump-dev/KNITRO.jl.

4.2 Implementation and Solver Features

The Julia implementation of Algorithm NCL, named NCL.jl [19], is in two parts. The first part defines a data type `NCLModel` that derives from the basic data type `AbstractNLPModel` mentioned earlier and represents subproblem NC$_k$. An `NCLModel` is a wrapper around the underlying problem NCO in which the current values of ρ_k and r can be updated efficiently. The second part is the solver itself, each iteration of which consists of a call to IPOPT or KNITRO, and parameter updates. The solver takes an `NCLModel` as input. If the input problem is not an `NCLModel`, it is first converted into one. Parameters are initialized as

$$\eta_0 = 10, \quad \omega_0 = 10, \quad \rho_0 = 100, \quad \mu_0 = 0.1,$$

where μ_0 is the initial barrier parameter for IPOPT or KNITRO. The initial values of x are those defined in the underlying model if any, or zero otherwise. We initialize r to zero and y to the vector of ones. When the subproblem solver returns with NC$_k$ solution $(x_k^*, r_k^*, y_k^*, z_k^*)$, we check whether $\|r_k^*\| \le \max(\eta_k, \eta_*)$. If so, we decide that good progress has been made toward feasibility and update

$$y_{k+1} = y_k - \rho_k r_k^*, \quad \eta_{k+1} = \eta_k/10, \quad \omega_{k+1} = \omega_k/10, \quad \rho_{k+1} = \rho_k,$$

where this definition of y_{k+1} is the first-order update of the multipliers. Otherwise, we keep most things the same but increase the penalty parameter:

$$y_{k+1} = y_k, \quad \eta_{k+1} = \eta_k, \quad \omega_{k+1} = \omega_k, \quad \rho_{k+1} = \min(10\rho_k, \rho^*),$$

where $\rho^* > 0$ is the threshold beyond which the user is alerted that the problem may be infeasible. In our implementation, we use $\rho^* = 10^{12}$.

Note that updating the multipliers based on $\|r_k^*\|$ instead of $\|c(x_k^*)\|$ is a departure from the classical augmented-Lagrangian update. From the optimality conditions for NC$_k$ we can prove that the first-order update is equivalent to choosing $y_{k+1} = y_k^*$ when NC$_k$ is solved accurately. We still have a choice between the two updates because we use low accuracy for the early NC$_k$. We could also "trim" y_{k+1} (i.e., for inequality constraints $c_i(x) + r_i \ge 0$ or ≤ 0, set components of y_{k+1} with non-optimal sign to zero). These are topics for future research.

With IPOPT as subproblem solver, we warm-start subproblem NC$_{k+1}$ with the options in Table 1 and (y_k^*, z_k^*) as initial values for the Lagrange multipliers. With KNITRO as subproblem solver, (y_k^*, z_k^*) as starting point did not help or harm KNITRO significantly. We allowed KNITRO to determine its own initial multipliers, and it proved to be significantly more reliable than IPOPT in solving the NC$_k$ subproblems for the optimal tax policy problems. In the next sections, Algorithm NCL means our Julia implementation with KNITRO as subproblem solver.

4.3 Results with Julia/NCL on the Tax Policy Problems

AMPL models of the optimal tax policy problems were input to the Julia implementation of Algorithm NCL. The notation 1D, 2D, 3D, 4D, 5D refers to problem parameters na, nb, nc, nd, ne that define the utility function appearing in the objective and constraints. The subproblem solver was KNITRO 12 [12].

Tables 3, 4, 5, 6 and 7 illustrate that, as with our AMPL implementation of Algorithm NCL, about 10 major iterations are needed independent of the problem size. (The problems have increasing numbers of variables and greatly increasing numbers of nonlinear inequality constraints.) In each iteration log,

`outer` and `inner` refer to the NCL major iteration number k and the total number of KNITRO iterations for subproblems NC_k;

`NCL obj` is the augmented Lagrangian objective value, which converges to the objective value for the model;

Table 3 Tax1D problem with realistic data. NCL with KNITRO solving subproblems

```
julia> using NCL

julia> using AmplNLReader

julia> tax1D = AmplModel("data/tax1D")
Maximization problem data/tax1D
nvar = 24, ncon = 133 (1 linear)

julia> NCLSolve(tax1D, outlev=0)

outer inner   NCL obj      ||r||         η     ||∇L||         ω         ρ   μ init     ||y||      ||x|| time
    1     5 -8.00e+02 9.7e-02 1.0e-02 7.6e-03 1.0e-02 1.0e+02 1.0e-01 1.0e+00 2.0e+02 0.13
    2    12 -7.89e+02 4.2e-02 1.0e-02 4.3e-03 1.0e-02 1.0e+03 1.0e-03 1.0e+00 1.9e+02 0.00
    3     7 -7.83e+02 5.7e-03 1.0e-02 1.0e-03 1.0e-02 1.0e+04 1.0e-03 1.0e+00 1.9e+02 0.00
    4     3 -7.82e+02 1.3e-04 1.0e-03 1.0e-05 1.0e-03 1.0e+04 1.0e-05 5.8e+01 1.9e+02 0.00
    5     2 -7.82e+02 2.3e-06 1.0e-04 1.0e-05 1.0e-04 1.0e+04 1.0e-05 5.9e+01 1.9e+02 0.00
    6     2 -7.82e+02 9.3e-08 1.0e-05 1.0e-06 1.0e-05 1.0e+04 1.0e-06 5.9e+01 1.9e+02 0.00
    7     2 -7.82e+02 7.7e-09 1.0e-06 1.0e-08 1.0e-06 1.0e+04 1.0e-06 5.9e+01 1.9e+02 0.00
```

Table 4 Tax2D problem. NCL with KNITRO solving subproblems

```
julia> tax2D = AmplModel("data/tax2D")
Maximization problem data/tax2D
nvar = 120, ncon = 3541 (1 linear)

julia> NCLSolve(tax2D, outlev=0)
outer inner   NCL obj      ||r||         η     ||∇L||         ω         ρ   μ init     ||y||      ||x|| time
    1    16 -4.35e+03 6.1e-02 1.0e-02 4.2e-03 1.0e-02 1.0e+02 1.0e-01 1.0e+00 4.0e+02 0.15
    2    15 -4.31e+03 2.5e-02 1.0e-02 2.7e-04 1.0e-02 1.0e+03 1.0e-03 1.0e+00 4.0e+02 0.13
    3    16 -4.29e+03 7.8e-03 1.0e-02 3.5e-04 1.0e-02 1.0e+04 1.0e-03 1.0e+00 4.0e+02 0.16
    4    15 -4.28e+03 5.1e-03 1.0e-03 1.0e-05 1.0e-03 1.0e+04 1.0e-05 7.9e+01 4.0e+02 0.14
    5    32 -4.28e+03 1.2e-03 1.0e-03 1.0e-05 1.0e-03 1.0e+05 1.0e-05 7.9e+01 4.0e+02 0.32
    6    12 -4.28e+03 1.5e-04 1.0e-03 1.5e-05 1.0e-03 1.0e+06 1.0e-06 7.9e+01 4.0e+02 0.15
    7     4 -4.28e+03 1.8e-05 1.0e-04 2.7e-06 1.0e-04 1.0e+06 1.0e-06 2.0e+02 4.0e+02 0.06
    8     4 -4.28e+03 1.2e-06 1.0e-05 1.3e-07 1.0e-05 1.0e+06 1.0e-07 2.0e+02 4.0e+02 0.05
    9     3 -4.28e+03 3.5e-07 1.0e-06 1.0e-07 1.0e-06 1.0e+06 1.0e-07 2.0e+02 4.0e+02 0.05
```

Table 5 Tax3D problem. NCL with KNITRO solving subproblems

```
julia> pTax3D = AmplModel("data/pTax3D")
Maximization problem data/pTax3D
nvar = 216, ncon = 11557 (1 linear)

julia> NCLSolve(pTax3D, outlev=0)
outer inner   NCL obj     ||r||       η     ||∇L||       ω        ρ  μ init     ||y||     ||x|| time
    1     9 -6.97e+03 4.5e-02 1.0e-02 9.1e-03 1.0e-02 1.0e+02 1.0e-01 1.0e+00 5.7e+02 0.54
    2    18 -6.87e+03 1.7e-02 1.0e-02 2.4e-04 1.0e-02 1.0e+03 1.0e-03 1.0e+00 5.6e+02 0.99
    3    16 -6.83e+03 7.8e-03 1.0e-02 1.7e-03 1.0e-02 1.0e+04 1.0e-03 1.0e+00 5.7e+02 1.01
    4    17 -6.81e+03 5.2e-03 1.0e-03 1.5e-05 1.0e-03 1.0e+04 1.0e-05 7.9e+01 5.6e+02 0.99
    5    54 -6.80e+03 2.6e-03 1.0e-03 1.2e-05 1.0e-03 1.0e+05 1.0e-05 7.9e+01 5.6e+02 3.15
    6    22 -6.80e+03 4.5e-04 1.0e-03 8.0e-05 1.0e-03 1.0e+06 1.0e-06 7.9e+01 5.6e+02 1.30
    7     9 -6.80e+03 1.1e-04 1.0e-04 1.0e-06 1.0e-04 1.0e+06 1.0e-06 5.2e+02 5.6e+02 0.56
    8     8 -6.80e+03 1.1e-05 1.0e-04 1.1e-07 1.0e-04 1.0e+07 1.0e-07 5.2e+02 5.6e+02 0.49
    9     5 -6.80e+03 1.1e-06 1.0e-05 1.0e-07 1.0e-05 1.0e+07 1.0e-07 5.3e+02 5.6e+02 0.32
   10     3 -6.80e+03 8.9e-08 1.0e-06 1.0e-08 1.0e-06 1.0e+07 1.0e-08 5.3e+02 5.6e+02 0.22
```

Table 6 Tax4D problem. NCL with KNITRO solving subproblems

```
julia> pTax4D = AmplModel("data/pTax4D")
Minimization problem data/pTax4D
nvar = 432, ncon = 46441 (1 linear)

julia> NCLSolve(pTax4D, outlev=0)
outer inner   NCL obj     ||r||       η     ||∇L||       ω        ρ  μ init     ||y||     ||x||  time
    1    12 -1.34e+04 3.3e-02 1.0e-02 5.4e-03 1.0e-02 1.0e+02 1.0e-01 1.0e+00 7.2e+02  3.38
    2    12 -1.31e+04 1.3e-02 1.0e-02 4.0e-03 1.0e-02 1.0e+03 1.0e-03 1.0e+00 7.2e+02  3.23
    3    15 -1.30e+04 5.1e-03 1.0e-02 2.1e-04 1.0e-02 1.0e+04 1.0e-03 1.0e+00 7.1e+02  3.86
    4    31 -1.30e+04 3.2e-03 1.0e-03 1.3e-05 1.0e-03 1.0e+04 1.0e-05 5.2e+01 7.0e+02  7.95
    5    37 -1.30e+04 1.8e-03 1.0e-03 1.2e-05 1.0e-03 1.0e+05 1.0e-05 5.2e+01 7.0e+02  9.89
    6    44 -1.29e+04 5.0e-04 1.0e-03 1.1e-06 1.0e-03 1.0e+06 1.0e-06 5.2e+01 7.0e+02 11.93
    7    16 -1.29e+04 2.6e-04 1.0e-04 1.2e-05 1.0e-04 1.0e+06 1.0e-06 5.3e+02 7.0e+02  3.74
    8    30 -1.29e+04 4.4e-05 1.0e-04 1.2e-07 1.0e-04 1.0e+07 1.0e-07 5.3e+02 7.0e+02  8.15
    9     9 -1.29e+04 2.3e-05 1.0e-05 1.2e-07 1.0e-05 1.0e+07 1.0e-07 8.2e+02 7.0e+02  2.49
   10    11 -1.29e+04 3.8e-06 1.0e-05 1.0e-08 1.0e-05 1.0e+08 1.0e-08 8.2e+02 7.0e+02  3.09
   11     6 -1.29e+04 1.7e-07 1.0e-06 1.3e-08 1.0e-06 1.0e+08 1.0e-08 9.4e+02 7.0e+02  1.74
```

Table 7 Tax5D problem. NCL with KNITRO solving subproblems

```
julia> pTax5D = AmplModel("data/pTax5D")
Minimization problem data/pTax5D
nvar = 864, ncon = 186193 (1 linear)

julia> NCLSolve(pTax5D, outlev=0)
outer inner   NCL obj     ||r||       η     ||∇L||       ω        ρ  μ init     ||y||     ||x||   time
    1    64 -1.76e+05 2.0e-01 1.0e-02 2.3e-03 1.0e-02 1.0e+02 1.0e-01 1.0e+00 1.1e+04  80.43
    2    29 -1.74e+05 4.9e-02 1.0e-02 1.2e-03 1.0e-02 1.0e+03 1.0e-03 1.0e+00 1.1e+04  35.02
    3    23 -1.74e+05 1.6e-02 1.0e-02 1.0e-03 1.0e-02 1.0e+04 1.0e-03 1.0e+00 1.1e+04  28.96
    4    46 -1.74e+05 4.1e-03 1.0e-02 3.6e-05 1.0e-02 1.0e+05 1.0e-05 1.0e+00 1.1e+04  54.50
    5    41 -1.74e+05 2.8e-03 1.0e-03 1.7e-05 1.0e-03 1.0e+05 1.0e-05 4.1e+02 1.1e+04  52.72
    6    28 -1.74e+05 6.1e-04 1.0e-03 1.0e-06 1.0e-03 1.0e+06 1.0e-06 4.1e+02 1.1e+04  34.38
    7    13 -1.74e+05 2.1e-04 1.0e-04 1.4e-06 1.0e-04 1.0e+06 1.0e-06 1.0e+03 1.1e+04  14.81
    8    12 -1.74e+05 5.3e-05 1.0e-04 1.2e-07 1.0e-04 1.0e+07 1.0e-07 1.0e+03 1.1e+04  14.80
    9     7 -1.74e+05 4.5e-06 1.0e-05 1.0e-07 1.0e-05 1.0e+07 1.0e-07 1.0e+03 1.1e+04   9.49
   10     5 -1.74e+05 8.0e-07 1.0e-06 1.2e-08 1.0e-06 1.0e+07 1.0e-08 1.0e+03 1.1e+04   7.02
```

η and ω show the KNITRO feasibility and optimality tolerances η_k and ω_k decreasing from 10^{-2} to 10^{-6};

$\|\nabla L\|$ is the size of the augmented Lagrangian gradient, namely $\|g(x_k^*) - J(x_k^*)^T y_{k+1}\|$ (a measure of the dual infeasibility at the end of major iteration k);

ρ is the penalty parameter ρ_k;

μ `init` is the initial value of KNITRO's barrier parameter;

$\|x\|$ is the size of the primal variable x_k^* at the (approximate) solution of NC$_k$;

$\|y\|$ is the size of the corresponding dual variable y_k^*;

`time` is the number of seconds to solve NC$_k$.

We see from the decreasing `inner` iteration counts that KNITRO was able to warm-start each subproblem, and from the decreasing $\|r\|$ and $\|\nabla L\|$ values that it is sufficient to solve the early subproblems with low (but steadily increasing) accuracy.

4.4 Results with Julia/NCL on CUTEst Test Set

Our Julia module CUTEst.jl [21] provides an interface with the CUTEst [9] environment and problem collection. Its main feature is to let users instantiate problems from CUTEst using the `CUTEstModel` constructor so they can be manipulated transparently or passed to a solver like any other `NLPModel`.

On a set of 166 constrained problems with at least 100 variables whose constraints are all nonlinear, KNITRO solves 147 and NCL solves 126. Although our simple implementation of NCL is not competitive with plain KNITRO in general, it does solve a few problems on which KNITRO fails. Those are summarized in Tables 8 and 9. The above results suggest that NCL's strength might reside in solving difficult problems (rather than being the fastest), and that more research is needed to improve its efficiency.

5 Nonlinear Least Squares

An important class of problems worthy of special attention is *nonlinear least-squares (NLS) problems* of the form

$$\min_x \tfrac{1}{2}\|c(x)\|^2 \text{ subject to } \ell \le x \le u, \tag{2}$$

where the Jacobian of $c(x)$ is again $J(x)$, and the bounds are often empty. Such problems are not immediately meaningful to Algorithm NCL, but if they are presented in the (probably infeasible) form

Table 8 KNITRO results on CUTEst constrained problems (a subset that failed)

Name	nvar	ncon	f	$\|\nabla L\|_2$	$\|c\|_2$	t	iter	$\#f$	$\#\nabla f$	$\#c$	$\#\nabla c$	$\#\nabla^2 L$	Status
CATENARY	3003	1000	−2.01e+10	4.9e+00	2.0e+09	18.50	2000	7835	2002	7835	2002	2000	Max_iter
COSHFUN	6001	2000	−9.82e+17	5.0e−01	0.0e+00	19.40	2000	2001	2001	2001	2001	2000	Max_iter
DRCAVTY1	4489	3969	0.00e+00	0.0e+00	2.2e−03	456.00	2000	9191	2002	9191	2002	2000	Max_iter
EG3	10001	20000	5.11e+05	2.0e+03	3.3e−01	7.18	51	55	52	55	52	52	Infeasible
JUNKTURN	10010	7000	1.78e−03	1.0e−02	6.4e−07	123.00	1913	15051	1915	15051	1915	1914	Unknown
LUKVLE11	9998	6664	5.12e+04	5.1e+02	4.1e−01	86.50	2000	6945	2001	6945	2001	2000	Max_iter
LUKVLE17	9997	7497	3.22e+04	1.8e−02	9.9e−07	47.00	2000	3190	2001	3190	2001	2000	Max_iter
LUKVLE18	9997	7497	1.12e+04	4.0e+01	2.5e−08	83.90	2000	4190	2001	4190	2001	2000	Max_iter
ORTHRDS2	5003	2500	7.62e+02	3.7e−01	5.0e−13	0.70	42	92	43	92	43	43	Unknown

Table 9 NCL results on the same problems (all successful)

Name	nvar	ncon	f	$\|\nabla L\|_2$	$\|c\|_2$	t	iter	$\#f$	$\#\nabla f$	$\#c$	$\#\nabla c$	$\#\nabla^2 L$	Status
CATENARY	3003	1000	−2.10e+06	1.54e−09	1.00e−07	1.76	183	430	195	430	206	194	First_order
COSHFUN	6001	2000	−7.81e−01	9.02e−07	1.00e−07	6.46	328	1712	337	1712	346	337	First_order
DRCAVTY1	4489	3969	0.00e+00	1.19e−08	1.00e−06	29.30	222	344	233	344	243	232	First_order
EG3	10001	20000	1.94e−07	1.00e−08	1.00e−07	4.94	37	47	47	47	57	47	First_order
JUNKTURN	10010	7000	9.94e−06	6.79e−07	1.00e−07	5.29	108	131	119	131	129	118	First_order
LUKVLE11	9998	6664	9.32e+02	4.57e−08	1.00e−06	1.86	37	63	47	63	57	47	First_order
LUKVLE17	9997	7497	3.24e+04	1.59e−08	1.00e−07	2.45	60	94	78	94	96	78	First_order
LUKVLE18	9997	7497	1.10e+04	2.00e−12	1.00e−09	3.43	60	81	79	81	98	79	First_order
ORTHRDS2	5003	2500	7.62e+02	9.96e−08	1.00e−07	1.23	47	63	62	63	77	62	First_order

$$\min_x \; 0 \text{ subject to } c(x) = 0, \quad \ell \le x \le u, \tag{3}$$

the first NCL subproblem will be

$$\boxed{\begin{array}{lll} \text{NC}_0 & \underset{x,r}{\text{minimize}} & y_0^T r + \tfrac{1}{2}\rho_0 \|r\|^2 \\ & \text{subject to} & c(x) + r = 0, \quad \ell \le x \le u, \end{array}}$$

which is well suited to KNITRO and is equivalent to (2) if $y_0 = 0$ and $\rho_0 > 0$. If we treat NLS problems as a special case, we can set $y_0 = 0$, $\rho_0 = 1$, $\eta_0 = \eta^*$, $\omega_0 = \omega^*$ and obtain an optimal solution in one NCL iteration. In this sense, Algorithm NCL is ideally suited to NLS problems (2).

The CUTEst collection features a number of NLS problems in both forms (2) and (3). While formulation (2) allows evaluation of the objective gradient $J(x)^T c(x)$, it does not give access to $J(x)$ itself. In contrast, a problem modeled as (3) allows solvers to access $J(x)$ directly.

The NLPModels modeling package allows us to formulate (2) from a problem given as (3) and fulfill requests for $J(x)$ in (2) by returning the constraint Jacobian of (3). Alternatively, problem NC$_0$ is easily created by the `NCLModel` constructor. The construction of both models is illustrated in Listing 1.1. Once a problem in the form (2) has been simulated in this way, it can be passed to KNITRO's nonlinear least-squares solver, which is a variant of the Levenberg-Marquardt method in which bound constraints are treated via an interior-point method.

Listing 1.1 Formulating (2) from (3)

```
julia> using CUTEst
julia> model = CUTEstModel("ARWHDNE")          #
problem in the form (*@(3)@*)
julia> nls_model = FeasibilityResidual(model)  # interpretation of
(*@(3)@*) as representing (*@(2)@*)
julia> knitro(nls_model)  # NLPModelsKnitro calls
KNITRO/Levenberg-Marquardt
julia> ncl_model = NCLModel(model,
y=zeros(model.meta.ncon), ρ=1.0)  # problem NC₀
julia> knitro(ncl_model)  # NLPModelsKnitro calls standard KNITRO
```

We identified 127 problems in the form (3) in CUTEst. We solve each problem in two ways:

Solver `knitro_nls` applies KNITRO's nonlinear least-squares method to (2).
Solver `ncl_nls` uses KNITRO to perform a single NCL iteration on NC$_0$.

In both cases, KNITRO is given a maximum of 500 iterations and 30 minutes of CPU time. Optimality and feasibility tolerances are set to 10^{-6}.

`knitro_nls` solved 101 problems to optimality, reached the iteration limit in 19 cases and the time limit in 3 cases, and failed for another reason in 4 cases. `ncl_nls` solved 119 problems to optimality, reached the iteration limit in 3 cases and the time limit in 3 cases, and failed for another reason in 2 cases.

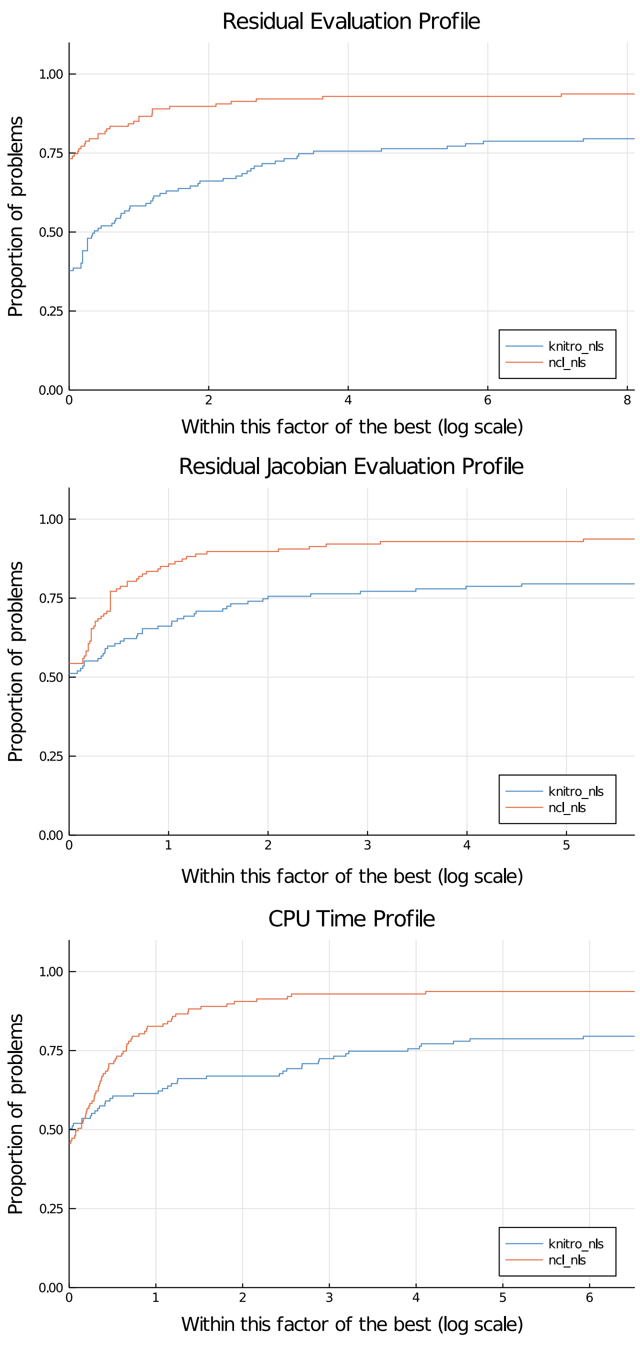


Fig. 1 Performance profiles comparing `knitro_nls` (KNITRO's NLS solver applied to (2)) and `ncl_nls` (KNITRO solving NC_0) on 127 nonlinear least squares problems from CUTEst. `ncl_nls` is more efficient

Figure 1 shows Dolan-Moré performance profiles comparing the two solvers. The top and middle plots use the number of residual and residual Jacobian evaluations as metric, which, in the case of (3), corresponds to the number of constraint and constraint Jacobian evaluations. The bottom plot uses time as metric. `ncl_nls` outperforms `knitro_nls` in all three measures and appears substantially more robust. It is important to keep in mind that a key difference between the two algorithms is that `ncl_nls` uses second-order information, and therefore performs Hessian evaluations. Nevertheless, those evaluations are not so costly as to put NCL at a disadvantage in terms of run-time. For reference, Tables 10 and 11 give the detailed results.

6 Summary

Our AMPL implementation of the tax policy models and Algorithm NCL has been the only way we could handle these particular problems reliably [15], with KNITRO solving each subproblem accurately. Our Julia implementation of NCL achieves greater efficiency on these AMPL models by gradually tightening the KNITRO feasibility and optimality tolerances. It also permits testing on a broad range of problems, as illustrated on nonlinear least-squares problems and other problems from the CUTEst test set. We believe Algorithm NCL could become an effective general-purpose optimization solver when first and second derivatives are available. It is especially useful when the LICQ is not satisfied at the solution. The current Julia implementation of NCL (with KNITRO as subproblem solver) is not quite competitive with KNITRO itself on the general CUTEst problems in terms of run-time or number of evaluations, but it does solve some problems on which KNITRO fails. An advantage is that the implementation is generic and may be applied to problems from any collection adhering to the interface of the NLPModels.jl package [24].

7 Detailed Results for Julia/NCL on NLS Problems

Table 10 reports the detailed results of KNITRO/Levenberg-Marquardt on problems of the form (2) using the modeling mechanism of Sect. 5. In the table headers, “nvar” is the number of variables, “ncon” is the number of constraints (i.e., the number of least-squares residuals), f is the final objective value, $\|\nabla L\|_2$ is the final dual residual, t is the run-time in seconds, “iter” is the number of iterations, “#c” is the number of constraint (i.e, residual) evaluations, “#∇c” is the number of constraint (i.e., residual) Jacobian evaluations, and “status” is the final solver status.

Table 11 reports the results of Julia/NCL solving Problem NC$_0$ for the same models. In the interest of space, the second table does not repeat problem dimensions. The other columns are as follows: $\|c\|_2$ is the final primal feasibility, and #$\nabla^2 L$ is the number of Hessian evaluations.

Table 10 `knitro_nls` results on 127 CUTEst nonlinear least-squares problems. 101 problems were solved successfully

Name	nvar	ncon	f	$\|\nabla L\|_2$	t	iter	$\#c$	$\#\nabla c$	Status
ARWHDNE	500	998	6.971e+01	8.5e−06	0.46	22	167	23	First_order
BA-L1	57	12	1.204e−25	3.8e−10	0.00	5	6	6	First_order
BA-L16	66462	167436	4.243e+05	4.2e−04	69.92	27	30	28	First_order
BA-L1SP	57	12	1.620e−23	5.1e−09	0.01	5	6	6	First_order
BA-L21	34134	72910	1.975e+05	7.2e−05	129.28	119	160	120	First_order
BA-L49	23769	63686	1.241e+05	4.0e+05	1809.44	121	728	122	Max_time
BA-L52	192627	694346	2.147e+06	3.9e+07	1808.91	37	200	38	Max_time
BA-L73	33753	92244	6.312e+05	9.5e+06	1801.24	396	2312	397	Max_time
BARDNE	3	15	4.107e−03	2.7e−09	0.00	5	6	6	First_order
BDQRTICNE	5000	9992	1.000e+04	1.4e−01	91.67	500	3668	501	Max_iter
BEALENE	2	3	1.491e−25	2.6e−12	0.00	6	8	7	First_order
BIGGS6NE	6	13	2.367e−17	5.9e−09	0.01	30	124	31	First_order
BOX3NE	3	10	3.259e−19	3.9e−10	0.00	5	6	6	First_order
BROWNBSNE	2	3	0.000e+00	0.0e+00	0.00	12	53	13	First_order
BROWNDENE	4	20	4.291e+04	2.4e+00	0.09	500	1697	501	Max_iter
BRYBNDNE	5000	5000	1.499e−21	8.2e−11	1.16	6	7	7	First_order
CHAINWOONE	4000	11994	4.699e+03	5.9e+01	63.96	500	1339	501	Max_iter
CHEBYQADNE	100	100	4.749e−03	1.5e−04	7.53	500	1994	501	Max_iter
CHNRSBNE	50	98	7.394e−18	3.2e−08	0.01	38	78	39	First_order
CHNRSNBMNE	50	98	1.452e−20	1.8e−09	0.02	54	132	55	First_order
COATINGNE	134	252	2.527e−01	5.5e−07	0.01	9	11	10	First_order
CUBENE	2	2	1.085e−26	1.1e−13	0.00	4	9	5	First_order

(continued)

Table 10 (continued)

Name	nvar	ncon	f	$\|\nabla L\|_2$	t	iter	$\#c$	$\#\nabla c$	Status
DECONVBNE	63	40	4.542e−10	6.0e−07	0.03	54	214	55	First_order
DECONVNE	63	40	2.245e−16	8.5e−10	0.00	2	3	3	First_order
DENSCHNBNE	2	3	8.573e−27	1.9e−13	0.00	5	6	6	First_order
DENSCHNCNE	2	2	2.838e−22	8.4e−11	0.00	7	8	8	First_order
DENSCHNDNE	3	3	8.307e−10	5.5e−07	0.00	17	18	18	First_order
DENSCHNENE	3	3	3.005e−22	2.4e−11	0.00	8	19	9	First_order
DENSCHNFNE	2	2	1.392e−26	2.1e−12	0.00	5	6	6	First_order
DEVGLA1NE	4	24	1.063e−13	2.4e−08	0.00	11	31	12	First_order
DEVGLA2NE	5	16	4.828e−15	5.9e−07	0.00	9	16	10	First_order
EGGCRATENE	2	4	4.744e+00	9.6e−07	0.00	5	6	6	First_order
ELATVIDUNE	2	3	2.738e+01	3.2e−06	0.00	15	30	16	First_order
ENGVAL2NE	3	5	2.465e−32	2.2e−16	0.00	10	14	11	First_order
ERRINROSNE	50	98	2.020e+01	1.8e−05	0.01	45	63	46	First_order
ERRINRSMNE	50	98	1.926e+01	1.7e−05	0.01	44	66	45	First_order
EXP2NE	2	10	1.618e−19	8.7e−12	0.00	5	6	6	First_order
EXPFITNE	2	10	1.203e−01	4.9e−07	0.00	10	12	11	First_order
EXTROSNBNE	1000	999	8.071e−31	1.3e−14	0.07	6	14	7	First_order
FBRAIN2NE	4	2211	1.842e−01	6.8e−07	0.08	8	11	9	First_order
FBRAINNE	2	2211	2.083e−01	5.7e−07	0.02	5	6	6	First_order
FREURONE	2	2	2.449e+01	1.8e−05	0.01	19	102	20	First_order
GENROSEBNE	500	998	7.965e+02	3.6e−06	0.04	8	10	10	First_order
GENROSENE	1000	1999	1.247e+02	6.3e+00	3.40	500	1567	501	Max_iter
GULFNE	3	99	2.107e−01	4.0e+06	0.27	500	2578	501	Max_iter

(continued)

Table 10 (continued)

Name	nvar	ncon	f	$\|\nabla L\|_2$	t	iter	$\#c$	$\#\nabla c$	Status
HATFLDANE	4	4	2.172e−17	3.4e−09	0.00	8	10	9	First_order
HATFLDBNE	4	4	2.786e−03	1.4e−07	0.00	7	8	8	First_order
HATFLDCNE	25	25	1.018e−15	1.5e−08	0.00	3	4	4	First_order
HATFLDDNE	3	10	1.273e−07	7.1e−11	0.00	6	9	7	First_order
HATFLDENE	3	21	1.364e−06	8.2e−12	0.00	5	6	6	First_order
HATFLDFLNE	3	3	3.254e−05	9.1e−07	0.01	11	69	12	First_order
HELIXNE	3	3	2.195e−20	2.8e−09	0.00	9	11	10	First_order
HIMMELBFNE	4	7	1.593e+06	4.4e−04	0.01	28	66	29	First_order
HS1NE	2	2	3.274e−17	4.0e−09	0.00	9	20	10	First_order
HS25NE	3	99	1.642e+01	9.2e−09	0.00	0	1	1	First_order
HS2NE	2	2	2.471e+00	2.1e−08	0.00	6	8	8	First_order
INTEQNE	12	12	6.938e−14	2.1e−07	0.00	2	3	3	First_order
JENSMPNE	2	10	6.218e+01	7.1e−06	0.01	10	82	11	First_order
JUDGENE	2	20	8.041e+00	1.2e−06	0.00	9	10	10	First_order
KOEBHELBNE	3	156	3.876e+01	7.5e−07	0.10	194	757	195	First_order
KOWOSBNE	4	11	1.539e−04	6.9e−07	0.00	16	30	17	First_order
LIARWHDNE	5000	10000	1.608e−27	4.4e−12	1.07	5	6	6	First_order
LINVERSENE	1999	2997	3.405e+02	1.2e−02	13.28	500	1753	502	Max_iter
MANCINONE	100	100	7.966e−22	2.2e−08	0.10	5	6	6	First_order
MANNE	6000	4000	3.261e+39	1.4e+19	95.71	167	168	168	Unknown
MARINE	11215	11192	3.922e−21	8.0e−08	8.51	9	10	10	First_order
MEYER3NE	3	16	4.397e+01	1.1e−03	0.00	11	19	12	Unknown

(continued)

Table 10 (continued)

Name	nvar	ncon	f	$\|\nabla L\|_2$	t	iter	$\#c$	$\#\nabla c$	Status
MODBEALENE	20000	39999	1.374e+00	8.3e−07	109.80	38	73	39	First_order
MOREBVNE	10	10	1.085e−14	1.9e−08	0.00	2	3	3	First_order
MUONSINE	1	512	2.194e+04	8.8e−04	0.02	36	37	37	First_order
NGONE	200	5048	7.203e−13	8.3e−07	36.90	221	914	223	First_order
NONDIANE	5000	5000	4.949e−01	1.9e−08	2.63	16	39	17	First_order
NONMSQRTNE	4900	4900	3.543e+02	1.1e+01	154.67	500	2624	501	Max_iter
NONSCOMPNE	5000	5000	1.378e−06	5.6e−07	11.31	80	217	81	First_order
OSCIGRNE	100000	100000	3.138e−24	8.3e−09	3.62	7	8	8	First_order
OSCIPANE	10	10	5.000e−01	1.1e−01	0.11	500	2880	501	Max_iter
PALMER1ANE	6	35	4.494e−02	6.7e−07	0.01	25	84	26	First_order
PALMER1BNE	4	35	1.724e+00	2.6e−08	0.00	6	7	7	First_order
PALMER1ENE	8	35	1.822e−01	1.3e+02	0.17	500	3471	501	Max_iter
PALMER1NE	4	31	5.877e+03	8.4e+00	0.13	500	2561	501	Max_iter
PALMER2ANE	6	23	8.555e−03	5.9e−07	0.02	58	248	59	First_order
PALMER2BNE	4	23	3.116e−01	5.2e−08	0.00	8	10	9	First_order
PALMER2ENE	8	23	1.952e−02	8.9e−07	0.01	9	68	10	First_order
PALMER2NE	4	23	1.826e+03	4.1e−06	0.01	26	92	27	Unknown
PALMER3ANE	6	23	1.022e−02	1.0e−07	0.00	18	39	19	First_order
PALMER3BNE	4	23	2.114e+00	1.5e−07	0.00	11	14	12	First_order
PALMER3ENE	8	23	2.303e−02	1.2e+01	0.15	500	3348	501	Max_iter
PALMER3NE	4	23	1.133e+03	5.7e−02	0.15	500	3303	501	Max_iter
PALMER4ANE	6	23	2.030e−02	3.8e−07	0.03	98	445	99	First_order

(continued)

Table 10 (continued)

Name	nvar	ncon	f	$\|\nabla L\|_2$	t	iter	#c	#∇c	Status
PALMER4BNE	4	23	3.418e+00	6.1e−07	0.00	14	18	15	First_order
PALMER4ENE	8	23	6.286e−02	9.3e−07	0.04	69	667	70	First_order
PALMER4NE	4	23	1.143e+03	2.2e−05	0.01	35	163	36	Unknown
PALMER5ANE	8	12	1.706e−01	1.2e+00	0.13	500	3000	501	Max_iter
PALMER5BNE	9	12	4.876e−03	3.4e−07	0.03	93	481	94	First_order
PALMER5ENE	8	12	1.699e−02	1.4e+00	0.16	500	3508	501	Max_iter
PALMER6ANE	6	13	2.797e−02	2.3e−07	0.00	17	30	18	First_order
PALMER6ENE	8	13	2.424e−02	5.8e−07	0.08	142	1654	143	First_order
PALMER7ANE	6	13	5.526e+00	2.4e+00	0.13	500	2806	501	Max_iter
PALMER7ENE	8	13	3.353e+00	1.4e+02	0.16	500	3243	500	Max_iter
PALMER8ANE	6	12	3.700e−02	5.8e−07	0.02	72	303	73	First_order
PALMER8ENE	8	12	1.687e−01	2.3e−04	0.22	500	5007	501	Max_iter
PENLT1NE	10	11	3.576e−10	8.1e−07	0.02	111	385	112	First_order
PENLT2NE	4	8	4.701e−11	9.6e−07	0.03	163	613	164	First_order
PINENE	8805	8795	5.184e−17	1.1e−07	1.72	2	10	3	First_order
POWERSUMNE	4	4	2.325e−17	7.6e−07	0.00	17	21	18	First_order
PRICE3NE	2	2	5.614e−22	3.9e−10	0.00	7	8	8	First_order
PRICE4NE	2	2	1.321e−14	9.8e−07	0.00	21	22	22	First_order
QINGNE	100	100	9.461e−20	1.2e−09	0.00	5	8	6	First_order
RSNBRNE	2	2	4.832e−30	3.1e−15	0.00	11	34	12	First_order
S308NE	2	3	3.866e−01	9.1e−07	0.00	34	36	35	First_order
SBRYBNDNE	5000	5000	1.790e−21	3.0e−07	1.16	6	7	7	First_order

(continued)

Table 10 (continued)

Name	nvar	ncon	f	$\|\nabla L\|_2$	t	iter	$\#c$	$\#\nabla c$	Status
SINVALNE	2	2	3.852e−32	2.8e−15	0.00	4	11	5	First_order
SPECANNE	9	15000	3.291e−13	5.3e−08	0.08	6	7	7	First_order
SROSENBRNE	5000	5000	5.547e−28	6.7e−16	0.49	2	3	3	First_order
SSBRYBNDNE	5000	5000	5.274e−22	4.0e−10	1.15	6	7	7	First_order
STREGNE	4	2	2.236e−03	4.7e−01	0.05	500	535	501	Max_iter
STRTCHDVNE	10	9	3.723e−10	4.4e−07	0.00	8	9	9	First_order
TQUARTICNE	5000	5000	2.631e−26	2.3e−13	0.39	1	2	2	First_order
TRIGON1NE	10	10	6.724e−16	1.0e−07	0.00	4	5	5	First_order
TRIGON2NE	10	31	1.620e+00	1.5e−09	0.00	7	8	8	First_order
VARDIMNE	10	12	7.960e−16	4.0e−07	0.00	9	10	10	First_order
VIBRBEAMNE	8	30	7.822e−02	6.9e−07	0.00	10	11	11	First_order
WATSONNE	12	31	1.429e−15	1.3e−13	0.00	4	5	5	First_order
WAYSEA1NE	2	2	3.683e−16	8.7e−07	0.00	7	8	8	First_order
WAYSEA2NE	2	2	3.388e−18	1.3e−08	0.00	11	17	12	First_order
WEEDSNE	3	12	1.294e+00	3.0e−07	0.00	15	25	16	First_order
WOODSNE	4000	3001	5.000e−01	0.0e+00	0.33	2	3	3	First_order

Table 11 `ncl_nls` results on 127 CUTEst nonlinear least-squares problems. 119 problems were solved successfully

Name	f	$\|\nabla L\|_2$	$\|c\|_2$	t	iter	#c	#∇c	#$\nabla^2 L$	Status
ARWHDNE	6.971e+01	1.5e−14	1.4e−16	2.65	30	105	32	31	First_order
BA-L1	4.400e−31	9.1e−16	2.0e−10	0.01	5	6	7	6	First_order
BA-L16	4.324e+05	1.4e+03	4.5e−07	937.15	88	454	90	89	Unknown
BA-L1SP	3.204e−23	7.4e−12	8.3e−05	0.01	4	5	6	5	First_order
BA-L21	1.975e+05	1.5e−02	3.0e−05	947.20	201	1093	203	203	Max_time
BA-L49	1.674e+04	1.2e−01	1.6e−04	869.15	217	1189	219	219	Max_time
BA-L52	3.860e+06	2.0e+02	8.0e−01	1778.45	31	50	33	32	Max_time
BA-L73	9.609e+05	1.1e+02	1.9e−06	693.06	135	635	137	136	Unknown
BARDNE	4.107e−03	2.3e−14	5.7e−12	0.00	5	6	7	6	First_order
BDQRTICNE	1.000e+04	3.4e−10	1.7e−11	0.47	17	25	19	18	First_order
BEALENE	4.598e−20	2.0e−10	1.6e−09	0.00	8	9	10	9	First_order
BIGGS6NE	2.153e−17	6.8e−12	7.2e−08	0.02	60	74	62	61	First_order
BOX3NE	2.641e−19	7.3e−12	1.8e−10	0.00	5	6	7	6	First_order
BROWNBSNE	5.933e−33	5.6e−12	1.1e−10	0.00	13	42	15	14	First_order
BROWNDENE	4.291e+04	9.7e−09	3.0e−11	0.00	14	22	16	15	First_order
BRYBNDNE	6.662e−20	2.6e−10	2.5e−10	0.20	6	7	8	7	First_order
CHAINWOONE	6.598e+03	1.8e+00	2.1e−01	14.40	500	546	502	501	Max_iter
CHEBYQADNE	4.358e−03	5.1e−07	2.9e−07	0.60	33	51	35	34	First_order
CHNRSBNE	2.913e−21	1.9e−10	7.2e−11	0.02	34	51	36	35	First_order
CHNRSNBMNE	6.544e−23	4.0e−11	3.9e−10	0.02	38	57	40	39	First_order
COATINGNE	2.527e−01	1.5e−09	3.9e−10	0.02	11	22	13	12	First_order
CUBENE	2.479e−33	9.6e−17	6.7e−15	0.00	2	7	4	3	First_order

(continued)

Table 11 (continued)

Name	f	$\|\nabla L\|_2$	$\|c\|_2$	t	iter	$\#c$	$\#\nabla c$	$\#\nabla^2 L$	Status
DECONVBNE	1.285e−03	3.1e−07	5.3e−06	0.03	20	34	23	21	First_order
DECONVNE	2.694e−15	3.8e−08	6.4e−10	0.00	2	3	4	3	First_order
DENSCHNBNE	1.225e−17	2.6e−09	5.8e−09	0.00	6	8	8	7	First_order
DENSCHNCNE	9.992e−38	4.3e−19	6.7e−06	0.00	6	7	8	7	First_order
DENSCHNDNE	4.012e−28	4.5e−17	3.6e−04	0.00	15	16	17	16	First_order
DENSCHNENE	4.540e−21	9.5e−11	1.7e−06	0.00	15	20	17	16	First_order
DENSCHNFNE	2.351e−38	2.2e−19	1.9e−06	0.00	4	5	6	5	First_order
DEVGLA1NE	1.063e−13	2.9e−08	4.0e−11	0.01	16	45	18	17	First_order
DEVGLA2NE	1.672e−14	6.3e−08	3.0e−07	0.00	9	12	11	10	First_order
EGGCRATENE	4.744e+00	3.1e−08	2.4e−09	0.00	4	5	6	5	First_order
ELATVIDUNE	2.738e+01	1.2e−09	5.0e−10	0.00	8	9	10	9	First_order
ENGVAL2NE	6.374e−15	1.3e−08	1.2e−07	0.00	12	20	14	13	First_order
ERRINROSNE	2.020e+01	1.4e−09	4.1e−10	0.01	17	24	19	18	First_order
ERRINRSMNE	1.926e+01	1.7e−08	1.2e−09	0.02	25	38	27	26	First_order
EXP2NE	1.617e−19	7.1e−13	1.9e−13	0.00	5	6	7	6	First_order
EXPFITNE	1.203e−01	2.4e−10	4.9e−12	0.01	21	149	23	22	First_order
EXTROSNBNE	−2.002e+00	1.0e−06	1.3e−09	1.16	250	1860	252	251	First_order
FBRAIN2NE	1.842e−01	6.3e−10	3.1e−11	0.11	5	6	7	6	First_order
FBRAINNE	2.083e−01	8.4e−08	1.7e−09	0.05	4	5	6	5	First_order
FREURONE	2.449e+01	1.5e−10	5.5e−12	0.00	7	15	9	8	First_order
GENROSEBNE	7.965e+02	1.3e−06	1.5e−06	0.04	6	8	9	7	First_order
GENROSENE	5.000e−01	8.9e−14	1.1e−14	2.07	436	639	438	437	First_order

(continued)

Table 11 (continued)

Name	f	$\|\nabla L\|_2$	$\|c\|_2$	t	iter	$\#c$	$\#\nabla c$	$\#\nabla^2 L$	Status
GULFNE	1.755e−20	3.4e−11	1.6e−10	0.02	20	28	22	21	First_order
HATFLDANE	8.445e−20	5.4e−11	6.2e−09	0.00	9	10	11	10	First_order
HATFLDBNE	2.786e−03	4.8e−12	1.1e−11	0.00	7	8	9	8	First_order
HATFLDCNE	2.674e−17	4.6e−09	7.8e−09	0.00	3	4	5	4	First_order
HATFLDDNE	1.273e−07	4.1e−09	9.7e−07	0.00	5	8	7	6	First_order
HATFLDENE	1.364e−06	5.8e−11	1.7e−09	0.00	5	6	7	6	First_order
HATFLDFLNE	3.008e−05	9.2e−12	7.3e−09	0.02	103	443	105	104	First_order
HELIXNE	5.661e−43	1.7e−21	1.7e−10	0.00	9	12	11	10	First_order
HIMMELBFNE	2.168e+06	2.1e+02	2.9e−04	0.09	500	720	502	501	Max_iter
HS1NE	3.359e−17	4.1e−09	5.8e−16	0.00	7	11	9	8	First_order
HS25NE	1.642e+01	8.1e−08	3.6e−11	0.00	3	5	6	4	First_order
HS2NE	2.471e+00	8.4e−12	1.3e−12	0.00	5	7	8	6	First_order
INTEQNE	6.816e−37	8.7e−19	1.6e−07	0.00	2	3	4	3	First_order
JENSMPNE	6.218e+01	1.6e−08	3.1e−09	0.00	8	14	10	9	First_order
JUDGENE	8.041e+00	4.8e−07	2.5e−07	0.00	5	6	7	6	First_order
KOEBHELBNE	3.876e+01	1.4e−08	4.5e−09	0.13	119	164	121	121	First_order
KOWOSBNE	1.539e−04	2.9e−10	2.6e−09	0.00	11	14	13	12	First_order
LIARWHDNE	1.113e−21	4.7e−11	4.0e−11	0.17	6	7	8	7	First_order
LINVERSENE	3.405e+02	3.3e−07	1.4e−07	0.31	19	22	22	20	First_order
MANCINONE	2.507e−32	1.1e−16	2.4e−09	0.12	4	5	6	5	First_order
MANNE	−9.698e−01	9.6e−07	0.0e+00	10.27	344	346	347	345	First_order
MARINE	2.009e+06	1.7e−08	1.1e−08	3.76	40	43	43	41	First_order

(continued)

Table 11 (continued)

Name	f	$\|\nabla L\|_2$	$\|c\|_2$	t	iter	$\#c$	$\#\nabla c$	$\#\nabla^2 L$	Status
MEYER3NE	4.397e+01	2.0e−06	1.1e−08	0.00	8	14	10	9	First_order
MODBEALENE	2.521e−18	1.5e−09	2.8e−09	1.80	11	12	13	12	First_order
MOREBVNE	2.674e−37	9.3e−19	7.6e−08	0.00	2	3	4	3	First_order
MUONSINE	2.194e+04	2.4e−11	6.2e−14	0.02	10	15	12	11	First_order
NGONE	−8.551e+00	4.0e−09	3.8e−11	2.46	106	108	109	107	First_order
NONDIANE	4.949e−01	1.3e−16	1.5e−07	0.16	6	7	8	7	First_order
NONMSQRTNE	3.543e+02	1.1e−07	2.5e−06	4.01	19	30	21	20	First_order
NONSCOMPNE	1.805e−07	1.8e−10	3.0e−06	0.52	19	28	21	20	First_order
OSCIGRNE	1.449e−35	4.0e−18	1.4e−07	3.78	6	7	8	7	First_order
OSCIPANE	5.000e−01	9.8e−06	5.5e−04	0.16	500	2698	502	501	Max_iter
PALMER1ANE	2.988e+01	2.4e−06	6.8e−08	0.01	12	16	14	13	First_order
PALMER1BNE	1.724e+00	1.9e−08	1.9e−10	0.01	11	16	13	12	First_order
PALMER1ENE	4.176e−04	7.7e−09	1.4e−06	0.00	7	8	9	8	First_order
PALMER1NE	5.877e+03	2.7e−06	2.4e−09	0.02	37	46	39	38	First_order
PALMER2ANE	8.555e−03	1.9e−10	4.1e−08	0.02	44	69	46	45	First_order
PALMER2BNE	3.116e−01	1.9e−07	3.1e−09	0.00	8	11	10	9	First_order
PALMER2ENE	5.815e−02	6.9e−08	3.2e−10	0.01	20	23	22	21	First_order
PALMER2NE	1.826e+03	3.4e−06	5.3e−08	0.02	49	64	51	50	First_order
PALMER3ANE	1.022e−02	2.0e−08	9.0e−07	0.01	13	17	15	14	First_order
PALMER3BNE	2.114e+00	6.1e−12	1.1e−14	0.01	27	38	29	28	First_order
PALMER3ENE	2.537e−05	4.1e−11	1.2e−09	0.01	14	19	16	15	First_order
PALMER3NE	1.208e+03	1.1e−05	2.4e−08	0.00	6	10	8	7	First_order

(continued)

Table 11 (continued)

Name	f	$\|\nabla L\|_2$	$\|c\|_2$	t	iter	$\#c$	$\#\nabla c$	$\#\nabla^2 L$	Status
PALMER4ANE	2.030e−02	2.8e−08	1.6e−07	0.01	11	20	13	12	First_order
PALMER4BNE	3.418e+00	1.3e−12	2.3e−14	0.01	32	41	34	33	First_order
PALMER4ENE	7.400e−05	4.7e−09	2.0e−07	0.01	11	13	13	12	First_order
PALMER4NE	1.212e+03	1.5e−05	4.2e−08	0.00	6	10	8	7	First_order
PALMER5ANE	1.452e−02	9.5e−07	4.3e−07	0.10	182	874	184	183	First_order
PALMER5BNE	4.876e−03	3.7e−09	1.1e−07	0.01	44	50	46	45	First_order
PALMER5ENE	1.036e−02	1.3e−09	3.3e−07	0.02	60	200	62	61	First_order
PALMER6ANE	2.797e−02	2.1e−08	7.5e−06	0.01	22	31	24	23	First_order
PALMER6ENE	6.462e−02	5.5e−07	2.3e−06	0.00	7	10	9	8	First_order
PALMER7ANE	5.168e+00	1.6e−07	2.6e−05	0.16	355	1551	357	358	First_order
PALMER7ENE	3.346e+00	1.5e−06	4.0e−10	0.03	54	274	56	55	First_order
PALMER8ANE	3.700e−02	9.3e−08	3.6e−07	0.01	19	31	21	20	First_order
PALMER8ENE	3.170e−01	6.3e−08	3.3e−09	0.01	25	35	27	26	First_order
PENLT1NE	3.544e−10	5.5e−14	3.8e−05	0.00	8	9	10	9	First_order
PENLT2NE	4.688e−11	1.4e−18	5.6e−08	0.00	5	10	7	6	First_order
PINENE	4.194e−05	4.9e−10	1.1e−06	0.72	13	18	16	14	First_order
POWERSUMNE	1.799e−22	1.2e−11	6.0e−10	0.01	45	48	47	46	First_order
PRICE3NE	1.505e−35	5.2e−18	1.1e−05	0.00	6	7	8	7	First_order
PRICE4NE	3.508e−32	3.4e−19	1.1e−04	0.00	13	14	15	14	First_order
QINGNE	1.318e−33	2.8e−17	9.2e−05	0.00	4	7	6	5	First_order
RSNBRNE	5.556e−32	3.3e−16	3.1e−14	0.00	1	3	3	2	First_order
S308NE	3.866e−01	1.7e−09	9.0e−10	0.00	10	16	12	11	First_order

(continued)

Table 11 (continued)

Name	f	$\|\nabla L\|_2$	$\|c\|_2$	t	iter	$\#c$	$\#\nabla c$	$\#\nabla^2 L$	Status
SBRYBNDNE	6.690e−20	2.6e−10	2.5e−10	0.18	6	7	8	7	First_order
SINVALNE	1.578e−30	1.8e−15	1.8e−15	0.00	1	3	3	2	First_order
SPECANNE	3.291e−13	3.5e−08	1.2e−12	0.49	10	14	12	11	First_order
SROSENBRNE	5.892e−33	1.1e−18	2.2e−18	0.05	2	3	4	3	First_order
SSBRYBNDNE	6.663e−20	2.6e−10	2.5e−10	0.21	6	7	8	7	First_order
STREGNE	2.204e−27	6.6e−22	4.4e−15	0.00	2	3	4	3	First_order
STRTCHDVNE	2.260e−15	1.8e−10	6.1e−07	0.00	10	11	12	11	First_order
TQUARTICNE	0.000e+00	0.0e+00	2.2e−16	0.05	1	2	3	2	First_order
TRIGON1NE	3.427e−39	5.7e−20	1.8e−08	0.00	4	5	6	5	First_order
TRIGON2NE	1.620e+00	2.7e−09	1.3e−09	0.00	11	12	13	12	First_order
VARDIMNE	4.873e−20	2.9e−09	1.2e−10	0.00	14	19	16	15	First_order
VIBRBEAMNE	7.822e−02	2.3e−11	1.7e−14	0.01	7	8	9	8	First_order
WATSONNE	1.430e−15	5.8e−16	5.6e−14	0.00	4	5	6	5	First_order
WAYSEA1NE	0.000e+00	0.0e+00	2.7e−08	0.00	7	8	9	8	First_order
WAYSEA2NE	8.386e−17	2.3e−09	1.0e−07	0.00	10	20	12	11	First_order
WEEDSNE	1.294e+00	8.6e−08	7.4e−07	0.01	17	24	19	18	First_order
WOODSNE	−1.906e+04	9.1e−13	8.0e−10	0.04	3	4	5	4	First_order

Acknowledgements We are deeply grateful to Professor Ken Judd and Dr Che-Lin Su for developing the AMPL tax policy model [11] that led to the development of Algorithm NCL [15], and to the developers of AMPL, Julia, IPOPT, and KNITRO for making the implementation and evaluation of Algorithm NCL possible. In particular, we thank Dr Richard Waltz of Artelys for his help in finding runtime options for warm-starting KNITRO. We also give sincere thanks to Pierre-Élie Personnaz for obtaining the Julia/NCL results in Tables 3, 4, 5, 6 and 7, and to Professor Mehiddin Al-Baali and other organizers of the NAO-V conference *Numerical Analysis and Optimization* at Sultan Qaboos University, Muscat, Oman, which brought the authors together in January 2020. Finally, we are very grateful to the referees for their constructive questions and comments, and to Michael Friedlander for his helpful discussions.

References

1. AMPL modeling system. http://www.ampl.com.
2. Jon Bentley. Programming pearls: Little languages. *Commun. ACM*, 29(8):711–721, 1986.
3. Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B Shah. Julia: A fresh approach to numerical computing. *SIAM Rev.*, 59(1):65–98, 2017.
4. A. R. Conn, N. I. M. Gould, and Ph. L. Toint. A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds. *SIAM J. Numer. Anal.*, 28:545–572, 1991.
5. A. R. Conn, N. I. M. Gould, and Ph. L. Toint. *LANCELOT: A Fortran Package for Large-scale Nonlinear Optimization (Release A)*. Lecture Notes in Computation Mathematics 17. Springer Verlag, Berlin, Heidelberg, New York, London, Paris and Tokyo, 1992.
6. R. Fourer, D. M. Gay, and B. W. Kernighan. *AMPL: A Modeling Language for Mathematical Programming*. Brooks/Cole, Pacific Grove, second edition, 2002.
7. GAMS modeling system. http://www.gams.com.
8. P. E. Gill, W. Murray, and M. A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Rev.*, 47(1):99–131, 2005. SIGEST article.
9. N. I. M. Gould, D. Orban, and Ph. L. Toint. CUTEst: a Constrained and Unconstrained Testing Environment with safe threads. *Comput. Optim. Appl.*, 60:545–557, 2015.
10. COIN-OR Interior Point Optimizer IPOPT. https://github.com/coin-or/Ipopt.
11. K. L. Judd and C.-L. Su. Optimal income taxation with multidimensional taxpayer types. Working paper, Hoover Institution, Stanford University, 2011.
12. KNITRO optimization software. https://www.artelys.com/tools/knitro_doc/2_userGuide.html.
13. LANCELOT optimization software. http://www.numerical.rl.ac.uk/lancelot/blurb.html.
14. M. Lubin and I. Dunning. Computing in operations research using julia. *INFORMS J. Comput.*, 27(2), 2015.
15. D. Ma, K. L. Judd, D. Orban, and M. A. Saunders. Stabilized optimization via an NCL algorithm. In M. Al-Baali et al., editor, *Numerical Analysis and Optimization, NAO-IV, Muscat, Oman, January 2017*, pages 173–191. Springer International Publishing AG, 2018.
16. B. A. Murtagh and M. A. Saunders. A projected Lagrangian algorithm and its implementation for sparse nonlinear constraints. *Math. Program. Study*, 16:84–117, 1982.
17. NCL AMPL models. http://stanford.edu/group/SOL/multiscale/models/NCL/.
18. NEOS server for optimization. http://www.neos-server.org/neos/.
19. D. Orban and P. E. Personnaz. NCL.jl: A nonlinearly-constrained augmented-Lagrangian method. https://github.com/JuliaSmoothOptimizers/NCL.jl, July 2020.
20. D. Orban, A. S. Siqueira, and contributors. AmplNLReader.jl: A Julia interface to AMPL. https://github.com/JuliaSmoothOptimizers/AmplNLReader.jl, July 2020.
21. D. Orban, A. S. Siqueira, and contributors. CUTEst.jl: Julia's CUTEst interface. https://github.com/JuliaSmoothOptimizers/CUTEst.jl, October 2020.

22. D. Orban, A. S. Siqueira, and contributors. JuliaSmoothOptimizers: Infrastructure and solvers for continuous optimization in Julia. https://github.com/JuliaSmoothOptimizers, July 2020.
23. D. Orban, A. S. Siqueira, and contributors. NLPModelsIpopt.jl: A thin IPOPT wrapper for NLPModels. https://github.com/JuliaSmoothOptimizers/NLPModelsIpopt.jl, July 2020.
24. D. Orban, A. S. Siqueira, and contributors. NLPModels.jl: Data structures for optimization models. https://github.com/JuliaSmoothOptimizers/NLPModels.jl, July 2020.
25. D. Orban, A. S. Siqueira, and contributors. NLPModelsKnitro.jl: A thin KNITRO wrapper for NLPModels. https://github.com/JuliaSmoothOptimizers/NLPModelsKnitro.jl, July 2020.

A Survey on Modeling Approaches for Generation and Transmission Expansion Planning Analysis

Giovanni Micheli and Maria Teresa Vespucci

Abstract Generation and transmission expansion planning (GTEP) models determine the evolution of power systems over a long-term planning horizon, by defining technology, capacity and location of new generating units, as well as new electrical interconnections to be built. The models required to plan investment decisions in the power sector are typically large-scale models, since many variables and constraints are needed to represent a great number of strategic and operating decisions: to compute a solution to GTEP models different approximations have to be introduced. This paper provides a comprehensive description of the GTEP analysis, by highlighting the characteristics and the challenges of this problem and by reviewing the main approaches proposed in the literature to answer to specific research questions. This paper provides also a formulation of the GTEP problem suited to address decarbonization challenges in the power sector.

Keywords Generation and transmission expansion planning · Renewable energy sources · Stochastic programming

1 Introduction

Generation and transmission expansion planning (GTEP) models determine the evolution over a long-term horizon of a power system, by defining the new power generation capacity and the new electrical interconnections to be built so as to minimize the total investment and operation cost. The definition of joint expansion plans is

G. Micheli · M. T. Vespucci (✉)
Department of Management, Information and Production Engineering, University of Bergamo, via Marconi 5, 24044 Dalmine, BG, Italy
e-mail: maria-teresa.vespucci@unibg.it

G. Micheli
e-mail: giovanni.micheli@unibg.it

one of the most relevant problems in the field of power systems. Indeed, GTEP models present many applications, allowing for instance to evaluate the possibility to achieve long-term policy goals such as decarbonization, integration of large shares of renewables or reduction of CO_2 emissions.

Addressing the GTEP problem for modern power systems requires considering three main characteristics: (i) the long-term horizon, (ii) the high level of temporal detail and (iii) the high uncertainty involved. Indeed, investments in both electrical power plants and transmission lines are capital intensive decisions, which are usually characterized by lifetimes greater than 30–50 years. Thus, when planning the expansion of a power system, planning horizons of several decades are typically applied. The second characteristic of the GTEP analysis is the need to employ a high level of temporal detail, evaluating power system operation with an hourly resolution. Specifically, the high level of temporal detail is needed to capture the fluctuation of solar and wind power generation, to model some technical constraints on thermal power production, such as the minimum uptime and the minimum downtime constraints, and to properly consider the dynamics of both hydro pumped storage and battery storage. Finally, the GTEP problem is also characterized by a high level of uncertainty. Indeed, since expansion plans are usually provided for a long-term planning horizon, the future system conditions are generally uncertain at the time the expansion plans are decided. Different sources of uncertainty may affect planning decisions and must be considered in the decision-making process. They can be divided into two groups, namely short-term and long-term uncertainties. Specifically, short-term uncertainties include the stochastic production from intermittent renewable energy sources and the demand variability throughout the hours of the day and the days of the week. By contrast, long-term uncertainty refers to long-term dynamics, including future values of investment costs, fossil fuel prices and policy constraints such as carbon prices.

Because of the long-term horizon, the hourly resolution and the high uncertainty, GTEP models are large-scale problems, whose solution is computationally challenging. When addressing GTEP problems, a trade-off between modeling accuracy and computational cost has to be considered: to compute a solution to real-scale GTEP models different approximations may be introduced. The aim of this paper is twofold: (i) provide a comprehensive review of the approaches proposed in the literature for the GTEP analysis, by highlighting the main characteristics and the challenges of this problem; (ii) present the GTEP model we have developed in the context of a research project aimed at planning the joint generation and transmission expansion of power systems to reach decarbonization targets set by the European Commission, which include an increasing penetration of renewable power sources and a reduction of CO_2 emissions.

The remainder of the paper proceeds as follows. In Sect. 2 we review the existing literature for GTEP, by discussing three main features: (i) the modeling choices related to the design of a GTEP model; (ii) the proposed approaches to include the uncertainty in the expansion planning framework; and (iii) the methods developed to provide an hourly solution to a GTEP model while maintaining the problem computationally tractable. Section 3 describes the assumptions we introduced in our analysis and formulates the GTEP problem as a two-stage stochastic MILP model. Finally, Sect. 4 concludes the paper.

2 Review of GTEP Models

2.1 Modeling Choices

This section introduces the basic choices associated with the construction of a GTEP model, discussing some important modeling issues such as the sectors to be included in the analysis, the problem formulation structure, the decision variables to introduce and the representation of the transmission network.

Sectors coverage. When designing a model for GTEP, the first decision to be taken concerns the choice of which sectors should be included in the analysis and which other sectors should be exogenously modeled. Indeed, there exist many interconnections between power systems and several sectors, such as the transportation sector, the heating sector or the fossil fuels sectors. In terms of sectors coverage, two different approaches are possible:

1. Perform a single sectoral analysis, focusing only on power sector and neglecting the interactions with other sectors.
2. Consider multiple sectors to study the high-level interactions within the whole energy system.

Considering multiple sectors allows evaluating the interdependencies between sectors, usually finding more efficient solutions for the whole energy system. However, the drawback of this approach is an increasing computational cost. To maintain the problem computationally tractable, multi-sector models usually introduce strong approximations in the way single sectors are modeled. For instance, reference [1] proposes an equilibrium model to plan infrastructure investments in the whole energy system to satisfy the future energy requirements of three different demand sectors, namely the industry sector, the residential sector and the transportation sector, while considering climate policies and governmental regulation. A very wide geographical scope is considered in this analysis, modeling the whole world through 30 nodes, of which 15 are European countries and the remaining 15 model the rest of the world, while the planning horizon of 40 years, from 2010 up to 2050, is discretized in 10-year steps. The energy requirements of demand sectors are satisfied by using various energy carriers (fuels) depending on relative costs, efficiencies, as well as regulatory and technical constraints. The fuels included in the analysis are: crude oil, oil products, natural gas, coal, lignite, electricity, biofuels, renewables, and (as an input for power generation only) nuclear and hydro. In this work, a very simplified representation of the power sector is considered: indeed, the power sector is modeled as a transformation process, converting input fuels into electricity, considering only capacity constraints stated in terms of output quantities.

When a high level of technical detail is required, the multi-sector models can be applied only by limiting the geographical scope. For instance, authors in [2] perform a long-term investment planning analysis on the Danish heat and electricity sector, considering the uncertainty related to the wind power production, which is modeled as a stochastic parameter. In this paper, a more detailed representation of the power sector is considered, including some technical constraints, such as peaking reserve constraints, and considering different scenarios for the wind power availability. However, the application of this analysis is limited to a single country, i.e., Denmark. From the previous works, it can be noted the existence of a trade-off between sectors coverage, geographical scope and technical detail: a multi-sector analysis can be performed only by limiting the geographical scope or by considering a low level of technical detail. Instead, by exogenously modeling all sectors but the power sector it is possible to consider technical and geographical characteristics in greater detail. The choice of the approach to adopt depends on the objective of the analysis: the multi-sector approach is to be applied when the research focus is to evaluate long-term trends, such as decarbonization pathways, taking into account the interactions between different economic sectors and the energy system; the single sectoral analysis is suited to plan investments in the power sector, since it allows a more detailed representation of the power system, as well as of the short-term operational dispatch, and therefore provides more reliable decisions to the actors involved in the GTEP problem. In this paper, we will focus on single sectoral analysis, by introducing several approaches designed to perform a GTEP analysis exogenously modeling all sectors but the power sector. In such models, the feedback effects between sectors are ignored and parameters like fuel prices and fuels availability are exogenous parameters rather than decision variables. However, this choice allows providing a more detailed representation of power systems, which is required when modeling both long-term investment decisions and short-term operational dispatch.

Formulation structure. Works that address the GTEP problem focusing only on power sector modeling can be further divided in two categories according to the problem formulation structure, which determines the decision variables to be introduced in the analysis. The first class includes decoupled models, i.e., researches that only address either generation expansion or transmission expansion. Specifically, many studies deal with only transmission expansion planning (TEP), a problem that is typically addressed by transmission planners to identify the optimal transmission reinforcements to be carried out with the aim of facilitating energy exchange among producers and consumers. In such models, generation expansion decisions are exogenous parameters rather than model variables. Pioneering work in this area is due to Garver [3], who in 1970 proposed a linear programming problem determining the transmission expansion plans based on the location of overloads. Since then, many relevant contributions based on mathematical programming have been produced. For instance, in [4] authors deal with the TEP problem considering the integration of large-scale wind power production, while in [5] a bi-level TEP model using conic AC power flow formulation is presented. Many other studies only address generation expansion planning (GEP), evaluating the adequacy of generating facilities used to supply load and analyzing whether it is necessary to build new power plants. The GEP problem is typically motivated by the growth of the demand for electricity and by the aging of existing generating facilities. As regards to the transmission network, in some GEP models transmission constraints are totally neglected. For instance, reference [6] proposes a mixed integer non-linear programming model to perform a generation expansion planning analysis minimizing the planning cost and environmental pollution at the same time, while considering energy storage systems. In this paper, the test system consists of an isolated system, which does not consider any representation of the transmission network. Instead, other works that address GEP include also the transmission network in the analysis by considering the network configuration as an exogenous parameter. Examples of these works are represented by references [7, 8]. Specifically, in [7] a multiobjective generation expansion problem is considered taking into account costs, environmental impacts and portfolio investment risk. The proposed model decides the location of the planned generation units in a multiperiod planning horizon by minimizing simultaneously costs, environmental impact, imported fuel and fuel price risks. In [8] the generation expansion problem is instead addressed considering a market framework, representing the strategic behavior of the producer through a bi-level model: the upper-level considers both investment decisions and strategic production actions and the lower-level corresponds to market clearing.

Studies in the second class optimize both the generation and the transmission expansion plan. Indeed, if the demand for electricity in a zone can be supplied with both local generation and transportation of power from other zones, generation and transmission decisions are substitutes and have to be simultaneously considered in the optimization process [9]. By endogenously modeling the interactions between generation and transmission expansion decisions, the joint GTEP analysis provides solutions that are less expensive than expansion plans provided by the decoupled models. Two main approaches have been proposed to address the joint GTEP: decentralized and centralized models. Specifically, decentralized models consider that power systems consist of multiple decision makers with different objectives. Decentralized models are usually formulated as equilibrium problems, with several participants maximizing their own objective. For instance, in [10] the GTEP problem is formulated as a tri-level problem, consisting of the pool-based market, the generation system and the transmission system. In [11], authors propose mathematical models for sequential coordination of transmission expansion planning with strategic generation investments. The interaction between transmission company and strategic generation companies is modeled using the sequential-move game, while the interaction between the strategic generation companies is modeled as a simultaneous-move game. Centralized models instead approach the GTEP problem as just one problem, with a unique decision maker, such as the authority or the ministry of energy, taking all the relevant decisions. Although this approach does not reflect the real structure of modern power systems, consisting of several actors involved, centralized models are widely used to perform the so-called anticipative planning, whose objective is to identify policies and incentives that could induce generation companies to invest in a socially efficient manner [12, 13]. Several contributions have been developed in this area. For example, reference [14] proposes a mixed integer linear programming formulation for generation and transmission planning considering the value of lost load. A centralized approach is also adopted in [15] to study the effect of wind speed's spatial distribution on the simultaneous generation and transmission expansion planning of power systems including wind farms. Specifically, in this work investment decisions are defined by means of a mixed integer linear programming model that minimizes the sum of total investment cost of thermal generation units, transmission lines and wind farms, operation cost of thermal generators and loss cost of transmission lines. The wind generation is also accurately considered in [16], where a mixed integer linear programming model is proposed to study the GTEP problem with a high wind power penetration rate in large-scale power grids. Also in [17–19], the co-optimization problem is formulated as a mixed integer programming model, with the addition of reliability constraints enforced iteratively. Finally, authors in [20] propose a multiobjective, multiarea and multistage model to long-term expansion planning of integrated generation and transmission facilities. The proposed model considers three objectives: (i) minimization of investments and operation costs of power generation and transmission facilities; (ii) minimization of Life-Cycle Greenhouse Gas Emissions; and (iii) maximization of the diversification of electricity generation mix. Both the centralized and the decentralized approaches are useful to analyze optimal policies in the power sector. Specifically, centralized models are suited to search for optimal policies, modeling the power system and the short-term operations in great details, but considering a very simplified representation of market aspects. By contrast, decentralized models allow to validate policies, by reducing the technical detail of the analysis to focus on the interactions between different agents involved in the liberalized power sector. Because of computational restrictions, models with a detailed representation of both power system technical operations and market aspects cannot be applied to plan investment decisions in real-scale power systems. Due to our interest in finding optimal policies for the evolution of power systems, rather than in validating the outcomes of already specified policies, this survey is focused on centralized models. For a detailed description of market aspects, we refer the reader to [21] and the references contained in it.

Beside the choice between centralized and decentralized models, when addressing the joint GTEP problem decision variables representing infrastructure investments in generation and transmission capacity for the power system have to be introduced. Different types of variables can be used according to the technical detail of the analysis. Specifically, some models introduce binary variables to represent the selection of discrete facilities within a set of candidate generating plants and candidate transmission lines. This approach is typically adopted in studies that consider a very detailed representation of power systems, such as [22], modeling the operations of every power plant. Instead, most of the works in the literature employ linear decision variables, representing the total aggregated capacity per technology at each node, meaning that traditional unit commitment constraints are neglected. According to the objective of the analysis, ignoring unit commitment constraints could be a very restrictive approximation for expansion planning models. Specifically, when studying power systems with large shares of renewables it is necessary to plan also the investments in new flexible resources that could respond to the variability and uncertainty of stochastic generation. Ignoring unit commitment constraints leads to the impossibility of properly evaluating such flexibility and, consequently, to underestimate the required new generation capacity as well as the system costs, as shown in [23].

Transmission network. In a GTEP analysis transmission network can be modeled in different ways. In the order of increasing realism, the available approaches include the transshipment model, the Direct Current (DC) power flow model and the Alternating Current (AC) power flow model [9]. In the transshipment model, transmission lines are characterized by an efficiency parameter describing transportation losses. Energy flows on transmission lines are subject to capacity limits and nodal power balances are imposed at every node, while Kirchhoff's voltage law is ignored. A more realistic modeling approach is the DC power flow model, which, in addition to transmission flow limits and nodal power balances, consists of a linear relation between power flows and voltage angle differences. Although the equations that describe the DC power flow model are linear, when integrating this formulation in the GTEP problem the resulting model is a non-linear and non-convex problem. Indeed, in such formulation binary variables are introduced to model investment decisions in candidate transmission lines and the voltage angles are divided by line reactances, which are both decision variables. Thus, the DC power flow model provides a greater fidelity but at the cost of an increasing computational complexity. Finally, the most realistic representation of the transmission network is the AC power flow model, which is strongly non-linear since it models both active and reactive power flows in the transmission grid through non-linear constraints that involve nodal voltages and angles and line impedances. Although the AC model provides the best representation of power flows, the complexity in this approach is further increased: the trade-off between modeling detail and computational complexity holds also for the choice of the transmission network modeling approach. Usually, when a detailed geographical scope is considered, modeling every bus of the transmission network, AC and DC models are more appropriate choices, while the transshipment approach is generally preferred with higher geographical scopes, such as when nodes of the transmission network represent zones or countries. In the literature there exist also some hybrid approaches. For instance, reference [24] uses an iterative approach where investment decisions are optimized using a transshipment model, followed by a separate load flow model for grid expansion to check the feasibility and the robustness of the solution provided.

2.2 *Uncertainty Inclusion*

Several approaches have been developed in the literature to accurately represent the uncertainty framework in the planning decisions. In particular, given the increasing generation from intermittent renewable power sources, many recent studies focus on the influence of these uncertainties. For instance, reference [25] develops an algorithm for TEP considering wind farm generation and combining Monte Carlo simulation and Point Estimation Method to investigate the effects of network uncertainties. In [26] a chance-constrained formulation is proposed to tackle the uncertainties of load and wind turbine generators in transmission network expansion planning. Also reference [27] considers the uncertainty of load and wind power generation by proposing an efficient approach for probabilistic TEP, dealt with by a Benders decomposition algorithm combined with a Monte Carlo method.

Other studies deal with long-term uncertainties. For instance, reference [28] considers the uncertainty related to future demand growth and the availability of generation facilities and proposes a robust optimization model to determine the investment decisions in transmission capacity expansion minimizing the system's total costs by anticipating the worst-case realization of the uncertain parameters within an uncertainty set. Instead, reference [29] focus on the uncertainty related to the annual net load duration curve by introducing a robust GTEP model, including flexible AC transmission systems (FACTS) devices. A robust optimization approach is also adopted in [30, 31] to address generation expansion and transmission expansion, respectively. Specifically, reference [30] models the uncertainties associated with forecasted electricity load demand, as well as estimated investment and operation costs, through distribution-free bounded intervals producing polyhedral uncertainty sets. Expansion plans for generating facilities are then determined for multiple years by applying a robust optimization model. Instead, in [31] two optimization criteria for the TEP problem under the robust optimization paradigm are studied, where maximum cost and maximum regret of the expansion plan overall uncertainties are minimized, respectively.

Adaptation programming represents another method developed to deal with uncertainty. First introduced in [32] and further explored in [33], adaptation programming designs a flexible system by minimizing the sum of investment and operational cost and system future adaptation cost to the conditions of other identified scenarios. Several uncertainty sources are considered in these works, including investment costs, carbon prices, fuel prices and demand growth.

Among all the techniques developed to consider the uncertainty in the GTEP problem, the most widely used is stochastic programming [34], a methodology introduced in the 1950s that uses a set of scenarios to model the future realization of the uncertain parameters in the considered planning horizon. Works in this field can be further classified in two groups: two-stage and multi-stage models. In a typical two-stage stochastic model, the investment decisions represent first-stage decisions, which are made before any uncertainty is revealed. Operational decisions are instead second-stage decisions, made after the realization of parameter values. For instance, in [35] a two-stage stochastic programming model for joint GTEP is presented, considering as random events the demand, the equivalent availability of the generating plants and the transmission capacity factor of the transmission lines. Reference [36] presents a stochastic two-stage optimization model to evaluate interregional grid reinforcements in Great Britain. The same approach is also used in [37] to determine the type and quantity of power plants to be constructed in each year of an extended planning horizon, considering the uncertainty regarding future demand and fuel prices.

The two-stage approach can be extended to a multi-stage method, constructing models that are both more flexible and complex. As explained in [38], in multi-stage approaches expansion decisions are made at different stages, i.e., different points in time of the planning horizon. The expansion decisions at each stage depend on the scenario realization of the previous periods, but they do not depend on the future scenario realizations. Examples of multi-stage models are represented by [39–41]. For instance, in [39] a multi-stage multi-scale linear stochastic model is presented to optimize electricity generation, storage and transmission investments over a long planning horizon. Both long-term uncertainties, such as investment and fuel-cost changes and long-run demand-growth rates, and short-term uncertainties, such as hour-to-hour demand and renewable-availability uncertainty, are considered in this analysis and the progressive hedging algorithm is applied to decompose the original model by scenario. Reference [40] proposes a risk-constrained multi-stage stochastic programming model to make optimal investment decisions on wind power facilities along a multi-stage horizon considering three major issues: the production variability and uncertainty of wind power facilities, the eventual future decline in wind power investment costs and the significant financial risk involved in such investment decisions. Finally, in [41] authors study how uncertain future renewable penetration levels impact the electricity system and try to quantify effects for the Central European power market, by applying a multi-stage stochastic investment and dispatch model to analyse the effects on investment choices, electricity generation, and system costs. Although the multi-stage approach better represents long-term dynamics than the two-stage method, the complexity of the problem in multi-stage models is further increased. Finding the right balance between modeling accuracy and computation tractability remains an open research topic for GTEP models.

2.3 *High Level of Temporal Detail*

To accurately study all the challenges related to integrating high shares of intermittent energy sources, the GTEP analysis requires evaluating power system operations with an hourly resolution. Indeed, as shown in [42], a low level of temporal detail in GTEP models could significantly affect the results, not allowing to capture the short-term volatility of renewable production and overestimating the renewable penetration. However, due to the long-term horizon, considering every hour of the planning horizon would be computationally infeasible. To provide an accurate representation of the short-term operation while maintaining the problem computationally tractable, some energy planning models use a small number of representative periods (i.e., days or weeks) instead of modeling every hour of the planning horizon. Different approaches have been proposed to identify representative periods. Some authors select representative days by using simple heuristics. For instance, to capture the fluctuations in demand during the year, in [43] three time periods are selected as the day that contains the minimum demand level of the year, the day that contains the maximum demand level and the day that contains the largest demand spread in 24 h. Each of these representative days is then weighted in such way that the weighted sum of the hourly demand of the three selected days equals the overall original demand of the entire year. Instead, reference [44] proposes to represent the operation of a power system by means of four representative weeks, one for every season. The load profile of each of these representative weeks is the average of the load profiles of all the weeks of a season. In the objective function, the operational cost of each representative week is weighted by the number of weeks in the season. To ensure that the power system has enough flexibility to handle extreme conditions, authors propose also to add to the previous four average weeks a fifth week representing extreme conditions. However, no explanations about the way the extreme week is selected are provided in the paper. Other works combine heuristic approaches with the random selection of some additional days. For instance, in [45] unit commitment decisions within each investment period are optimized by considering a set of 12 days: two for each even-numbered month, with one day corresponding to the peak-load day and the second day randomly selected from days belonging to the same month. In [46], in order to significantly reduce the size of the planning problem, authors propose to consider 28 representative days in the whole planning horizon that are obtained by selecting 20 random days to characterize typical system behavior and 8 specific days that contain hours with extreme meteorological and load events. More advanced methods are based on clustering algorithms in order to group days with similar load, wind power production or solar power production into clusters: cluster's centroid or a specific historical day for each group is then taken as representative day. Different clustering algorithms are proposed in the literature. For example, reference [47] employs k-means algorithm, in [48] the Ward's hierarchical clustering algorithm is used, while reference [49] suggests to apply k-means algorithm using median representatives. Finally, some works select representative days by considering the historical load duration curve and the one obtained from the load in the representative days (see e.g. [50, 51]). Specifically, authors in [51] design a mixed integer linear programming model to select representative days and determine their weights so as to minimize the distance between the historical load duration curve and the one obtained by using the selected days. From this brief review about the selection of representative days, four main observations can be drawn:

1. Clustering algorithms generally provide better approximations than heuristic approaches [49, 51].
2. Using as representative days historical days rather than clusters means provide better results, especially if power systems have large shares of renewables [48, 49].
3. Representative days selected using clustering algorithms characterize typical system behavior. However, even the occurrence of extreme events should be taken into account to properly design the expansion of a power system [44–46].

4. The distance between the historical load duration curve and the one approximated by representative days can provide useful information about the goodness of the representation [50, 51].

Beside the method applied for representative days selection, the use of representative days raises the crucial issue regarding how these days should be linked in the expansion planning model. Some of the existing methods, such as [52], consider the representative days as temporally consecutive, linking these days according to an arbitrary order, from which, however, the model results may be affected. Other works, such as [44], assume that each representative day is followed by similar days and, thus, state that the initial status of each representative day should be equal to its final status. Specifically, in this approach new decision variables are introduced to represent the initial status of each generating plants and constraints are imposed to enforce the equality between initial and final status of each representative period. However, this approach is not appropriate for extreme days, which represent unique conditions. Extreme days are usually preceded by average days, thus considering the initial status of an extreme day equal to its final status could bias the calculation of the optimal amount of flexibility. More sophisticated approaches, such as [53], connect representative days by computing the transition matrix, which gives the number of transitions between each pair of representative days. However, this approach implies the introduction of many constraints and the interconnection among days, which can consistently increase computational costs. In order to provide an accurate solution to the unit commitment problem while maintaining representative days separate, reference [54] assigns to each thermal power plant an initial ON/OFF status in every representative day by means of a decision tree built on historical data. In this way, representative days are kept separate in the expansion planning problem, allowing the implementation of decomposition techniques such as Benders algorithms to decompose the expansion planning model by representative day.

The use of representative days introduces also a strong limitation in storage operation modeling. Indeed, representative days allow providing a good representation of storage operation within a day, but, since the chronology among representative days is not preserved, any energy storage system with a cycle longer than 24 h cannot be modeled with great accuracy. To model the operation of medium-term or long-term storage facilities, two different approaches can be applied:

1. Representative days are connected in order to catch the seasonality of storage operation. While some works consider representative days as temporal consecutive [52], some recent studies model the long-term storage by considering the Cluster Indices, which is a numeric column vector where each row indicates the cluster assignment of the corresponding day of the year [53].
2. According to historical data or the results of a preliminary model, initial and final energy contents are assigned to each representative day.

By imposing constraints that connect representative days, the drawback of the first approach is an increasing computational complexity and the impossibility to decompose the problem by representative day, which is a strategy commonly applied to pursue scalability in GTEP models [23]. Instead, the disadvantage of the second approach is the need to apply a statistical analysis or an optimization model to determine the medium-term or long-term energy content of storage facilities. However, decisions about the use of storage facilities depend on investments in new generation and transmission and, thus, they cannot be determined by considering historical data or by applying a preliminary model on input data.

3 A GTEP Model for the Decarbonization of Power Systems

In this section we present a GTEP model designed to reach in the power sector the decarbonization targets set by the European Commission, which include an increasing penetration of renewable power sources and a reduction of CO_2 emissions. The model hereafter introduced is a simplified version of the model proposed in [22] to plan the joint expansion of generation and transmission capacity in the Italian power system. The distinct feature of this model is a high level of both technical and temporal detail, which is required to properly evaluate all the challenges related to integrating high shares of renewables to reach decarbonization targets. Taking into account the trade-off between modeling accuracy and computational tractability described in the previous section, a zonal representation of the power system is adopted, with the set of zones denoted by Z, and only a small number of representative days is considered to evaluate the short-term operational dispatch, with the set of representative days denoted by C.

The structure of the power system at the beginning of the planning horizon is described by set L_E of transmission lines connecting zones, set K_E of thermal power plants, set H_E of hydropower plants and parameters $sol_{z,0}$ and $wind_{z,0}$ representing the solar power capacity and the wind power capacity, respectively, installed in zone $z \in Z$. The decisions to be taken concern decommissioning of existing thermal power plants as well as investments in new lines, new thermal units, new hydropower plants, new RES generation capacity and batteries, whose installed capacity is supposed to be negligible at the beginning of the planning horizon (as it is, for instance, in the Italian power system). Decommissioning and investment decisions are defined for every year of the planning horizon, with the set of years denoted by Y. Given the zonal representation of the power system and the long-term planning horizon, a transshipment approach is introduced to model the transmission network.

As regards to generation expansion decisions, the investments in new RES power generation capacity are represented by continuous variables. While it is possible to build wind and solar power plants of any capacity, thermal units usually present specified size, which does not allow modeling thermal power capacity expansion by means of continuous variables. Thus, investments in new thermal plants are managed through a set of candidate projects and binary variables describing whether the units are realized or not. Similarly, binary variables are introduced also to represent investments in new hydropower plants and new transmission lines, while investments in batteries capacity are modeled through continuous variables.

Since the thermal power plants production costs are uncertain parameters that play an important role in the GTEP analysis by affecting the merit order of thermal plants and the economic viability of renewable generation, different scenarios for these parameters are considered, with the set of scenarios denoted by W. The joint GTEP problem is then formulated as a two-stage stochastic programming model, with the first stage representing the investment problem and the second stage being the operational problem. The following paragraphs provide the nomenclature and the mathematical formulation of the GTEP model.

Sets

Y	set of years, indexed by y
Z	set of zones, indexed by z
M	set of macro-areas, indexed by m
K	set of thermal power plants, indexed by k
$K_E \subseteq K$	set of existing thermal power plants
$K_C \subseteq K$	set of candidate thermal power plants
$\Omega_z^k \subseteq K$	set of thermal power plants located in zone z
L	set of lines, indexed by l
$L_E \subseteq L$	set of existing lines
$L_C \subseteq L$	set of candidate lines
F	set of fuels, indexed by f
$\Phi_{f,z} \subseteq \Omega_z^k$	set of thermal power plants located in zone z using fuel f
H	set of hydropower plants, indexed by h
$H_E \subseteq H$	set of existing hydropower plants
$H_C \subseteq H$	set of candidate hydropower plants
$\Omega_z^h \subseteq H$	set of hydropower plants located in zone z
B	set of batteries, indexed by b
$\Omega_z^b \subseteq B$	set of batteries located in zone z
C	set of representative days, from 1 to $n_{cluster}$, indexed by c
T	set of hours, from 1 to 24 $n_{cluster}$ indexed by t
W	set of scenarios, indexed by w
$ma(z)$	macro-area that contains zone z
$UP(k)$	upgrade project of existing thermal power plant k
$r(l)$	receiving-end zone of transmission line l
$s(l)$	sending-end zone of transmission line l
$fuel(k)$	fuel used in thermal power plant k
$year(c)$	year that contains representative day c
$cl(t)$	representative day that contains hour t
$yr(t)$	year that contains hour t

Parameters

y_0	reference year to which all investment costs are discounted
r	annual discount rate
c_{ENP}	penalty for energy not provided
c_{OG}	penalty for over-generation
pr_w	probability of scenario w
wg_c	weight of cluster c
DC_k	decommissioning cost of existing thermal power plant $k \in K_E$
IC_k^{th}	investment cost of candidate thermal power plant $k \in K_C$
FC_k	annual fixed costs of thermal power plant $k \in K$
$CM_{k,y,w}$	marginal production cost of thermal power plant k in year y in scenario w
$\underline{\tau}_k$	earliest date for construction/decommission of thermal power plant k
$\overline{\tau}_k$	latest date for construction/decommission of thermal power plant k
$\underline{P}_k$	minimum power output of thermal power plant k
$\overline{P}_k$	maximum power output of thermal power plant k
SUC_k	start-up cost of thermal power plant k
MUT_k	minimum up time of thermal power plant k
MDT_k	minimum down time of thermal power plant k
$\gamma_{k,c,w0}$	initial status of thermal power plant k in cluster c in scenario w
hh_t	hour of the day (from 1 to 24) related to hour t (e.g. $hh_{25} = 1, hh_{48} = 24$)
$IC_{z,y}^{sol}$	investment cost of new solar power capacity in zone z in year y
$sol_{z,0}$	solar power capacity installed in zone z at the beginning of the planning horizon
$\underline{PV}_{z,y}$	lower bound for solar power capacity in zone z in year y
$\overline{PV}_{z,y}$	upper bound for solar power capacity in zone z in year y
$IC_{z,y}^{wind}$	investment cost of new wind power capacity in zone z in year y
$wind_{z,0}$	wind power capacity installed in zone z at the beginning of the planning horizon
$\underline{W}_{z,y}$	lower bound for wind power capacity in zone z in year y
$\overline{W}_{z,y}$	upper bound for wind power capacity in zone z in year y
$D_{z,t}$	load in zone z in hour t
$R_{z,t}$	reserve requirement for zone z in hour t
$\mu_{z,t}$	solar power capacity factor for zone z in hour t
$\rho_{z,t}$	wind power capacity factor for zone z in hour t
$IC_{h,y}^{hydro}$	investment cost of new hydropower plant $h \in H_C$ in year y
$Cvar_h$	operating cost of hydropower plant h
$\underline{\tau}_h$	earliest date for construction of hydropower plant h
$\overline{\tau}_h$	latest date for construction of hydropower plant h
$E_{h,c0}$	initial energy content of hydropower plant h in cluster c
$E_{h,c24}$	final energy content of hydropower plant h in cluster c
$F_{h,t}$	hourly energy inflows for hydropower plant h at time t
λ_h	loss coefficient for energy stored by hydropower plant h $(0 \leq \lambda_h \leq 1)$
λ_h^{in}	loss coefficient for hydroplant h pumping $(0 \leq \lambda_h^{in} \leq 1)$
λ_h^{out}	loss coefficient for hydroplant h power generation $(\lambda_h^{out} \geq 1)$

$\overline{E}_h^{in}$	upper bound on hydropower plant h pumping power
$\overline{E}_h^{out}$	upper bound on hydropower plant h power output
EPR_h	maximum energy to power ratio (in hours) for hydropower plant h
$IC_{b,y}^{batt}$	investment cost of battery b in year y
$Cvar_b$	operating cost of battery b
$\overline{CAP}_b$	upper bound on battery b installed capacity
λ_b	loss coefficient for energy stored by battery b ($0 \leq \lambda_b \leq 1$)
λ_b^{in}	loss coefficient for battery b charge ($0 \leq \lambda_b^{in} \leq 1$)
λ_b^{out}	loss coefficient for battery b discharge ($\lambda_b^{out} \geq 1$)
$\overline{E}_b^{in}$	upper bound on battery b charge
$\overline{E}_b^{out}$	upper bound on battery b discharge
EPR_b	maximum energy to power ratio (in hours) for battery b
IC_l^{line}	investment cost of candidate line $l \in L_C$
$\underline{\tau}_l$	earliest date for construction of candidate line l
$\overline{\tau}_l$	latest date for construction/of candidate line l
$\underline{F}_l$	minimum capacity of line l
$\overline{F}_l$	maximum capacity of line l
HR_k	heat rate of thermal power plant k
$ECnt_f$	energy content of fuel f
co_{2f}	CO_2 emission factor of fuel f
$\overline{FA}_{f,m,y}$	upper bound on availability of fuel f for macro-area m in year y
$\overline{CO}_{2m,y}$	CO_2 emission limit for macro-area m in year y
$\varphi_{m,y}$	lower bound for renewables penetration in macro-area m in year y

Decision Variables

$sol_{z,y}$	new solar capacity installed in zone z in year y
$wind_{z,y}$	new wind capacity installed in zone z in year y
$cap_{b,y}$	storage capacity of battery b installed in year y
$\delta_{k,y}^-$	1: thermal power plant $k \in K_E$ is decommissioned in year y; 0: otherwise
$\delta_{k,y}^+$	1: thermal power plant $k \in K_C$ is built in year y; 0: otherwise
$\delta_{h,y}$	1: hydropower plant $h \in H_C$ is built in year y; 0: otherwise
$\delta_{l,y}$	1: line $l \in L_C$ is built in year y; 0: otherwise
$\theta_{k,y}^-$	1: thermal power plant $k \in K_E$ is decommissioned within year y; 0: otherwise
$\theta_{k,y}^+$	1: thermal power plant $k \in K_C$ is built within year y; 0: otherwise
$\theta_{h,y}$	1: hydropower plant $h \in H_C$ is built within year y; 0: otherwise
$\theta_{l,y}$	1: line $l \in L_C$ is built within year y; 0: otherwise
$p_{k,t,w}$	power production of thermal power plant k in hour t in scenario w above its minimum output $\underline{P}_k$
$E_{h,t,w}^{in}$	pumping power of hydro reservoir h in hour t in scenario w
$E_{h,t,w}^{out}$	power output of hydro reservoir h in hour t in scenario w
$E_{h,t,w}$	energy level of hydro reservoir h in hour t in scenario w

$sl_{h,t,w}$	spillage from hydro reservoir h in hour t in scenario w
$E^{in}_{b,t,w}$	charge of battery b in hour t in scenario w
$E^{out}_{b,t,w}$	discharge of battery b in hour t in scenario w
$E_{b,t,w}$	energy level of battery b in hour t in scenario w
$x_{l,t,w}$	energy flow on transmission line l in hour t in scenario w
$ENP_{z,t,w}$	energy not provided in zone z in hour t in scenario w
$OG_{z,t,w}$	over-generation in zone z in hour t in scenario w
$RES_{z,t,w}$	renewable generation in zone z in hour t in scenario w
$\gamma_{k,t,w}$	1: thermal power plant k is ON in hour t in scenario w; 0: otherwise
$\alpha_{k,t,w}$	1: thermal power plant k is started up in hour t in scenario w; 0: otherwise
$\beta_{k,t,w}$	1: thermal power plant k is shut down in hour t in scenario w; 0: otherwise

Mathematical Model

$$
\begin{aligned}
min\ z = &\sum_{y\in Y}\left[\sum_{k\in K_E}\frac{DC_k\delta^-_{k,y}}{(1+r)^{y-y_0}}+\sum_{k\in K_C}\frac{IC^{th}_k\delta^+_{k,y}}{(1+r)^{y-y_0}}\right]\\
&+\sum_{y\in Y}\left[\sum_{z\in Z}\frac{IC^{sol}_{z,y}sol_{z,y}}{(1+r)^{y-y_0}}+\sum_{z\in Z}\frac{IC^{wind}_{z,y}wind_{z,y}}{(1+r)^{y-y_0}}\right]\\
&+\sum_{y\in Y}\sum_{h\in H_C}\frac{IC^{hydro}_{h,y}\delta_{h,y}}{(1+r)^{y-y_0}}+\sum_{y\in Y}\sum_{b\in B}\frac{IC^{batt}_{b,y}cap_{b,y}}{(1+r)^{y-y_0}}+\sum_{y\in Y}\sum_{l\in L_C}\frac{IC^{line}_l\delta_{l,y}}{(1+r)^{y-y_0}}\\
&+\sum_{y\in Y}\left[\sum_{k\in K_E}FC_k\left(1-\theta^-_{k,y}\right)+\sum_{k\in K_C}FC_k\theta^+_{k,y}\right]\\
&+\sum_{w\in W}pr_w\left[\sum_{y\in Y}\sum_{c|year(c)=y}wg_c\sum_{t|cl(t)=c}\left(\sum_{k\in K}CM_{k,y,w}\left(\underline{P}_k\gamma_{k,t,w}+p_{k,t,w}\right)\right.\right.\\
&\qquad+\sum_{k\in K}SUC_k\alpha_{k,t,w}+\sum_{h\in H}Cvar_hE^{out}_{h,t,w}+\sum_{b\in B}Cvar_bE^{out}_{b,t,w}\\
&\qquad\left.\left.+c_{ENP}\sum_{z\in Z}ENP_{z,t,w}+c_{OG}\sum_{z\in Z}OG_{z,t,w}\right)\right]
\end{aligned}
\tag{1}
$$

subject to

$$\delta^-_{k,y}=0 \qquad k\in K_E,\ \ y\notin\left[\underline{\tau}_k,\overline{\tau}_k\right] \tag{2}$$

$$\delta^+_{k,y}=0 \qquad k\in K_C,\ \ y\notin\left[\underline{\tau}_k,\overline{\tau}_k\right] \tag{3}$$

$$\delta_{h,y}=0 \qquad h\in H_C,\ \ y\notin\left[\underline{\tau}_h,\overline{\tau}_h\right] \tag{4}$$

$$\delta_{l,y}=0 \qquad l\in L_C,\ \ y\notin\left[\underline{\tau}_l,\overline{\tau}_l\right] \tag{5}$$

$$\underline{PV}_{z,y}\le sol_{z,0}+\sum_{i=1}^{y}sol_{z,i}\le\overline{PV}_{z,y} \qquad z\in Z,\ y\in Y \tag{6}$$

$$\underline{W}_{z,y} \leq wind_{z,0} + \sum_{i=1}^{y} wind_{z,i} \leq \overline{W}_{z,y} \qquad z \in Z,\ y \in Y \tag{7}$$

$$\sum_{y \in Y} cap_{b,y} \leq \overline{CAP}_b \qquad b \in B \tag{8}$$

$$\theta^{-}_{k,y} = \sum_{i=1}^{y} \delta^{-}_{k,y} \qquad k \in K_E,\ y \in Y \tag{9}$$

$$\theta^{+}_{k,y} = \sum_{i=1}^{y} \delta^{+}_{k,y} \qquad k \in K_C,\ y \in Y \tag{10}$$

$$\theta_{h,y} = \sum_{i=1}^{y} \delta_{h,y} \qquad h \in H_C,\ y \in Y \tag{11}$$

$$\theta_{l,y} = \sum_{i=1}^{y} \delta_{l,y} \qquad l \in L_C,\ y \in Y \tag{12}$$

$$\gamma_{k,t,w} \leq 1 - \theta^{-}_{k,y} \qquad k \in K_E,\ t \in T,\ y = yr(t),\ w \in W \tag{13}$$

$$\gamma_{k,t,w} \leq \theta^{+}_{k,y} \qquad k \in K_C,\ t \in T,\ y = yr(t),\ w \in W \tag{14}$$

$$\gamma_{k,t,w} \leq 1 - \theta^{+}_{UP(k),y} \qquad k \in K_E,\ t \in T,\ y = yr(t),\ w \in W \tag{15}$$

$$0 \leq p_{k,t,w} \leq (\overline{P}_k - \underline{P}_k)\gamma_{k,t,w} \qquad k \in K,\ t \in T,\ w \in W \tag{16}$$

$$\gamma_{k,t,w} - \gamma_{k,t-1,w} = \alpha_{k,t,w} - \beta_{k,t,w} \qquad k \in K,\ t : hh_t > 1,\ w \in W \tag{17}$$

$$\gamma_{k,t,w} - \gamma_{k,c,w_0} = \alpha_{k,t,w} - \beta_{k,t,w} \qquad k \in K,\ t : hh_t = 1,\ c = cl(t),\ w \in W \tag{18}$$

$$\sum_{i=t-MUT_k+1}^{t} \alpha_{k,i,w} \leq \gamma_{k,t,w} \qquad k \in K,\ \ t : hh_t \geq MUT_k,\ w \in W \tag{19}$$

$$\sum_{i=t-MDT_k+1}^{t} \beta_{k,i,w} \leq 1 - \gamma_{k,t,w} \qquad k \in K,\ \ t : hh_t \geq MDT_k,\ w \in W \tag{20}$$

$$0 \leq E^{in}_{h,t,w} \leq \overline{E}^{in}_h \qquad h \in H_E,\ \ t \in T,\ w \in W \tag{21}$$

$$0 \leq E^{out}_{h,t,w} \leq \overline{E}^{out}_h \qquad h \in H_E,\ \ t \in T,\ w \in W \tag{22}$$

$$0 \leq E_{h,t,w} \leq EPR_h \overline{E}^{in}_h \qquad h \in H_E,\ \ t \in T,\ w \in W \tag{23}$$

$$0 \leq E^{in}_{h,t,w} \leq \overline{E}^{in}_h \theta_{h,y} \qquad h \in H_C,\ \ t \in T,\ y = yr(t),\ w \in W \tag{24}$$

$$0 \leq E^{out}_{h,t,w} \leq \overline{E}^{out}_h \theta_{h,y} \qquad h \in H_C,\ \ t \in T,\ y = yr(t),\ w \in W \tag{25}$$

$$0 \leq E_{h,t,w} \leq EPR_h\, \overline{E}^{in}_h\, \theta_{h,y} \qquad h \in H_C,\ \ t \in T,\ y = yr(t),\ w \in W \tag{26}$$

$$\begin{aligned} E_{h,t,w} &= (1 - \lambda_h) E_{h,t-1,w} + F_{h,t} \\ &+ \lambda^{in}_h E^{in}_{h,t,w} - \lambda^{out}_h E^{out}_{h,t,w} - sl_{h,t,w} \end{aligned} \qquad h \in H,\ \ t : hh_t > 1,\ w \in W \tag{27}$$

$$\begin{aligned} E_{h,t,w} &= (1 - \lambda_h) E_{h,c_0} + F_{h,t} \\ &+ \lambda^{in}_h E^{in}_{h,t,w} - \lambda^{out}_h E^{out}_{h,t,w} - sl_{h,t,w} \end{aligned} \qquad h \in H,\ \ t : hh_t = 1,\ c = cl(t),\ w \in W \tag{28}$$

$$E_{h,t,w} = E_{h,c_{24}} \qquad h \in H,\ \ t : hh_t = 24,\ c = cl(t),\ w \in W \tag{29}$$

$$0 \leq E^{in}_{b,t,w} \leq \sum_{i=1}^{y} cap_{b,i} \qquad b \in B,\ \ t \in T,\ y = yr(t),\ w \in W \tag{30}$$

$$0 \leq E^{out}_{b,t,w} \leq \sum_{i=1}^{y} cap_{b,i} \qquad b \in B,\ t \in T,\ y = yr(t),\ w \in W \tag{31}$$

$$0 \leq E_{b,t,w} \leq EPR_b \sum_{i=1}^{y} cap_{b,i} \qquad b \in B,\ t \in T,\ y = yr(t),\ w \in W \tag{32}$$

$$E_{b,t,w} = (1 - \lambda_b)E_{b,t-1,w} + \lambda^{in}_b E^{in}_{b,t,w} - \lambda^{out}_b E^{out}_{b,t,w} \qquad b \in B,\ t : hh_t > 1,\ w \in W \tag{33}$$

$$E_{b,t,w} = (1 - \lambda_b)E_{b,c0} + \lambda^{in}_b E^{in}_{b,t,w} - \lambda^{out}_b E^{out}_{b,t,w} \qquad b \in B,\ t : hh_t = 1,\ c = cl(t),\ w \in W \tag{34}$$

$$E_{b,t,w} = E_{b,c0} \qquad b \in B,\ t : hh_t = 24,\ c = cl(t),\ w \in W \tag{35}$$

$$\underline{F}_l \leq x_{l,t,w} \leq \overline{F}_l \qquad l \in L_E,\ t \in T,\ w \in W \tag{36}$$

$$\underline{F}_l \theta_{l,y} \leq x_{l,t,w} \leq \overline{F}_l \theta_{l,y} \qquad l \in L_C,\ t \in T,\ w \in W \tag{37}$$

$$\begin{aligned}
&\sum_{k \in \Omega^k_z} \left(\underline{P}_k \gamma_{k,t,w} + p_{k,t,w}\right) + \mu_{z,t} \left(sol_{z,0} + \sum_{i=1}^{yr(t)} sol_{z,i}\right) \\
&+ \rho_{z,t} \left(wind_{z,0} + \sum_{i=1}^{yr(t)} wind_{z,i}\right) + \sum_{l|r(l)=z} x_{l,t,w} \\
&+ \sum_{h \in \Omega^h_z} E^{out}_{h,t,w} + \sum_{b \in \Omega^b_z} E^{out}_{b,t,w} + ENP_{z,t,w} = D_{z,t} \\
&+ \sum_{l|s(l)=z} x_{l,t,w} + \sum_{h \in \Omega^h_z} E^{in}_{h,t,w} + \sum_{b \in \Omega^b_z} E^{in}_{b,t,w} + OG_{z,t,w}
\end{aligned} \qquad z \in Z,\ t \in T,\ w \in W \tag{38}$$

$$\begin{aligned}
&\sum_{k \in K_E \cap \Omega^k_z} \left(\overline{P}_k (1 - \theta^-_{k,y}) - \underline{P}_k \gamma_{k,t,w} - p_{k,t,w}\right) + \\
&\sum_{k \in K_C \cap \Omega^k_z} \left(\overline{P}_k \theta^+_{k,y} - \underline{P}_k \gamma_{k,t,w} - p_{k,t,w}\right) \geq R_{z,t}
\end{aligned} \qquad z \in Z,\ t \in T,\ w \in W \tag{39}$$

$$\begin{aligned}
&\sum_{z|ma(z)=m} \sum_{f \in F} \sum_{k \in \Phi_{z,f}} \sum_{c|year(c)=y} wg_c \\
&\sum_{t|cl(t)=c} HR_k(\underline{P}_k \gamma_{k,t,w} + p_{k,t,w}) co2_f \leq \overline{CO2}_{m,y}
\end{aligned} \qquad m \in M,\ y \in Y,\ w \in W \tag{40}$$

$$\begin{aligned}
&\sum_{z|ma(z)=m} \sum_{k \in \Phi_{z,f}} \sum_{c|year(c)=y} wg_c \\
&\sum_{t|cl(t)=c} \frac{HR_k(\underline{P}_k \gamma_{k,t,w} + p_{k,t,w})}{ECnt_f} \leq \overline{FA}_{f,m,y}
\end{aligned} \qquad f \in F,\ m \in M,\ y \in Y,\ w \in W \tag{41}$$

$$RES_{z,t,w} = \mu_{z,t}\left(sol_{z,0} + \sum_{i=1}^{yr(t)} sol_{z,i}\right) + \rho_{z,t}\left(wind_{z,0} + \sum_{i=1}^{yr(t)} wind_{z,i}\right) + \sum_{h\in\Omega_z^h} E^{out}_{h,t,w} \qquad z \in Z,\ t \in T,\ w \in W \qquad (42)$$

$$\sum_{z|ma(z)=m}\ \sum_{c|year(c)=y} wg_c \sum_{t|cl(t)=c} RES_{z,t,w} \geq \varphi_{m,y} \sum_{z|ma(z)=m}\ \sum_{c|year(c)=y} wg_c \sum_{t|cl(t)=c} D_{z,t} \qquad m \in M,\ y \in Y,\ w \in W \qquad (43)$$

$$\delta^-_{k,y}, \theta^-_{k,y} \in \{0,1\} \qquad k \in K_E,\ \ y \in Y \qquad (44)$$

$$\delta^+_{k,y}, \theta^+_{k,y} \in \{0,1\} \qquad k \in K_C,\ \ y \in Y \qquad (45)$$

$$\delta_{h,y}, \theta_{h,y} \in \{0,1\} \qquad h \in H_C,\ \ y \in Y \qquad (46)$$

$$\delta_{l,y}, \theta_{l,y} \in \{0,1\} \qquad l \in L_C,\ \ y \in Y \qquad (47)$$

$$sol_{z,y}, wind_{z,y} \geq 0 \qquad z \in Z,\ \ y \in Y \qquad (48)$$

$$cap_{b,y} \geq 0 \qquad b \in B,\ \ y \in Y \qquad (49)$$

$$\alpha_{k,t,w}, \beta_{k,t,w}, \gamma_{k,t,w} \in \{0,1\} \qquad k \in K,\ \ t \in T,\ \ w \in W \qquad (50)$$

$$ENP_{z,t,w}, OG_{z,t,w}, RES_{z,t,w} \geq 0 \qquad z \in Z,\ \ t \in T,\ \ w \in W. \qquad (51)$$

The objective function (1) comprises the seven terms below:

1. $\sum_{y\in Y}\left[\sum_{k\in K_E} \frac{DC_k\delta^-_{k,y}}{(1+r)^{y-y_0}} + \sum_{k\in K_C} \frac{IC_k^{th}\delta^+_{k,y}}{(1+r)^{y-y_0}}\right]$ are the annualized decommissioning costs of existing thermal power plants and investment costs in new thermal power generation;
2. $\sum_{y\in Y}\left[\sum_{z\in Z} \frac{IC^{sol}_{z,y} sol_{z,y}}{(1+r)^{y-y_0}} + \sum_{z\in Z} \frac{IC^{wind}_{z,y} wind_{z,y}}{(1+r)^{y-y_0}}\right]$ are the annualized investment costs in new solar and wind capacity;
3. $\sum_{y\in Y}\sum_{h\in H_C} \frac{IC^{hydro}_{h,y}\delta_{h,y}}{(1+r)^{y-y_0}}$ are the annualized investment costs in new hydropower plants;
4. $\sum_{y\in Y}\sum_{b\in B} \frac{IC^{batt}_{b,y} cap_{b,y}}{(1+r)^{y-y_0}}$ are the annualized investment costs in new batteries capacity;
5. $\sum_{y\in Y}\sum_{l\in L_C} \frac{IC^{line}_l \delta_{l,y}}{(1+r)^{y-y_0}}$ are the annualized investment costs in new transmission lines;
6. $\sum_{y\in Y}\left[\sum_{k\in K_E} FC_k\left(1-\theta^-_{k,y}\right) + \sum_{k\in K_C} FC_k\theta^+_{k,y}\right]$ are the fixed costs for the available thermal power plants, i.e., not decommissioned existing plants and already constructed candidate plants;
7. $\sum_{w\in W} pr_w \Big[\sum_{y\in Y}\sum_{c|year(c)=y} wg_c \sum_{t|cl(t)=c}\big(\sum_{k\in K} CM_{k,y,w}\big(\underline{P}_k\gamma_{k,t,w} + p_{k,t,w}\big) + \sum_{k\in K} SUC_k\alpha_{k,t,w} + \sum_{h\in H} Cvar_h E^{out}_{h,t,w} + \sum_{b\in B} Cvar_b E^{out}_{b,t,w} + c_{ENP}\sum_{z\in Z} ENP_{z,t,w} + c_{OG}\sum_{z\in Z} OG_{z,t,w}\big)\Big]$ are the second-stage costs, i.e., the operating costs.

Specifically, item 7 above considers for each representative day the sum of production costs, start-up costs, hydro and batteries operational costs and penalties for energy not provided and over-generation. Production costs are supposed to be linear functions of the power output, being $CM_{k,y,w}$ the slopes of these linear relationships. In each scenario w, the marginal cost of thermal plant k in year y is computed as:

$$CM_{k,y,w} = OM_k + HR_k(Price_{y,w}^{fuel(k)} + co_{2\,fuel(k)} Price_{y,w}^{co_2}) \tag{52}$$

with OM_k [€/MWh] being the operative and maintenance costs of plant k, HR_k [Gcal/MWh] the heat rate of thermal power plant k, $Price_{y,w}^{fuel(k)}$ [€/Gcal] the price in year y under scenario w of fuel used by unit k, $co_{2\,fuel(k)}$ [ton/Gcal] the CO$_2$ emission factor of fuel used by unit k and $Price_{y,w}^{co_2}$ [€/ton] the emission cost in year y under scenario w.

In the proposed model there are three groups of constraints, namely investment constraints (2)–(12), operational constraints (13)–(39) and target constraints (40)–(43). Investment constraints model decommissioning and investment decisions. In particular, assignment constraints (2) impose earliest and latest dates for decommissioning of existing thermal power plants. Similarly, constraints (3)–(5) impose a temporal window for investment decisions on candidate thermal plants, hydropower plants and lines, respectively. Inequalities (6) and (7) enforce lower and upper bounds on solar and wind installed capacity, respectively. Inequalities (8) impose an upper bound on batteries installed capacity at the end of the planning horizon. Finally, equations (9)–(12) compute the values of the binary variables $\theta_{k,y}^-$, $\theta_{k,y}^+$, $\theta_{h,y}$ and $\theta_{l,y}$ that express if decommissioning or investment decisions have been made within every year y of the planning horizon.

Operational constraints express all the technical conditions for operating thermal and hydropower plants, transmission lines and storages and consider the flexibility provided to the energy system by the hydro-thermal dispatch and the storage units. In particular, the block of equations (13)–(20) models the thermal component of the energy system. Constraints (13)–(15) ensure consistency between the binary variables representing the commitment status and those representing the decommissioning and investment decisions. Specifically, inequalities (13) force the existing power plants decommissioned within year y to be offline in all hours after decommission. Constraints (14) impose that projects built within year y can be used to supply load, while thermal units not yet constructed are forced to be offline in all hours of year y. Inequalities (15) model the reinforcements of existing thermal power plants. Specifically, let $k' = UP(k)$ denote the new project that replaces the existing unit k when it starts operating: building project k' within year y implies the permanent offline status of unit k. Inequalities (16) state that the non-negative variables $p_{k,t,w}$ representing the power output above the minimum power output $\underline{P}_k$ are either bounded above by $\overline{P}_k - \underline{P}_k$, if unit k is online, or zero if unit k is offline. Constraints (17) enforce consistency between the binary variables representing start-up, shut down and status in adjacent hours in all hours of the representative days, except the first hour of the day. Constraints (18) enforce consistency between statuses and maneuvers for the first hour of every representative day, where the parameter $\gamma_{k,c,w0}$ represents the status of unit k at the beginning of representative day c under scenario w. The minimum uptime constraints (19) impose that unit k can be started-up at most once in an interval of MUT_k consecutive time periods. For each representative day, the minimum uptime constraints are enforced on the hours in the range from MUT_k to the final (the 24th) hour of the day, to keep the representative days separate in the description of future scenarios. The minimum downtime constraints (20) work similarly to the shut-down of thermal power plants. Constraints (21)–(29) model the operation of hydro plants. Specifically, constraints (21)–(23) bound the non-negative variables corresponding to the pumping power, the power output and the energy level of the existing hydroelectric reservoirs below their respective upper limits. If candidate hydro plant $h \in H_c$ is built within year y, inequalities (24)–(26) define the upper bounds to pumping power, power output and energy level of new hydroelectric reservoirs, otherwise set the corresponding variables to zero. Energy balances (27) apply to all hours but the first of the representative days: they ensure that the energy stored by hydro plant h at the end of hour t equals the energy stored at the end of hour $t-1$ (reduced by the loss coefficient $\lambda_h \leq 1$), plus the natural inflows, plus the energy injected in h (reduced by the coefficient $\lambda_h^{in} \leq 1$), minus the energy released from the reservoir (reduced by the coefficient $\lambda_h^{out} \geq 1$), minus the spillage. Equations (28) impose the energy balance for the first hour of each representative day, while equations (29) set the energy levels of each reservoir h at the end of each representative day to the value $E_{h,c24}$.

Constraints (30)–(35) model the operation of batteries. Specifically, inequalities (30)–(32) impose upper bounds to non-negative variables representing charge, discharge and stored energy and enforce consistency between the values of the first-stage and second-stage variables. Similarly to constraints (27) and (28), equations (33) and (34) impose energy balances for batteries. Constraints (35) state the equality for each battery b between energy levels at the beginning and the end of each representative day: in this model, batteries are supposed to have a cycle of 24 h. Inequalities (36) restrict the energy flows on the existing transmission lines. Constraints (37) impose lower and upper bounds to the power exchanges among zones and enforce consistency between the energy flows on candidate transmission lines and the binary variables related to investment decisions, not allowing energy flows on candidate lines which have not been built. The zonal balance equations (38) impose equality between energy sources and use in every zone and every hour. Specifically, the hourly energy sources of zone z are given by thermal, solar and wind generation, incoming energy flows, hydro generation and energy released by batteries (the left-hand side of the equations), while and the energy uses are represented by the load, outgoing energy flows, pumping power and energy absorbed by batteries (the right-hand side of the equations). The variables $ENP_{z,t,w}$ and $OG_{z,t,w}$ allow detecting and evaluating problems in the simulated system that can cause a mismatch between supply and demand. Inequalities (39) ensure the fulfillment of zonal reserve requirements provided by available thermal plants.

Target constraints (40)–(43) model decarbonization targets. Specifically, inequalities (40) impose limits for thermal energy production due to CO_2 emissions. These constraints are imposed for each macro-area m, each year y, and each scenario w. In particular, they are computed by multiplying the daily CO_2 emissions under scenario w in all the zones belonging to macro-area m by the weight of each cluster, to take into account the different occurrences of representative days. Inequalities (41) impose limits on thermal power generation employing fossil fuels whose availability could be limited in time. Equations (42) compute the zonal hourly renewable production, by considering solar, wind and hydroelectric generation. Constraints (43) control the renewable penetration, forcing the total renewable generation in macro-area m in year y to cover at least ratio $\varphi_{m,y}$ of the total yearly demand for electricity. Finally, constraints (44)–(51) define the binary variables and the non-negative variables whose lower bound has not been specified in the previous equations.

Given the high dimensionality, providing a solution to the proposed model is computationally challenging. In our numerical experiments, we applied a multi-cut Benders decomposition algorithm, decomposing the stochastic model both by year and by scenario, and solving at each iteration master problem and subproblems with solver Gurobi under GAMS 24.7.4. We refer the reader to [22] for the details about solution algorithm and computational times.

4 Conclusions

In this paper, a detailed description of the GTEP problem is provided, by highlighting the main characteristics of the problem and by reviewing the most relevant works developed in the literature. First, the different modeling choices associated with the design of a GTEP model are presented, highlighting which approaches are suited to address specific research questions. In general, performing a GTEP analysis implies dealing with large-scale models, whose solution could be computationally challenging. Thus, when designing a model for GTEP a trade-off between modeling accuracy and computational tractability has always to be considered.

Then, GTEP models are classified according to the way uncertainty is represented in the expansion planning framework. Indeed, given the long-term planning horizon, a high level of uncertainty affects investments in the power sector. Including this uncertainty in the GTEP analysis provides more solid results for decision makers.

This paper discusses also the need to evaluate short-term operational condition by considering representative days, so as to catch the fluctuation of intermittent renewable power sources and properly evaluate the flexibility needs of power systems, while maintaining the problem computationally tractable. Therefore, different approaches to select representative days are compared, discussing also the challenges related to these methods.

Finally, a GTEP model designed to reach the decarbonization targets in the power sector is presented in the paper. Such model co-optimizes strategic and operational decisions for transmission, generation and storage facilities, providing a very detailed representation of the power system and including constraints to limit the CO_2 emissions and to increase the generation from renewable power sources. However, this model does not consider the interactions between the electricity and the gas systems, which is a very relevant topic for modern power systems, given the increasing deployment of gas-fueled thermal units and Power-to-Gas plants. The integration of electricity and gas systems will be investigated in a future work.

References

1. D. Huppmann and R. Egging-Bratseth, “Market power, fuel substitution and infrastructure – A large-scale equilibrium model of global energy markets,” *Energy*, vol. 75, no. C, pp. 483–500, 2014.
2. P. Seljom and A. Tomasgard, “Short-term uncertainty in long-term energy system models – A case study of wind power in Denmark,” *Energy Economics*, vol. 49, pp. 157–167, 2015.
3. L. Garver, “Transmission network estimation using linear programming,” *IEEE Transactions on Power Apparatus and Systems*, vol. 89, no. 7, pp. 1686–1697, 1970.
4. Y. Gu, J. McCalley, and M. Ni, “Coordinating large-scale wind integration and transmission planning,” *IEEE Transactions on Sustainable Energy*, vol. 3, no. 4, pp. 652–659, 2011.
5. H. Haghighat and B. Zneg, “Bilevel conic transmission expansion planning,” *IIEEE Transactions on Power Systems*, vol. 33, no. 4, pp. 4640–4642, 2018.
6. R. Hemmati, H. Saboori, and M. Jirdehi, “Multistage generation expansion planning incorporating large scale energy storage systems and environmental pollution,” *Renewable Energy*, vol. 97, pp. 636–645, 2016.
7. J. Meza, M. Yildirim, and A. Masud, “A model for the multiperiod multiobjective power generation expansion problem,” *IEEE Transactions on Power Systems*, vol. 22, no. 2, pp. 871–878, 2007.
8. S. Kazempour, A. Conejo, and C. Ruiz, “Strategic generation investment using a complementarity approach,” *IEEE Transactions on Power Systems*, vol. 26, no. 2, pp. 940–948, 2011.
9. V. Krishnan, J. Ho, B. Hobbs, A. Liu, J. McCalley, M. Shahidehpour, and Q. Zheng, “Co-optimization of electricity transmission and generation resources for planning and policy analysis: review of concepts and modeling approaches,” *Energy Systems*, vol. 7, no. 2, pp. 297–332, 2016.
10. D. Pozo, E. Sauma, and J. Contreras, “A three-level static MILP model for generation and transmission expansion planning,” *IEEE Transactions on Power Systems*, vol. 28, pp. 201–210, 2013.
11. Y. Tohidi, M. Hesamzadeh, and F. Regairaz, “Sequential coordination of transmission expansion planning with strategic generation investments,” *IEEE Transactions on Power Systems*, vol. 32, no. 4, pp. 2521–2534, 2017.
12. E. Sauma and S. Oren, “Economic criteria for planning transmission investment in restructured electricity markets,” *IEEE Transactions on Power Systems*, vol. 22, no. 4, pp. 1394–1405, 2007.
13. D. Pozo, J. Contreras, and E. Sauma, “If you build it, he will come: Anticipative power transmission planning,” *Energy Economics*, vol. 36, pp. 135–146, 2013.
14. J. Aghaei, N. Amjady, A. Baharvandi, and M. Akbari, “Generation and transmission expansion planning: MILP-based probabilistic model,” *IEEE Transactions on Power Systems*, vol. 29, pp. 1592–1601, 2014.
15. S. Moghaddam, “Generation and transmission expansion planning with high penetration of wind farms considering spatial distribution of wind speed,” *International Journal of Electrical Power & Energy Systems*, vol. 106, pp. 232–241, 2019.
16. S. You, S. Hadley, M. Shankar, and Y. Liu, “Co-optimizing generation and transmission expansion with wind power in large-scale power grids – implementation in the us eastern interconnection,” *Electric Power System Research*, vol. 133, pp. 209–218, 2016.

17. B. Alizadeh and S. Jadid, “Reliability constrained coordination of generation and transmission expansion planning in power systems using mixed integer programming,” *IET Generation, Transmission & Distribution*, vol. 5, pp. 948–960, 2011.
18. B. Alizadeh and S. Jadid, “A dynamic model for coordination of generation and transmission expansion planning in power systems,” *International Journal of Electrical Power & Energy Systems*, vol. 65, pp. 408–418, 2015.
19. F. Fallahi, M. Nick, G. Riahy, S. Hosseinian, and A. Doroudi, “The value of energy storage in optimal non-firm wind capacity connection to power systems,” *Renewable Energy*, vol. 64, pp. 34–42, 2014.
20. C. Unsihuay-Vila, J. Marangon-Lima, A. Zambroni de Souza, and I. Perez-Arriaga, “Multistage expansion planning of generation and interconnections with sustainable energy development criteria: A multiobjective model,” *International Journal of Electrical Power & Energy Systems*, vol. 33, no. 2, pp. 258–270, 2011.
21. M. Hesamzadeh, J. Rosellon, and I. Vogelsang, *Transmission Network Investment in Liberalized Power Markets*. Springer, 2020.
22. G. Micheli, M. Vespucci, M. Stabile, C. Puglisi, and A. Ramos, “A two-stage stochastic MILP model for generation and transmission expansion planning with high shares of renewables,” *Energy Systems*, 2020.
23. A. Schwele, J. Kazempour, and P. Pinson, “Do unit commitment constraints affect generation expansion? A scalable stochastic model,” *Energy Systems*, 2019.
24. M. Fürsch, S. Hagspiel, C. Jägemann, S. Nagl, D. Lindenberger, and E. Tröster, “The role of grid extensions in a cost-efficient transformation of the european electricity system until 2050,” *Applied Energy*, vol. 104, pp. 642 – 652, 2013.
25. M. Moeini-Aghtaie, A. Abbaspour, and M. Fotuhi-Firuzabad, “Incorporating large-scale distant wind farms in probabilistic transmission expansion planning–part i: Theory and algorithm,” *IEEE Transactions on Power Systems*, vol. 27, no. 3, pp. 1585–1593, 2012.
26. H. Yu, C. Chung, K. Wong, and J. Zhang, “A chance constrained transmission network expansion planning method with consideration of load and wind farm uncertainties,” *IEEE Transactions on Power Systems*, vol. 24, no. 3, pp. 1568–1576, 2009.
27. G. Orfanos, P. Georgilakis, and N. Hatziargyriou, “Transmission expansion planning of systems with increasing wind power integration,” *IEEE Transactions on Power Systems*, vol. 28, no. 2, pp. 1355–1362, 2012.
28. C. Ruiz and A. Conejo, “Robust transmission expansion planning,” *European Journal of Operational Research*, vol. 242, no. 2, pp. 390–401, 2015.
29. Z. Li, J. Li, F. Liu, H. Ye, X. Zhang, S. Mei, and N. Chang, “Robust coordinated transmission and generation expansion planning considering ramping requirements and construction periods,” *IEEE Transactions on Power Systems*, vol. 33, no. 1, pp. 268–280, 2018.
30. S. Dehghan, N. Amjady, and K. Ahad, “Two-stage robust generation expansion planning: A mixed integer linear programming model,” *IEEE Transactions on Power Systems*, vol. 29, no. 2, pp. 584–597, 2014.
31. S. Chen, J. Wang, Y. He, and Z. Wang, “Robust optimization for transmission expansion planning: Minimax cost vs. minimax regret,” *IEEE Transactions on Power Systems*, vol. 29, no. 6, pp. 3069–3077, 2014.
32. D. Mejía-Giraldo and J. McCalley, “Maximizing future flexibility in electric generation portfolios,” *IEEE Transactions on Power Systems*, vol. 29, pp. 279–288, 2014.
33. P. Maloney, O. Olatujoye, A. Ardakani, D. Mejía-Giraldo, and J. McCalley, “A comparison of stochastic and adaptation programming methods for long term generation and transmission co-optimization under uncertainty,” in *2016 North American Power Symposium (NAPS), Denver, CO, USA*, 2016.
34. J. Birge and F. Louveaux, *Introduction to Stochastic Programming*. Springer, New York: Springer Series in Operations Research and Financial Engineering, 2011.
35. J. Alvarez Lopez, K. Ponnambalam, and V. Quintana, “Generation and transmission expansion under risk using stochastic programming,” *IEEE Transactions on Power Systems*, vol. 22, no. 3, pp. 1369–1378, 2007.

36. A. Van der Weijde and B. Hobbs, "The economics of planning electricity transmission to accommodate renewables: Using two-stage optimisation to evaluate flexibility and the cost of disregarding uncertainty," *Energy Economics*, vol. 34, no. 6, pp. 2089–2101, 2012.
37. S. Jin, S. Ryan, J. Watson, and D. Woodruff, "Modeling and solving a large-scale generation expansion planning problem under uncertainty," *Energy Systems*, vol. 2, pp. 209–242, 2011.
38. A. Conejo, L. Baringo, S. Kazempour, and A. Siddiqui, *Investment in electricity generation and transmission: Decision making under uncertainty*. Springer, 2016.
39. Y. Liu, R. Sioshansi, and A. Conejo, "Multistage stochastic investment planning with multiscale representation of uncertainties and decisions," *IEEE Transactions on Power Systems*, vol. 33, no. 1, pp. 781–791, 2018.
40. L. Baringo and A. Conejo, "Risk-constrained multi-stage wind power investment," *IEEE Transactions on Power Systems*, vol. 28, no. 1, pp. 401–411, 2013.
41. M. Fürsch, S. Nagl, and D. Lindenberg, "Optimization of power plant investments under uncertain renewable energy deployment paths: a multistage stochastic programming approach," *Energy Systems*, vol. 5, pp. 85–121, 2014.
42. K. Poncelet, E. Delarue, D. Six, J. Duerinck, and W. D'haeseleer, "Impact of the level of temporal and operational detail in energy-system planning models," *Applied Energy*, vol. 162, pp. 631–643, 2016.
43. A. Belderbos and E. Delarue, "Accounting for flexibility in power system planning with renewables," *International Journal of Electrical Power & Energy Systems*, vol. 71, pp. 33–41, 2015.
44. D. Kirschen, J. Ma, V. Silva, and R. Belhomme, "Optimizing the flexibility of a portfolio of generating plants to deal with wind generation," in *2011 IEEE Power and Energy Society General Meeting, Detroit, MI, USA*, 2011.
45. M. Fripp, "Switch: A planning tool for power systems with large shares of intermittent renewable energy," *Environmental Science & Technology*, vol. 46, no. 11, pp. 6371–6378, 2012.
46. E. Hart and M. Jacobson, "A monte carlo approach to generator portfolio planning and carbon emissions," *Renewable Energy*, vol. 36, no. 8, pp. 2278–2286, 2011.
47. M. Nick, R. Cherkaoui, and M. Paolone, "Optimal allocation of dispersed energy storage systems in active distribution networks for energy balance and grid support," *IEEE Transactions on Power Systems*, vol. 29, no. 5, pp. 2300–2310, 2014.
48. P. Nahmmacher, E. Schmid, L. Hirth, and B. Knopf, "Carpe diem: A novel approach to select representative days for long-term power system models with high shares of renewable energy sources," *Energy*, vol. 112, pp. 430–442, 2016.
49. M. ElNozahy, M. Salama, and R. Seethapathy, "A probabilistic load modelling approach using clustering algorithms," in *2013 IEEE Power & Energy Society General Meeting*, 2013.
50. S. Fazlollahi, S. Bungener, P. Mandel, G. Becker, and F. Maréchal, "Multi-objectives, multi-period optimization of district energy systems: I. selection of typical operating periods," *Computers & Chemical Engineering*, vol. 65, pp. 54–662, 2014.
51. K. Poncelet, H. Höschle, E. Delaru, A. Virag, and W. D'haeseleer, "Selecting representative days for capturing the implications of integrating intermittent renewables in generation expansion planning problems," *IEEE Transactions on Power Systems*, vol. 32, no. 3, pp. 1936–1948, 2017.
52. K. Poncelet, A. van Stiphout, J. Meus, E. Delarue, and W. D'haeseleer, "Lusym invest: a generation expansion planning model with a high level of temporal and technical detail," tech. rep., KU Leuven, 2018. TME Working Paper WP EN2018-07.
53. D. Tejada-Arango, M. Domeshek, S. Wogrin, and E. Centeno, "Enhanced representative days and system states modeling for energy storage investment analysis," *IEEE Transactions on Power Systems*, vol. 33, no. 6, pp. 6534–6544, 2018.
54. G. Micheli, M. Vespucci, M. Stabile, and A. Cortazzi, "Selecting and initializing representative days for generation and transmission expansion planning with high shares of renewables," in *Graphs and combinatorial optimization: from theory to applications - CTW2020 proceedings* (C. Gentile, G. Stecca, and P. Ventura, eds.), pp. 321–334, Springer, 2020.

Second Order Adjoints in Optimization

Noémi Petra and Ekkehard W. Sachs

Abstract Second order, Newton-like algorithms exhibit convergence properties superior to gradient-based or derivative-free optimization algorithms. However, deriving and computing second order derivatives—-needed for the Hessian-vector product in a Krylov iteration for the Newton step—often is not trivial. Second order adjoints provide a systematic and efficient tool to derive second derivative information. In this paper, we consider equality constrained optimization problems in an infinite-dimensional setting. We phrase the optimization problem in a general Banach space framework and derive second order sensitivities and second order adjoints in a rigorous and general way. We apply the developed framework to a partial differential equation-constrained optimization problem.

Keywords Adjoint-based methods · Second order adjoints · Optimization in infinite dimensions · Newton method · PDE-constrained optimization

1 Introduction

We consider second order methods for equality constrained optimization problems in an infinite-dimensional setting. These methods are locally very fast, but on the downside require substantial computational effort for each iteration. To alleviate this disadvantage, variants of Newton's methods have been developed, e.g., Newton-Shamanskii, where the Jacobian is fixed throughout iterations, or inexact Newton methods, where the Newton step is computed inexactly. In this paper we devote ourselves to an adjoint-based framework that extends from the usual gradient computation all the way to an efficient way to compute Hessian-vector products. This opens up a venue to use iterative solvers for the computation of the Newton step.

N. Petra (✉)
Department of Applied Mathematics, University of California, Merced, CA 95340, USA
e-mail: npetra@ucmerced.edu

E. W. Sachs
Department of Mathematics, Trier University, 54286 Trier, Germany
e-mail: sachs@uni-trier.de

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*, Springer Proceedings in Mathematics & Statistics 354, https://doi.org/10.1007/978-3-030-72040-7_10

Unlike in the case of approximating the Hessian-vector product by a finite-difference quotient, which carries issues about its accuracy, the use of second order adjoints do not exhibit this problem and can be computed to required accuracy, if needed.

Related work. The concept of second order adjoints can be found in different fields in the literature, also under different names, for example incremental adjoints and adjoint sensitivities. The earliest work using second order adjoints goes back to papers in the sixties [29, 32]. Second order adjoints have been widely applied especially to inverse problems governed by differential equations. Here we give a sample of the literature on applications to various optimization problems constrained by differential equations: seismology [41], structural optimization [20, 21, 27], unconstrained discrete-time and continuous-time optimal control problems with Bolza objective functions [13], ordinary differential equations (ODEs) and partial differential equations (PDEs)-constrained optimization problems in the context of air traffic flow [38], optimization problems governed by PDEs with inequality constraints [16, 17] or parabolic PDEs [6], optimal semiconductor design based on the standard drift diffusioxn model [26], open loop optimal control problems governed by the two-dimensional stationary Navier-Stokes equations [25], calculation of directional derivatives of stiff ODE embedded functionals [35], data assimilation for numerical weather prediction and ocean models [2, 3, 11, 12, 31, 40, 42], inversion of the initial concentration of the airborne contaminant in a convection-diffusion transport model [1], full wave form or global seismic inversion [7, 8, 14, 15, 33], inverse ice sheet modeling [28, 34, 37, 43], in the context of optimal experimental design [4], and optimal control of systems governed by PDEs with random parameter fields [5, 10], and inexact Hessian-vector products computed using approximate second order adjoints [24]. It is also worth mentioning the following very useful technical reports targeting model (academic) problems [19, 22, 36].

Contributions. The derivation of second order adjoints in infinite-dimensions are often motivated using ad-hoc analytic arguments, hence there is a need for a rigorous in-depth investigation of such derivation and adjoint expressions. The main goal of this paper is therefore to present a general and rigorous theory for second order adjoints which then can be applied to various applications. In addition, when second order optimization algorithms (e.g., Newton's method or variants) are applied to solve large-scale optimization problems often iterative solvers are used for the solution of the systems of linearized equations. Therefore, motivated by the need to avoid the computation of explicit Hessian information, we show in the general framework how the adjoint calculus can be applied to compute Hessian-vector products. We note that since we have two well studied pathways to compute first-order derivative information, e.g., via a sensitivity approach or via an adjoint approach, there are four pathways to obtain the second order derivatives. In this paper, we present all these four pathways and show that in fact three of them are the same. Finally, we apply our framework to an inverse problem that seeks to reconstruct a coefficient field in an elliptic PDE from observational data. This problem is formulated as a nonlinear least squares optimization problem governed by the Poisson problem. In this paper, we use the proposed second order adjoint derivation and Hessian-vector product expression to derive derivative information that can be used by fast optimization methods to solve complex optimization problems. We note that in [1, 5–8, 10, 14, 15, 28, 34, 36, 37, 43] second order adjoints are also considered in an infinite-dimensional setting, however, using the Lagrangian approach. Here, we recast the problem as an unconstrained optimization problem and derive the Lagrange multipliers, i.e., adjoint variables, directly.

Problem formulation. To set the stage, we choose a general framework: Let Y, U, Z be Banach spaces, e.g., Y is the space variable or dependent variable, U the space of control or design variables and Z the range space of the equality constraint. The distinction of the variable into y and u is essential for our approach, but can be found in many applications, see e.g., PDE-constrained optimization. The optimization problem is formulated as follows:

Problem 1

$$\begin{array}{ll} \min & \phi(y,u), \quad (y,u) \in Y \times U, \\ \text{s.t.} & g(y,u) = 0, \\ \text{where } \phi : Y \times U \to I\!R, & g : Y \times U \to Z. \end{array} \tag{1}$$

If we assume that for each control variable u we have a unique solution $y = s(u)$ of the equality constraint $g(s(u), u) = 0$, then we can rewrite the constrained optimization problem as an unconstrained optimization problem:

$$\min \quad \Phi(u) = \phi(s(u), u), \quad u \in U, \quad \Phi : U \to I\!R.$$

In Newton's method, the correction step is defined as the solution of

$$\Phi''(u)v = -\Phi'(u), \quad v \in U, \tag{2}$$

where $\Phi'(u) \in U^*$ and $\Phi''(u) \in L(U, U^*)$, the space of linear operators mapping U into U^*. In many applications where the second derivative $\Phi''(u)$ is prohibitively expensive to compute in an explicit manner, the Eq. (2) is solved by an iterative technique, see for example Krylov methods [23]. In order to implement these methods efficiently, one needs a fast evaluation of the Hessian-vector product, in our notation $\Phi''(u)v$ for some $v \in U$. One way to achieve this is in the inexact Newton framework, where one uses the approximation

$$\Phi''(u)v \approx (\Phi'(u + hv) - \Phi'(u))/h. \tag{3}$$

The use of this approximation introduces an error at each iteration of the Krylov method which has to be handled with care, see e.g., [30].

However, if the optimization problem is of the type like shown in (1) there is another way of computing the Hessian-vector product. The reason for this lies in the fact that the variables are grouped into two groups y and u and the introduction of another adjoint variable, which we call the *second order adjoint*. This means that we have to solve two adjoint equations, i.e., equations of the type

$$g_y^* p = r_1, \qquad g_y^* \pi = r_2$$

for the first and second order adjoint variables p and π with different right hand sides r_1, r_2. The advantage of such an approach lies in the fact that we do not introduce an additional error by computing the Hessian-vector product exactly. Furthermore, the computational cost for the second order adjoint is not higher than an additional evaluation of the derivative $\Phi'(u + hv)$ as in (3).

It is well known that the derivative of the function $\phi(y, u)$ can be expressed in two ways, namely

– the sensitivity equations or
– the adjoint equations.

The first approach is considered reasonable for a small number of variables, whereas the second one requires a bit more analysis in the derivation, but shows to be highly efficient for large-scale problems [9].

If we turn to the second derivative applied to a vector as this required for iterative solvers like CG or GMRES, we are free to choose for this purpose again either the adjoint or sensitivity approach. Hence we have four different routes available which we could follow, namely

– first the sensitivity, then the adjoint approach or
– first the sensitivity, then the sensitivity approach or
– first the adjoint, then the sensitivity approach or
– first the adjoint, then the adjoint approach.

Content. In this paper we carefully analyze these four approaches and prove rigorously the results following these venues. It turns out that not four, but two different ways exist to compute the Hessian-vector products for this optimization problem. One approach is based on the sensitivity framework which is amenable for a small number of variables. The second one relies on the adjoint approach which leads to the concept of a second order adjoint that has to be computed for a Hessian-vector product. This approach is usually much more efficient, especially for problems with a large number of variables. The effort per iteration is comparable to if not lower than that of an inexact Newton's method where the matrix-vector multiplication is approximated by a finite difference quotient, yet it gives the precise result rather than an approximation.

The remaining sections of this paper are organized as follows. We begin by providing two venues to obtain representations of the first-order derivatives in Sect. 2. Next, in Sect. 3 we set the stage for the second order derivatives followed by the fourth section containing the results including proofs of the four approaches mentioned above. Section 5 is devoted to an application in PDE-constrained optimization, where we illustrate some of the theoretical results.

2 Representation of First-Order Fréchet-Derivative

For notational purposes we recall that a map $g : X \to Z$ from a Banach space X to another Banach space Z is called Fréchet-differentiable at $x \in X$, if there exists a linear operator denoted by $g'(x) : X \to Z$ such that

$$\|g(x+h) - g(x) - g'(x)h\|_Z \le \alpha(\|h\|_X)\|h\|_X,$$

with a function $\alpha(r)$ satisfying $\alpha(r) \to 0$ for $r \to 0$. The partial Fréchet-derivatives of e.g., $g(y,u)$ is denoted by $g_y(y,u)$ or $g_u(y,u)$ with a subscript indicating the variable with respect to which the derivative is taken. The adjoint operator of $g'(x)$ is denoted by $g'(x)^* : Y^* \to X^*$. We note that Fréchet-derivatives of second order like $g_{yu}(y,u)$ are linear operators in the spaces $L(U, L(Y,Z)) = L(U \times Y, Z)$.

In what follows, we impose the following smoothness assumptions on the functions in the problem formulation.

Assumption 1 *Let the function ϕ and the mapping g be continuously Fréchet-differentiable on $Y \times U$.*

Furthermore we assume the following constraint qualification to hold at a later to be specified point $(y,u) \in Y \times U$.

Assumption 2 *For $(y,u) \in Y \times U$ let the partial Fréchet-derivative $g_y(y,u) : Y \to Z$ be surjective and invertible.*

With these assumptions we can apply the implicit function theorem [39].

Theorem 1 *Let Assumptions 1 and 2 hold at $(y_*, u_*) \in Y \times U$. Then there exist neighborhoods $B_Y \subset Y$ at y_* and $B_U \subset U$ at u_* and a Fréchet-differentiable map $s : B_U \to B_Y$ such that*

$$g(s(u), u) = 0 \quad \textit{and} \quad g_y(s(u),u)s'(u) = -g_u(s(u),u). \tag{4}$$

This theorem can be used to reformulate the constrained optimization problem from above as an unconstrained optimization problem in a neighborhood around a local minimizer.

Corollary 1 *Let $(y_*, u_*) \in Y \times U$ be a local minimizer of optimization problem 1 and let Assumptions 1 and 2 hold at (y_*, u_*). Then u_* is also a local minimizer of the unconstrained optimization problem*

$$\min_{u \in B_U} \Phi(u), \quad \Phi(u) := \phi(s(u), u), \tag{5}$$

where $\Phi : B_U \to I\!R$, with $B_U \in U$ a neighborhood of u_*, and $s(u)$ given by the implicit function theorem.

In some applications the computation of $s(u)$ is theoretically possible, but numerically only feasible within a certain error tolerance. This happens, for example, for a constraint described by partial differential equations where $s(u)$ is the solution of a PDE. This would introduce an additional error into $s(u)$ and $s'(u)$ as well. The purpose of this paper is to derive in a rigorous way the expression for second derivatives without this additional complexity. This aspect, however, opens interesting venues for future research.

The necessary optimality conditions of first-order require various derivatives which are well defined under the statements above. Therefore the first derivative of the objective function of the unconstrained problem can be computed as follows. We note that this approach is usually denoted as the approach using the sensitivity equations.

Theorem 2 *Let Assumptions 1 and 2 hold at $(y, u) \in Y \times U$. Then the Fréchet-derivative of $\Phi(u)$ applied to $\Delta u \in U$ is given by*

$$\Phi'(u)\Delta u = \phi_y(s(u), u)\xi + \phi_u(s(u), u)\Delta u, \tag{6}$$

where $\xi = s'(u)\Delta u \in Y$ is the unique solution of the <u>*sensitivity equation*</u>

$$g_y(s(u), u)\xi = -g_u(s(u), u)\Delta u. \tag{7}$$

Proof The proof follows from the implicit function theorem, an application of the chain rule and

$$\begin{aligned} \Phi'(u)\Delta u &= \phi_y(s(u), u)s'(u)\Delta u + \phi_u(s(u), u)\Delta u \\ &= \phi_y(s(u), u)\xi + \phi_u(s(u), u)\Delta u. \end{aligned} \tag{8}$$

If U is not a Banach but a Hilbert space, one would expect for the Fréchet-derivative $\Phi'(u)\Delta u$ a gradient representation. In order to achieve this one would need an explicit representation of the derivative in a $\Phi(u)'\Delta u = \langle \nabla\Phi(u), \Delta u\rangle$ in the proper duality pairing. Such a representation clearly cannot be derived from Eq. (6) since ξ as shown in (7) depends in an implicit way on Δu. The only possibility to obtain this consists in computing the whole sensitivity operator $s'(u) \in L(U, Y)$ which in finite dimensions results in the computation of the sensitivity matrix. This requires repeated solves of the sensitivity Eq. (7), which for high dimensions is not a feasible approach. For completeness we formulate this in the following theorem.

Theorem 3 *Let Assumptions 1 and 2 hold at $(y, u) \in Y \times U$. The linear map $s'(u) : U \to Y$ is well defined by the equation*

$$g_y(s(u), u)s'(u) = -g_u(s(u), u).$$

Then we obtain

$$\Phi'(u) = s'(u)^*\phi_y(s(u), u) + \phi_u(s(u), u) \in U^*. \tag{9}$$

We can avoid this difficulty by using the adjoint operator $s'(u)^*$ of $s'(u) \in L(U, Y)$ and a so-called adjoint equation. The adjoint approach is outlined in the following theorem.

Theorem 4 *Let Assumptions 1 and 2 hold at $(y, u) \in Y \times U$. Then we obtain*

$$\Phi'(u) = g_u(s(u), u)^* p + \phi_u(s(u), u) \in U^*, \tag{10}$$

where $p \in Z^*$ is defined as the solution of the $\underline{\text{adjoint equation}}$

$$g_y(s(u), u)^* p = -\phi_y(s(u), u) \in Y^*. \tag{11}$$

Alternatively, the action of the adjoint variable $p \in Z^*$ is given by

$$p(z) = -\phi_y(s(u), u) g_y(s(u), u)^{-1} z \quad \forall z \in Z. \tag{12}$$

Proof We know from (4)

$$s'(u) = -g_y(s(u), u)^{-1} g_u(s(u), u) \in L(U, Y)$$

and hence for $s'(u)^* : Y^* \to U^*$

$$s'(u)^* = -g_u(s(u), u)^* (g_y(s(u), u)^*)^{-1}.$$

Then the first term in (9) of the derivative of Φ can be rewritten as

$$s'(u)^* \phi_y(s(u), u) = -g_u(s(u), u)^* (g_y(s(u), u)^*)^{-1} \phi_y(s(u), u) = g_u(s(u), u)^* p,$$

where $p \in Z^*$ solves the adjoint equation (11).

Furthermore, we have for an arbitrary $z \in Z$ with $\Delta v := g_y(s(u), u)^{-1} z$ using (11)

$$\begin{aligned} -\phi_y(s(u), u) g_y(s(u), u)^{-1} z &= -\phi_y(s(u), u)\Delta v = [g_y(s(u), u)^* p]\Delta v \\ &= p(g_y(s(u), u)\Delta v) = p(z), \end{aligned}$$

which shows (12). Here we used the definition of the adjoint, namely

$$< g_y(s(u), u)^* p, \Delta v >_{Y^*, Y} = < p, g_y(s(u), u)\Delta v >_{Z^*, Z} = p(g_y(s(u), u)\Delta v).$$

To synchronize the representation with the sensitivity approach we give also a version where the Fréchet-derivative is applied to a vector Δv also for the adjoint version.

Corollary 2 *Let Assumptions 1 and 2 hold at $(y, u) \in Y \times U$. Then*

$$\Phi'(u)\Delta u = p(g_u(s(u), u)\Delta u) + \phi_u(s(u), u)\Delta u$$

with the adjoint functional p from Theorem 4 defined in (12).

The main advantage of this approach is that we do not need to solve for the sensitivity operator or matrix but rather need to solve only one Eq. (11) in order to compute the adjoint variable p.

3 Representation of Second Order Fréchet-Derivative

In this section we devote our efforts to a rigorous derivation of the second derivative of the function Φ. Since in many algorithmic applications, e.g., the use of iterative solvers for the Newton step, the complete Hessian information $\Phi''(u)$ is not needed, we concentrate on the computation of the Hessian-vector product $\Phi''(u)\Delta u$. Obviously we have to strengthen Assumption 1 as follows.

Assumption 3 *Let the functional ϕ and the map g be twice continuously Fréchet-differentiable on $Y \times U$.*

As noted before, the notation using second derivatives can be somewhat complex, since the e.g., the second partial derivative $g_{yu}(y, u)$ can be interpreted as a map in $L(U, L(Y, Z))$ or $L(U \times Y, Z)$, etc.

In order to compute the second derivative we proceed in the following way: For fixed $\Delta v \in U$ consider $\Phi'(u)\Delta v$ as a function of u. Then we differentiate this new functional with respect to u which gives us the second derivative. Since $\Phi(u) = \phi(s(u), u)$ contains variables $y = s(u)$ that are implicitly defined, this will also be the case for $\Phi'(u)\Delta v$. However, there are even more implicitly defined variables like ξ in the sensitivity approach or p in the adjoint approach. Both ξ and p also depend on u in an implicit way. All this has to be kept in mind for a careful computation of the second derivatives.

In some applications, the derivative $\Phi'(u)$ is no longer differentiable in a classical sense, but only in a generalized sense where generalized derivatives come into play. Those problems can sometimes be solved efficiently using semi-smooth Newton methods. In that case at each iteration, one has to select an element from the possibly set-valued generalized second derivative of Φ. The theory of this section can be extended in such a case, if this representor exhibits a certain structure close to the problem in this paper. A similar question arises when one wants to use a Gauss-Newton method, where several terms in the second derivative operator are omitted. These issues of generalization will be addressed in a forthcoming paper.

Let us outline the approach we take here for the next sections. In the previous section we found two routes to obtain a representation of the first derivative, i.e., via the sensitivity equation or the adjoint equation. If we calculate the second derivative as outlined above, we need to decide to use either the adjoint or sensitivity approach in the computation of the derivative of $\Phi'(\cdot)\Delta v$, i.e., the second derivative of $\Phi(u)$. Therefore, we have four different routes available which we could follow, as outlined at the end of Sect. 1.

In the following, we proceed along these routes in rigorous mathematical terms. Recall that in (6) the first derivative is given by $\Phi'(u)\Delta v = \phi_y(y,u)\xi + \phi_u(y,u)\Delta v$. These terms contain two variables y and ξ which depend on u and which are defined by the original equality constraint (1) and the sensitivity Eq. 7. Hence we expand the y-variables to $\tilde{y} := (y, \xi) \in Y \times Y$ and define in analogy to $\phi(y,u)$ the function

$$\tilde{\phi}(\tilde{y}, u) = \Phi'(u)\Delta v = \phi_y(y,u)\xi + \phi_u(y,u)\Delta v. \tag{13}$$

The vector $\tilde{y} := (y, \xi) \in Y \times Y$ is the solution of the state and sensitivity equation which we combine into

$$\tilde{g}(\tilde{y}, u) := \begin{pmatrix} g(y,u) \\ g_y(y,u)\xi + g_u(y,u)\Delta v \end{pmatrix} = 0. \tag{14}$$

If we want to proceed similar to the case of the first-order derivatives, i.e., eliminate the equality constraint due to an application of the implicit function theorem, then we need to check the invertibility of $\tilde{g}_{\tilde{y}}$.

Lemma 1 *We define $\tilde{\phi} : Y \times Y \times U \to I\!R$ by (13) and $\tilde{g} : Y \times Y \times U \to Z \times Z$ by (14). Then its Fréchet-derivatives are given by*

$$\tilde{g}_{\tilde{y}}(\tilde{y}, u) = \begin{pmatrix} g_y(y,u) & 0 \\ g_{yy}(y,u)\xi + g_{uy}(y,u)\Delta v & g_y(y,u) \end{pmatrix} \in L(Y \times Y, Z \times Z), \tag{15}$$

which is invertible if Assumption 2 holds. Furthermore,

$$\tilde{g}_u(\tilde{y}, u) = \begin{pmatrix} g_u(y,u) \\ g_{yu}(y,u)\xi + g_{uu}(y,u)\Delta v \end{pmatrix} \in L(U, Z \times Z) \tag{16}$$

and for the objective function $\tilde{\phi}$ we obtain

$$\tilde{\phi}_{\tilde{y}}(\tilde{y}, u) = \begin{pmatrix} \phi_{yy}(y,u)\xi + \phi_{uy}(y,u)\Delta v \\ \phi_y(y,u) \end{pmatrix} \in Y^* \times Y^* \tag{17}$$

and

$$\tilde{\phi}_u(\tilde{y}, u) = \phi_{yu}(y,u)\xi + \phi_{uu}(y,u)\Delta v \in U^*. \tag{18}$$

The statements of the lemma can be easily obtained from the definitions (13) and (14).

Since $\tilde{g}_{\tilde{y}}(\tilde{y}, u)$ from (15) is invertible under Assumption 2 we can apply the implicit function theorem, Theorem 1, to derive in the same way as in Theorem 1 the following:

Theorem 5 *Let the Assumptions 2 and 3 hold at $(\tilde{y}_*, u_*) \in Y \times Y \times U$ with $\tilde{g}(\tilde{y}_*, u_*) = 0$. Then there exist neighborhoods $\tilde{B}_U \subset U$ of u_* and $\tilde{B}_{Y \times Y} \subset Y \times Y$ of $\tilde{y}^*$ and a map*

$$\tilde{s}(\cdot) : \tilde{B}_U \to \tilde{B}_{Y\times Y} \quad \textit{such that} \quad \tilde{g}(\tilde{s}(u), u) = 0 \quad \textit{on} \quad \tilde{B}_U$$

with its derivative defined by

$$\tilde{g}_{\tilde{y}}(\tilde{s}(u), u)\tilde{s}'(u) = -\tilde{g}_u(\tilde{s}(u), u).$$

In the case of the adjoint approach we proceed in a similar fashion by augmenting the variable y to $\hat{y} = (y, p) \in Y \times Z^*$. Hence Eq. (10) becomes

$$\hat{\phi}(\hat{y}, u) := \Phi'(u)\Delta v = [g_u(y, u)^* p + \phi_u(y, u)]\Delta v. \tag{19}$$

The equality constraints then are defined as follows using (11)

$$\hat{g}(\hat{y}, u) := \hat{g}(y, p, u) = \begin{pmatrix} g(y, u) \\ g_y(y, u)^* p + \phi_y(y, u) \end{pmatrix} = 0. \tag{20}$$

The derivatives for these mappings are computed as follows:

Lemma 2 *We define $\hat{\phi} : Y \times Z^* \times U \to I\!R$ by (19) and $\hat{g} : Y \times Z^* \times U \to Z \times Y^*$ by (20). Then the Fréchet-derivatives for $\hat{\phi}$ are given by*

$$\hat{\phi}_u(\hat{y}, u) = (g_{uu}(y, u)\Delta v)^* p + \phi_{uu}(y, u)\Delta v \in U^* \tag{21}$$

$$\hat{\phi}_{\hat{y}}(\hat{y}, u) = \begin{pmatrix} (g_{uy}(y, u)\Delta v)^* p + \phi_{uy}(y, u)\Delta v \\ g_u(y, u)\Delta v \end{pmatrix} \in Y^* \times Z \tag{22}$$

and for $\hat{g}$ we have

$$\hat{g}_u(\hat{y}, u) = \begin{pmatrix} g_u(y, u) \\ (g_{yu}(y, u)(\cdot))^* p + \phi_{yu}(y, u) \end{pmatrix} \in L(U, Z \times Y^*) \tag{23}$$

$$\hat{g}_{\hat{y}}(\hat{y}, u) = \begin{pmatrix} g_y(y, u) & 0 \\ (g_{yy}(y, u)(\cdot))^* p + \phi_{yy}(y, u) & g_y(y, u)^* \end{pmatrix} \in L(Y \times Z^*, Z \times Y^*). \tag{24}$$

Note, that also in this case Assumption 2 implies that $\hat{g}_{\hat{y}}$ is invertible. Therefore we can apply the implicit function theorem also to this setting.

Theorem 6 *For fixed Δv let the Assumptions 2 and 3 hold at $(\hat{y}_*, u_*) \in Y \times Z^* \times U$ with $\hat{g}(\hat{y}_*, u_*) = 0$. Then there exist neighborhoods $\hat{B}_U \in U$ of u_* and $\hat{B}_{Y\times Z^*} \subset Y \times Z^*$ of $\hat{y}$ and a map*

$$\hat{s} : \hat{B}_U \to \hat{B}_{Y\times Z^*} \quad \textit{with} \quad \hat{g}(\hat{s}(u), u) = 0 \textit{ on } \hat{B}_U$$

with its derivative defined by

$$\hat{g}_{\hat{y}}(\hat{s}(u), u)\hat{s}'(u) = -\hat{g}_u(\hat{s}(u), u).$$

With these technicalities resolved we turn to the computation of the second derivative using the four strategies outlined before.

4 Second Order Sensitivities and Second Order Adjoints

4.1 Sensitivity-Sensitivity Approach

If we apply the sensitivity equation approach to both the first and the second derivative, then we obtain the following theorem.

Theorem 7 *Let Assumptions 2 and 3 hold at* (y, u) *with* $g(y, u) = 0$. *Then for* $\Delta v, \Delta w \in U$

$$\begin{aligned}\Phi''(u)(\Delta v, \Delta w) = {} & \phi_{uu}(y, u)(\Delta v, \Delta w) + \phi_{uy}(y, u)(\Delta v, \eta) \\ & + \phi_{yu}(y, u)(\xi, \Delta w) + \phi_{yy}(y, u)(\xi, \eta) + \phi_y(y, u)\rho,\end{aligned}$$

where $\xi, \eta \in Y$ *solve the following first-order sensitivity equations*

$$\begin{aligned} g_y(y, u)\xi &= -g_u(y, u)\Delta v \\ g_y(y, u)\eta &= -g_u(y, u)\Delta w,\end{aligned}$$

and $\rho \in Y$ *solves the second order sensitivity equation*

$$\begin{aligned} g_y(y, u)\rho = {} & -g_{yy}(y, u)(\xi, \eta) - g_{uy}(y, u)(\Delta v, \eta) \\ & - g_{yu}(y, u)(\xi, \Delta w) - g_{uu}(y, u)(\Delta v, \Delta w).\end{aligned}$$

Proof Theorem 2 applied to the problem formulation in (13) and (14) yields

$$\tilde{\Phi}'(u)\Delta w = \tilde{\phi}_{\tilde{y}}(\tilde{s}(u), u)\tilde{\xi} + \tilde{\phi}_u(\tilde{s}(u), u)\Delta w,$$

where $\tilde{\xi} = (\eta, \rho)^T$ is the unique solution of the sensitivity equation

$$\tilde{g}_{\tilde{y}}(\tilde{s}(u), u)\tilde{\xi} = -\tilde{g}_u(\tilde{s}(u), u)\Delta w.$$

Using (17)–(18) implies

$$\begin{aligned}\tilde{\Phi}'(u)\Delta w &= \tilde{\phi}_{\tilde{y}}(\tilde{s}(u), u)\tilde{\xi} + \tilde{\phi}_u(\tilde{s}(u), u)\Delta w \\ &= \phi_{yy}(y, u)(\xi, \eta) + \phi_{uy}(y, u)(\Delta v, \eta) \\ &\quad + \phi_y(y, u)\rho + \phi_{yu}(y, u)(\xi, \Delta w) + \phi_{uu}(y, u)(\Delta v, \Delta w),\end{aligned}$$

where $\tilde{\xi} = (\eta, \rho)^T$ solve

$$\tilde{g}_{\tilde{y}}(\tilde{s}(u), u)\tilde{\xi} = -\tilde{g}_u(\tilde{s}(u), u)\Delta w.$$

With (15)–(16) this implies

$$\begin{pmatrix} g_y(y,u)\eta \\ g_{yy}(y,u)(\xi,\eta) + g_{uy}(y,u)(\Delta v,\eta) + g_y(y,u)\rho \end{pmatrix}$$
$$= -\begin{pmatrix} g_u(y,u)\Delta w \\ g_{yu}(y,u)(\xi,\Delta w) + g_{uu}(y,u)(\Delta v,\Delta w) \end{pmatrix}.$$

Rearranging the terms leads to the formulation in the theorem.

This theorem shows that in order to perform a full Hessian evaluation one needs three solves (25) of the linearized equality constraint equation, i.e. sensitivity equation. This is an interesting observation by itself, especially since only the right hand side of the sensitivity equation is changed. However, note that this is true only for an evaluation of the second order term in a Taylor expansion of the objective function, i.e. the Hessian applied to two arguments, here Δv and Δw. The computation of a Hessian vector product resulting into a vector, as it is required for a Newton step, is still problematic. Here we would have to scan the whole space U, i.e. let Δw run through all basis vectors, which is quite expensive.

In a similar way, one can derive a representation of the Hessian operator itself, without any application to arguments Δv, Δw. This can be obtained easily with mappings $s'(u) \in L(U,Y)$ and $\sigma(u) \in L(U,Y)$ and

$$\xi = s'(u)\Delta v, \quad \eta = s'(u)\Delta w, \quad \rho = \sigma(u)\Delta v.$$

Theorem 8 *Let Assumptions 2 and 3 hold at (y,u) with $g(y,u) = 0$. Then $\Phi''(u) : U \to U^*$ can be represented as*

$$\begin{aligned}\Phi''(u) = {} & \phi_{uu}(s(u),u) + \phi_{uy}(s(u),u)s'(u) \\ & + s'(u)^*\phi_{yu}(s(u),u) + s'(u)^*\phi_{yy}(s(u),u)s'(u) + \sigma(u)^*\phi_y(s(u),u),\end{aligned}$$

where $s(u) \in Y$ solves the system equation $g(s(u),u) = 0$. The operators $s'(u) \in L(U,Y)$ and $\sigma(u) \in L(U,Y)$ are the solutions to the following first and second order sensitivity equations

$$\begin{aligned} g_y(s(u),u)s'(u) &= -g_u(s(u),u) \\ g_y(s(u),u)\sigma(u) &= -s'(u)^*g_{yy}(y,u)s'(u) - g_{yu}(y,u)s'(u) \\ &\quad -s'(u)^*g_{uy}(y,u) - g_{uu}(y,u).\end{aligned}$$

Also for this theorem, one would need knowledge of the full operator $s'(u)$ which is computationally not available. We can compute an evaluation of $s'(u)\Delta v$ by the solve of one sensitivity equation, but to get information of the full operator $s'(u)$ one would need to solve it for all basis vectors Δv.

4.2 Sensitivity-Adjoint Approach

We start with the first-order derivative in sensitivity form as written up in Lemma 1 and used in the previous subsection. However, here we apply the adjoint approach for the calculation of the second derivative as outlined in Theorem 4.

In this theorem one sees that two adjoint equation equations need to be solved, both with the system matrix or operator $g_y(y,u)^*$. The first solution p comes from the adjoint equation (25) which we know from the computation of the gradient or first derivative, therefore we call it the first-order adjoint. The other solution π needs to be computed as a solution of (26) in order to obtain the information about the second derivative, therefore we call it the second order adjoint.

Theorem 9 *Let Assumptions 2 and 3 hold at (y,u) with $g(y,u)=0$. Then $\Phi''(u)\Delta v \in U^*$ can be represented as*

$$\begin{aligned}\Phi''(u)\Delta v = g_u(y,u)^*\pi + (g_{uy}(y,u)\xi)^*p + (g_{uu}(y,u)\Delta v)^*p \\ + \phi_{uy}(y,u)\xi + \phi_{uu}(y,u)\Delta v.\end{aligned}$$

Here $\xi \in Y$ solves the first-order sensitivity equation

$$g_y(y,u)\xi = -g_u(y,u)\Delta v \in Z,$$

$p \in Z^*$ is a solution of the first-order adjoint equation

$$g_y(y,u)^*p = -\phi_y(y,u) \in Y^*, \tag{25}$$

and $\pi \in Z^*$ solves the second order adjoint equation

$$g_y(y,u)^*\pi = -(g_{yy}(y,u)\xi)^*p - (g_{yu}(y,u)\Delta v)^*p - \phi_{yy}(y,u)\xi - \phi_{yu}(y,u)\Delta v \in Y^*. \tag{26}$$

Proof Due to the definition of $\tilde{g}$ in (14) the corresponding multiplier is of the form $\tilde{p} := (\pi, p) \in Z^* \times Z^*$. Then inserting (16) and (18) into (10) gives

$$\begin{aligned}\tilde{\Phi}'(u) &= \tilde{g}_u(\tilde{s}(u),u)^*\tilde{p} + \tilde{\phi}_u(\tilde{s}(u),u) \\ &= \begin{pmatrix} g_u(y,u) \\ g_{yu}(y,u)\xi + g_{uu}(y,u)\Delta v \end{pmatrix}^* \begin{pmatrix} \pi \\ p \end{pmatrix} + \phi_{yu}(y,u)\xi + \phi_{uu}(y,u)\Delta v \\ &= g_u(y,u)^*\pi + (g_{yu}(y,u)\xi)^*p + (g_{uu}(y,u)\Delta v)^*p + \phi_{yu}(y,u)\xi + \phi_{uu}(y,u)\Delta v.\end{aligned}$$

The adjoint equation for the second derivative is obtained by inserting (15) and (17) into Eq. (11) which reads as

$$\tilde{g}_{\tilde{y}}(\tilde{s}(u),u)^*\tilde{p} = -\tilde{\phi}_{\tilde{y}}(\tilde{s}(u),u)$$

or

$$\begin{pmatrix} g_y(y,u) & 0 \\ g_{yy}(y,u)\xi + g_{yu}(y,u)\Delta v & g_y(y,u) \end{pmatrix}^* \begin{pmatrix} \pi \\ p \end{pmatrix} = -\begin{pmatrix} \phi_{yy}(y,u)\xi + \phi_{yu}(y,u)\Delta v \\ \phi_y(y,u) \end{pmatrix}$$

and finally

$$g_y(y,u)^*\pi + (g_{yy}(y,u)\xi)^* p + (g_{yu}(y,u)\Delta v)^* p = -\phi_{yy}((y,u)\xi - \phi_{uy}(y,u)\Delta v$$

and

$$g_y(y,u)^* p = -\phi_y(y,u),$$

which yields the statements of the theorem.

This result shows that it is possible to compute the Hessian-vector product by simply solving the (first-order) sensitivity equation for ξ and the second order adjoint equation (26) for π. The operator or matrix for the second order adjoint solve is the same as for the first-order adjoint. In contrast to the sensitivity approach in the previous section we obtain the full information of the vector that represents the Hessian-vector product. This is an highly efficient way to compute the information needed in each step of a Krylov method to solve for the Newton step.

4.3 Adjoint-Sensitivity Approach

In this subsection we start with the first derivative represented by the adjoint equation. For the computation of the second derivative we apply the sensitivity approach. In what follows, we use the notation $\hat{y} = (p,u)$ as outlined in (19) and (20). By Theorem 2 we have from Eq. (20) that

$$\hat{g}_{\hat{y}}(\hat{y},u)\hat{\xi} = -\hat{g}_u(\hat{y},u)\Delta w$$

or with $\hat{\xi} = (\xi,\pi)$ using (23) and (24)

$$\begin{pmatrix} g_y(y,u) & 0 \\ (g_{yy}(y,u)(\cdot))^* p + \phi_{yy}(y,u) & g_y(y,u)^* \end{pmatrix} \begin{pmatrix} \xi \\ \pi \end{pmatrix}$$

$$= -\begin{pmatrix} g_u(y,u) \\ (g_{yu}(y,u)(\cdot))^* p + \phi_{yu}(y,u) \end{pmatrix} \Delta w,$$

which leads to the first-order sensitivity equation for ξ

$$g_y(y,u)\xi = -g_u(y,u)\Delta w$$

and the second order adjoint equation for π

$$(g_{yy}(y,u)\xi)^* p + \phi_{yy}(y,u)\xi + g_y(y,u)^*\pi = -(g_{yu}(y,u)\Delta w)^* p - \phi_{yu}(y,u)\Delta w.$$

Furthermore, the Hessian-vector product can be obtained from (6)

$$\begin{aligned}\hat{\phi}_{\hat{y}}(\hat{y},u)\hat{\xi} + \hat{\phi}_u(\hat{y},u)\Delta w = &\begin{pmatrix} (g_{uy}(y,u)\Delta v)^* p + \phi_{uy}(y,u)\Delta v \\ g_u(y,u)\Delta v \end{pmatrix} \begin{pmatrix} \xi \\ \pi \end{pmatrix} \\ &+[(g_{uu}(y,u)\Delta v)^* p + \phi_{uu}(y,u)\Delta v]\Delta w \\ = &[(g_{yu}(y,u)\xi)^* p + \phi_{yu}(y,u))\xi + (g_{uu}(y,u)\Delta w)^* p \\ &+\phi_{uu}(y,u))\Delta w + g_u(y,u)^*\pi]\Delta v,\end{aligned}$$

which is the same expression as in Theorem 9.

In summary, we obtain the following remark.

Remark 1 For fixed Δv let the Assumptions 2 and 3 hold at $(\hat{y}_*, u_*)$. Then

$$\begin{aligned}\Phi''(u)(\Delta v, \Delta w) = &[(g_{yu}(y,u)\xi)^* p + \phi_{yu}(y,u))\xi + (g_{uu}(y,u)\Delta w)^* p \\ &+\phi_{uu}(y,u))\Delta w + g_u(y,u)^*\pi]\Delta v,\end{aligned} \tag{27}$$

where (y,u) solve $g(y,u) = 0$, p solves the adjoint equation and ξ solves the first-order sensitivity equation

$$g_y(y,u)\xi = -g_u(y,u)\Delta w,$$

and π solves the second order adjoint equation

$$g_y(y,u)\pi = -(g_{yy}(y,u)\xi)^* p - \phi_{yy}(y,u)\xi - (g_{yu}(y,u)\Delta w)^* p - \phi_{yu}(y,u)\Delta w.$$

Since in (27) the vector Δv is separated outside of the parentheses, we have an expression for the Hessian-vector product. This result is identical with the findings in Theorem 9.

4.4 Adjoint-Adjoint Approach

Finally we apply the adjoint approach to compute the second derivative when the first derivative is also calculated by the adjoint approach.

The adjoint equation for the extended system reads as in (11)

$$\hat{g}_{\hat{y}}(y,u)^* \hat{p} = -\hat{\phi}_{\hat{y}}(\hat{y},u) \quad \hat{p} \in Z^* \times Y$$

and $\hat{g}_{\hat{y}}(y,u)^* \in L(Z^* \times Y, Y^* \times Z)$. Inserting the terms from (22) and (24) we obtain for $\hat{p} = (\pi, \xi) \in Z^* \times Y$

$$\begin{aligned}&\begin{pmatrix} g_y(y,u) & 0 \\ (g_{yy}(y,u)(\cdot))^*p + \phi_{yy}(y,u) & g_y(y,u)^* \end{pmatrix}^* \begin{pmatrix} \pi \\ \xi \end{pmatrix} \\ &= -\begin{pmatrix} (g_{uy}(y,u)\Delta v)^*p + \phi_{uy}(y,u)\Delta v \\ g_u(y,u)\Delta v \end{pmatrix} \in Y^* \times Z,\end{aligned}$$

which results in

$$g_y(y,u)^*\pi + (g_{yy}(y,u)\xi)^*p + \phi_{yy}(y,u)\xi = -(g_{uy}(y,u)\Delta v)^*p - \phi_{uy}(y,u)\Delta v$$

and

$$g_y(y,u)^*\xi = -g_u(y,u)\Delta v.$$

The Hessian-vector product can be computed from Eq. (10) as

$$\hat{g}_u(\hat{y},u)^*\hat{p} + \hat{\phi}_u(\hat{y},u)$$

and inserting (21) and (23)

$$\begin{pmatrix} g_u(y,u) \\ (g_{yu}(y,u)(\cdot))^*p + \phi_{yu}(y,u) \end{pmatrix}^* \begin{pmatrix} \pi \\ \xi \end{pmatrix} + (g_{uu}(y,u)\Delta v)^*p + \phi_{uu}(y,u)\Delta v$$

and

$$g_u(y,u)^*\pi + (g_{yu}(y,u)\xi)^*p + \phi_{yu}(y,u)\xi + (g_{uu}(y,u)\Delta v)^*p + \phi_{uu}(y,u)\Delta v.$$

Therefore, we have the following remark.

Remark 2 The second derivative reads

$$\begin{aligned}\Phi''(u)\Delta v = g_u(y,u)^*\pi + (g_{yu}(y,u)\xi)^*p + \phi_{yu}(y,u)\xi + (g_{uu}(y,u)\Delta v)^*p \\ + \phi_{uu}(y,u)\Delta v,\end{aligned}$$

where y, u satisfy $g(y,u) = 0$ and p solves the adjoint equation. Furthermore ξ solves the sensitivity equation of first-order

$$g_y(y,u)\xi = -g_u(y,u)\Delta v,$$

and π the second order adjoint equation

$$g_y(y,u)^*\pi = -(g_{yy}(y,u)\xi)^*p - \phi_{yy}(y,u)\xi - (g_{uy}(y,u)\Delta v)^*p - \phi_{uy}(y,u)\Delta v.$$

From the results of the theorems above we realize that the four different approaches outlined in the beginning of Sect. 3 lead to two different results: one where a second order adjoint equation comes into play and another one where a second sensitivity equation has to be solved. We can also see that this can be derived in a fairly general setting for infinite-dimensional spaces and hence can be applied to all kinds of optimization problems. In this paper we will concentrate on two applications in the context of Newton's method.

5 PDE-Constrained Optimization

We consider as an application an optimization problem with a partial differential equation and control in the coefficient. In particular, we consider an inverse problem where the partial differential equation is given by

$$-\nabla \cdot ((\exp(u) + \epsilon)\nabla y) = f \quad \text{in } \Omega, \qquad y = 0 \quad \text{on } \partial\Omega \tag{28}$$

on a bounded and closed domain $\Omega \subset I\!R^2$ for a given right-hand side $f \in L^2(\Omega)$ and some small $\epsilon > 0$. The inverse problem consists of finding a proper $u \in L^\infty(\Omega)$ such that the corresponding solution $y \in H_0^1(\Omega)$ is close to a given observed output $w^{obs} \in H_0^1(\Omega_1)$, with $\Omega_1 \subseteq \Omega$.

To formulate this PDE-constrained optimization problem in the form problem 1 was posed, we set

$$Y = H_0^1(\Omega),\ U = L^\infty(\Omega),\ W = L^2(\Omega_1),\ Z = L^2(\Omega).$$

Next we define the constraint as

$$g(y,u) = -\nabla \cdot ((\exp(u) + \epsilon)\nabla y) - f, \quad g : Y \times U \to Z,$$

and the objective function including a regularization term with $\alpha > 0$,

$$\phi(y,u) = \frac{1}{2}\int\limits_{\Omega_1} (\mathcal{B}y(x) - w^{obs}(x))^2 dx + \frac{\alpha}{2}\int\limits_{\Omega} u(x)^2 dx,$$

where $\mathcal{B} : L^2(\Omega) \to L^2(\Omega_1)$ is a linear observation operator that extracts measurements from y. One can show that the Fréchet-derivatives up to second order exist and have the following form for the objective function

$$\begin{aligned}
\phi_y(y,u)\bar{y} &= \int\limits_{\Omega} [\mathcal{B}^*(\mathcal{B}y(x) - w^{obs}(x))]\bar{y}(x)dx,\\
\phi_u(y,u)\bar{u} &= \alpha \int\limits_{\Omega} u(x)\bar{u}(x)dx,\\
\phi_{yy}(y,u)(\bar{y},\bar{z}) &= \int\limits_{\Omega} \mathcal{B}^*\mathcal{B}\bar{y}(x)\bar{z}(x)dx,
\end{aligned}$$

$$\phi_{yu}(y,u)(\bar{y},\bar{u}) = \phi_{uy}(y,u)(\bar{u},\bar{y}) = 0,$$
$$\phi_{uu}(y,u)(\bar{u},\bar{v}) = \alpha \int_{\Omega} \bar{u}(x)\bar{v}(x)dx.$$

Similarly we obtain for the Fréchet-derivatives of the constraint

$$\begin{aligned} g_y(y,u)\bar{y} &= -\nabla\cdot((\exp(u)+\epsilon)\nabla\bar{y}),\\ g_u(y,u)\bar{u} &= -\nabla\cdot(\exp(u)\bar{u}\nabla y),\\ g_{yy}(y,u)(\bar{y},\bar{z}) &= 0,\\ g_{yu}(y,u)(\bar{y},\bar{u}) &= -\nabla\cdot(\exp(u)\bar{u}\nabla\bar{y}),\\ g_{uu}(y,u)(\bar{u},\bar{v}) &= -\nabla\cdot(\exp(u)\bar{u}\bar{v}\nabla y). \end{aligned}$$

Since the two adjoints are computed by an application of the operator g_y^*, it is straightforward to derive a representation for the solution $p \in Z^*$ of the equation $g_y^* p = r$ with given right hand side $r \in Z$ along the following lines.

Lemma 3 *The solution $p \in Z^*$ of $g_y^* p = r$ for given $r \in Y^* = H^{-1}(\Omega)$ is represented by $p(\zeta) = \langle \bar{p}, \zeta\rangle_Z$, $\zeta \in Z = L^2(\Omega)$, where $\bar{p} \in Y = H_0^1(\Omega)$ is a weak solution of*

$$-\nabla\cdot(\exp(u)+\epsilon)\nabla\bar{p} = r.$$

Proof Equivalently, $g_y^* p = r$ can be written as

$$(g_y^* p)(\eta) = p(g_y\eta) = \langle r,\eta\rangle_Z \quad \forall\, \eta \in Y.$$

Let us make an ansatz for the solution $p \in Z^* = L^2(\Omega)^*$, i.e., assume the linear functional p is represented by a function $\bar{p} \in H_0^1(\Omega)$ such that $p(\zeta) = \langle \bar{p},\zeta\rangle_Z$ for all $\zeta \in Z$. Then

$$\begin{aligned}(g_y^* p)(\eta) = p(g_y\eta) &= -\langle \bar{p}, \nabla\cdot(\exp(u)+\epsilon)\nabla\eta)\rangle_Z = \langle \nabla\bar{p}, (\exp(u)+\epsilon)\nabla\eta)\rangle_Z\\ &= \langle(\exp(u)+\epsilon)\nabla\bar{p}, \nabla\eta)\rangle_Z \quad \forall\eta \in Y,\end{aligned}$$

and

$$(g_y^* p)(\eta) = \langle r,\eta\rangle_Z \iff \langle(\exp(u)+\epsilon)\nabla\bar{p},\nabla\eta)\rangle_Z = \langle r,\eta\rangle_Z \quad \forall\eta\in Y.$$

This is the definition in weak form of a solution of the PDE

$$-\nabla\cdot(\exp(u)+\epsilon)\nabla\bar{p} = r.$$

Therefore, the first-order adjoint $p \in H_0^1(\Omega)$ according to (11) is given as the solution of

$$-\nabla\cdot(\exp(u)+\epsilon)\nabla p) = -\mathcal{B}^*(\mathcal{B}y - w^{obs}). \quad (29)$$

The second order adjoint $\pi \in H_0^1(\Omega)$ according to (26) is the solution of

$$-\nabla \cdot ((\exp(u) + \epsilon)\nabla \pi) = \nabla \cdot (\exp(u)\bar{u}\nabla p) - \mathcal{B}^*\mathcal{B}\xi. \tag{30}$$

The solution $\xi \in H_0^1(\Omega)$ of the first-order sensitivity equation can be obtained following (7) by solving

$$-\nabla \cdot ((\exp(u) + \epsilon)\nabla \xi) = \nabla \cdot (\exp(u)\bar{u}\nabla y). \tag{31}$$

Given all the derivatives above, we can apply Theorem 9 to see which partial differential equations need to be solved for an evaluation of a Hessian of Φ applied to a vector.

Theorem 10 *For $\Phi(u) = \phi(s(u), u)$ the application of the Hessian of ϕ to a vector $\bar{u}$, $\Phi''(u)\bar{u}$, can be obtained with $y = s(u)$ from*

$$\Phi''(u)\bar{u} = -\exp(u)[(\nabla\pi)^T\nabla y + (\nabla p)^T\nabla\xi] + [-\exp(u)(\nabla p)^T\nabla y + \alpha]\bar{u},$$

where $y \in H_0^1(\Omega)$ is the solution of the state Eq. 28, $p \in H_0^1(\Omega)$ is the solution of the first-order adjoint Eq. 29, $\xi \in H_0^1(\Omega)$ the solution of the first-order sensitivity Eq. 31, and $\pi \in H_0^1(\Omega)$ is the solution of the second order adjoint Eq. 30.

6 Summary and Conclusions

In this paper we derived rigorously second order adjoints for equality constrained optimization problems in a general, infinite-dimensional setting which are used for the Hessian-vector products in Newton's method. We showed that while there are four routes to arrive to the second order adjoints (e.g., via combinations of sensitivity or adjoint equations), except the sensitivity-sensitivity approach, all the other three of these coincide, i.e., they give the same Hessian-apply expression. This finding suggests that one can choose whichever route is more convenient without compromising the underlying computational effort. However, as discussed the sensitivity-sensitivity approach is feasible only in the case of small number parameters.

We have applied this general framework to a PDE-constrained optimization problem formulated as a nonlinear least squares problem governed by an elliptic PDE. This application revealed the ability to derive the second order adjoints (and Hessian-vector apply) in a straightforward manner when following the general framework established in this paper.

In this paper we chose to derive the expressions for the second order adjoints at the infinite-dimensional level for a number of reasons. First, the derivation and final results do not depend on any particular discretization of the underlying PDEs. Second, the derivation is clean and reveal similar structure. Third, the form of the boundary conditions for the adjoints falls out cleanly from the infinite-dimensional expressions. However, in certain cases working in finite-dimensions is beneficial, for instance in the case when the optimize-then-discretize (OTD) and discretize-then-optimize (DTO) approaches do not commute [18]. In the case of Hilbert spaces, the role of the adjoint solves could be simplified when the domain and range space are identical and the operator g_y turns out to be self-adjoint. Furthermore, the standard formulation of a Gauss-Newton method for a nonlinear least squares problem can be used in a Hilbert space setting and it can be shown that the adjoint solve is in fact a second order adjoint as defined in this context. The derivation of these equations in finite-dimension and the framework for Gauss-Newton methods is the subject of future work.

Acknowledgements This work was partially supported by the US National Science Foundation grant CAREER-1654311. E. S. acknowledges partial support from the University of California, Merced, and Lawrence Livermore National Laboratory.

References

1. Akçelik, V., Biros, G., Drăgănescu, A., Ghattas, O., Hill, J., van Bloeman Waanders, B.: Dynamic data-driven inversion for terascale simulations: Real-time identification of airborne contaminants. In: Proceedings of SC2005. Seattle (2005)
2. Alekseev, A.K., Navon, I.M.: The analysis of an ill-posed problem using multi-scale resolution and second-order adjoint techniques. Computer Methods in Applied Mechanics and Engineering **190**, 1937–1953 (2001)
3. Alekseev, A.K., Navon, I.M., Steward, J.: Comparison of advanced large-scale minimization algorithms for the solution of inverse ill-posed problems. Optimization Methods & Software **24**(1), 63–87 (2009)
4. Alexanderian, A., Petra, N., Stadler, G., Ghattas, O.: A fast and scalable method for A-optimal design of experiments for infinite-dimensional Bayesian nonlinear inverse problems. SIAM Journal on Scientific Computing **38**(1), A243–A272 (2016)
5. Alexanderian, A., Petra, N., Stadler, G., Ghattas, O.: Mean-variance risk-averse optimal control of systems governed by PDEs with random parameter fields using quadratic approximations. SIAM/ASA Journal on Uncertainty Quantification **5**(1), 1166–1192 (2017)
6. Becker, R., Meidner, D., Vexler, B.: Efficient numerical solution of parabolic optimization problems by finite element methods. Optimization Methods Software **22**, 813–833 (2007)
7. Bui-Thanh, T., Burstedde, C., Ghattas, O., Martin, J., Stadler, G., Wilcox, L.C.: Extreme-scale UQ for Bayesian inverse problems governed by PDEs. In: SC12: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (2012). Gordon Bell Prize finalist
8. Bui-Thanh, T., Ghattas, O., Martin, J., Stadler, G.: A computational framework for infinite-dimensional Bayesian inverse problems: Part I. The linearized case, with application to global seismic inversion. SIAM Journal on Scientific Computing **35**(6), A2494–A2523 (2013)
9. Cao, Y., Li, S., Petzold, L., Serban, R.: Adjoint sensitivity analysis for differential-algebraic equations: the adjoint DAE system and its numerical solution. SIAM Journal on Scientific Computing **24**(3), 1076–1089 (electronic) (2002)
10. Chen, P., Villa, U., Ghattas, O.: Taylor approximation and variance reduction for PDE-constrained optimal control under uncertainty. Journal of Computational Physics **385**, 163–186 (2019)
11. Cioaca, A., Alexe, M., Sandu, A.: Second-order adjoints for solving PDE-constrained optimization problems. Optimization Methods and Software **27**(4-5), 625–653 (2012)

12. Daescu, D.N., Navon, I.M.: An analysis of a hybrid optimization method for variational data assimilation. International Journal of Computational Fluid Dynamics **17**(4), 299–306 (2003).
13. Dunn, J.C., Bertsekas, D.P.: Efficient dynamic programming implementations of Newton's method for unconstrained optimal control problems. Journal of Optimization Theory and Applications **63**(1), 23–38 (1989)
14. Epanomeritakis, I., Akçelik, V., Ghattas, O., Bielak, J.: A Newton-CG method for large-scale three-dimensional elastic full-waveform seismic inversion. Inverse Problems **24**(3), 034015 (26pp) (2008)
15. Fichtner, A., Trampert, J.: Hessian kernels of seismic data functionals based upon adjoint techniques. Geophysical Journal International **185**(2), 775–798 (2011)
16. Griesse, R.: 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)
17. Griesse, R., Vexler, B.: Numerical sensitivity analysis for the quantity of interest in PDE-constrained optimization. SIAM Journal on Scientific Computing **29**(1), 22–48 (2007)
18. Gunzburger, M.D.: Perspectives in Flow Control and Optimization. SIAM, Philadelphia (2003)
19. Haber, E., Hanson, L.: Model problems in PDE-constrained optimization. Tech. Rep. TR-2007-009, Emory University (2007)
20. Haftka, R.T., Mróz, Z.: First- and second-order sensitivity analysis of linear and nonlinear structures. AIAA journal **24**(7), 1187–1192 (1986)
21. Haug, E.J.: Second-order design sensitivity analysis of structural systems. AIAA Journal **19**(8), 1087–1088 (1981)
22. Heinkenschloss, M.: Numerical solution of implicitly constrained optimization problems. Tech. Rep. TR08-05, Department of Computational and Applied Mathematics, Rice University (2008)
23. Herzog, R., Sachs, E.: Preconditioned conjugate gradient method for optimal control problems with control and state constraints. SIAM Journal on Matrix Analysis and Applications **31**(5), 2291–2317 (2010)
24. Hicken, J.E.: Inexact Hessian-vector products in reduced-space differential-equation constrained optimization. Optimization and Engineering **15**(3), 575–608 (2014)
25. Hinze, M., Kunisch, K.: Second order methods for optimal control of time–dependent fluid flow. SIAM Journal on Control and Optimization **40**, 925–946 (2001)
26. Hinze, M., Pinnau, R.: Second-order approach to optimal semiconductor design. Journal of Optimization Theory and Applications **133**(2), 179–199 (2007)
27. Hou, G.J.W., Sheen, J.: Numerical methods for second-order shape sensitivity analysis with applications to heat conduction problems. International Journal for Numerical Methods in Engineering **36**(3), 417–435 (1993)
28. Isaac, T., Petra, N., Stadler, G., Ghattas, O.: Scalable and efficient algorithms for the propagation of uncertainty from data through inference to prediction for large-scale problems, with application to flow of the Antarctic ice sheet. Journal of Computational Physics **296**, 348–368 (2015)
29. Jacobson, D.H.: Second-order and second-variation methods for determining optimal control: A comparative study using differential dynamic programming. International Journal of Control **7**(2), 175–196 (1968)
30. Kelley, C.T.: Iterative Methods for Optimization. SIAM, Philadelphia (1999)
31. Le Dimet, F.X., Navon, I.M., Daescu, D.N.: Second-order information in data assimilation. Monthly Weather Review **130**(3), 629–648 (2002)
32. Mayne, D.: A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems. International Journal of Control **3**(1), 85–95 (1966)
33. Métivier, L., Brossier, R., Operto, S., Virieux, J.: Second-order adjoint state methods for full waveform inversion. In: EAGE 2012-74th European Association of Geoscientists and Engineers Conference and Exhibition (2012)
34. Nicholson, R., Petra, N., Kaipio, J.P.: Estimation of the Robin coefficient field in a Poisson problem with uncertain conductivity field. Inverse Problems **34**(11), 115005 (2018)

35. Özyurt, D.B., Barton, P.I.: Cheap second order directional derivatives of stiff ODE embedded functionals. SIAM Journal on Scientific Computing **26**(5), 1725–1743 (2005)
36. Petra, N., Stadler, G.: Model variational inverse problems governed by partial differential equations. Tech. Rep. 11-05, The Institute for Computational Engineering and Sciences, The University of Texas at Austin (2011)
37. Petra, N., Zhu, H., Stadler, G., Hughes, T.J.R., Ghattas, O.: An inexact Gauss-Newton method for inversion of basal sliding and rheology parameters in a nonlinear Stokes ice sheet model. Journal of Glaciology **58**(211), 889–903 (2012)
38. Raffard, R.L., Tomlin, C.J.: Second order adjoint-based optimization of ordinary and partial differential equations with application to air traffic flow. In: American Control Conference, pp. 798–803. IEEE (2005)
39. Rudin, W.: Principles of mathematical analysis, third edn. McGraw-Hill , Inc., New York (1976)
40. Sandu, A., Zhang, L.: Discrete second order adjoints in atmospheric chemical transport modeling. Journal of Computational Physics **227**(12), 5949–5983 (2008)
41. Santosa, F., Symes, W.W.: An analysis of least squares velocity inversion. Society of Exploration Geophysicists (1989)
42. Wang, Z., Navon, I.M., Le Dimet, F.X., Zou, X.: The second order adjoint analysis: theory and applications. Meteorology and Atmospheric Physics **50**(1-3), 3–20 (1992)
43. Zhu, H., Petra, N., Stadler, G., Isaac, T., Hughes, T.J.R., Ghattas, O.: Inversion of geothermal heat flux in a thermomechanically coupled nonlinear Stokes ice sheet model. The Cryosphere **10**, 1477–1494 (2016)

Largest Small n-polygons: Numerical Optimum Estimates for $n \geq 6$

János D. Pintér

Abstract The diameter of a convex planar polygon is defined as the maximum of the distances measured between all of its vertex pairs. $LSP(n)$, the *largest small polygon* with n vertices, is the polygon of unit diameter that has maximal area $A(n)$. It has been known for almost a century that for all odd values $n \geq 3$, $LSP(n)$ is the regular n-polygon. Perhaps surprisingly, this statement is not valid for even values of n. Finding the polygon $LSP(n)$ and $A(n)$ for even $n \geq 6$ has been a long-standing "puzzle" that can be considered as a class of global optimization problems. We present numerical solution estimates for all even values $6 \leq n \leq 80$, using the AMPL model development environment with the LGO global–local solver engine option. Based on these results, we also present a regression model-based estimate of the optimal area sequence $\{A(n)\}$.

Keywords Largest small polygons · Global optimization model · Numerical optimization by AMPL-LGO and other solvers · Illustrative results and comparisons · Regression model

1 Introduction

The diameter of a convex planar polygon is defined as the maximum of the distances measured between all of its vertex pairs. In other words, the diameter of the polygon is the length of its longest diagonal. The *largest small polygon* with n vertices is the polygon of unit diameter that has maximal area. For a given integer $n \geq 3$, we will refer to this polygon as $LSP(n)$ with corresponding area $A(n)$. To illustrate, see Fig. 1 that depicts the largest small hexagon $LSP(6)$; in this case, all polygon diagonals are of unit length.

For unambiguity, we will consider all $LSP(n)$ instances with a fixed position corresponding to appropriate modifications of Fig. 1 for even values $n \geq 6$. Specifically, we

J. D. Pintér (✉)
Department of Management Science and Information Systems, Rutgers University,
100 Rockafeller Rd, Piscataway, NJ 08854, USA
e-mail: jpinter@business.rutgers.edu
URL: https://www.business.rutgers.edu/faculty/janos-pinter

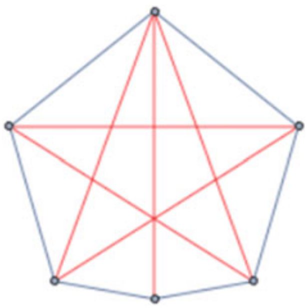


Fig. 1 Largest small hexagon *LSP*(6)

define a planar Cartesian coordinate system in which the *LSP*(n) polygons are positioned and express the "height" of each vertex by its coordinate on the vertical axis. Following a standard assumption, each even n-polygon considered here is symmetrical with respect to its diagonal that connects its "lowest" vertex (which is placed at the origin) with its "highest" vertex.

Reinhardt [27] proved that for all *odd* values $n \geq 3$, *LSP*(n) is the regular n-polygon. Perhaps surprisingly, this statement is not valid for *even* values of n. For $n = 4$ (tetragon), the square with diameter 1 has maximum area $A(4) = 0.5$, but infinitely many other tetragons with diameter 1 have the same area. The case $n = 6$ (hexagon) was analyzed and solved by Graham [11]; the case $n = 8$ (octagon) was solved by Audet et al. [2]. More recently, Henrion and Messine [14] found the largest small polygons for $n = 10$ (decagon) and $n = 12$ (dodecagon) and for $n \leq 16$ presented rigorous bounds for the optimum value. We refer to these studies and cited works therein for theoretical background and for further details regarding the analysis of the problem class $\{LSP(n)\}$. We will also review the results obtained by using general-purpose nonlinear optimization software, as reported in [4] and [5]. In addition to the publications cited in our study, we refer to the topical webpages [29–31] for concise discussions with further references.

In this work, we follow a numerical global optimization approach, in order to find *LSP*(n) configurations and corresponding estimated values $A(n)$. Following this introduction, one of the standard optimization model forms is reviewed in Sect. 2. Earlier alternative solution approaches and best-known results are reviewed in Sect. 3. The AMPL model development environment [1] and the AMPL-LGO solver option [22] are briefly discussed in Sect. 4, followed by AMPL-LGO results compared to results by other researchers and using also other solvers (Sect. 5). A regression model based on our numerical results is presented in Sect. 6, together with corresponding optimum estimates $\{A(n)\}$ for $n \geq 6$. Conclusions are presented in Sect. 7, followed by references.

2 A Standard Optimization Model for Finding *LSP*(*n*)

Our objective is to find numerically optimized $LSP(n)$ configurations with $n \geq 6$ vertices, n being an input parameter of the general model. The model formulation presented here is cited from Bondarenko et al. [4] who refer to Gay's model [8], discussed also in [9]. The AMPL model pgon.mod [8] refers to a GAMS model developed by Francisco J. Prieto (noting that a more accurate reference to Prieto's original work is unknown to this author). The corresponding GAMS model library item polygon.gms [7] refers to [8, 11] and the benchmarking study [5].

Following the model formulations referred to above, we consider polar coordinates to describe $LSP(n)$, assuming that vertex i is positioned at polar radius r_i and at angle θ_i. For unambiguity, we assume that the polygon vertices i = 1,…, n–1 are arranged (indexed) according to increasing angles θ_i. Placing the last vertex position at the origin, we have $r_n = 0, \theta_n = \pi$. Please refer to Fig. 1 for the hexagon instance $LSP(6)$ that corresponds to this standardized position.

2.1 Model Formulation

Maximize total area of the n-polygon:

$$\max\ A(n) = 1/2 \sum_{i=1,\ldots,n-1} r_i r_{i+1} \sin(\theta_{i+1} - \theta_i). \tag{1}$$

Bounds for pairwise distance between vertices i and j:

$$\begin{aligned} & r_i^2 + r_j^2 - 2 r_i r_j \cos(\theta_i - \theta_j) \leq 1, \\ & \text{for } 1 \leq i \leq n-2, i+1 \leq j \leq n-1. \end{aligned} \tag{2}$$

Vertex angle ordering relations:

$$\theta_{i+1} - \theta_i \geq 0,\ \text{for } 1 \leq i \leq n-2. \tag{3}$$

Decision variable bounds, including the two fixed settings:

$$0 \leq \theta_i \leq \pi \text{ and } 0 \leq r_i \leq 1,\ \text{for } 1 \leq i \leq n{-}1;\ r_n = 0, \theta_n = \pi. \tag{4}$$

2.2 Numerical Challenges

Difficulties can be expected to arise, due to the nonconvex objective function (1) and the nonconvex constraints (2). The number of these nonlinear constraints increases *quadratically* as a function of n. For example, the $LSP(80)$ model instance has 158 decision variables (since r_n and θ_n are fixed) with corresponding bound constraints; and it has 3241 constraints of which 3161 are nonconvex (adding to the 78 linear constraints (3) the two fixed value constraints from (4)). As conjectured by other researchers and numerically supported also by the present study, while the standardized $LSP(n)$ model instances have a unique globally optimal solution, the number of local optima increases with n. Many of the local optima are close in quality to the (unknown or only approximately known) global optimum. These features make the $\{LSP(n)\}$ problem-class numerically challenging, similarly to many other object configuration design problems arising e.g. in computational physics, chemistry and biology.

3 Related Earlier Studies

3.1 Analytical Approaches

Following Graham [11]—who combines geometric insight with results based on [33]—finding $LSP(6)$ requires the *exact* solution of a 10th order irreducible polynomial equation. More specifically, the area $A(6)$ of $LSP(6)$ can be found as the second-largest real root r of the equation

$$\begin{aligned}11993 - 78488r + 144464r^2 + 1232r^3 - 221360r^4 + 146496r^5 \\ + 21056r^6 - 30848r^7 - 3008r^8 + 8192r^9 + 4096r^{10} = 0.\end{aligned}$$

Audet et al. [2], Henrion and Messine [14] follow a different approach: in their studies finding $LSP(n)$ requires the *exact* solution of a corresponding nonconvex quadratic programming problem with quadratic constraints, combined with geometric analysis. These solution strategies, based on a different model from the one cited in Sect. 2, also brings the $LSP(n)$ problem-class into the realm of global optimization.

In [14] it is conjectured that $LSP(n)$ for all even values $n \geq 4$ has a symmetry axis, as indicated by Fig. 1 for $LSP(6)$. This conjecture was proven by Reinhardt [27] for $n = 4$, and by Yuan [34] for $n = 6$. As noted by Henrion and Messine, Graham used this conjecture to find $LSP(6)$; the $LSP(8)$ configuration found in [2] also supports the conjecture. The software packages SeDuMi [28], VSDP [16], and GloptiPoly [12, 13] are used in [14] to solve $LSP(n)$ instances for $6 \leq n \leq 16$. Henrion

Table 1 Numerical results based on analytical approaches

n	$LSP(n)$ area $A(n)$	References
4	0.5	Reinhardt [27]
6	0.674981	Graham [11][a]
8	0.726867	Audet et al. [2][a]
8	$0.72686845 \leq A(8) \leq 0.72686849$	Henrion and Messine [14][b]
10	$0.74913721 \leq A(10) \leq 0.74913736$	Henrion and Messine [14]
12	$0.76072986 \leq A(12) \leq 0.76072997$	Henrion and Messine [14]
14	$0.76753100 \leq A(14) \leq 0.76893595$	Henrion and Messine [14]
16	$0.77185969 \leq A(16) \leq 0.77279135$	Henrion and Messine [14]

[a]The results given with lower (6-decimal digit) precision are cited from [2, 11, 29]
[b]Notice the slight numerical discrepancy between the results of [2, 14] for the case $n = 8$

and Messine also discuss the current computational limitations of this approach, as runtimes increase rapidly from seconds to tens of minutes in their numerical tests.

Table 1 summarizes *all* currently known validated numerical results, including also the bounds reported in [14].

3.2 Numerical Solution Approaches

The COPS technical report [4] by Bondarenko et al. presents comparative numerical results for several $LSP(n)$ instances as shown in Table 2. These results were obtained by using the *local* nonlinear optimization software packages DONLP2, LANCELOT, LOQO, MINOS, and SNOPT (as of September 1998) linked to the AMPL modeling environment.

Table 2 summarizes the *best* numerical solution—obtained by at least one of the above listed solvers—cited from the COPS report. The term *best* refers to the solution

Table 2 Numerical results obtained by using local nonlinear optimization software [4, 5]. Best results attained by one of DONLP2, LANCELOT, LOQO, MINOS, SNOPT

n	$LSP(n)$ area $A(n)$
6	0.6749814429
10	0.7491373458
20	0.7768587560
25	0.779740[a]
50	0.7840161480
75	0.784769[a]
100	0.7850565708

[a]The numerical results given with 6 decimal digit precision are cited from [5]

Table 3 Numerical results reported by Mossinghoff [17]

n	$LSP(n)$ area $A(n)$
6	0.6749814429
8	0.7268684828
10	0.7491373459
12	0.7607298734
14	0.7675310111
16	0.7718613220
18	0.7747881651
20	0.7768587560

which has the highest objective function value, while meeting all model constraints with at least 10^{-8} precision. For completeness, we also added results for $n = 25, 50, 75, 100$ from the subsequent benchmarking study [5] in which LANCELOT, LOQO, MINOS, and SNOPT were tested: again, we cite only the *best* results.

Mossinghoff [17] also studied the $\{LSP(n)\}$ problem-class. He describes an approach to search for polygons with an even number of sides n and fixed diameter d (here $d = 1$), aiming at the largest possible area. The construction is based on optimizing a parameterized polygon model, leading to an apparently difficult numerical problem to handle. For arbitrary even $n \geq 6$, Mossinghoff's construction leads to a polygon, denoted by Q_n which provably has a larger area than that of the regular polygon P_n. *Mathematica*, by Wolfram Research [32], has been used by Mossinghoff to find Q_n for $6 \leq n \leq 20$: see Table 3. The required calculations are far from trivial: consult [17] for further details and alternatives. To the author's knowledge, this approach has not been applied to find Q_n for $n > 20$. Arguably—and similarly to work performed by other researchers—this is due to the rapidly increasing difficulties to carry out the calculations required.

3.3 The Asymptotic Behaviour of A(n)

According to the optimization model presented in Sect. 2, the numerical solution of $LSP(n)$ instances requires the handling of a nonlinear programming problem with $O(n^2)$ nonconvex constraints, and a nonconvex objective function. In spite of the implied difficulty, $LSP(n)$ problems are thought not to become dramatically more difficult to handle as n increases—at least in terms of finding reasonable numerical optimum estimates. This opinion is based on the generally postulated structural similarity and symmetry of the sequence of $\{LSP(n)\}$ configurations. As noted e.g., in [4], the optimal $LSP(n)$ configurations approach the circle of unit diameter as $n \to \infty$ consult [17] for related asymptotic results. Consequently,

$$A(\infty) = \lim_{n\to\infty} A(n) = \pi/4 \sim 0.7853981634.$$

Based on this conjectured structure, in [4, 5] "a polygon with almost equal sides" is used as the initial solution guess in all numerical tests. This approach is implemented in both the AMPL and GAMS model codes referred to earlier. Such solution strategy illustrates the point that, in optimization models good insight and resulting initial solution guess can become a valuable step towards finding credible numerical solutions efficiently. This heuristic approach, however, does not guarantee provable global optimality in many nonconvex models, including the $\{LSP(n)\}$ model-class discussed here. In spite of such "hand-crafted" initial solutions, the high-quality local nonlinear solvers mentioned above often failed to find solutions, the solutions returned were typically somewhat different, and in a number of cases evidently suboptimal. For further details regarding this point, consult [4, 5], and the results presented later on in Tables 4 and 5.

4 Solving LSP Problems Numerically by AMPL-LGO

4.1 Solution Approach

In this study, we follow a numerical optimization approach, keeping in mind also the cautionary notes presented above. Specifically, using the LGO global–local optimization solver engine linked to the AMPL modeling environment, we present *numerical optimum estimates* for all even values $6 \leq n \leq 80$. Our results, presented in Sect. 5, are in close agreement with the best results reported in Tables 1, 2 and 3—when comparable results are available. For comparison, we also report results obtained by using the currently available alternative solver options MINOS [18], SNOPT [10], and IPOPT [15] linked to AMPL.

4.2 The AMPL Model Development Environment

AMPL is a powerful modeling language that facilitates the formulation of optimization models. AMPL enables model development in a natural, concise, and scalable manner. AMPL also supports model analysis, the usage of different data sets for the same model, the seamless invocation of various solver engine options to handle optimization models, together with report generation and many other useful features not discussed here. AMPL has been extensively documented elsewhere: we refer to the AMPL book [6], and to the resources available at the AMPL website [1].

4.3 LGO Solver Suite for Nonlinear Optimization

Nonlinear optimization models frequently have multiple—local and global—optima: the objective of global optimization is to find the best possible solution under such circumstances. LGO is an integrated global–local solver suite for constrained nonlinear optimization. The model-class addressed by LGO is concisely defined by the vector of decision variables $x \in R^n$; the explicit, finite n-vector variable bounds l and u; the continuous objective function $f(x)$; and the (possibly absent) m-vector of continuous constraint functions $g(x)$. Applying these notations, LGO is aimed at numerically solving models of the general form.

$$\min f(x) \text{ subject to } x \in D := \{l \leq x \leq u,\ g(x) \leq 0\}. \tag{5}$$

In (5) all vector inequalities are interpreted component-wise: l, x, u, are n-component vectors and 0 denotes the m-component zero vector. Formally more general optimization models that include also = and constraint relations and/or explicit lower and upper bounds on the constraint function values can be directly deduced to the model form (5). If D is non-empty, then the stated key analytical assumptions guarantee that the optimal solution set X^* of the model is non-empty; however, finding X^* could still remain a formidable analytical and/or numerical challenge. Clearly, the $\{LSP(n)\}$ problem-class is encompassed by the optimization model form (5).

Without going into further details regarding LGO, we mention that the foundations of the LGO software development project are discussed by Pintér [19]; further implementation aspects are discussed e.g., in [20, 21]. Here we utilize the LGO solver option available for use with AMPL [22]; the current stand-alone LGO implementation is documented in [23]. In addition to these references, the studies [24, 25] present numerical results using LGO to solve a range of nonlinear optimization problems, from relatively simple standard test problems to well-known challenges. LGO also has been applied to handle a broad range of business, engineering and scientific optimization problems.

5 Numerical Results and Comparisons

5.1 AMPL-LGO Results

In our numerical tests reported here, the AMPL code implementation pgon.mod was used. All test runs were conducted on a several years old laptop PC with Intel Core i5-3337-U CPU @ 1.80 GHz (x-64 processor), 16 Gb RAM, running under the Windows 10 (64-bit) operating system.

The results of a *single, completely reproducible* run-sequence are summarized in Table 4 for all even values $6 \leq n \leq 80$, with a single default setting of all solver options, using LGO in global search mode. In several (seemingly more difficult) cases, we received somewhat better numerical results in additional tests, at the expense of longer runtimes: however, for consistency, we did not include those results here. Let us also mention that LGO in its local search mode often found optimum estimates that are numerically identical or close to the global search-based solution, at a fraction of the runtimes reported below. (Recall the related comment from Sect. 3.3.)

The results summarized in Table 4 are directly cited from the AMPL-LGO solver output; the corresponding $LSP(n)$ configurations can be optionally reported in the AMPL command window, and/or written to a result text file. To avoid reporting such excessive details, the optimized configurations found are not presented here. An illustrative collection of detailed results has been kept for documentation and archival purposes, and all results can be reproduced in a few minutes. The reported precision of our numerical results is set to 10 digits after the decimal point. Arguably, this is a bit of "overkill", but it is in line with the required constraint satisfaction precision as shown below. The results reported also support an in-depth comparison with the results cited earlier, as well as with the results obtained using alternative AMPL solvers (noting that in some cases the differences between the optimum estimates found are rather small).

Our numerical results for $n \leq 20$ are in fairly close agreement with the *best* results displayed in Tables 1, 2 and 3. In several cases, we found somewhat better *conjectured* optimum estimates compared to the earlier results in Tables 1 and 2; and our results up to $n = 20$ are in close agreement (up to 8 decimal digits) with those reported by Mossinghoff, see Table 3. The runtimes appear to scale rather well for $6 \leq n \leq 80$, mostly (but not always) increasing with n. The entire sequence of the 38 optimization runs reported here took a little over 10 min.

Although AMPL-LGO seems to perform reasonably well in comparison to the other solvers tested by us or by others (as reported above), its numerical limitations start to show around $n = 64$ when used in a pre-set default mode. The results presented in Table 4 for $n = 64$, 74, and 78 are clearly suboptimal, while all other $A(n)$ values are monotonically increasing with n, as expected. Instead of "tweaking" the LGO option parameters—e.g., by increasing the pre-set global search effort limit (which was actually reached in several cases reported above, for some of the larger n values), or increasing the runtime limit (set to 5 min for each run, but never reached)—here we very simply use linear interpolation to "adjust" the clearly suboptimal results based on the bracketing values in Table 4. For example, $A(64)$ is estimated on the basis of the results obtained for $A(62)$ and $A(66)$. Applying such simple interpolation leads to the following estimated values: $A(64) \sim 0.7845510976$, $A(74) \sim 0.7847519869$, $A(78) \sim 0.7847919330$.

The reason to produce these simple estimates is to use them in Sect. 6, to develop a regression model for the entire sequence $A(n)$.

Table 4 AMPL-LGO numerical results

n	$LSP(n)$ area $A(n)$	Runtime (seconds)	Maximum constraint violation
6	0.6749814433	0.55	2.21e-09 (i.e., $2.21 \cdot 10^{-9}$, etc.)
8	0.7268684830	0.70	6.47e-09
10	0.7491373457	0.95	2.96e-10
12	0.7607298709	1.30	8.9e-10
14	0.7675310106	1.69	3.91e-09
16	0.7718613224	2.55	4.09e-09
18	0.7747881650	2.63	9.78e-09
20	0.7768587506	3.02	2.23e-09
22	0.7783773308	3.95	9.08e-09
24	0.7795240461	5.22	7.73e-09
26	0.7804111201	5.34	6.34e-09
28	0.7811114192	6.05	9.83e-09
30	0.7816739255	6.98	3.67e-09
32	0.7818946320	5.72	6.29e-10
34	0.7823103007	7.61	9.03e-09
36	0.7826513767	9.50	9.75e-09
38	0.7829526627	9.34	5.08e-09
40	0.7832011589	9.55	8.47e-11
42	0.7834135187	12.06	4.62e-09
44	0.7835966860	13.22	1.42e-09
46	0.7837554636	16.88	3.43e-09
48	0.7838942710	17.95	8.31e-09
50	0.7840161496	16.53	9.99e-09
52	0.7841233641	20.61	8.78e-09
54	0.7842192995	21.38	9.18e-09
56	0.7843044654	23.91	3.87e-09
58	0.7843807534	22.95	8.43e-09
60	0.7844492943	27.97	9.79e-09
62	0.7845111362	21.22	8.93e-09
64	0.7834620877	30.48	9.82e-09
66	0.7845910589	34.17	1.19e-09
68	0.7846139029	35.84	9.00e-09
70	0.7846403575	22.33	6.45e-09
72	0.7847454020	42.75	7.34e-09

(continued)

Table 4 (continued)

n	$LSP(n)$ area $A(n)$	Runtime (seconds)	Maximum constraint violation
74	0.7845564840	26.25	3.54e-09
76	0.7847585719	49.19	8.95e-09
78	0.7845160579	49.47	9.64e-09
80	0.7848252941	51.45	7.25e-09

5.2 An Illustrative Comparison with Results Obtained by Several AMPL Solvers

For a somewhat more comprehensive picture, we also generated a set of comparative results using several currently available AMPL solvers, namely: MINOS, SNOPT, IPOPT, and LGO. All solvers are used with their default settings. We did not include all even values $6 \leq n \leq 80$, only a representative subset (starting from $n = 30$, we increased n by 10), since—based on the results obtained—the solver performance tendencies seem rather clear. Table 5 summarizes these numerical results; the LGO results are directly imported from Table 4.

In several cases, MINOS and SNOPT issued interim (runtime) warning messages—while in most runs they report optimal solutions on return—but all runs were properly terminated with the results shown in Table 5. All solvers return close, but slightly different results for $n = 6$ and $n = 8$. The first clearly notable difference appears at $n = 10$, where IPOPT returns a somewhat inferior result compared to the other three solvers.

The numerical limitations of *all* tested solvers become more apparent as n increases. In several cases, MINOS returns clearly inferior results, and except for small values of n, it produces inferior results that are a few percent below the best solution returned considering all solvers. IPOPT consistently returns somewhat inferior results, but still within a few percent of the best solution returned by at least one of the solvers. SNOPT works well for the considered range of n values, in several cases returning slightly better objective function value estimates than LGO. In the other cases, LGO returns best results, with relatively little difference between LGO and SNOPT results. Let us point out that the constraint satisfaction levels attained by these solvers are a bit different, depending also on the $LSP(n)$ instance solved: therefore, it would be inappropriate to draw far-reaching conclusions based on rather small differences in the reported objective function values.

The LSP model-class clearly poses a challenge to the high-quality solvers tested here. Arguably, the same conclusion remains valid for other solver engines which could not be considered in the present study. To support this statement, consult also [4, 5] for the numerical results included for LSP models.

Table 5 Comparative numerical results obtained by using several AMPL solvers *nLSP*(*n*) area A(*n*)

	MINOS	SNOPT	IPOPT	LGO
6	0.6749814429	0.6749814429	0.6749814308	0.6749814433
8	0.7268684828	0.7268684827	0.7268684678	0.7268684830
10	0.7491373459	0.7491373459	0.7371215901	0.7491373457
12	0.7607298734	0.7607298734	0.7542668597	0.7607298709
14	0.7521931121	0.7675310112	0.7675309793	0.7675310106
16	0.7625954979	0.7718613220	0.7696844715	0.7718613224
18	0.7554106917	0.7747881651	0.7491373424	0.7747881650
20	0.7649920891	0.7768587560	0.7732071277	0.7768587506
22	0.7640946468	0.7783773301	0.7607298336	0.7783773308
24	0.7640946468	0.7795240452	0.7548403603	0.7795240461
26	0.7636943870	0.7804111199	0.7523851367	0.7804111201
28	0.7641232665	0.7807502582	0.7523851373	0.7811114192
30	0.4738428148	0.7813775853	0.7491373081	0.7816739255
40	0.7740433310	0.7832011593	0.7268684622	0.7832011589
50	0.5591307889	0.7820205034	0.7197409051	0.7840161496
60	0.7403488333	0.7827992931	0.6749814462	0.7844492943
70	0.7605166660	0.7846838685	0.7268685003	0.7846403575
80	0.5070738413	0.7848417622	0.7197409068	0.7848252941

6 Regression Model Development

Based on the AMPL-LGO numerical results, we present a simple nonlinear regression model that enables the estimation of the optimal area $A(n)$, for all *even* values of $n \geq 6$. Obviously, the same type of regression model could be used to estimate $A(n)$ also for *odd* values. However, based on the optimality of regular n-polygons [27], one could *exactly* compute $A(n)$ for all odd values of n.

Given that $A(n)$ is a monotonically increasing function of n, and $A(\infty) = \pi/4$, the following model form is conjectured for even values $n \geq 6$:

$$A(n) = \pi/4 - c_1/n - c_2/n^2 - c_3/n^3. \tag{6}$$

In (6), the parameters c_1, c_2, c_3 are expected to be positive. To calibrate this model, we use the estimated $A(n)$ $6 \leq n \leq 80$ values found by AMPL-LGO, substituting the three apparently suboptimal calculated $A(n)$ values by their interpolated approximation as discussed earlier.

The regression model parameters were determined using the NonlinearModelFit function of *Mathematica* [32]. This leads to the following model, rounding the coefficients to five digits after the decimal point (to reflect the expected model

Table 6 Estimated $A(n)$ values based on the regression model (7) *vs.* best known results

n	6	8	10	20	30	40
A(n) est	0.674983	0.726829	0.749185	0.776816	0.781572	0.783209
Best res	0.674981	0.726868	0.749137	0.776859	0.781674	0.783201
n	50	60	70	80	90	100
A(n) est	0.783965	0.784378	0.784629	0.784794	0.784908	0.784991
Best res	0.784016	0.784449	0.784684	0.784842	0.784946	0.785028
n	200	300	400	500	1000	2000
A(n) est	0.785270	0.785329	0.785352	0.785364	0.785384	0.785392
Best res	0.785316	0.785309	0.785356	???	???	???

accuracy):

$$A(n) \sim \pi/4 - 0.01098/n - 2.91512/n^2 - 5.96369/n^3. \tag{7}$$

Applying this regression model, in Table 6 we present an illustrative set of $A(n)$ estimates *vs.* the best known numerical results from Tables 1, 2, 3, 4 and 5. All values are rounded to 6-digit precision after the decimal point.

All estimated values shown in Table 6 are in reasonable agreement with the best numerical results presented in Tables 1, 2, 3, 4 and 5 whenever such values are available. The relative difference between the calculated best values and estimated values is less than 10^{-4} in all examples included in Table 5, except for $n = 30$, where the relative difference approximately equals $1.3 \cdot 10^{-4}$.

The calculated optima shown for $n = 90, 100, 200, 300, 400$ were found using SNOPT. (Due to the current pre-set model size limitations of AMPL-LGO, it could not be used for doing these calculations; and the solvers MINOS and IPOPT were deemed unsuitable due to their inferior performance experienced for smaller n values). The SNOPT runtimes—which were below one second in most cases for $n < 100$ (reaching about 5 s for $n = 100$) – increased rapidly when n was sequentially set to 200, 300, and 400. SNOPT was running for more than an hour on the computer mentioned earlier to solve the $n = 400$ model instance. The entries ??? for n = 500, 1000, 2000 indicate that we did not attempt to calculate these, since none of the solvers used in this study seemed capable to return numerical results within an acceptable timeframe.

Let us point out that while the *LSP*(100) model-instance has "only" 198 decision variables and 5049 constraints (omitting the prefixed values), the *LSP*(400) model-instance has 798 variables, and the number of constraints is 80199. It seems clear that solving *LSP*(n) models directly, for *arbitrarily large even* values n, is beyond the capability of current (and perhaps also of future) numerical optimization tools. This aspect makes the regression model based estimation approach a simple viable alternative.

Since the data used to develop the regression model (2) are likely to be at least slightly suboptimal, one can expect that—generally speaking—the estimated $A(n)$ values could be also suboptimal. This tendency can be observed in Table 6, but one can see also some exceptions, indicating regression model error and/or optimization inaccuracy.

Let us also note that—for the purpose of developing a regression model—the exact values $\{A(n)\}$ for odd n could also be used. However, this approach would be based on "mixing" numerically *exact* and *estimated* optima, and hence would give less indication of the quality of the numerical solutions found for even values of n. For this reason, we used only the results obtained in the present study, and we did not attempt to find adjusted optimum estimates based on further information.

To support a more complete comparative analysis, first and second order regression models (with $1/n$ as their input argument) were also calculated, but the third order model (6) clearly resulted in a superior fit to the entire data set used. Obviously, within reason, higher order models (or perhaps other model types) could give even more precise fit to the data, but—considering also the inherent data inaccuracies—the third order model (6) already gives a fairly good fit. Figure 2 displays the model function curve defined by (7) together with the adjusted data set (represented by dots) that includes the interpolated data.

Figure 3 displays the regression model residuals. With a few exceptions, the residual errors are less than $1 \cdot 10^{-4}$; the absolute value of the singularly largest estimated error is around $8 \cdot 10^{-4}$. All estimated error values are fairly small, compared to the approximate range [0.674981, 0.784825] of the observed data.

One can observe that most of the residuals seem to follow an interesting cyclical pattern that—in the author's opinion—seems more due to the inherent structure of the $LSP(n)$ problem-class than to numerical fluctuations and other "noise" induced by the computational environment.

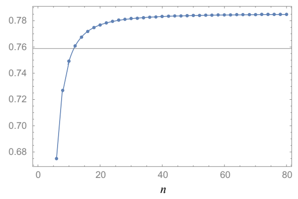


Fig. 2 The nonlinear regression model (7), *vs.* the adjusted data set $A(n)$ (dots) for $6 \leq n \leq 80$

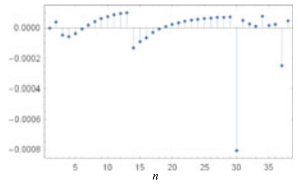


Fig. 3 Residuals (see dots) in the regression model (7) of $A(n)$, for $3 \leq n \leq 40$

7 Concluding Remarks

In this study, we address the problem of finding numerically the sequence of largest small n-polygons $LSP(n)$ with unit diameter and maximal area $A(n)$. Finding $LSP(n)$ and $A(n)$ for even values of $n \geq 6$ has been a long-standing challenge, leading to an interesting class of nonlinear optimization problems with different formulations by a number of researchers.

The structural properties of this problem, and of similar optimization challenges—e.g., atomic structure models, potential energy models, regular object packings, and other problems in which the goal is to find the best configuration of identical objects—often support the proposition of "credible" initial solutions and solution guesses. However, finding the true global solution typically remains difficult, as the cited earlier studies and our present work illustrate.

Using the AMPL modeling environment with the LGO solver option, we present global search based numerical solutions for all even values $6 \leq n \leq 80$, in a matter of minutes. Our results are comparable to (and in a number of cases are somewhat better than) the best results obtained earlier by other authors and by other solver software options, before our results were produced. Based on the results obtained, we also propose a regression model that enables the simple estimation of the optimal area sequence $\{A(n)\}$, for arbitrary integer values of n.

Upon revising the manuscript of this work, it was brought to our attention by a helpful reviewer that a recently posted (September 2020) study [3] presents *numerical optimum estimates* which are somewhat better for $n \geq 32$ than the *numerical optimum estimates* presented in our work. To illustrate, [3] reports for $n = 32$ the estimate 0.7821325276 *versus* our estimate 0.7818946320: the approximate ratio of these values is 0.9996958. To produce the results reported in [3] for a selection of even values $n \leq 128$, a different global optimization model was used; MATLAB and CVX serve as the modeling environment, and the MOSEK Optimization Suite was used (with a default precision setting which, without delving into further details, seems to be similar to the numerical feasibility tolerance used in our study).

In response to the above, we point out that [3] cites our numerical results obtained in 2018 (and left unchanged for the present study). We never claimed more than producing credible numerical optimum estimates using off-the-shelf optimization software which returns results in seconds or minutes for the model instances discussed here.

Let us add that in a recently completed study [26] we present numerical results for an illustrative sequence of even values of n, up to $n \leq 1000$. Our results are in close agreement with (or surpass) the best results reported in all earlier studies known to us, including the results presented in [3]. For completeness, we also calculated numerically optimized results for a selection of odd values of n, up to $n \leq 999$. In this study, we used a tighter model formulation; to handle this model, *Mathematica* was used with the IPOPT solver option. Following up by corresponding regression models (similarly to our present work), we present *numerical solution estimates* for the entire LSP model-class.

The motto of the benchmarking studies [4, 5] is, arguably, somewhat provocative and funny, but the message is worth quoting: "*COPS: Keeping optimization software honest.*" In line with this message, let us conclude with some honest and pragmatic advice, not driven by unconditional "*software developer's pride*". Facing the vast universe of nonlinear optimization problems, it is advisable to refrain from confident blanket statements regarding the superiority of any particular solver software over others. Instead, it is good practice to use a repertoire of appropriate model versions and solver options whenever possible, especially since it may not be obvious a priori which model type or solver engine will work best for a novel or unusually hard optimization challenge.

Acknowledgements The author thanks David M. Gay (AMPL Optimization, Inc.) for his contributions to the AMPL-LGO solver link development, and for many useful discussions. Thanks are due also to AMPL Optimization and Wolfram Research for their long-standing support of the author's research.

References

1. AMPL Optimization, AMPL. www.ampl.com, 2020.
2. C. Audet, P. Hansen, F. Messine, and J. Xiong, The Largest Small Octagon, *Journal of Combinatorial Theory, Series A* 98 (2002), pp. 46–59.
3. Bingane, Largest Small Polygons: A Sequential Convex Optimization Approach. Available at https://arxiv.org/pdf/2009.07893.pdf. (Manuscript dated September 18, 2020.)
4. A.S. Bondarenko, D.M. Bortz, and J.J. Moré, *COPS: Large-Scale Nonlinearly Constrained Optimization Problems*, *Technical Report ANL/MCS-TM-237*, Argonne National Laboratory, Argonne, IL, 1998.
5. E.D. Dolan, and J.J. Moré, *Benchmarking Optimization Software with COPS*, *Technical Report ANL/MCS-TM-246*, Argonne National Laboratory, Argonne, IL, 2000.
6. R. Fourer, D.M. Gay, and B.W. Kernighan, *AMPL: A Modeling Language for Mathematical Programming* (2$^{\text{nd}}$ Edition), Brooks/Cole-Thomson Learning, Pacific Grove, CA, 2003.
7. GAMS Development Corporation, *Model libraries; polygon.gms: Largest small polygon COPS 2.0 #1*, https://www.gams.com/latest/gamslib_ml/libhtml/gamslib_polygon.html; 2018.
8. D.M. Gay, *pgon.mod*, https://netlib.org/ampl/models/pgon.mod; 1998.
9. D.M. Gay, The AMPL Modeling Language: An Aid to Formulating and Solving Optimization Problems, in M. Al-Baali, L. Grandinetti, and A. Purnama, Eds., *Numerical Analysis and Optimization: NAO-III* (Muscat, Oman, January 2014), pp. 95–116. Springer International Publishing, Switzerland, 2015.
10. P.E. Gill, W. Murray, and M.A. Saunders, *User's Guide for SNOPT Version 7: Software for Large-Scale Nonlinear Programming,* Systems Optimization Laboratory, Department of Management Science and Engineering, Stanford University, Stanford, CA, 2006.
11. R.L. Graham, The Largest Small Hexagon, *Journal of Combinatorial Theory, Series A* 18 (1975) pp. 165–170.
12. D. Henrion, and J.B. Lasserre, GloptiPoly: Global Optimization over Polynomials with Matlab and SeDuMi, *ACM Transactions on Mathematical Software* 29 (2003) 2, pp. 165–194.
13. D. Henrion, J.B. Lasserre, and Loefberg, J., GloptiPoly 3: Moments, Optimization and Semidefinite Programming, *Optimization Methods and Software* 24 (2009), 4–5, pp. 761–779.
14. D. Henrion, and F. Messine, Finding Largest Small Polygons with GloptiPoly. Available at https://arxiv.org/pdf/1103.4456.pdf. (Current manuscript version dated June 17, 2018.)

15. IPOPT, *Ipopt, a Library for Large-scale Nonlinear Optimization*; https://projects.coin-or.org/Ipopt. The authors of Ipopt are listed at https://projects.coin-or.org/Ipopt/browser/trunk/Ipopt/AUTHORS; 2018.
16. C. Jansson, VSDP: A Matlab Software Package for Verified Semidefinite Programming, *Nonlinear Theory and its Applications* (2006) pp. 327–330.
17. M.J. Mossinghoff, Isodiametric Problems for Polygons. *Discrete and Computational Geometry*, 36 (2006) 2, pp. 363–379.
18. B.A. Murtagh, and M.A. Saunders, *MINOS 5.5 User's Guide. Technical Report SOL 83–20R.* Systems Optimization Laboratory, Department of Management Science and Engineering, Stanford University, Stanford, CA, 1983; revised version 1998.
19. J.D. Pintér, *Global Optimization in Action.* Kluwer Academic Publishers, Dordrecht, 1996.
20. J.D. Pintér, Global Optimization: Software, Test Problems, and Applications, in Pardalos, P.M. and Romeijn, H.E., Eds. *Handbook of Global Optimization*, *Volume 2*, pp. 515-569. Kluwer Academic Publishers, Dordrecht, 2002.
21. J.D. Pintér, Software Development for Global Optimization, in Pardalos, P.M. and T. F. Coleman, Eds. *Global Optimization: Methods and Applications*, pp. 183–204. Fields Institute Communications Volume 55. Published by the American Mathematical Society, Providence, RI, 2009.
22. J.D. Pintér, *AMPL-LGO User's Guide*, AMPL Optimization, Inc.
23. J.D. Pintér, *LGO Solver Suite for Global and Local Nonlinear Optimization: User's Guide,* PCS, Inc.
24. J.D. Pintér, How Difficult is Nonlinear Optimization? A Practical Solver Tuning Approach, with Illustrative Results, *Annals of Operations Research* 265 (2018), pp. 119–141.
25. J.D. Pintér, F.J. Kampas, and I. Castillo, Globally Optimized Packings of Non-uniform Size Spheres in R^d: A Computational Study, *Optimization Letters* 12 (2018) 3, pp. 585–613.
26. J.D. Pintér, F.J. Kampas, and I. Castillo, Finding the Sequence of Largest Small n-Polygons by Numerical Optimization. https://www.optimization-online.org/DB_FILE/2020/11/8112.pdf.
27. K. Reinhardt, Extremale Polygone gegebenen Durchmessers, *Jahresbericht der Deutschen Mathematiker-Vereinigung* 31(1922), pp. 251–270.
28. J.F. Sturm, Using SeDuMi 1.02, A Matlab Toolbox for Optimization over Symmetric Cones, *Optimization Methods and Software* 11 (1999) (1–4), 625–653.
29. E.W. Weisstein, "*Biggest Little Polygon.*" From MathWorld – A Wolfram Web Resource. https://mathworld.wolfram.com/BiggestLittlePolygon.html. (Retrieved on June 28, 2020)
30. E.W. Weisstein, "*Graham's Biggest Little Hexagon.*" From MathWorld—A Wolfram Web Resource. https://mathworld.wolfram.com/GrahamsBiggestLittleHexagon.html. (Retrieved on June 28, 2020)
31. E.W. Weisstein, "*Regular Polygon.* " From MathWorld—A Wolfram Web Resource. https://mathworld.wolfram.com/RegularPolygon.html. (Retrieved on June 28, 2020)
32. Wolfram Research, *Mathematica (Version 11.0)*, Wolfram Research, Inc., Champaign, IL, 2018.
33. D.R. Woodall, Thrackles and Deadlock, in D.J.A. Welsh, Ed. *Combinatorial Mathematics and Its Applications*, pp. 335-347, Academic Press, New York, 1971.
34. B. Yuan, *The Largest Small Hexagon,* M.Sc. Thesis, Department of Mathematics, National University of Singapore, 2004.

Computational Science in the 17th Century. Numerical Solution of Algebraic Equations: Digit–by–Digit Computation

Trond Steihaug

Abstract In this paper we give a complete overview of test–problems by Viète from 1600, Harriot from 1631 and Oughtred from 1647. The original material is not easily accessible due to archaic language and lack of conciseness. Viéte's method was gradually elucidated by the subsequent writers Harriot and Oughtred using symbols and being more concise. However, the method is presented in tables and from the layout of the tables it is difficult to find the general principle. Many authors have therefore described Viète's process inaccurately and in this paper we give a precise description of the divisor used in the process which has been verified on all the test–problems. The process of Viète is an iterative method computing one digit of the root in each iteration and has a linear rate of convergence and we argue that the digit–by–digit process lost its attractiveness with the publications in 1685 and 1690 of the Newton-Raphson method which doubles the number of digits for each iteration.

Keywords History of mathematics in the 17th century · History of numerical analysis

Mathematics Subject Classification: 01A45 · 65-03

1 Introduction

Viète wrote two treatises on solving equations: one theoretical and the other numerical. The second treatise *De numerosa potestatum ad exegesim resolutione* or *On the numerical resolution of powers* was published in 1600 and offered something quite new. Here Viète took equations that could be solved only with difficulty, or not at all, by standard methods and showed how numerical solutions could be found to whatever degree of accuracy was required [26]. Viète exemplifies his technique on solving equations on numerous examples more like what we find in more modern papers on numerical solution of nonlinear equations. All examples by Viète have

T. Steihaug (✉)
Department of Informatics, University of Bergen, Box 7803, 5020 Bergen, Norway
e-mail: Trond.Steihaug@ii.uib.no

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*, Springer Proceedings in Mathematics & Statistics 354, https://doi.org/10.1007/978-3-030-72040-7_12

integer solutions. Viète's work was the first comprehensive method of solving such equations that had been attempted, and it involved no restrictions as to terms, signs, or degree [17]. Viète's method closely resembles the method of Šaraf–al–Din al-Tūsī (died in the last quarter of the 12th century) [21]. This method is described in a manuscript on algebra entitled *On equations*. However, Viète's treatise from 1600 contains the first printed version of such a method.

The *Appendice Algébraique* of 1594, an appendix to *L'arithmetique* from 1585, Simon Stevin writes that after the publication of L' Arithmétique he has found a general rule to solve all equations either perfectly or with any degree of approximation. The appendix itself was reproduced in French and Latin in 1608 and in the reprint of L'Arithmétique by A. Girard of 1625. The processes presented by Stevin and Viète compute the solution or root one digit at time. There are basically two stages in such a process, first ascertain the number of digits in the solution and determine the first digit of the solution. The next stage is to determine one digit at a time. If the sought root is an integer, the process terminates after a few steps.

The first printed method for numerical solution of equations is that of Gerolamo Cardano (1501–1576) in *Ars Magna* from 1545 under the title *De regula aurea*. This is the first successful general methods of approximating roots of algebraic equations. The method was known in manuscripts and commonly referred to as the Rule of Double False Position since the 11th century. In *Ars Magna* there are four examples using the double false position. The double false position is a bracketing methods where the solution of the equation will be in the interval. The secant method uses the same linear interpolation as the double false position, but is not a bracketing technique. In "Newton's Waste Book" ([39, p. 489-49] and there tentatively dated to early 1665) Ypma [40] identifies the method used by Newton to be the secant method. It is well known that these techniques are not digit–by–digit computation.

The invention of decimal fractions is usually ascribed Simon Stevin [2, p. 314], but most importantly he introduced their use in mathematics in Europe. Simon Stevin wrote a booklet called *De Thiende* or "the art of tenths", first published in Dutch in 1585, translated into French the same year and to English in 1608. With the work of Simon Stevin, the classical restriction of "numbers" to integers or to rational fractions was eliminated. For Stevin, the real numbers formed a continuum. His general notion of a real number was accepted, tacitly or explicitly, by all later scientists [31, p. 69].

Viète does not use decimals. In [32, 33] Viète writes (translation by Witmer [34])

> Thus if you are seeking the root of 2, a square, extract, if you wish, the root of[1] 2 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00, [which is] 141, 421, 356, 237, 309, 505. So the root of 2 is said to be approximately $\frac{141{,}421{,}356{,}237{,}309{,}505}{100{,}000{,}000{,}000{,}000{,}000}$.

Both Harriot [6] and Oughtred [18] use the same process as Viète but make different arrangement of the computation. Harriot shows the use of fractions and Oughtred also shows computing a solution with decimals. Wallis [35] gives a better estimate of the new digit than Viète, Harriot, and Oughtred by not excluding terms in the divisor.

The mathematical notation of equations changes in the period we consider. In addition the methods or processes are illustrated using examples leaving some details

[1] The integer $2 \cdot 10^{28}$ in Viète is replaced by $2 \cdot 10^{34}$.

Viète	$x^5 + 500x = 254832$	1QC+500N, aequetur 254832
Harriot	$x^5 + 500x = 254832$	$aaaaa + 500a$ = = = 254832
Oughtred	$x^5 - 15x^4 + 160x^3 - 1250x^2$ $+6480x = 170304782$	1qc-15qq+160c-1250q+6480ℓ = 170304782
Newton MS	$x^3 + 30x = 14356197$	$\mathcal{L}$c+30$\mathcal{L}$=14356197
Wallis	$x^3 - 2x^2 = 186494880$	Rc-2Rq=186494880
Newton [14]	$x^4 - x^3 - 19x^2 + 49x - 30 = 0$	$x^4 - x^3 - 19xx + 49x - 30 = 0$

Fig. 1 Examples of mathematical notation in the 17th century

to interpretations and an example of the arrangement is given in Fig. 2. Figure 1 contains examples of the notation used by the cited authors in the different sections of the paper.

Cajori [1] in 1916 was one of the first to point out that the Viète process has been described inaccurately by leading historians at that time, including Cantor in 1900.[2] Due to the inexplicitness of Viète process, writers like Augustus De Morgan in 1847,[3] Cajori [1, p. 40], and Nordgaard [17, p. 28] have misinterpreted the process. Rashed in 1974 [21], Goldstin in 1977 [4, p. 66] and Ypma in 1995 [40] make the error of stating an explicit formula to determine a digit. As will be shown the general equations stated in Sect. 3 serve as estimates to determine the digits used by Viète, Harriot and Oughtred. Section 3 contains a description of Viète's method and all test examples. However, the first section on solving equations is on Stevin's method in Sect. 2. The next two sections contains the examples used by Harriot in Sect. 4 and Oughtred in Sect. 5. Compared to extensive reuse of the test problems [27] of Joseph Raphson [20] from 1690, there are few authors that reuse the test problems of Viète.

In Sect. 6 is Newton's annotations from Viète and Oughtred. The annotation represents 'state of the art' in mid 17th century but the manuscript was never published. Newton's method was first published in print in Wallis' algebra in 1685, but the algebra book also introduces a modification of the divisor used by Viète, Harriot and Oughtred. This is treated in Sect. 7.

Already in 1670 we find evidence in a letter from Collins to Leibniz, that the computational work using the Viète process *unfit for a Christian, and more proper to one that can undertake to remove the Italian Alps into England* [22]. In Sect. 8 we argue that with the presence of higher order methods at the end of the 17th century the use of digit–by–digit calculation for algebraic equation diminished. However, new variations of digit–by–digit methods for algebraic equations appears in the two books by John Ward [37, 38], a new method by Horner [8] and by Holdred [7]. The digit–by–digit process survived in textbooks for hand calculation until the age of

[2] M. Cantor, Vorlesungen über Geschichte der Mathematik, II, 1900, p. 640–641.

[3] Augustus De Morgan, Involution and Evolution, in The Penny Cyclopaedia of the Society for the Diffusion of Useful Knowledge, London 1846, Volume 2 p. 103.

calculators and was practised a lot to compute square roots. This process is discussed in the final Sect. 9.

2 Stevin's Method 1594

The *Appendice Algébraique* of 1594, Stevin uses two examples. The first equation has an integer solution. The second equation has a root that is not integral, but Stevin does not use the decimal notation of his *De Thiende*. The two examples are $x^3 = 300x + 33915024$ and $x^3 = 300x + 33900000$. To find a first approximation for x, try $x = 10^k$, for $k = 0, 1, 2 \ldots$. The result is that for $k = 2$ the value of x^3 is less than that of $300x + 33915024$, but for $k = 3$, the value of x^3 is larger. Hence there will be 3 digits in the root (if integer) To find the first digit, or approximation for x, he now substitutes $x = 100, 200, 300, 400$ and finds $300 < x < 400$. The first digit is then 3. Now he tries $x = 3 \cdot 10^2 + 10,\ 3 \cdot 10^2 + 20,\ 3 \cdot 10^2 + 30$ and finds $320 < x < 330$ and the second digit is 2. Then $x = 3 \cdot 10^2 + 2 \cdot 10 + 1,\ 3 \cdot 10^2 + 2 \cdot 10 + 2,\ 3 \cdot 10^2 + 2 \cdot 10 + 3,\ 3\ 10^2 + 2\ 10 + 4$. It appears that for $x = 3 \cdot 10^2 + 2 \cdot 10 + 4 = 324$ both sides of the equation finally are equal so that $x = 324$ is the root.

Stevin points out that the method can also be applied if the root is not an integral number. Consider $x^3 = 300x + 33900000$ and we find $323 < x < 324$. Then write $x = 323 + \frac{d_1}{10}$ and test for $d_1 = 0, 1, \ldots, 9$ and we find the first decimal digit $d_1 = 9$. Proceed with $x = 323.9 + \frac{d_2}{100}$ as above, with $d_2 = 0, 1, 2, 3, 4, 5, 6$ to find $d_2 = 5$, then $x = 323.95 + d_3 \cdot 10^{-3}$ etc. This can go on indefinitely.

Stevin's method was made popular in the algebra in four volumes by John Kersey in 1673 and 1674 where four examples are given with quadratic, cubic and fourth order polynomials and using decimals [13, Book II, Ch.X].

3 Viète's Method 1600

Viète's method is an extension of Stevin's method. Where Stevin systematic examines all digits $0, 1, \ldots, 9$, Viète makes an estimate of the digit and either increases or decreases it. Hutton in 1795 [9, Algebra, p. 87] and [10, Tract 33, p. 270] writes:

> The method is very laborious, and is but little more than what was before done by Stevinus on this subject, depending not a little upon trials.

However, it is today agreed that François Viète was not familiar with the works of Stevin [34, Translator's introduction].

Viète divides the examples in two; pure and affected equations. For the pure equations Viète uses the technique presented in Sect. 9.

Table 1 Pure equations in Viète 1600 [32, p. 3r-6v] and 1646 [33, p. 166–172]

Name	$p(x) = N$	Solution
Problem I	$x^2 = 2916$	54
Problem II	$x^3 = 157464$	54
Problem III	$x^4 = 331776$	24
Problem IV	$x^5 = 7962624$	24
Problem V	$x^6 = 191102976$	24

3.1 *Pure Equations*

In the section *Purarum resolutione* in *De numerosa potestatum purarum, atque adfectarum ad exegesin resolutione tractatus* Viète [33, p. 163–172] demonstrates digit-by-digit computation on five problems using the technique in Sect. 9. Table 1 contains problem number used by Viète and the solution.

In Nordgaard [17, p. 25] is the arrangement of the computation in Viète's Problem II $x^3 = 157464$ with a close paraphrase in modern notation.

3.2 *Affected Equations*

In the section *Adfectarum resolutione* in *De numerosa potestatum purarum, atque adfectarum ad exegesin resolutione tractatus* Viète [33, p. 173–223] gives numerous examples of digit–by–digit computation for positive roots of polynomials. It is generally agreed that the language used by Viète is archaic and there is an absence of clear symbolism and conciseness [3]. Taking Problem IX which in the notation of Viète is

> Quidam numerus ductus in sui Quadrato-cubum, & in 6000 facit 191,246,976. Queritur quis fit numerus ille. In notis 1CC+6000N æquatur 191,246,976 & fit 1N unitatatum quout?

will in modern language be *A certain number multiplied by its sixth power and by 6000 makes 191,246,976. The question is what that number is. In symbols*, $x^6 + 6000x = 191,246,976$. *What is* x?[4] An equation is "duly prepared" according to Viète if the coefficients of the polynomial are integers and N is positive and Viète partition the equations in affected positively (I to IX), affected negatively (X to XII), mixed (XIII to XV) and avulsed (XVI to XX). The positively and negatively affected equations have only one positive root.

[4] Translated by T. Richard Witmer [34].

Table 2 Quadratic equations in Viète 1600 [32] and 1646 [33]

Name	$p(x) = N$	Solution	1600	1646
Problem (Ia)	$x^2 + 7x = 60750$	243	p. 7v	p. 174
Example (b)	$x^2 + 954x = 18487$	19	p. 8r	p. 175
Problem (Xa)	$x^2 - 7x = 60750$	250	p. 18v	p. 195
Example (b)	$x^2 - 240x = 484$	242	p. 19v	p. 196
Example (c)	$x^2 - 60x = 1600$	80	p. 20r	p. 197
Example (d)	$x^2 + 8x = 128$	8	p. 20r	p. 197
Problem (XVIa)	$-x^2 + 370x = 9261$	27	p. 27v	p. 211
Example (b)	$-x^2 + 370x = 9261$	343	p. 28r	p. 212

Consider the quadratic equation $x^2 + cx + d = 0$. Let $f(x) = x^2 + cx + d$ and consider $f(x + h) = 0$ for given $x > 0$. A bound on h can be found from $h(2x + c) \le -f(x)$, and provided $2x + c > 0$ the upper bound on h is

$$h \le \frac{-f(x)}{2x + c}. \tag{1}$$

All the coefficients in the examples used by Viète are positive. For the quadratic equation $-x^2 + cx + d = 0$ the corresponding bound will be an upper bound on h, $h \le \frac{-f(x)}{-2x+c}$ when $-2x + c \ge 0$. We can observe that the bound in the quadratic case is the correction to the current iterate x in the Newton-Raphson method $-\frac{f(x)}{f'(x)}$. Rashed [21, p. 269] points out that the method of Šaraf–al–Din al-Tūsī and Viète are identical for quadratic equations. Problem 1a) is reproduced in [21, p. 266–267] and compared to the method of Šaraf–al–Din al-Tūsī. In Table 2 the first column gives the name of the problems which are found in the second column. The third column is the solutions and the two last columns give the page numbers in the 1600 and 1646 editions.

For Problem XVIa the bound on $h = \alpha_0$ (the second and last digit) will be a lower bound. Problem XVIa) and b) are Problem 4 in Harriot [6, p. 128].

We now consider the cubic equations given in Table 3. Let $f(x) = x^3 + bx^2 + cx + d$ and consider the cubic equation $f(x + h) = 0$ for given x. Then

$$f(x + h) = f(x) + h(3x^2 + 2bx + c) + h^2(3x + b) + h^3 = 0. \tag{2}$$

Viète eliminates h^3 so $-f(x) \ge h(3x^2 + 2bx + c) + h^2(3x + b)$. If $3x + b \ge 0$, initial $\underline{h} \le h$ and $3x^2 + 2bx + c \ge 0$ then an upper bound on h is,

$$h \le \frac{-f(x)}{3x^2 + 2bx + c + (3x + b)\underline{h}}.$$

Table 3 Cubic equations in Viète 1600 [32] and 1646 [33]

Name	$p(x) = N$	Solution	1600	1646
Problem (IIa)	$x^3 + 30x = 14356197$	243	p. 9r	p. 176
Example (b)	$x^3 + 95400x = 1819459$	19	p. 10r	p. 178
Problem (IIIa)	$x^3 + 30x^2 = 86220288$	432	p. 10v	p. 180
Example (b)	$x^3 + 10000x^2 = 5773824$	24	p. 11v	p. 182
Problem (XIa)	$x^3 - 10x = 13584$	24	p. 20r	p. 198
Example (b)	$x^3 - 116620x = 352947$	343	p. 21r	p. 199
Example (c)	$x^3 - 6400x = 153000$	90	p. 22r	p. 200
Example (d)	$x^3 + 64x = 1024$	8	p. 22r	p. 201
Problem (XIIa)	$x^3 - 7x^2 = 14580$	27	p. 22r	p. 201
Example (b)	$x^3 - 10x^2 = 288$	12	p. 22v	p. 202
Example (c)	$x^3 - 7x^2 = 720$	12	p. 23v	p. 203
Example (d)	$x^3 + 8x^2 = 1024$	8	p. 24r	p. 204
Problem (XVIIa)	$-x^3 + 13104x = 155520$	12	p. 29r	p. 214
Example (b)	$-x^3 + 13104x = 155520$	108	p. 29v	p. 215
Problem (XVIIIa)	$-x^3 + 57x^2 = 24300$	30	p. 30r	p. 216
Example (b)	$-x^3 + 57x^2 = 24300$	45	p. 30v	p. 217

In Sect. 6 on Newton's annotation the above bound is the same as the bound (4) when $b = 0$. $\underline{h}$ will depend on the number of digits in the solution, k, and the order $j > 1$, 10^{k-j}. For the cubic equation, the bound is the Newton-Raphson correction only when $\underline{h} = 0$. The method of Šaraf–al–Din al-Tūsī and Viète deviates for cubic equations where the estimate of the next digit is the Newton-Raphson correction $-f(x)/f'(x)$ scaled [21]. To show the differences, Rashed [21, p. 268–270] used Problem IIa.

The arrangement of Viète's Problem IIa) and IIb) and a close paraphrase in modern notation is found in Nordgaard [17, p. 26–27].

Let f be a polynomial of degree n on the form $f(x) = x^n + q(x)$ where q is a polynomial of degree $n - 1$. We can write

$$q(x + h) = \sum_{i=0}^{n-1} \frac{h^i}{i!} q^{(i)}(x)$$

where $q^{(i)}$ is the ith derivative of q. Further

$$(x + h)^n = x^n + \sum_{i=1}^{n-1} \binom{n}{i} x^{n-i} h^i + h^n \geq x^n + h \sum_{i=1}^{n-1} \binom{n}{i} x^{n-i} \underline{h}^{i-1}$$

for $0 \leq \underline{h} \leq h$. So if all $q^{(i)}(x) \geq 0$ then

Table 4 Higher order algebraic equations in Viète 1600 [32] and 1646 [33]

Name	$p(x) = N$	Solution	1600	1646
Problem (IVa)	$x^4 + 1000x = 355776$	24	p. 12v	p. 183
Example (b)	$x^4 + 100000x = 2731776$	24	p. 13v	p. 185
Problem V	$x^4 + 10x^3 = 470016$	24	p. 14r	p. 186
Problem (VIa)	$x^4 + 200x^2 = 446976$	24	p. 14v	p. 187
Example (b)	$x^4 + 200x^2 + 100x = 449376$	24	p. 15r	p. 188
Problem VII	$x^5 + 500x = 254832$	12	p. 16r	p. 190
Problem VIII	$x^5 + 5x^3 = 257472$	12	p. 16v	p. 191
Problem IX	$x^6 + 6000x = 191246976$	24	p. 17v	p. 193
Problem XIII	$x^4 - 68x^3 + 202752x = 5308416$	32	p.24r	p. 205
Problem XIV	$x^4 + 10x^3 - 200x = 1369856$	32	p.25r	p. 207
Problem XV	$x^5 - 5x^3 + 500x = 7905504$	24	p. 26r	p. 208
Problem (XIXa)	$-x^4 + 27755x = 217944$	8	p. 31v	p. 219
Example (b)	$-x^4 + 27755x = 217944$	27	p. 32r	p. 220
Problem XX	$-x^4 + 65x^3 = 1481544$	38	p. 32v	p. 221
Example (b)	$-x^4 + 65x^3 = 1481544$	57	p. 33v	p. 222

$$- f(x) \geq h \left[\sum_{i=1}^{n-1} \binom{n}{i} x^{n-i} \underline{h}^{i-1} + \sum_{i=1}^{n-1} \frac{\underline{h}^{i-1}}{i!} q^{(i)}(x) \right]. \tag{3}$$

This general case (3) will give an estimate which is either an upper bound or lower bound on h. The first sum is by Viète called the lower part and second sum is upper part based on the table Viète uses. To recognize the computation in the tables of Viète, Harriot and Oughtred note that $f(x_{j+1}) = f(x_j) + (f(x_{j+1}) - f(x_j))$ and some of the terms in $f(x_{j+1}) - f(x_j)$ are also needed in computing the divisor. Further details are given in Sect. 7.

Consider $f(x) = x^4 + ax^3 + bx^2 + cx + d$. Then the two parts will be

$$\text{(lower) } 4x^3 + 6x^2\underline{h} + 4x\underline{h}^2 \text{ and (upper) } 3ax^2 + 2bx + c + \underline{h}(3ax + b).$$

Viète computes the sum of the lower and upper part to find an estimate on the (next) digit.

An annotated version of Problem XV is found in [3, p. 214–216] and with explanations omitted in [15, p. 37]. The last section in Viète's book [33, p. 228] there is an example on how to transform the equation to get the root correct to the tenths and to the hundredths by scaling the variables. Given $x^3 + 6x = 8$. Substitute x by $\frac{x}{10}$ and the equation will be $x^3 + 6 \cdot 10^2 x = 8 \cdot 10^3$ and solve the new equation using the Viète process and the approximate root of the original equation will be $\frac{11}{10} = 1\frac{1}{10}$ which will be correct to the tenths.

4 Test Examples from Harriot 1631

The test examples by Harriot are from the chapter *Exegetice numerosa* [6, p. 117–167] or Numerical Exegesis [23, p. 129–182] in *Praxis* [6]. In the manuscripts Harriot refers all his examples to Viète and each manuscript page with an example is marked *De numerosa potestatum resolutione* [23]. However, only three of the examples in *Praxis* are from Viète. Where Viète splits the computation of the divisor into a lower part (corresponding to x^n) and the upper part (the remaining divisor), Harriot also splits the order of the terms in two parts, without the correcting term $\underline{h}$ and the part with the correction $\underline{h}$. Further the tables in *Praxis* contains symbols commenting the computation where Viète has a verbal description.

In one example, Problem 6 in Table 7, Harriot suggests another table corresponding to $\underline{h} = 0$. This will also yield an upper bound. Three problems in Praxis in Table 7 are identical to problems in Viète and two of these problems are used by Newton in his manuscript [39, p. 63–71]:

– Problem 4 (p. 128) in Table 7 is Viète's Problem XVI a) and b) in Table 3
– Example (p. 138) in Table 7 is Viète's Problem IIb in Table 3 and also used by Newton in Table 5.
– Example (p. 143) in Table 7 is Viète's Problem XIa in Table 3 and also used by Newton in Table 5.

Problem 13 (p. 155) in Table 7 is reproduced in [15, pp. 38–39] and in [17, p. 30] using Harriot's original formulation and notation

$$aaaa - 1024aa + 6254a = 19633735875.$$

Hankel's book on history of mathematics from 1874 illustrates Viète's method using one of Harriots's examples [6, p. 164] $x^2 + 14x = 7929$ in Table 7 computing the approximate root 82.319 [5, p. 370] using decimals.

The four 'pure' powers, Problem 1, 5, 11, and 15 in Table 7, Harriot uses the same technique as for 'affected' equations. The computation will be the same as described in Sect. 9 where we always get an upper bound on the digit.

5 Test Examples from Oughtred 1647/48

No one did more to popularize the new method of Viète than did the clergyman mathematician William Oughtred. This he accomplished by giving private tuition to ambitious young men and these spread his teachings throughout Great Britain;

Table 5 Examples in Newton's note [39, pp. 63–71]

$f(x)$	Reference(s)
$x^2 - 2916$	Problem I in Viète
$x^3 - 157464$	Problem II in Viète
$x^5 - 7962624$	Problem IIII in Viète
$x^3 + 30x - 14356197$	Problem (IIa) in Viète and in MS
$x^3 + 95400x - 1819459$	Example (IIb) in Viète, and in Harriot and MS
$x^3 - 10x - 13584$	Problem (XIa) in Viète and MS
$x^3 - 116620x - 352947$	Example (XIb) in Viète, and in Harriot and MS

among them were Seth Ward, Christopher Wren, and John Wallis [17, p. 31]. John Wallis [35] devotes a chapter on Mr. Oughtred and his Clavis [35, Ch. xv] and points out that Oughtred's contributions to Viète's method were in the simplification of the notation.

The second edition of William Oughtred (1574–1660) *Clavis Mathematicæ* (The Key to Mathematics) was published in 1648 and an English translation in 1647. In the chapter *Some examples of equations resolved in numbers* Oughtred [19, p. 139–172] considers 16 examples using a digit-by-digit computation. In Table 6 the page numbers refer to the translated version from 1647. The first edition from 1631 contains only two examples using digit–by–digit computation, $\sqrt{3272869681} = 57209$ and $\sqrt[3]{187237601580329} = 57209$. These two examples are also in the second edition. Before the year 1700 five editions of this little volume had been published. The Clavis opens with an explanation of the Hindu-Arabic notation of decimal fractions. Oughtred would write $15|\underline{7}$ for 15.7.

The 16 examples are also used by Jeake [11, 12] which also includes some additional examples and comments on the computation. The same arrangement of the computation of Example 1 in Table 6 is found in De Morgan [15, p. 39–40] using the Oughtred's notation in Fig. 1. Oughtred's computation in Example 2 in Table 6 is discussed by Caljori [1, pp. 458–459]. This example has the same form as discussed in Sect. 6 and in this case (5) is an equality.

6 On Newton's Annotations 1664

In an unpublished note from 1664(?) reproduced in [39, p. 63–71] Newton annotates Viète's *Opera Mathematica* from 1646 using the simplified notation in Oughtred's *Clavis Mathematicæ* from 1648. Newton gives 7 examples computing the root digit–by–digit. This unpublished note represents the 'state of the art' in mid 17th century. In the table below the references are to Viète 1600 and 1646 [32, 33], to Harriot [6] and MS is Harriot's manuscripts collected by Stedall [24]. The first column in Table 5 gives the function and the second contains the references.

Table 6 Test examples from oughtred 1647/48

Name	$p(x) = N$	Solution
Example 1 (p. 140)	$x^5 - 15x^4 + 160x^3 - 1250x^2 + 6480x = 170304782$	47
Example 2 (p. 142)	$x^3 + 420000x = 247651713$	417
Example 3 (p. 143)	$x^3 + 1007x^2 = 247617936$	417
Example 4 (p. 145)	$x^4 - 44299005x = 22252086$	354
Example 5 (p. 146)	$x^4 - 124600x^2 = 89726256$	354
Example 6 (p. 147)	$x^4 - 340x^3 = 621066096$	354
Example 7 (p. 149)	$x^4 - 77108000x = 85530576$	426
Example 8 (p. 150)	$-x^3 + 3200x = 46577$	47
Example 9 (p. 151)	$-x^3 + 3200x = 46577$	15.7
Example 10 (p. 152)	$-x^3 + 53x^2 = 13254$	47
Example 11 (p. 153)	$-x^3 + 53x^2 = 13254$	20.05
Example 12 (p. 154)	$-x^3 + 60034x = 1023768$	236
Example 13 (p. 155)	$-x^3 + 60034x = 1023768$	17.135
Example 14 (p. 156)	$x^4 - 72x^3 + 238600x = 8725815.7056$	47.6
Example 15 (p. 158)	$-x^3 + 3x = 1.258640782100$	0.4499
Example 16 (p. 154)	$x^5 - 5x^3 + 5x = 1.147152872702092$	0.2437

For the first three problems the algorithm is the one used in Sect. 9. The other problems are all on the form $x^3 + cx = d$ and a meta description of the algorithm is:

– Step 1: Determine the number of digits in the root, say $x = \alpha_2 10^2 + \alpha_1 10 + \alpha_0$.
– Step 2: Determine the first digit α_2: Choose the largest $0 < \alpha_2 \leq 9$ so that

$$(\alpha_2 10^2)^3 + c(\alpha_2 10^2) \leq d$$

– Step 3: Determine the second digit α_1: Choose the largest $0 \leq \alpha_1 \leq 9$ so that

Table 7 Test examples from Harriot 1631*Praxis*

Name	$p(x) = N$	Solution
Problem 1 (p.117)	$x^2 - 48233025$	6945
Problem 2 (p.119)	$x^2 + 432x = 13584208$	3476
Example 1 (p.121)	$x^2 + 75325x = 41501984$	547
Example 2 (p.122)	$x^2 + 675325x = 369701984$	547
Problem 3 (p.124)	$x^2 - 624x = 16305126$	4362
Example A (p.125)	$x^2 - 6253x = 6254$	6254
Example R (p.127)	$x^2 - 732x = 86005$	835
Problem 4 (p.128)	$-x^2 + 370x = 9261$	27 and 343
Problem 5 (p.131)	$x^3 = 105689636352$	4728
Problem 6 (p.132)	$x^3 + 68x^2 + 4352x = 186394079$	547
Problem 7 (p.134)	$x^3 + 45796x = 449324752$	746
Example (p.138)	$x^3 + 95400x = 1819459$	19
Example (p.139)	$x^3 + 274576x = 301163392$	536
Problem 8 (p.141)	$x^3 - 2648x = 91148512$	452
Example (p.143)	$x^3 - 116620x = 352947$	343
Example (p.145)	$x^3 - 127296x = 85700000$	536
Problem 9 (p.146)	$-x^3 + 52416x = 1244160$	216 and 24
Problem 10 (p.149)	$x^3 - 68x = 134454528$	536
Problem 11 (p.151)	$x^4 = 19565295376$	374
Problem 12 (p.153)	$x^4 - 426x = 2068948$	38
Example (p.154)	$x^4 - 43602354x = 4172008$	352
Problem 13 (p.155)	$x^4 - 1024x^2 + 6254x = 19633735875$	375
Problem 14 (p.158)	$x^4 - 1024x^2 - 6254x = 19629045375$	1375
Problem 15 (p.160)	$x^5 = 15755509298176$	436
Problem 16 (p.162)	$x^5 - 57x^3 + 5263x = 900050558322$	246
Example (p.164)	$x^2 + 14x = 7929$	$82\frac{319}{1000}$
Example (p.166)	$x^3 + 135x = 98754$	$45\frac{24}{100}$

$$(\alpha_2 10^2 + \alpha_1 10)^3 + c(\alpha_2 10^2 + \alpha_1 10) \leq d$$

– Step 4: Determine the last digit α_0: Choose the largest $0 \leq \alpha_0 \leq 9$ so that

$$(\alpha_2 10^2 + \alpha_1 10 + \alpha_0)^3 + c(\alpha_2 10^2 + \alpha_1 10 + \alpha_0) \leq d$$

The convergence of this technique follows from the observation that this is a bracketing process where the root will be in an interval on the form $[\cdot, \cdot)$ (the right end is open) and the assumed existence of a root and monotonicity of $x^3 + cx$ in the interval. The first interval will be $[\alpha_2 10^2, (\alpha_2 + 1)10^2)$, then $[\alpha_2 10^2 + \alpha_1 10,\ \alpha_2 10^2 + (\alpha_1 + 1)10)$ and the final $[\alpha_2 10^2 + \alpha_1 10 + \alpha_0,\ \alpha_2 10^2 + \alpha_1 10 + \alpha_0 + 1)$.

Consider $f(x) = x^3 + cx - d$ and $f(x + h) = 0$ for given $x > 0$ and unknown h. Then

$$f(x + h) = f(x) + h(3x^2 + 3xh + h^2 + c) = 0.$$

Let $h \geq \underline{h} \geq 0$ be an initial estimate, then an upper bound on h will be

$$h \leq \frac{-f(x)}{3x^2 + 3x\underline{h} + c} = \hat{h}, \tag{4}$$

provided c is not too negative. To determine digit number $j > 1$, α_{k-j}, consider

$$x_j = \sum_{i=k-j}^{k-1} 10^i \alpha_i = x_{j-1} + 10^{k-j}\alpha_{k-j}.$$

Define $h_{k-j} = 10^{k-j}$ then from (4)

$$\alpha_{k-j} \leq \left\lfloor \frac{1}{h_{k-j}} \quad \frac{-f(x_{j-1})}{3x_{j-1}^2 + 3x_{j-1}h_{k-j} + c} \right\rfloor. \tag{5}$$

In the following table we show the actual computation of the digits in the solution. We assume that the magnitude of the root and the first digit are known.

	$x^3 + 30x - 14356197$		$x^3 + 95400x - 1819459$
	$x^* = 243$		$x^* = 19$
x	200	240	10
$-f(x)$	6350197	524997	864459
$\underline{h}$	10	1	1
$3x^2 + 3x\underline{h} + c$	126030	173550	95730
$\hat{h}$	50.4	3.03	9.03
α_i	4	3	9

Consider finding root x_* of $x^3 + 30x - 14356197$. The number of digits in x_*, when $(x_*)^3 >> 30x_* > 0$, will be the number of digits in $\sqrt[3]{14\ 356\ 197}$ which is 3 and the leading digit will be $\alpha_2 = 2$. To find the next digit of x_* use (4) with $x = 200$ and $\underline{h} = 10$. An upper bound of $\alpha_1 \le \lfloor 5.04 \rfloor = 5$. However, 5 is too large, and the second digit is found to be 4.

	$x^3 - 10x - 13584$	$x^3 - 116620x - 352947$	
	$x^* = 24$	$x^* = 343$	
x	20	300	340
$-f(x)$	5784	8338947	699747
$\underline{h}$	1	10	1
$3x^2 + 3x\underline{h} + c$	1250	162380	231200
$\hat{h}$	4.63	51.4	3.03
α_i	4	4	3

Newton's transcripts of Viète's solution of $x^3 + 30x = 14356197$ is found in [39, p. 66] and reproduced in [40, p. 534]. The notebook (MS Add. 4000) with transcripts is available online.[5]

7 Contributions of John Wallis 1685

In his algebra and history of algebra book [35] from 1685, John Wallis discusses the work by Viète, Harriot and Oughtred and summarizes the method in one example. In [35, p. 103–105] he gives the example $x^3 - 2x^2 = 186494880$ and computes the root 572 following the same basic principle as in [6, 18, 32] to compute the solution digit by digit. Consider $f(x) = x^3 + bx^2 + d$. Contrary to Viète, Harriot and Oughtred, Wallis does not exclude the h^3 term in (2) and uses

$$\alpha_{k-j} \le \left\lfloor \frac{1}{h_{k-j}} \quad \frac{-f(x_{j-1})}{3x_{j-1}^2 + 2bx_{j-1} + (3x_{j-1} + b)h_{j-k} + h_{k-j}^2} \right\rfloor, \tag{6}$$

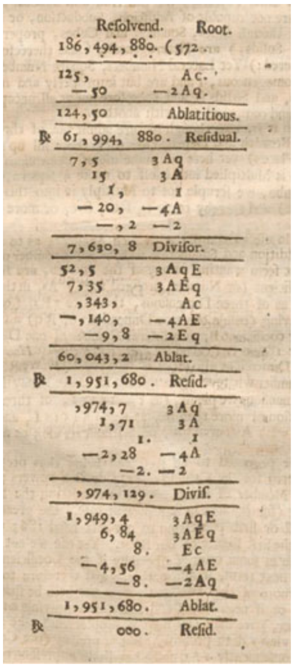
where $h_{k-j} = 10^{k-j}$ as in Sect. 6. Further,

$$f(x+h) = f(x) + (f(x+h) - f(x)) = 3x^2h + 3xh^2 + h^3 + 2bxh + bh^2. \tag{7}$$

Since $500 < \sqrt[3]{186000000} \le \sqrt[3]{186494880 + 2x^2}$ it follows that the sought root has three digits (α_2, α_1, and α_0) and the first digit will be 5. So $\alpha_2 = 5$, $x_1 = 500$, and $h_1 = 10$ and $h_0 = 1$. In Wallis the known is denoted A and E is to be determined which corresponds to x and h properly scaled. In the first column in Fig. 2 is the computation in [35, p. 104] and in the next column the same computation using the notation in this paper.

A problem proposed by Pell and later proposed to Wallis by Colonel Silas Titus is to find a, b, and c so that [25]

[5] https://cudl.lib.cam.ac.uk/view/MS-ADD-04000/1.



```
        Reſolvend.        Root.
    186, 494, 880. (572
    125,              Ac.
     —50            —2Aq.
    124, 50         Ablatitious.
℞  61, 994,  880.   Reſidual.
     7, 5            3Aq
       15            3A
        1,           1
    — 20,          —4A
      —, 2         —2
    7, 630, 8   Diviſor.
    52, 5           3AqE
     7, 35          3AEq
      , 343,          Ac
    —, 140,         —4AE
      —9, 8         —2Eq
    60, 043, 2   Ablat.
℞   1, 951, 680.  Reſid.
      , 974, 7       3Aq
        1, 71        3A
           1.        1
      —2, 28       —4A
          —2.      —2
      , 974, 129.  Diviſ.
    1, 949, 4       3AqE
        6, 84       3AEq
            8.       Ec
      —4, 56        —4AE
          —8.       —2Aq
    1, 951, 680.    Ablat.
℞          000.     Reſid.
```

	186 494 880	$-d$
Ac	125 000 000	x_1^3
-2Aq	-50 000	bx_1^2
Residual	61 994 880	$-f(x_1)$
3Aq	7 500 000	$3x_1^2 \cdot h_1$
3A	150 000	$3x_1h_1 \cdot h_1$
I	1 000	$h_1^2 \cdot h_1$
-4A	-20 000	$2bx_1 \cdot h_1$
-2	-200	$bh_1 \cdot h_1$
Divisor	7 630 800	$\alpha_1 \leq 8, x_2 = 570, h = \alpha_1 h_1 = 70$
3AqE	52 500 000	$3x_1^2h$
3AEq	7 350 000	$3x_1h^2$
Ec	343 000	h^3
-4AE	-140 000	$2bx_1h$
-2Eq	-9 800	$3x_1h^2$
Residual	1 951 680	$-f(x_2)$,
3Aq	974 700	$3x_2^2 \cdot h_0$
3A	1 710	$3x_2h_0 \cdot h_0$
I	1	$h_0^2 \cdot h_0$
-4A	-2 280	$2bx_2 \cdot h_0$
-2	-2	$bh_0 \cdot h_0$
Divisor	974 129	$\alpha_0 \leq 2, x_3 = 572$

Fig. 2 Wallis 1685 $x^3 - 2x^2 = 186494880$

$$a^2 + bc = 16, \quad b^2 + ac = 17, \text{ and } c^2 + ab = 18.$$

Wallis [35, p. 225–252] treats this problem and in [35, Ch. 62] reduces the three equations to a fourth order algebraic equation

$$x^4 - 80x^3 + 1998x^2 - 14937x + 5000 = 0$$

to determine $a = \sqrt{x^*/2}$ using Viète's method. Wallis computes

$$x^* = 12.756441794480744$$

with 17 correct digits. The second equations follows from multiplying the first quadratic equation by a and the second by b and eliminate abc to get the equation

$$17b - b^3 = 16a - a^3, \text{ where } a = \sqrt{\frac{1}{2}x^*}.$$

This equation is solved for b to 16 digits again using Viète's method. Having found a and b, c is found from the first quadratic $a^2 + bc = 16$. The third example of Viète's method is found using synthetic division

$$f(x) = \frac{x^4 - 80x^3 + 1998x^2 - 14937x + 5000}{x - x^*}$$

and finding a second root of the quartic polynomial 0.350987046. In all three examples Wallis is using all terms in the divisor.

If Newton's method is applied to the system $F(a, b, c) = (a^2 + bc - 16, b^2 + ac - 17, c^2 + ab - 18)$ with the starting point $(a, b, c) = (2, 3, 4)$ the error in F is of order 10^{-14} after 5 iterations.

8 End of an Era

In the1670 s John Collins (1625–1683) wrote an account of Pell's achievements for Leibniz, and after describing one of Pell's table (a yard long, according to Collins) and its use, he made the remark that in an attempt to solve the equations with Viète's method, *Mr Warner used to call work unfit for a Christian, and more proper to one that can undertake to remove the Italian Alps into England* [22, Ch. LXXXV, p. 248]. Similar statement from 1758 on Viète's method, Montucla [14, p. 492] regards the calculation of the root of a biquadratic polynomial to eleven decimal places as a work of the most extravagant labour or as Hutton says in 1795 the method is very laborious.

On Wednesday, 17 December 1690, in a meeting of the Royal Society we find the following announcement of Raphson's book [20] (quote from [30])

> Mr Ralpson's Book was this day produced by E Halley, wherein he gives a Notable Improvemt of ye method of Resolution of all sorts of Equations Shewing, how to Extract their Roots by a General Rule, which doubles the known figures of the Root known by each Operation, So yt by repeating 3 or 4 times he finds them true to Numbers of 8 or 10 places. The Society being highly pleased with this his performance Ordered him their thanks with their Desires, that he would please to Continue to prosecute those Studys, wherein he hath been so Successful.

This marks the end of an active area on numerical solution of algebraic equations using digit–by–digit computations. As mentioned in the introduction improved methods appeared, but these methods were soon replaced by the Newton-Raphson method, the Rule of Double False Position or the secant method. However, the digit–by–digit computation of square roots continued to be popular and was used in schools right up to the 1960s [3, 29].

9 Computing the Square Root

Why no one before Viète should have thought of applying to the solution of algebraic equations the classical method of finding roots of large numbers may seem strange [17, p. 24]. In this section we discuss this classical approach to compute square root of any positive integer digit by digit. The history of the method goes back in

Europe to the 13th century with the method of Ibn al–Bannã [3]. Already in 1695 Wallis pointed out that the digit–by–digit computation advocated by Viète, Harriot and Oughtred was not an efficient method [36]. Other iterative methods that are not digit-by-digit based method are based on repeated approximation of the root [28].

Let N be a positive integer and assume that $\sqrt{N}$ is an integer. Assume for $k \geq 1$ that

$$N = \sum_{i=0}^{2k-1} \beta_i \; 10^i = \sum_{i=0}^{k-1} (\beta_{2i} + 10\beta_{2i+1})\, 10^{2i}, \;\; \beta_i \in \{0, 1, 2, \ldots, 9\}$$

where not both $\beta_{2(k-1)}$ and $\beta_{2(k-1)+1}$ are equal 0. The number of digits in $\sqrt{N}$ will then be k, say

$$\sqrt{N} = \sum_{i=0}^{k-1} \alpha_i \; 10^i, \;\; \alpha_i \in \{0, 1, \ldots, 9\}$$

In the following an approximation x_j of $\sqrt{N}$ will be the number with k digits where the j leftmost digits $\alpha_{k-1}, \ldots, \alpha_{k-j}$ are determined and $\alpha_{k-j-1} = \ldots = \alpha_0 = 0$,

$$x_j = \sum_{i=k-j}^{k-1} \alpha_i 10^i = 10^{k-j} \sum_{i=0}^{j-1} \alpha_{i+k-j} \; 10^i, \quad j = 1, 2, \ldots, k$$

and the remaining $k - j$ digits are 0. Let

$$a_j = \sum_{i=0}^{j-1} \alpha_{i+k-j} \; 10^i, \text{ then } x_j = 10^{k-j} a_j, \quad j = 1, \ldots, k-1.$$

Let $d_j = \alpha_{k-j-1} 10^{k-j-1}$ where α_{k-j-1} is the digit to be determined. Since $x_{j+1} = x_j + d_j$ we have $a_{j+1} = 10a_j + \alpha_{k-j-1}$.

To determine α_{k-j-1} choose largest α_{k-j-1} so that

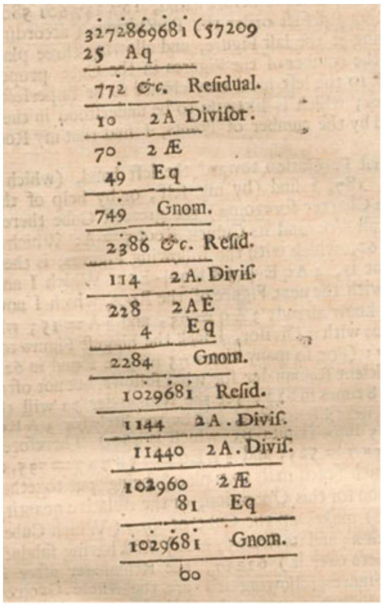
$$(x_j + d_j)^2 \leq N \text{ or } d_j(2x_j + d_j) \leq N - x_j^2$$

Then we have

$$10^{2(k-j-1)}(20a_j + \alpha_{k-j-1})\alpha_{k-j-1} \leq N - x_j^2$$

Now use the assumption that the last $k - j$ digits in x_j are 0. Hence

$$\begin{aligned} N - x_j^2 &= 10^{2(k-j)} r_j + \sum_{i=0}^{2(k-j)-1} \beta_i 10^i \\ &= 10^{2(k-j-1)} \left(10^2 r_j + 10\beta_{2(k-j-1)+1} + \beta_{2(k-j-1)}\right) + \sum_{i=0}^{2(k-j-1)-1} \beta_i 10^i \end{aligned}$$



```
3272869681 (57209
25   Aq
 772 &c.  Reſidual.
 10     2A Diviſor.
 70     2Æ
 49     Eq
 749     Gnom.
 2386 &c. Reſid.
 114    2A. Diviſ.
 228    2AE
   4     Eq
 2284     Gnom.
  1029681   Reſid.
  1144    2A. Diviſ.
  11440   2A. Diviſ.
  102960    2Æ
      81    Eq
  1029681    Gnom.
       00
```

j	α_{k-j}	$\hat{r}_{j-1}$	a_{j-1}	$(20a_{j-1}+\alpha_{k-j})\alpha_{k-j}$	r_j
1	5	32	0	25	7
2	7	772	5	749	23
3	2	2386	57	2284	102
4	0	10296	572	0	10296
5	9	1029681	5720	1029681	0

Fig. 3 Computing the square root of 3272869681 in Wallis 1685 [35, p. 99]

Then α_{k-j-1} is the largest integer so that

$$(20a_j + \alpha_{k-j-1})\alpha_{k-j-1} \le \hat{r}_j,$$

where

$$\hat{r}_j = 10^2\, r_j + 10\beta_{2(k-j-1)+1} + \beta_{2(k-j-1)}.$$

Further, we have

$$r_{j+1} = \hat{r}_j - (20a_j + \alpha_{k-j-1})\alpha_{k-j-1}, \quad j = 1, \dots k-1.$$

To find the largest α_{k-j-1}, Wallis [35, p. 98] chooses $\alpha_{k-j-1} \approx \lfloor \frac{(\hat{r}_j - \beta_{2(k-j-1)})/10}{2a_j} \rfloor$ and increase or decrease if needed. To determine an approximation to the second digit in Fig. 3 this will be $\lfloor \frac{77}{10} \rfloor$ and for the third digit $\lfloor \frac{(2386-6)/10}{57 \cdot 2} \rfloor$.

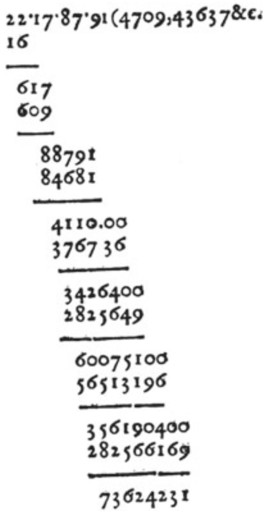
To determine the first digit α_{k-1} we note that

$$d_0^2 \le N, \text{ or } \alpha_{k-1}^2 \le 10\beta_{2k-1} + \beta_{2(k-1)},$$

so the first digit can be easily be determined directly.

Then we have the following digit by digit square root algorithm

$r := 0$



```
22'17'87'91(4709,43637&c.
16
617
609
 88791
 84681
  4110.00
  376736
   3426400
   2825649
    60075100
    56513196
     356190400
     282566169
      73624231
```

j	α_{k-j}	$\hat{r}_{j-1}$	a_{j-1}	$(20a_{j-1}+\alpha_{k-j})\alpha_{k-j}$	r_j
1	4	22	0	16	6
2	7	617	4	609	8
3	0	887	47	0	887
4	9	88791	470	84681	4110
5	4	411000	4709	376736	34264
6	3	3426400	47094	2825649	600751
7	6	60075100	470943	56513196	3561904
8	3	356190400	4709436	282566169	73624231
9	7	7362423100	47094363	6593210869	76921223

Fig. 4 Square root of 22178791 with five decimals in Newton 1707 [16, p. 32]

```
$a := 0$
for $j = 1, 2, \ldots, k$
    $r := 100r + 10\beta_{2(k-j)+1} + \beta_{2(k-j)}$
    Find the largest $\alpha_{k-j}$ so that
            $\alpha_{k-j}(20a + \alpha_{k-j}) \leq r$
    $a := 10a + \alpha_{k-j}$
    $r := r - \alpha_{k-j}(20a + \alpha_{k-j})$
```

We give two examples computing the square root by Wallis in 1685 [35, p. 99] in Fig. 3 and Newton in 1707 [16, p. 32] in Fig. 4.

References

1. Florian Cajori. *William Oughtred, a great seventeenth-century teacher of mathematics*. Chicago Open Court Pub. Co, 1916.
2. Florian Cajori. *A history of mathematical notations, Volume 1. Notations in elementary mathematics*. The Open court publishing company, Chicago, 1928.
3. Jean-Luc Chabert (Ed.). *A History of Algorithms. From the Pebble to the Microchip*. Springer-Verlag, Berlin, 1999.
4. Herman Heine Goldstine. *A History of Numerical Analysis from the 16th through the 19th Century*. Springer-Verlag, New York, 1977.
5. Hermann Hankel. *Geschichte der Mathematik in Alterthum und Mittelalter*. Leipzig, 1874.
6. Thomas Harriot. *Artis Analyticae Praxis*. London, 1631.
7. Theophilus Holdred. *A New Method of Solving Equations with Ease and Expedition; by which the True Value of the Unknown Quantity is Found without Previous Reduction. With a Supple-*

ment, Containing Two Other Methods of Solving Equations, Derived from the Same Principle. London, 1820.

8. William George Horner. A new method of solving numerical equations of all orders, by continuous approximation. *Philosophical Transactions*, 109:308–335, 1819.
9. Charles Hutton. *A mathematical and philosophical dictionary. Volume 1*. London, 1795.
10. Charles Hutton. *Tracts on mathematical and philosophical subjects. Volume 2*. London, 1812.
11. Samuel Jeake. *A Compleat Body of Arithmetic*. London, 1701.
12. Samuel Jeake, Senior. *ΛΟΓΙΣΤΙΚΗΛΟΓΙ'Α*, or Arithmetick surveighed and reviewed: in four books, etc. [Edited by Samuel Jeake, the Younger.]
13. John Kersey. *The Elements of That Mathematical Art Commonly Called Algebra*. London, 1673.
14. Jean Étienne Montucla. *Histoire des mathématiques, Volume I*. Paris, 1758.
15. Augustus De Morgan. Notices of the progress of the problem of evolution. *The companion to the almanac*, pages 34–52, 1839.
16. Isaac Newton. *Arithmetica universalis; Sive de compositione et resolutione arithmetica liber*. London, 1707. Edited by William Whiston.
17. Martin Andrew Nordgaard. *A historical survey of algebraic methods of approximating the roots of numerical higher equations up to the year 1819*. PhD thesis, Columbia University, New York City, 1922.
18. William Oughtred. *Clavis mathematicae*. London, 1631.
19. William Oughtred. *The key of the mathematics new forged and filed*. London, 1647.
20. Joseph Raphson. *Analysis Æquationum Universalis, seu ad Æquationes Algebraicas Resolvendas Methodus Generalis, et Expedita, Ex nova Infinitarum serierum Doctrina, Deducta ac Demonstrata*. London, 1690.
21. Roshdi Rashed. Résolution des équations numérique et algèbre: Šaraf–al–Din al-Tūsī, Viéte. *Archive for History of Exact Sciences*, 12(3):244–290, 1974.
22. Stephen Jordan Rigaud. *Correspondence of Scientific Men of the Seventeenth Century, Vol.1*. Oxford, 1841.
23. Muriel Seltman and Robert Goulding. *Thomas Harriot's Artis Analyticae Praxis. An English Translation with Commentary*. Springer, 2007.
24. Jacqueline Anne Stedall. *The Greate Invention of Algebra: Thomas Harriot's Treatise on equations*. Oxford University Press, Oxford, 2003.
25. Jacqueline Anne Stedall. Tracing mathematical networks in seventeenth-century England. In Eleanor Robson and Jacqueline Stedall, editors, *The Oxford handbook of the history of mathematics*, pages 133–151. Oxford University Press, 2009.
26. Jacqueline Anne Stedall. Reconstructing Thomas Harriot's treatise on equations. In Robert Fox, editor, *Thomas Harriot and His World: Mathematics, Exploration, and Natural Philosophy in Early Modern England*, pages 53–63. Ashgate Publishing (Routledge), 2012.
27. Trond Steihaug. Computational science in the eighteenth century. Test cases for the methods of Newton, Raphson, and Halley: 1685 to 1745. *Numerical Algorithms*, 83:1259–1275, 2020.
28. Trond Steihaug and D. G. Rogers. Approximating cube roots of integers, after Heron's Metrica III.20. *Normat*, 61(2):87–110, 2013.
29. Philip D. Straffin. Liu Hui and the first golden age of Chinese mathematics. *Mathematics Magazine*, 71(3):163–181, 1998.
30. David J. Thomas and Judith M. Smith. Joseph Raphson, F.R.S. *Notes and Records of the Royal Society of London*, 44(2):151–167, 1990.
31. Bartel Leendert van der Waerden. *A History of Algebra. From al-Khwarizmi to Emmy Noether*. Springer, Berlin, 1985.
32. François Viète. *De numerosa potestatum ad exegesim resolutione*. Paris, 1600.
33. François Viète. *Opera mathematica, edited by Frans van Schooten*. Leiden, 1646.
34. François Viète. *The analytic art. Translated by T.Richard Witmer*. The Kent State University Press, 1983.
35. John Wallis. *A treatise of algebra, both historical and practical*. London, 1685.

36. John Wallis. A discourse concerning the methods of approximation in the extraction of surd roots. *Philosophical Transactions (1683-1775)*, 19:2–11, 1695.
37. John Ward. *A Compendium of Algebra*. London, 1695.
38. John Ward. *The Young Mathematician's Guide. Being a Plain and Easie Introduction to the Mathematicks*. London, 1707. Reprinted 1709.
39. Derek Thomas Whiteside. *The Mathematical Papers of Isaac Newton 1664-1666, Volume I*. Cambridge University Press, Cambridge, 1967.
40. Tjalling J. Ypma. Historical development of the Newton-Raphson method. *SIAM Review*, 37(4):531–551, 1995.

NAOV-2020 Conference Participants

Invited Speakers

- Adil M. Bagirov, Federation University, Australia
- Yu-Hong Dai, Chinese Academy of Sciences, China
- Iain Duff, Rutherford Appleton Laboratory, UK
- Francisco Facchinei, Sapienza University of Rome, Italy
- David M. Gay, AMPL Optimization Inc., USA
- Desmond J. Higham, University of Edinburgh, UK
- Michael Hintermüller, Humboldt-Universitat zu Berlin, Germany
- Sven Leyffer, Argonne National Laboratory, USA
- Nezam Mahdavi-Amiri, Sharif University of Technology, Iran
- Pammy Manchanda, Guru Nanak Dev University, India
- Dominique Orban, Ecole Polytechnique de Montreal, Canada
- Michael L. Overton, New York University, USA
- Amiya Kumar Pani, India Institute of Technology Bombay, India
- Jànos D. Pintèr, Lehigh University, Pennsylvania, USA
- Cornelis Roos, Delft University of Technology, The Netherlands
- Ekkehard W. Sachs, University of Trier, Germany
- Jesus M. Sanz-Serna, Universidad Carlos iii de Madrid, Spain
- Michael Saunders, Stanford University, USA

- Abul Hasan Siddiqi, Sharda University, India
- Trond Steihaug, University of Bergen, Norway
- Philippe L. Toint, University of Namur, Belgium
- Maria Teresa Vespucci, Bergamo University, Italy
- Andy Wathen, Oxford University, UK
- Ya-xiang Yuan, Chinese Academy of Sciences, China

Organizing Committee

- Mehiddin Al-Baali (Chair), Sultan Qaboos University, Oman
- Magda Al-Hinai, Sultan Qaboos University and Oman Mathematics Committee, Oman
- Fatma Al-Kharousi, Sultan Qaboos University and Oman Mathematics Committee, Oman
- Nasser Al-Salti, Sultan Qaboos University, Oman
- Ibrahim Dweib, Sultan Qaboos University, Oman
- Easwaran Balakrishnan, Sultan Qaboos University, Oman
- Sebti Kerbal, Sultan Qaboos University, Oman
- Amar Oukil, Sultan Qaboos University, Oman
- Anton Purnama (Co-Chair), Sultan Qaboos University, Oman
- Bernhard Heim, German University of Technology, Oman
- Muhammad Syam, United Arab Emirates University, UAE
- Chefi Triki, HKBU, Qatar
- Issam Moghrabi, Gulf University for Science and Technology, Kuwait
- Lucio Grandinetti, Calabria University, Italy
- Hamed S. Al-Asmi (Superintendent), Sultan Qaboos University, Oman
- Wasila Al-Busaidi (WebMaster), Sultan Qaboos University, Oman

Liaison Committee

- Ahmed Al-Rawas (Chair), Dean of College of Science
- Ahmad Al-Salman, Head of Department of Mathematics
- Mehiddin Al-Baali, Chair of the Organizing Committee
- Anton Purnama, Co-Chair of the Organizing Committee
- Khamis Al Hadhrami, Public Relations Office, Sultan Qaboos University
- Mohammed A. Al-Harthi, Administrative Director of College of Science

International Programme Committee

- Mehiddin Al-Baali, Sultan Qaboos University, Oman
- Adil M. Bagirov, Federation University, Australia
- Oleg Burdakov, Linkoping University, Sweden
- Yu-Hong Dai, Chinese Academy of Sciences, China
- Iain Duff, Rutherford Appleton Laboratory, UK
- Masao Fukushima, Kyoto University, Japan

- David M. Gay, AMPL Optimization, Inc., USA
- Lucio Grandinetti, Calabria University, Italy
- Luigi Grippo, Rome University, Italy
- Nezam Mahdavi-Amiri, Sharif University of Technology, Iran
- Jorge Moré, Argonne National Laboratory, USA
- Dominique Orban, Ecole Polytechnique de Montreal, Canada
- Martin Reed, University of Bath, UK
- Cornelis Roos, Delft University of Technology, The Netherlands
- Ekkehard W. Sachs, University of Trier, Germany, and North Carolina State University, USA
- Michael Saunders, Stanford University, USA
- Robert C. Sharpley, University of South Carolina, USA
- Emilio Spedicato, University of Bergamo, Italy
- Brian Straughan, University of Durham, UK
- Tamas Terlaky, Lehigh University, USA
- Philippe L. Toint, University of Namur, Belgium

Supporting Committee

- Sulaiman Said Al-Alawi
- Mohammed Said Al-Ghafri
- Adnan Aziz Al-Hadhrami
- Majid Mubarak Al-Hamadani
- Mahmoud Mohammed Al-Hashami
- Ahmed Ali Al-Kasbi
- Amal Al-Kharousi
- Aasem Thabit Al-Maamari
- Abdulrahman Ali Al-Muqbali
- Azza Al-Salahi
- Mubarak Musabeh Al-Shekaili
- Afraa Mohammed Al-Shihi
- Nayif Khamis Al-Sinani
- Ahmed Azan Al-Siyabi
- Shamsa Mohamed Al-Wahaibi

Participants

Name	Affiliation	Email
Chetouani Abdelaziz	University of Oujda, Morocco	a.chetouani@ump.ac.ma
Usama Abdelsalam	Rustaq College of Education, Oman	usamaahmad.rus@cas.edu.om

(continued)

(continued)

Name	Affiliation	Email
Najeeb Abdulaleem	University of Lodz , Poland	nabbas985@gmail.com
Rosni Abdullah	Universiti Sains Malaysia, Malaysia	rosni@usm.my
Reda Abu-Elwan	Sultan Qaboos University, Oman	abuelwan@squ.edu.om
Auwal Bala Abubakar	King Mongkut’s University of Technology, Thailand	ababubakar.mth@buk.edu.ng
Noura Hussein Abusalih	Sultan Qaboos University, Oman	nabusalih@squ.edu.om
Fola Adeyeye	Kampala International University, Uganda	adeyeye.john@kiu.ac.ug
Afaq Ahmad	Sultan Qaboos University, Oman	afaq@squ.edu.om
Eqbal Ahmad	Diwan of Royal Court, Oman	
Naveed Ahmed	Gulf University for Science and Technology, Kuwait	ahmed.n@gust.edu.kw
Al-Jalila Al-Abri	Sultan Qaboos University, Oman	aljalila@squ.edu.om
Ibtisam Al-Abri	Sultan Qaboos University, Oman	ialabri@squ.edu.om
Hussain Ali Al-Ajmi	Ministry of Education, Oman	h.ajmi@moe.om
Sulaiman Said Al-Alawi	Sultan Qaboos University, Oman	s45454@student.squ.edu.om
Younis Awadh Al-Alawi	Ministry of Education, Oman	yo.a.m@moe.om
Masood Alam	Sultan Qaboos University, Oman	malam@squ.edu.om
Said Juma Al-Araimi	Ministry of Education, Oman	saidj.aloraimi@moe.om
Hamad Al-Asmi	Sultan Qaboos University, Oman	hasmi@squ.edu.om
Ibrahim Al-Ayyoub	Sultan Qaboos University, Oman	i.alayyoub@squ.edu.om
Nasser Hamood Al-Azri	Ministry of Education, Oman	nasser.azri@moe.om
Khalid Nasser Al-Azri	Ministry of Education, Oman	Khalidazri79@gmail.com
Mehiddin Al-Baali	Sultan Qaboos University, Oman	albaali@squ.edu.om
Amro Al-Baali	McGill University, Canada	amro.al-baali@mail.mcgill.ca

(continued)

(continued)

Name	Affiliation	Email
Slama Said Al-Badri	Ministry of Education, Oman	Slama.badri@moe.om
Khalid Khalifa Al-Bahri	Ministry of Education, Oman	K13k@moe.om
Salim Khalfan Al-Barashdi	Ministry of Education, Oman	Salim.brasde@moe.om
Wasila Al-Busaidi	Sultan Qaboos University, Oman	wasila@squ.edu.om
Mohammed Said Al-Ghafri	Sultan Qaboos University, Oman	s104802@student.squ.edu.om
Sukina Obaid Al-Habsi	Ministry of Education, Oman	Al-HABSI-2011@moe.om
Adnan Aziz Al-Hadhrami	Sultan Qaboos University, Oman	s85292@student.squ.edu.om
Ruqaiya Hamood Al-Hadi	Ministry of Education, Oman	ss.325@moe.om
Saeed Abdullah Al-Hadi	Ministry of Education, Oman	Said22.hadi@moe.om
Majid Mubarak Al-Hamadani	Sultan Qaboos University, Oman	s104806@student.squ.edu.om
Abdullah Musallam Al-Hasani	Ministry of Education, Oman	
Mahmoud M. Al-Hashami	Sultan Qaboos University, Oman	s100111@student.squ.edu.om
Mohammed Mattar Al-Hatmi	Sultan Qaboos University, Oman	m.alhatmi@squ.edu.om
Magda Al-Hinai	Sultan Qaboos University, Oman	magda@squ.edu.om
Ahmed Salim Al-Hrassi	Ministry of Education, Oman	aaa. alharasi@moe.om
Asmaa Salim Al-Hrassi	Ministry of Education, Oman	
Salim Al-Hudaify	Sultan Qaboos University, Oman	salimsaid@squ.edu.om
Mohammed Majid Ali	Ibra College of Technology, Oman	mmali@ict.edu.om
Ishtiaq Ali	Ibra College of Technology, Oman	ishtiaqali@comsats.edu.pk
Yousef Aljarrah	Tafila Technical University, Jordan	yjarrah@ttu.edu.jo
Ahmed Ali Al-Kasbi	Sultan Qaboos University, Oman	s95573@student.squ.edu.om

(continued)

(continued)

Name	Affiliation	Email
Fatma Al-Kharousi	Sultan Qaboos University, Oman	fatma9@squ.edu.om
Safiya Al-Kindi	Sultan Qaboos University, Oman	s116769@student.squ.edu.om
Aasem Thabit Al-Maamari	Sultan Qaboos University, Oman	s34525@student.squ.edu.om
Khaled Al-Mashrafi	Ministry of Education, Oman	khaled2014om@gmail.com
Mariam Al-Maskari	Sultan Qaboos University, Oman	u070036@hotmail.com
Abdulrahman Ali Al-Muqbali	Sultan Qaboos University, Oman	s91616@student.squ.edu.om
Qasim Ali Al-Muqbali	Ministry of Education, Oman	qassim.al-meqbali@moe.om
Fatma Al-Musalhi	Sultan Qaboos University, Oman	fatma@squ.edu.om
Mohamed Khamis Al-Oufi	Ministry of Education, Oman	
Ahmed Al-Qassabi	Sultan Qaboos University, Oman	s98158@student.squ.edu.om
Fatima Al-Raisi	Sultan Qaboos University, Oman	fraisi@andrew.cmu.edu
Mohammed Alrizeiqi	Sultan Qaboos University, Oman	ruzeiki@squ.edu.om
Munira Al-Ruzaqi	Ministry of Education, Oman	ruzaqi@gmail.com
Amal Al-Saidi	Sultan Qaboos University, Oman	Amal2.ALSaeedi@moe.om
Abdul-Sattar Al-Saif	Basrah University, Iraq	sattaralsaif@yahoo.com
Azza Al-Salahi	Sultan Qaboos University Hospital, Oman	azzasalahi@gmail.com
Nasser Al-Salti	Sultan Qaboos University, Oman	nalsalti@squ.edu.om
Mubark Said Al-Salti	Ministry of Education, Oman	Mubarak.alsalti@moe.om
Nasser Salim Al-Shabibi	Ministry of Education, Oman	naser.salem@moe.om
Ebrahim Mansoor Al-Shamsi	Ministry of Education, Oman	abrnet@moe.om
Mubarak M. Al-Shekaili	Sultan Qaboos University, Oman	s29026@student.squ.edu.om

(continued)

(continued)

Name	Affiliation	Email
Afraa Mohammed Al-Shihi	Sultan Qaboos University, Oman	s38933@student.squ.edu.om
Nayif Khamis Al-Sinani	Sultan Qaboos University, Oman	s47590@student.squ.edu.om
Ahmed Azan Al-Siyabi	Sultan Qaboos University, Oman	s45018@student.squ.edu.om
Nisreen Althweib	Ministry of Education, Palestine	thweib@hotmail.com
Talib Mousa Al-Toubi	Ministry of Education, Oman	Talip.altobi@moe.om
Shamsa Mohamed Al-Wahaibi	Sultan Qaboos University, Oman	s35894@student.squ.edu.om
Khalid Alzebdeh	Sultan Qaboos University, Oman	alzebdeh@squ.edu.om
Talal Shaban Amer	Sultan Qaboos University, Oman	talal@squ.edu.om
Gamal Talal Shaban Amer	Sultan Qaboos University, Oman	s117418@student.squ.edu.om
Neculai Andrei	Research Institute for Informatics, Romania	nandrei@ici.ro
Gunarathna W. Arachchilage	Rajarata University, Sri Lanka	gunarathnawa@yahoo.com
Yasir Arfat	King Mongkut's University of Technology, Thailand	yasir.arfat@mail.kmutt.ac.th
Arunkumar Arumugam	Yangzhou University, China	arunapm@yahoo.com
Medhat Awadalla	Sultan Qaboos University, Oman	medhatha@squ.edu.om
Isa Abdullahi Baba	Bayero University, Nigeria	isababa7@yahoo.com
Adil M. Bagirov	Federation University, Australia	a.bagirov@federation.edu.au
Easwaran Balakrishnan	Sultan Qaboos University, Oman	balak@squ.edu.om
Eihab Bashier	Dhofar University, Oman	eihabbashier@gmail.com
Azzeddine Bellour	Ecole Normale Superieure de Constantine, Algeria	bellourazze123@yahoo.com
Mohammad U. Bokhari	Aligarh Muslim University, India	mubokhari@gmail.com
Messaoud Boulbrachene	Sultan Qaboos University, Oman	boulbrac@squ.edu.om
Oleg Burdakov	Linkoping University, Sweden	oleg.burdakov@liu.se

(continued)

(continued)

Name	Affiliation	Email
Pallath Chandran	Sultan Qaboos University, Oman	chandran@squ.edu.om
Maryam Chilan	Shahid Beheshti University of Tehran, Iran	maryamchilan.tmu@gmail.com
Yu-Hong Dai	Chinese Academy of Sciences, China	dyh@lsec.cc.ac.cn
Majid Darehmiraki	Behbahan Khatam Alanbi University of technology, Iran	darehmiraki@gmail.com
Musa Ahmed Demba	King Mongkut's University of Technology, Thailand	musa.demba@mail.kmutt.ac.th
El Amir Djeffal	University of Batna 2, Algeria	l.djeffal@univ-batna2.dz
Iain Duff	STFC RAL, UK and Cerfacs, France	iain.duff@stfc.ac.uk
Ibrahim Dweib	Sultan Qaboos University, Oman	dweib@squ.edu.om
Adel Abd Elaziz El-Sayed	Rustaq College of Education, Oman	aelsayed.rus@cas.edu.om
Ibrahim Eltayeb	University of Nizwa, Oman	ieltayeb@unizwa.edu.om
Francisco Facchinei	Sapienza University, Italy	francisco.facchinei@uniroma1.it
Mohammad Fares	Rustaq College of Education, Oman	mohammad.fares.rus@cas.edu.om
Reza Farzipoor Saen	Sohar University, Oman	rfarzipoorsaen@su.edu.om
Giovanni Fasano	University of Venice Ca Foscari, Italy	fasano@unive.it
Daniele Funaro	University of Modena and Reggio Emilia, Italy	daniele.funaro@unimore.it
Chandrashekhar Gangipelli	Ibra College of Technology, Oman	chandu.724@gmail.com
David M. Gay	AMPL Optimization Inc, USA	dmg@ampl.com
Ahmed Ghaleb	Cairo University, Egypt	afghaleb@sci.cu.edu.eg
Mohammed Amine Ghezzar	Mostaganem University, Algeria	Amine.ghezzar@univ-mosta.dz
Asghar Ghorbani	Ferdowsi University of Mashhad, Iran	aghorbani@um.ac.ir
Gopakumar	Sultan Qaboos University, Oman	g.gopakumar@gmail.com
Lucio Grandinetti	Calabria University, Italy	lucio.grandinetti@unical.it
Francesca Guerriero	University of Calabria, Italy	francesca.guerriero@unical.it

(continued)

(continued)

Name	Affiliation	Email
Sanjiv Gupta	Sultan Qaboos University, Oman	gupta@squ.edu.om
Yousaf Habib	COMSATS University, Pakistan	yhabib@gmail.com
Tayeb Hamaizia	Vente ambulant de vÃªtement, Algeria	el_tayyeb@yahoo.fr
Amira Hamdi	Mohamed Cherif Messaadia University, Algeria	smalika335@gmail.com
Mohammed Hanaki	Faculty of Science, Ibn Tofail University, Morocco	hanaki.mohammed@gmail.com
Bernhard Heim	German Univeristy of Technology, Oman	bernhard.heim@gutech.edu.om
Desmond J. Higham	University of Edinburgh, UK	d.j.higham@ed.ac.uk
Michael Hintermuller	Humboldt-Universitat zu Berlin, Germany	hint@math.hu-berlin.de
Afzal Hussain	Sultan Qaboos University, Oman	afzal19@squ.edu.om
Usman Jatto	University of Ibadan, Nigeria	universityofibandan@gmail.com
Z.A.M.S. Juman	University of Peradeniya, Sri Lanka	jumanabdeen@yahoo.com
Mohammad Kafini	King Fahd University of Petroleum and Minerals, Dahran, Saudi Arabia	mkafini@kfupm.edu.sa
Aref Kamal	Sultan Qaboos University, Oman	akamal@squ.edu.om
Samir Karaa	Sultan Qaboos University, Oman	skaraa@squ.edu.om
Sebti Kerbal	Sultan Qaboos University, Oman	sebti@squ.edu.om
Anees Khadom	University of Diyala, Iraq	aneesdr@gmail.com
Qamar Khan	Sultan Qaboos University, Oman	qjalil@squ.edu.om
Mohammad Khan	Sultan Qaboos University, Oman	mohammad@squ.edu.om
Shamsher Khan	Mazoon College, Oman	shamsher.khan@mazcol.edu.om
Morteza Kimiaei	University of Vienna, Austria	kimiaeim83@univie.ac.at
Mokhtar Kirane	Université de La Rochelle, France	mkirane@univ-lr.fr

(continued)

(continued)

Name	Affiliation	Email
Edamana Krishnan	Sultan Qaboos University, Oman	krish@squ.edu.om
Vijay Kumar Vyas	Sur University College, Oman	vmathsvyas@gmail.com
Arunima Kumari	Bhagwan Parshuram Institute of Technology, India	aru68ram@gmail.com
Kenneth K. Kwikiriza	Sultan Qaboos University, Oman	kkwikk@squ.edu.om
Sven Leyffer	Argonne National Laboratory, USA	leyffer@anl.gov
Ding Ma	City University of Hong Kong, Hong Kong	dingma@cityu.hk.edu
Aliyu Ishaku Ma'ali	Ibrahim Badamasi Babangida University, Nigeria	aaimaali4real@gmail.com
Desmond C. Maduneme	University of Cocody, Nigeria	fransico88@gmail.com
Nezam Mahdavi-Amiri	Sharif University of Technology, Iran	nezamm@sharif.ir
Morteza Maleknia	Amirkabir University of Technology, Iran	maleknia.morteza@gmail.com
Pammy Manchanda	Guru Nanak Dev University, India	pmanch2k1@yahoo.co.in
Jasbir Manhas	Sultan Qaboos University, Oman	manhas@squ.edu.om
Annadurai Manickam Nadar	Ibra College of Technology, Oman	annadurai@ict.edu.om
Khaled Melkemi	University of Batna 2, Algeria	k.melkemi@univ-batna2.dz
Khaled Moaddy	Shaqra University, Saudi Arabia	Moaddy@su.edu.sa
Issam Moghrabi	Gulf University for Science and Technology, Kuwait	moughrabi.i@gust.edu.kw
Dalah Mohamed	Freres Mentouri Constantine University, Algeria	dalah.mohamed@yahoo.fr
Mutaz Mohammad	Zayed University, UAE	Mutaz.Mohammad@zu.ac.ae
Hassan Mohammad	Zayed University, UAE	hmuhd.mth@buk.edu.ng
Ahmed Mohiuddin Mohammed	Sultan Qaboos University, Oman	ahmedmm@squ.edu.om
Touati Brahim Mohammed Said	University of Eloued, Algeria	said-touatibrahim@univ-eloued.dz

(continued)

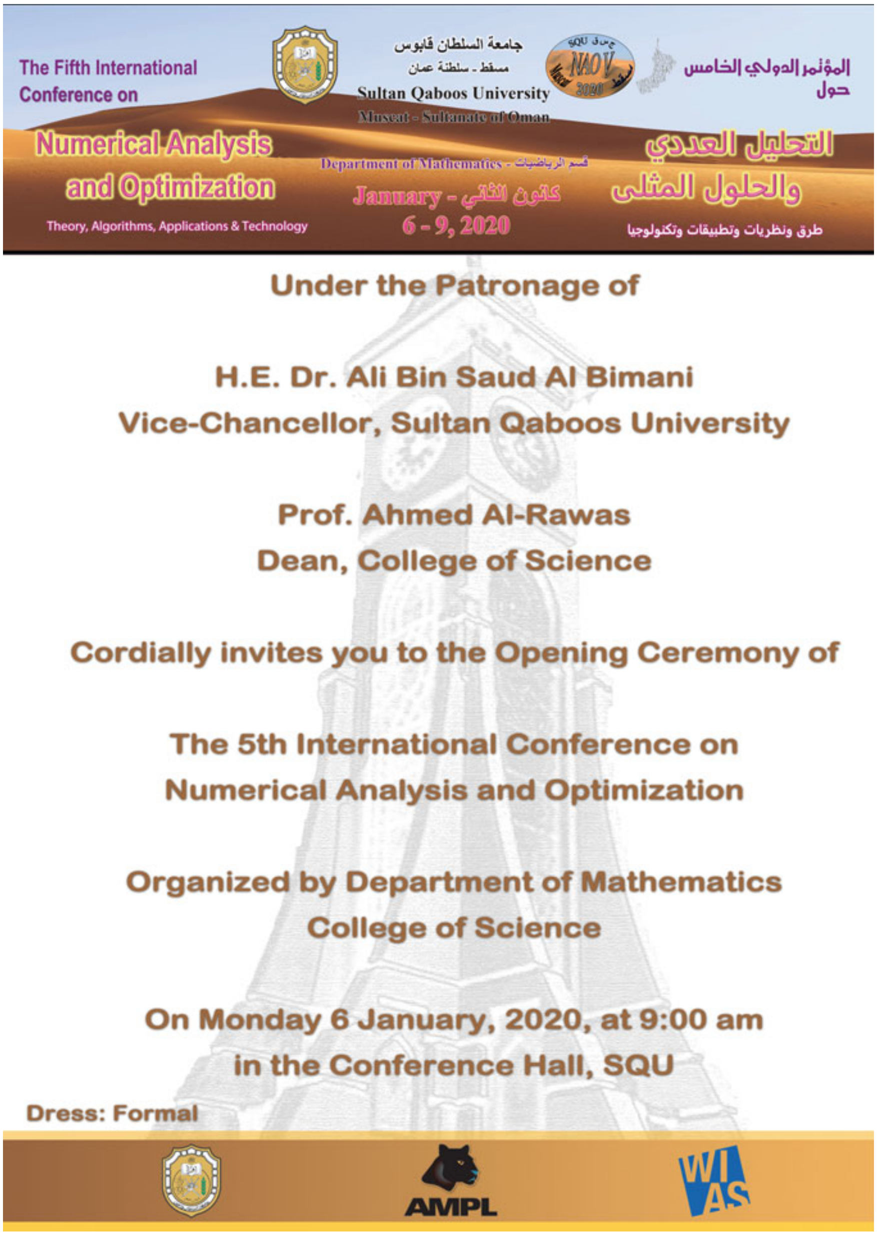
(continued)

Name	Affiliation	Email
Zouhir Mokhtari	University of Biskra, Algeria	z.mokhtari@univ-biskra.dz
Aliyu Awwal Muhammed	King Mongkut's University of Technology, Thailand	aliyumagsu@gmail.com
Haniffa Mohamed Nasir	Sultan Qaboos University, Oman	nasirh@squ.edu.om
Abdesselam Nawel	Laghouat University, Algeria	nawelabedess@gmail.com
Mohamed Yasin Noor Mohamed	Sultan Qaboos University, Oman	mohyasin@squ.edu.om
Dominique Orban	Ecole Polytechnique de Montreal, Canada	dominique.orban@polymtl.ca
Hassen Ouakad	Sultan Qaboos University, Oman	houakad@squ.edu.om
Amar Oukil	Sultan Qaboos University, Oman	aoukil@squ.edu.om
Michael Overton	New York University, USA	mo1@nyu.edu
Danumjaya Palla	BITS Pilani Birla Goa Campus, India	danu@goa.bits-pilani.ac.in
Amiya Kumar Pani	Indian Institute of Technology Bombay, India	akp@math.iitb.ac.in
Ambit Kumar Pany	Siksha O Anusandhan, India	ambit.pany@gmail.com
Janos D. Pinter	Lehigh University, USA	jdp416@lehigh.edu
Anton Purnama	Sultan Qaboos University, Oman	antonp@squ.edu.om
V.P. Ramesh	Central University of Tamil Nadu, India	vpramesh@gmail.com
Cornelis Roos	Delft University of Technology, The Netherlands	c.roos@tudelft.nl
Farouk Saad	Yusuf Maimata Sule University, Nigeria	farouksaaad@yahoo.co.uk
Nirmal Sacheti	Sultan Qaboos University, Oman	nirmal@squ.edu.om
Ekkehard W. Sachs	University of Trier, Germany	sachs@uni-trier.de
Tokuei Sako	Nihon University, Japan	sako.tokuei@nihon-u.ac.jp
Ahmed Salam	University Lille Nord de France, France	salam@univ-littoral.fr
Jesus Maria Sanz-Serna	University Carlos III de Madrid, Spain	jmsanzserna@gmail.com
Michael Saunders	Stanford University, USA	saunders@stanford.edu

(continued)

(continued)

Name	Affiliation	Email
Djamila Seba	Boumerdes University, Algeria	sebadjamila@gmail.com
Syed Muhammed Shavalliuddin	Al-Musanna College of Technology, Oman	syedmdshaval@gmail.com
Abul Hasan Siddiqi	Sharda University, India	Siddiqi.abulhasan@gmail.com
Ahsan Siddiqui	Sultan Qaboos University, Oman	ahsan@squ.edu.om
Md Ashraf Siddiqui	Sultan Qaboos University, Oman	ashrafkfu@gmail.com
Irina Skhomenko	Sultan Qaboos University, Oman	irinashk@squ.edu.om
Ola Hajj Sleiman	American University of Beirut, Lebanon	ola-sleiman@hotmail.com
Dimitri Sotiropoulos	University of the Peloponnese, Greece	dgs@eap.gr
Emilio Spedicato	University of Bergamo, Italy	emilio.spedicato@unibg.it
Trond Steihaug	Unversity of Bergen, Norway	trond.steihaug@ii.uib.no
Bashar Swaid	University of Kalamoon, Syria	bashar.swaid@uok.edu.sy
Muhammed I. Syam	United Arab Emirates University, UAE	m.syam@uaeu.ac.ae
Nasser-eddine Tatar	King Fahd University of Petroleum and Minerals, Saudi Arabia	tatarn@kfupm.edu.sa
Tamas Terlaky	Lehigh University, USA	tat208@Lehigh.edu
Philippe L. Toint	University of Namur, Belgium	philippe.toint@unamur.be
Chefi Triki	Hamad bin Khalifa University, Qatar	ctriki@hbku.edu.qa
Maria Teresa Vespucci	Bergamo University, Italy	maria@unibg.it
Vladimir Vladimirov	Sultan Qaboos University, Oman	vladimir@squ.edu.om
Abdul Wahab	National University of Technology, Pakistan	abdulwahabmalik@gmail.com
Andrew Wathen	Oxford University, UK	wathen@maths.ox.ac.uk
Ya-xiang Yuan	Chinese Academy of Sciences, China	yyx@lsec.cc.ac.cn
Muhammad Ziad	Sultan Qaboos University, Oman	mziad@squ.edu.om
Riadh Zorgati	EDF Lab Paris Saclay, France	riadh.zorgati@edf.fr



The Fifth International Conference on

Numerical Analysis and Optimization

Theory, Algorithms, Applications & Technology

جامعة السلطان قابوس

مسقط - سلطنة عمان

Sultan Qaboos University

Muscat - Sultanate of Oman

Department of Mathematics - قسم الرياضيات

January - كانون الثاني

6 - 9, 2020

المؤتمر الدولي الخامس حول

التحليل العددي والحلول المثلى

طرق ونظريات وتطبيقات وتكنولوجيا

Under the Patronage of

H.E. Dr. Ali Bin Saud Al Bimani
Vice-Chancellor, Sultan Qaboos University

Prof. Ahmed Al-Rawas
Dean, College of Science

Cordially invites you to the Opening Ceremony of

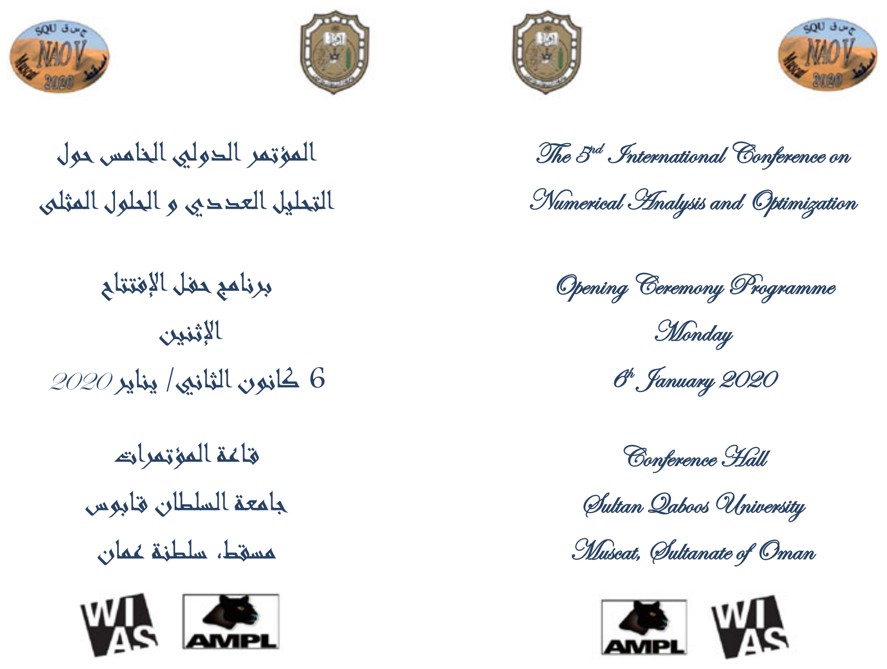
The 5th International Conference on
Numerical Analysis and Optimization

Organized by Department of Mathematics
College of Science

On Monday 6 January, 2020, at 9:00 am
in the Conference Hall, SQU

Dress: Formal

Fig. 1 Invitation to the NAO-V Opening Ceremony



المؤتمر الدولي الخامس حول
التحليل العددي و الحلول المثلى

برنامج حفل الإفتتاح
الإثنين
6 كانون الثاني/ يناير 2020

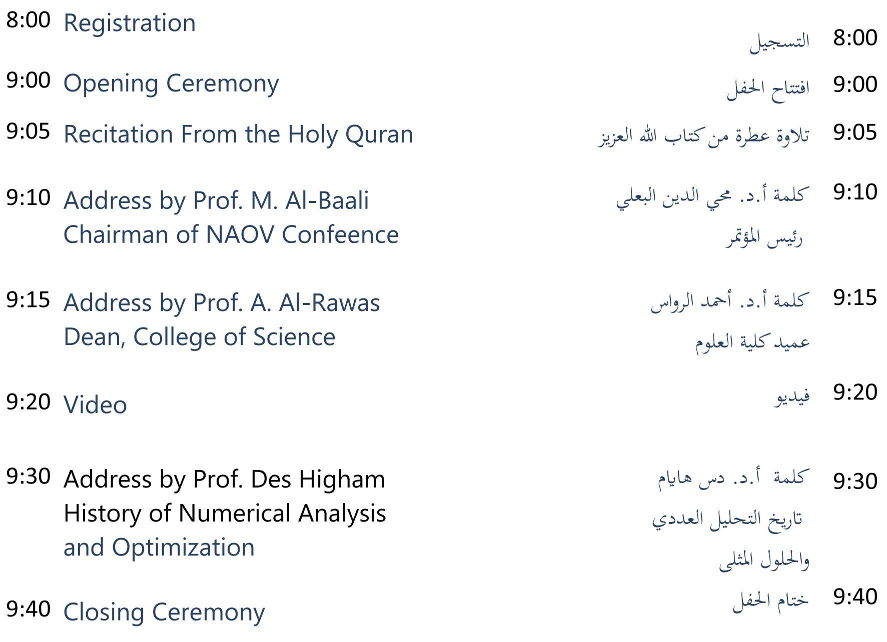
قاعة المؤتمرات
جامعة السلطان قابوس
مسقط، سلطنة عمان

The 5^{nd} International Conference on
Numerical Analysis and Optimization

Opening Ceremony Programme
Monday
6^{th} January 2020

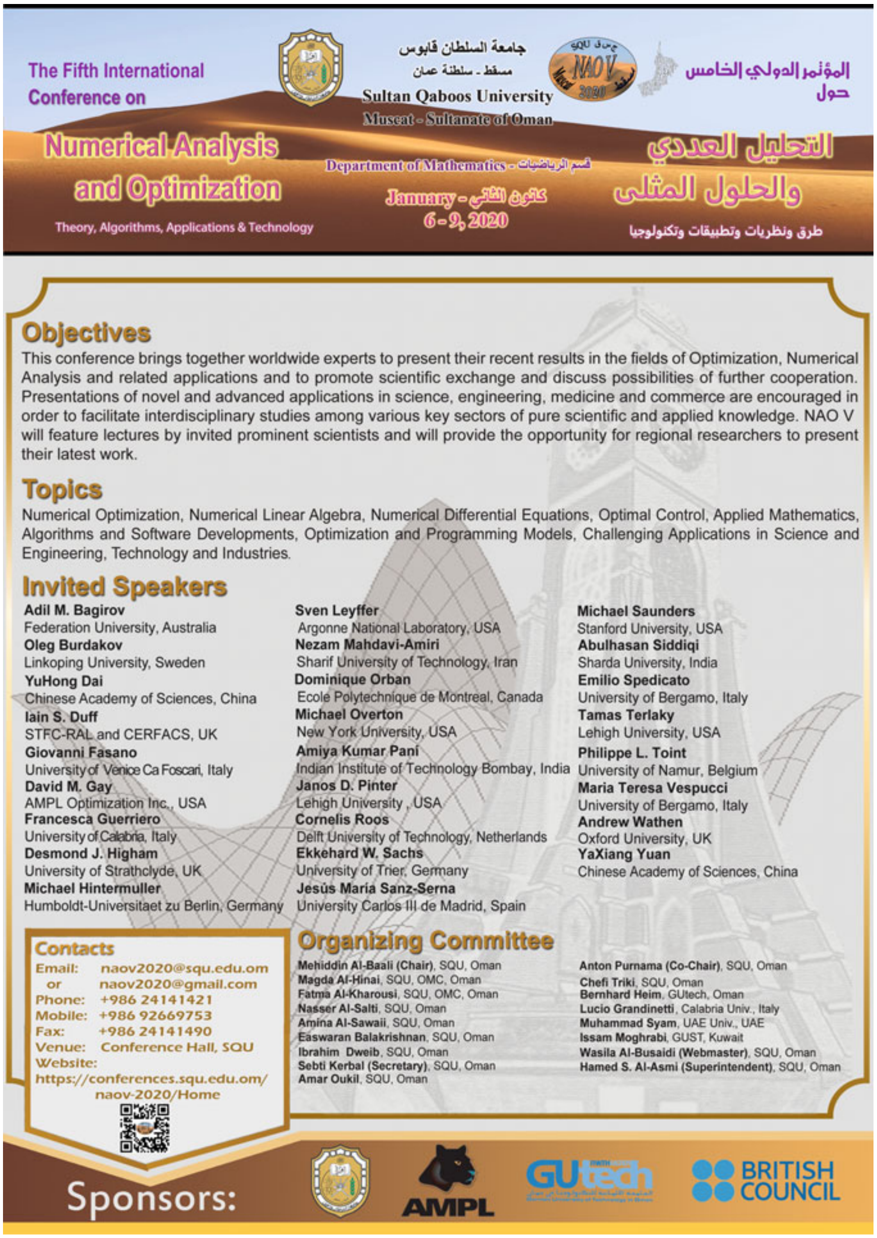
Conference Hall
Sultan Qaboos University
Muscat, Sultanate of Oman

Fig. 2 NAO-V Opening Ceremony Program-Cover



8:00	Registration	التسجيل	8:00
9:00	Opening Ceremony	افتتاح الحفل	9:00
9:05	Recitation From the Holy Quran	تلاوة عطرة من كتاب الله العزيز	9:05
9:10	Address by Prof. M. Al-Baali Chairman of NAOV Confeence	كلمة أ.د. محي الدين البعلي رئيس المؤتمر	9:10
9:15	Address by Prof. A. Al-Rawas Dean, College of Science	كلمة أ.د. أحمد الرواس عميد كلية العلوم	9:15
9:20	Video	فيديو	9:20
9:30	Address by Prof. Des Higham History of Numerical Analysis and Optimization	كلمة أ.د. دس هايام تاريخ التحليل العددي والحلول المثلى	9:30
9:40	Closing Ceremony	ختام الحفل	9:40

Fig. 3 NAO-V Opening Ceremony Program



The Fifth International Conference on

Numerical Analysis and Optimization

Theory, Algorithms, Applications & Technology

جامعة السلطان قابوس
مسقط - سلطنة عمان

Sultan Qaboos University

Muscat - Sultanate of Oman

Department of Mathematics - قسم الرياضيات

January - كانون الثاني
6 - 9, 2020

المؤتمر الدولي الخامس حول

التحليل العددي والحلول المثلى

طرق ونظريات وتطبيقات وتكنولوجيا

Objectives

This conference brings together worldwide experts to present their recent results in the fields of Optimization, Numerical Analysis and related applications and to promote scientific exchange and discuss possibilities of further cooperation. Presentations of novel and advanced applications in science, engineering, medicine and commerce are encouraged in order to facilitate interdisciplinary studies among various key sectors of pure scientific and applied knowledge. NAO V will feature lectures by invited prominent scientists and will provide the opportunity for regional researchers to present their latest work.

Topics

Numerical Optimization, Numerical Linear Algebra, Numerical Differential Equations, Optimal Control, Applied Mathematics, Algorithms and Software Developments, Optimization and Programming Models, Challenging Applications in Science and Engineering, Technology and Industries.

Invited Speakers

Adil M. Bagirov
Federation University, Australia
Oleg Burdakov
Linkoping University, Sweden
YuHong Dai
Chinese Academy of Sciences, China
Iain S. Duff
STFC-RAL and CERFACS, UK
Giovanni Fasano
University of Venice Ca Foscari, Italy
David M. Gay
AMPL Optimization Inc., USA
Francesca Guerriero
University of Calabria, Italy
Desmond J. Higham
University of Strathclyde, UK
Michael Hintermuller
Humboldt-Universitaet zu Berlin, Germany
Sven Leyffer
Argonne National Laboratory, USA
Nezam Mahdavi-Amiri
Sharif University of Technology, Iran
Dominique Orban
Ecole Polytechnique de Montreal, Canada
Michael Overton
New York University, USA
Amiya Kumar Pani
Indian Institute of Technology Bombay, India
Janos D. Pinter
Lehigh University , USA
Cornelis Roos
Delft University of Technology, Netherlands
Ekkehard W. Sachs
University of Trier, Germany
Jesús María Sanz-Serna
University Carlos III de Madrid, Spain
Michael Saunders
Stanford University, USA
Abulhasan Siddiqi
Sharda University, India
Emilio Spedicato
University of Bergamo, Italy
Tamas Terlaky
Lehigh University, USA
Philippe L. Toint
University of Namur, Belgium
Maria Teresa Vespucci
University of Bergamo, Italy
Andrew Wathen
Oxford University, UK
YaXiang Yuan
Chinese Academy of Sciences, China

Contacts

Email:	naov2020@squ.edu.om
or	naov2020@gmail.com
Phone:	+986 24141421
Mobile:	+986 92669753
Fax:	+986 24141490
Venue:	Conference Hall, SQU

Website:
https://conferences.squ.edu.om/naov-2020/Home

Organizing Committee

Mehiddin Al-Baali (Chair), SQU, Oman
Magda Al-Hinai, SQU, OMC, Oman
Fatma Al-Kharousi, SQU, OMC, Oman
Nasser Al-Salti, SQU, Oman
Amina Al-Sawaii, SQU, Oman
Easwaran Balakrishnan, SQU, Oman
Ibrahim Dweib, SQU, Oman
Sebti Kerbal (Secretary), SQU, Oman
Amar Oukil, SQU, Oman
Anton Purnama (Co-Chair), SQU, Oman
Chefi Triki, SQU, Oman
Bernhard Heim, GUtech, Oman
Lucio Grandinetti, Calabria Univ., Italy
Muhammad Syam, UAE Univ., UAE
Issam Moghrabi, GUST, Kuwait
Wasila Al-Busaidi (Webmaster), SQU, Oman
Hamed S. Al-Asmi (Superintendent), SQU, Oman

Sponsors:

AMPL

GUtech

BRITISH COUNCIL

Fig. 4 NAO-V Conference Poster

SQU ج س ق
NAO V
Muscat مسقط
2020
المؤتمر الدولي الخامس حول
التحليل العددي والحلول المثلى
The Fifth International Conference on
Numerical Analysis and Optimization
Programme
January - كانون الثاني
6 - 9, 2020
Sponsors:
AMPL
WIAS

Fig. 5 NAO-V Program

Monday, January 6

Time	Conference Hall
8:00 - 9:00	Registration
9:00 - 10:00	Opening Ceremony

10:00 - 10:20 - Coffee / Tea Break

10:30 - Photo Session

Time	Lecture Theater 1 **Chair**: Sven Leyffer
10:40 - 11:20	Philippe Toint
11:20 - 12:00	Jànos D. Pintèr
12:00 - 12:40	Cornelis Roos

12:40 - 14:00 - Lunch Break

Time	Lecture Theater 1 **Chair**: Ekkehard Sachs
14:00 - 14:40	Michael Saunders
14:40 - 15:20	Ya-xiang Yuan
15:20 - 16:00	Michael Hintermüller

16:00 - 16:20 - Coffee / Tea Break

Time	Lecture Theater 2 **Chair**: Nezam Mahdavi-Amiri	Lecture Theater 3 **Chair**: Abul Hasan Siddiqi
16:20 - 17:00	Adil Bagirov	Pammy Manchanda
17:00 - 17:20	Ding Ma	Daniele Funaro
17:20 - 17:40	Mariam Al-Maskari	Ibtisam Al-Abri

Fig. 5 (continued)

Tuesday, January 7

Time	Lecture Theater 1 Chair: Jesus Sanz-Serna
9:00 - 9:40	Iain Duff
9:40 - 10:20	Francisco Facchinei
10:20 - 11:00	Sven Leyffer

11:00 - 11:20 - Coffee / Tea Break

Time	Lecture Theater 2 Chair: Jànos Pintèr	Lecture Theater 3 Chair: Philippe Toint
11:20 - 12:00	David Gay	Trond Steihaug

Time	Lecture Theater 1 Chair: Pammy Manchanda	Lecture Theater 2 Chair: Jànos D. Pintèr	Lecture Theater 3 Chair: Philippe Toint
12:00 - 12:20	Zouhir Mokhtari	Usama Abdelsalam	Riadh Zorgati
12:20 - 12:40	Ahmed Salam	Morteza Maleknia	Muhammed Syam
12:40 - 13:00	Najeeb Abdulaleem	Musa Demba	Rosni Abdullah

13:00 - 14:00 - Lunch Break

Time	Lecture Theater 1 Chair: Cornelis Roos
14:00 - 14:40	Michael Overton
14:40 - 15:20	Ekkehard Sachs
15:20 - 16:00	Dominique Orban

16:00 - 16:20 - Coffee / Tea Break

Time	Lecture Theater 1 Chair: Maria Teresa Vespucci	Lecture Theater 2 Chair: Amiya Kumar Pani	Lecture Theater 3 Chair: Andy Wathen
16:20 - 16:40	Issam Moghrabi	W. A. Gunarathna	V. P. Ramesh
16:40 - 17:00	Ahmed Al-Siyabi	Khaled Al-Mashrafi	Palla Danumjaya
17:00 - 17:20	Amal Al-Saidi	Adel El-Sayed	Azzeddine Bellour
17:20 - 17:40	Auwal Bala Abubakar	Ahmed Al-Kasbi	Salim Al-Hudaify

Conference Dinner (buses leave at 18:15 near faculty club)

Fig. 5 (continued)

Wednesday, January 8

Time	Lecture Theater 1 Chair: Iain Duff
9:00 - 9:40	Desmond Higham
9:40 - 10:20	Andy Wathen
10:20 - 11:00	Jesus Sanz-Serna

11:00 - 11:20 - Coffee / Tea Break

Time	Lecture Theater 2 Chair: Adil Bagirov	Lecture Theater 3 Chair: Francisco Facchinei
11:20 - 12:00	Nezam Mahdavi-Amiri	Maria Teresa Vespucci

Time	Lecture Theater 1 Chair: Issam Moghrabi	Lecture Theater 2 Chair: Adil Bagirov	Lecture Theater 3 Chair: Francisco Facchinei
12:00 - 12:20	Nasser-eddine Tatar	Hassan Mohammad	Afaq Ahmad
12:20 - 12:40	Abdul Wahab	Morteza Kimiaei	Medhat Awadalla
12:40 - 13:00	Abeer Al-Siyabi	Aliyu Awwal	Hassen M. Ouakad

13:00 - 14:00 - Lunch Break

Time	Lecture Theater 1 Chair: Ya-xiang Yuan
14:00 - 14:40	Yu-Hong Dai

Time	Lecture Theater 2 Chair: Trond Steihaug	Lecture Theater 3 Chair: Dominique Orban
14:40 - 15:20	Amiya Kumar Pani	Abul Hasan Siddiqi

Time	Lecture Theater 1 Chair: Muhammed Syam	Lecture Theater 2 Chair: Trond Steihaug	Lecture Theater 3 Chair: Dominique Orban
15:20 - 15:40	Naveed Ahmed	Amar Oukil	Khaled Melkemi
15:40 - 16:00	Abdul-Sattar Al-Saif	Reza Farzipoor Saen	Afzal Husain
16:00 - 16:20	Bashir Zargar	Fatima Al-Raisi	Mohammad Bokhari

16:20 - 16:30 - Coffee / Tea Break

Time	Lecture Theater 1
16:30	Closing Ceremony

Fig. 5 (continued)

Thursday, January 9

8:30 -	Muscat city sightseeing and excursions to the interior of Oman

Fig. 5 (continued)

Invited Speaker **Title**	**Abstract Page**
1. **Adil Bagirov** - Federation University Australia, Australia	
Piecewise linear regression: A nonsmooth optimization approach	
2. **Yu-Hong Dai** - Chinese Academy of Sciences, China	
On the complexity of sequentially lifting cover inequalities for the Knapsack polytope	
3. **Iain Duff** - STFC RAL, UK and Cerfacs, France	
Using shared and distributed memory in the solution of large sparse systems	
4. **Francisco Facchinei** - Sapienza University of Rome, Italy	
Ghost penalties in nonconvex constrained optimization: Diminishing stepsizes and iteration complexity	
5. **David M. Gay** - AMPL Optimization Inc., USA	
Functions in AMPL	
6. **Desmond J. Higham** - University of Edinburgh, UK	
A nonlinear spectral method for network core-periphery detection	
7. **Michael Hintermüller** - Humboldt-Universitt zu Berlin, Germany	
Functional-analytic and numerical issues in splitting methods for total variation-based image reconstruction	
8. **Sven Leyffer** - Argonne National Laboratory, USA	
Mixed-integer PDE-constrained optimization	
9. **Nezam Mahdavi-Amiri** - Sharif University of Technology, Iran	
A new inexact nonmonotone filter SQP algorithm	
10. **Pammy Manchanda** - Guru Nanak Dev University, India	
Wavelet Methods in Data Mining	
11. **Dominique Orban** - GERAD and Ecole Polytechnique, Canada	
Implementing a smooth exact penalty function for nonlinear optimization	
12. **Michael L. Overton** - New York University, USA	
Behavior of L-BFGS on nonsmooth optimization problems in theory and practice	
13. **Amiya Kumar Pani** - IIT Bombay, India	
Global stabilization of viscous Burgers equation by nonlinear Neumann boundary feedback control: theory and finite element analysis	
14. **Jànos D. Pintèr** - Lehigh University, Pennsylvania, USA	
Largest small n-polygons: Numerical results and estimated optima for all even values n	
15. **Cornelis Roos** - Delft University of Technology, Netherlands	
On some 'easy' optimization problems	
16. **Ekkehard Sachs** - Trier University, Germany	
Second order adjoints in optimization	
17. **Jesus M. Sanz-Serna** - Universidad Carlos iii de Madrid, Spain	
The Hamiltonian Monte Carlo method and geometric integration	
18. **Michael Saunders** - Stanford University, USA	
Computing Hamiltonian cycles in random graphs	
19. **Abul Hasan Siddiqi** - Sharda University, India	
General conformable fractional derivatives with applications	

Fig. 5 (continued)

Invited Speaker **Title**	**Abstract** **Page**
20. **Trond Steihaug** - University of Bergen, Norway	
Computational science in the late 1600s up to the mid-1700s: From digit–by–digit computation to second order rate of convergence	
21. **Philippe Toint** - University of Namur, Belgium	
Recent results in worst-case evaluation complexity for smooth and non-smooth, exact and inexact, nonconvex optimization	
22. **Maria Teresa Vespucci** - University of Bergamo, Italy	
Power generation and transmission expansion planning with high shares of renewables: A two-stage stochastic programming approach	
23. **Andy Wathen** - Oxford University, UK	
Preconditioning for nonsymmetric Toeplitz matrices with application to time-dependent partial differential equations	
24. **Ya-xiang Yuan** - Chinese Academy of Sciences, China	
Efficient optimization algorithms for large-scale data analysis	

Contributed Speaker **Title**	**Abstract** **Page**
1. **Afaq Ahmad** - Sultan Qaboos University, Oman	
Algorithmic computation of mean time between failures (MTBF) of a solar array	
2. **Usama M. Abdelsalam** - Rustaq College of Education, Oman	
Travelling wave solutions for Fisher's equation using extended homogeneous balance method	
3. **Najeeb Abdulaleem** - Hadhramout University, Yemen	
Sufficiency and duality for E-differentiable multiobjective programming problems involving generalized V-E-invex functions	
4. **Rosni Abdullah** - Universiti Sains Malaysia, Malaysia	
A parallel island-based genetic algorithm for inferring metabolic network	
5. **Auwal Bala Abubakar** - King Mongkut's University of Technology Thonburi, Thailand	
New three-term conjugate gradient algorithm for solving monotone nonlinear equations and signal recovery problems	
6. **Naveed Ahmed** - Gulf University for Science and Technology, Kuwait	
An assessment of two classes of variational multiscale methods for the simulation of incompressible turbulent flows	
7. **Ibtisam Al-Abri** - Sultan Qaboos University, Oman	
Evaluating wildfire management policies using stochastic dynamic model and numerical optimization	
8. **Salim S. Al-Hudaify** - Sultan Qaboos University, Oman	
Du-Fort Frankel and Lax-Wendroff schemes for one dimensional transport equation	
9. **Ahmed Al-Kasbi** - Sultan Qaboos University, Oman	
Spreading of brine effluents into spring-neap tidal currents: modelling and optimum conditions	

Fig. 5 (continued)

Contributed Speaker **Title**	**Abstract** **Page**
10. **Khaled S. M. Al-Mashrafi** - Ministry of Education, Oman	
The mathematical model of the dynamics of bounded cartesian plumes	
11. **Mariam Al-Maskari** - Sultan Qaboos University, Oman	
FEM for nonlinear subdiffusion equations with a local Lipschitz condition	
12. **Fatima Al-Raisi** - Sultan Qaboos University, Oman	
Regularization for DAG learning	
13. **Amal A.M. Al-Saidi** - Sultan Qaboos University, Oman	
Computational studies of conjugate gradient methods for large-scale unconstrained optimization	
14. **Abdul-Sattar J. Al-Saif** - Basrah University, Iraq	
Analytical approximate solutions of $2D$ incompressible Navier-Stokes equations using Adomian decomposition method	
15. **Abeer Al-Siyabi** - Sultan Qaboos University, Oman	
Quasi crystallography originating from Voronoi and Delone cells of the root lattice A_n	
16. **Ahmed Al-Siyabi** - Sultan Qaboos University, Oman	
Numerical experiments with new basic hessian approximations for large-scale nonlinear least-squares optimization	
17. **Medhat Awadalla** - Sultan Qaboos University, Oman	
An evolutionary approach for data exchange optimization in parallel architectures	
18. **Aliyu Awwal** - King Mongkut's University of Technology Thonburi, Thailand	
A modified Perry algorithm for solving convex constrained nonlinear monotone equations and minimizing ℓ_1 regularized problem	
19. **Azzeddine Bellour** - Ecole Normale Superieure de Constantine, Algeria	
Numerical solution of high-order linear Volterra integro-differential equations by using Taylor collocation method	
20. **Mohammad U. Bokhari** - Aligarh Muslim University, India	
Objective measures to evaluate retrieval effectiveness of news search systems for parameters of relevance and freshness	
21. **Palla Danumjaya** - BITS-Pilani K K Birla Goa Campus, India	
Mixed virtual element methods for the fourth order nonlinear parabolic problems	
22. **Musa Demba** - King Mongkut's University of Technology Thonburi, Thailand	
A high order optimized explicit Runge-Kutta-Nyström method for solving periodic problems	
23. **Adel Abd Elaziz El-Sayed** - Rustaq College of Education, Oman	
Numerical solution of the fractional order advection-dispersion equation via shifted Vieta-Fibonacci polynomials	
24. **Daniele Funaro** - Universit'a di Modena e Reggio Emilia, Italy	
Electrodynamical ring resonators	
25. **W. A. Gunarathna** - Rajarata University of Sri Lanka, Sri Lanka	
A unified explicit form for difference approximations of fractional and integral order derivatives	
26. **Afzal Husain** - Sultan Qaboos University, Oman	
Numerical analysis and performance optimization of a hybrid microchannel and jet impingement heat sink	

Fig. 5 (continued)

Contributed Speaker Title	Abstract Page
27. **Morteza Kimiaei** - Universität Wien, Austria	
A limited memory method for unconstrained nonlinear least squares	
28. **Ding Ma** - City University of Hong Kong, Hong Kong	
Optimal income taxation with multidimensional taxpayer types	
29. **Morteza Maleknia** - Amirkabir University of Technology, Iran	
A gradient sampling method based on ideal direction for solving nonsmooth nonconvex optimization problems	
30. **Khaled Melkemi** - University of Batna 2, Algeria	
Chebyshev approximation, non stationary wavelets and applications	
31. **Issam A.R. Moghrabi** - Gulf University for Science and Technology, Kuwait	
A new non-Secant quasi-Newton method	
32. **Hassan Mohammad** - Bayero University, Nigeria	
A multivariate spectral method for nonlinear monotone equations with applications to sparse recovery	
33. **Zouhir Mokhtari** - University of Biskra, Algeria	
An adapted integration method for volterra integral equation of the second kind with weakly singular kernel	
34. **Hassen M. Ouakad** - Sultan Qaboos University, Oman	
Efficient numerical approaches for the nonlinear nonlocal behavior of electrically actuated carbon nanotube resonators	
35. **Amar Oukil** - Sultan Qaboos University, Oman	
A robust ranking approach based on the value system embedded under DEA cross-efficiency	
36. **V. P. Ramesh** - Central University of Tamil Nadu, India	
A priori mesh refinement strategy for singularly perturbed turning point problem	
37. **Reza Farzipoor Saen** - Sohar University, Oman	
How to assess sustainability of supply chains by fuzzy Malmquist productivity index?	
38. **Ahmed Salam** - University Lille Nord de France, Calais, France	
On theoretical and numerical aspects of symplectic reductions of a matrix to upper J-Hessenberg form	
39. **Muhammed I. Syam** - UAE University, United Arab Emirates	
Optimization of one step block methods hybrid points for solving first-order ordinary differential equations	
40. **Nasser-eddine Tatar** - KFUPM, Saudi Arabia	
The main challenges in studying fractional viscoelastic problems	
41. **Abdul Wahab** - Natio **??**	
A model based joint sparsity approach for inverse elastic medium scattering	
42. **Bashir Ahmad Zargar** - University of Kashmir, India	
Bernstein type inequalities for polynomials	
43. **Riadh Zorgati** - EDF Lab Paris-Saclay, France	
Solving ill-conditioned linear systems with a preconditionning stochastic matrice-based approach	

Fig. 5 (continued)

The Fifth International Conference on Numerical Analysis and Optimization
January
6-9, 2020

Fig. 6 NAO-V Chairman, Mehiddin Al-Baali. Photo Courtesy of Photography Department, Sultan Qaboos University

Fig. 7 NAO-V Co-Chairman, Anton Purnama. Photo Courtesy of Photography Department, Sultan Qaboos University

Fig. 8 NAO-V Opening Ceremony. Photo Courtesy of Photography Department, Sultan Qaboos University

Fig. 9 NAO-V Opening Ceremony. Photo Courtesy of Photography Department, Sultan Qaboos University

Fig. 10 NAO-V Participants. Photo Courtesy of Photography Department, Sultan Qaboos University