Structural and Multidisciplinary Optimization (2020) 61:343–352
https://doi.org/10.1007/s00158-019-02363-y

RESEARCH PAPER

A separable augmented Lagrangian algorithm for optimal structural design

Kemal M. Palanduz[1] · Albert A. Groenwold[1]

Received: 23 November 2018 / Revised: 10 July 2019 / Accepted: 15 July 2019 / Published online: 23 August 2019

Abstract
We propose an iterative separable augmented Lagrangian algorithm (SALA) for optimal structural design, with SALA being a subset of the alternating directions of multiplier method (ADMM)–type algorithms. Our algorithm solves a sequence of separable quadratic-like programs, able to capture reciprocal- and exponential-like behavior, which is desirable in structural optimization. A salient feature of the algorithm is that the primal and dual variable updates are all updated using closed-form expressions. Since algorithms in the ADMM class are known to be very sensitive to scaling, we propose a scaling method inspired by the well-known ALGENCAN algorithm. Comparative results for SALA, ALGENCAN, and the Galahad LSQP solver are presented for selected test problems. Finally, although we do not exploit this feature herein, the primal and dual updates are both embarrassingly parallel, which makes the algorithm suitable for implementation on massively parallel computational devices, including general purpose graphical processor units (GPGPUs).

Keywords Structural optimization · Separable augmented Lagrangian algorithm (SALA) · Alternating directions of multiplier method (ADMM) · Separable quadratic program (QP) · Scaling

1 Introduction

In this paper, we consider an equality constrained nonlinear optimization problem $\mathcal{P}_{\text{NLP}}$ of the form

$$\begin{aligned} &\min_{\boldsymbol{x}} && f(\boldsymbol{x}) \\ &\text{subject to} && g_j(\boldsymbol{x}) = 0, && j = 1, 2, \dots, m, \\ &&& l_i \le x_i \le u_i, && i = 1, 2, \dots, n, \end{aligned} \qquad (1)$$

where $f(\boldsymbol{x})$ is a real-valued scalar objective function and the $g_j(\boldsymbol{x})$, $j = 1, 2, \cdots, m$ are m equality constraint functions, which depend on the n real (design) variables $\boldsymbol{x} = \{x_1, x_2, \cdots, x_n\}^T \in \mathcal{X} \subset \mathcal{R}^n$, with l_i and u_i, respectively, the lower and upper bounds on variable x_i. The functions $f(\boldsymbol{x})$ and $g_j(\boldsymbol{x})$ are assumed to be (at least) once continuously differentiable.

Although many or possibly even most interesting problems in structural optimization are normally formulated using inequality constraints only,[1] it is convenient to herein use only equality constraint functions, for reasons that will become clear in sections to follow. Accordingly, we will assume that any inequality constraints present may be reformulated as equality constraints, with the aid of so-called slack variables s_j. In this approach, an inequality constraint $g_j(\boldsymbol{x}) \le 0$ is rewritten as

$$g_j(\boldsymbol{x}) + s_j = 0,$$

subject to $s_j \ge 0$. Similarly, a separable inequality constraint can be rewritten as

$$g_{ji}(x_i) + s_{ji} = 0,$$

subject to $s_{ji} \ge 0$. This simple technique is not only well known but also often used, even in successful commercial codes, e.g., see Nocedal and Wright (2006) and Conn et al. (1992). For the sake of brevity, we will in the following not

Responsible Editor: Michael Kokkolaras

✉ Albert A. Groenwold
albertg@sun.ac.za

[1] Department of Mechanical and Mechatronic Engineering, University of Stellenbosch, Stellenbosch, South Africa

[1] A notable exception being the so-called simulated analysis and design (SAND) methodology.

even mention the presence of the slack variables s_j and s_{ji} for inequality constraints; their use is implied.

Arguably, the state-of-the-art in structural optimization is to use a sequential approximate optimization (SAO) algorithm to solve problem $\mathcal{P}_{\text{NLP}}$. SAO relies on the iterative solution of a sequence of approximate optimization problems $\mathcal{P}_{\text{P}}[k]$, $k = 0, 1, 2, \cdots$. In turn, the approximate optimization problems, or subproblems, are normally based on approximation functions $\tilde{g}_j(\boldsymbol{x})$ which have a relatively *simple structure*, albeit that they may all be nonlinear. Then, at some iterate $\boldsymbol{x}^k$, we obtain continuous primal approximate subproblem $\mathcal{P}_{\text{P}}[k]$, written as

$$\begin{aligned} &\min_{\boldsymbol{x}} && \tilde{f}(\boldsymbol{x}) && \\ &\text{subject to} && \tilde{g}_j(\boldsymbol{x}) = 0, && j = 1, 2, \ldots, m, \\ & && l_i \le x_i \le u_i, && i = 1, 2, \ldots, n. \end{aligned} \tag{2}$$

This primal problem contains n unknowns, m equality constraints, and $2n$ side or bound constraints (not counting any slack or relaxation variables that may have been introduced). The side constraints are normally handled in an efficient way which does not require additional computational effort.

In SAO, a most notable feature of primal approximate subproblem $\mathcal{P}_{\text{P}}[k]$ is that the approximation functions $\tilde{f}(\boldsymbol{x})$ and $\tilde{g}_j(\boldsymbol{x})$ are *separable*. Possibly surprising at first, this is routinely done since the evaluation and storage of second-order information in structural optimization is considered prohibitively expensive on the computational devices available to us today. Instead, so-called intermediate or intervening variables are relied upon which, when substituted into a linear Taylor series expansion, reveal behavior that is representative of the underlying physics of nonlinear optimization problem $\mathcal{P}_{\text{NLP}}$ (assuming that the physics is understood in the first place). The approximations then become linear in the intervening variables used.

In structural optimization for example, the reciprocal intermediate variables $z_i = x_i^{-1}$ are important and often used, since they capture the inverse relationship between stress and area well. Invariably, the intermediate variables used themselves are separable; when substituted into a linear or first-order Taylor series expansion, this of course in turn results in separable approximations and hence separable approximate subproblems $\mathcal{P}_{\text{P}}[k]$.

Examples of popular algorithms that use separable approximations include the convex linearization algorithm (CONLIN) of Fleury and Braibant (1986) and its generalization, the method of moving asymptotes (MMA) of Svanberg (1987b, 1995). Groenwold and Etman (2011) have proposed the use of separable diagonal quadratic approximations, which rely on the so-called "approximated approximations" approach to capture reciprocal-like behavior (Groenwold et al. 2010). In this approach, the quadratic approximation to the reciprocal approximation itself is constructed, or even the quadratic approximation to the CONLIN and MMA approximations already mentioned.

Given the dominance of separable approximations, it is convenient to rewrite subproblems (2) as

$$\begin{aligned} &\min_{\boldsymbol{x}} && \sum_{i=1}^{n} \tilde{f}_i(x_i) && \\ &\text{subject to} && \sum_{i=1}^{n} \tilde{g}_{ji}(x_i) = 0, && j = 1, 2, \ldots, m, \\ & && l_i \le x_i \le u_i, && i = 1, 2, \ldots, n. \end{aligned} \tag{3}$$

It is prudent to here mention that the SAO subproblems used in structural optimization are often solved in the dual space, which is problem-free from a theoretical point of view, if the approximations are *convex* and separable (although solution of the subproblems may still be demanding). The use of pure dual methods is particularly popular when inequality constraints only are present.

Of course, primal separability does not imply that the associated *dual problem* often favored in structural optimization will be separable; in general, this is indeed not the case. Hence, in pure dual methods, we have the disappointing situation that even though the primal approximations are all separable, the maximization over the dual variables is not separable. (Notwithstanding, the primal-dual relationships may of course benefit from separability if the approximations are simple enough.)

From a loss of separability point of view, augmented Lagrangian (AL) methods or sequential unconstrained minimization techniques (SUMT) do not help. Consider for example the popular augmented Lagrangian statement due to Rockafellar (1973):

$$\tilde{L}_\rho(\boldsymbol{x}, \boldsymbol{\mu}) = \tilde{f}(\boldsymbol{x}) + \sum_{j=1}^{m} \mu_j \tilde{g}_j(\boldsymbol{x}) + \frac{\rho}{2} \sum_{j=1}^{m} \left(\tilde{g}_j(\boldsymbol{x})\right)^2, \tag{4}$$

where μ_j represent the Lagrangian multipliers, $\tilde{g}_j(\boldsymbol{x})$ the separable constraint approximation functions, and ρ a nonzero (positive) penalty parameter (it is often actually common practice to use m penalty parameters ρ_j). For subproblem k, the optimal duo $(\boldsymbol{x}^{k*}, \boldsymbol{\mu}^{k*})$ is found by iteratively minimizing (4) w.r.t. $\boldsymbol{x}$ with ρ^k and $\boldsymbol{\mu}^k$ fixed, then updating the multipliers $\boldsymbol{\mu}$ and the penalty parameter ρ—the latter often conditionally—until convergence occurs on the subproblem level. The multipliers are updated using the famous Hestenes–Powell formula, e.g., see Hestenes (1969) and Powell (1969). The minimization of (4) w.r.t. $\boldsymbol{x}$ may be done using any suitable minimizer that is able to accommodate the bound constraints $\boldsymbol{l}$, $\boldsymbol{u}$. Unfortunately, a disappointing if obvious feature of (4) is that the separable nature of primal approximate subproblem $\mathcal{P}_{\text{P}}[k]$ is not preserved, due to the effects of the squared terms. Penalty-based SUMT suffer from the same drawback.

Enter the so-called *separable* augmented Lagrangian algorithm, or SALA, popularized by Hamdi et al. (1997), Hamdi and Mahey (2000), Hamdi (2005a, b, c), Boyd et al. (2010), and many others. The SALA framework preserves the separable nature of the subproblems, and the minimizations over the primal variables x_i result in n uncoupled searches, with the obvious feature that this is an *embarrassingly parallel* operation, while the m dual variable updates are also uncoupled. What is more, the separable nature begs the question whether it is possible to find the *primal minimizers in closed-form*. For subproblems that are simple enough, this is indeed the case; this includes the important class of separable quadratic programs (QPs).

Although the SALA paradigm has to the author's knowledge not been applied in structural optimization, they are quite general and are receiving some attention in the mathematical programming community. A SALA may be considered to be an extension of proximal decomposition methods and derive from the class of splitting algorithms of Douglas and Rachford (1956) and Lions and Mercier (1984); they are often known as *alternating directions*–type methods. An immediate word of warning though: although the SALA framework seems very attractive given the combination of uncoupled updates of both the primal and dual variables with the dominance of separable approximations in optimal structural design, algorithm SALA suffers from sensitivity to a subproblem scaling parameter. To address this, we will herein use an update strategy for this parameter inspired by recent efforts of Lenoir and Mahey (2007) and Boyd et al. (2010), combined with a function and constraint scaling strategy inspired by the ALGENCAN solver (Birgin and Martínez 2007).

What is more, to take full benefit from the separable nature of algorithm SALA, suitable hardware like general purpose graphical processor units (GPGPUs) should be exploited, but we will not do so herein. Without this option, it is not clear if algorithm SALA will be competitive with the classical dual methods currently so popular in optimal structural design. Nevertheless, application of algorithm SALA to problems in optimal structural design is interesting in its own right and, in addition seems to have educational value. Our paper makes a few salient contributions: Algorithm SALA is free from any line search, and the primal and dual updates are embarrassingly parallel. Indeed, the primal and dual variable updates are available in closed-form. The importance of the algorithm using closed-form primal and dual updates cannot be overstated in the parallel context, where search methods can be difficult to implement on a massively parallel scale. Furthermore, the algorithm can easily exploit the use of intervening variables, which are popular and proven in structural optimization.

Our paper is arranged as follows: In Section 2, we present some diagonal quadratic approximations that are necessary for the separability of SALA. In Section 3, we outline the alternating directions–type method that we rely upon. We again emphasize that we rely on (strictly) convex approximations in doing so. We then proceed with numerical experiments in Section 4, where we compare SALA with the well-known ALGENCAN (Birgin and Martínez 2007) and the Gould et al. (2003) LSQP solvers, followed by conclusions and recommendations in Section 5.

2 Some diagonal quadratic approximations

The approximate subproblems used are based on an incomplete series expansion (ISE) (Groenwold and Etman 2008). We construct approximations $\tilde{f}(\boldsymbol{x})^k$ to the objective function $f_0(\boldsymbol{x})$ and all the constraint functions $f_j(\boldsymbol{x})$ at the point x^k, such that

$$\tilde{f}_j^k(\boldsymbol{x}) = f_j^k + \nabla^T f_j^k \boldsymbol{s} + \frac{1}{2}\boldsymbol{s}^T \boldsymbol{C}_j^k \boldsymbol{s}, \tag{5}$$

with $\boldsymbol{s} = (\boldsymbol{x} - \boldsymbol{x}^k)$ and each $\boldsymbol{C}_j^k$ an appropriate approximate *diagonal* Hessian matrix, for $j = 0, 1, 2, \ldots, m$. We will occasionally use the abbreviated notation

$$f_j^k = f_j(\boldsymbol{x}^k).$$

For the sake of clarity, we rewrite (5) using summation convention as

$$\tilde{f}_j^k(\boldsymbol{x}) = f_j^k + \sum_{i=1}^n \left(\frac{\partial f_j}{\partial x_i}\right)^k (x_i - x_i^k) + \frac{1}{2}\sum_{i=1}^n c_{2i_j}^k (x_i - x_i^k)^2, \tag{6}$$

with $c_{2i_j}^k$ approximate second-order diagonal Hessian terms or curvatures. We will herein consider two very simple instances of (6), in which the the curvatures are chosen as follows:

1. Such that the approximate function value at the previous iterate $\tilde{f}^{k-1}$ is equal to the real function value f^{k-1} at the previous iterate, being a spherical quadratic approximation (denoted SPH-QDR)
2. Such that the approximation becomes the quadratic approximation to the reciprocal approximation (denoted T2:R), being closely related to the very popular MMA (Svanberg 1987a) approximations

While many other possibilities exist, we only outline the above approximations in Appendix B. For the sake of brevity, the reader is referred to Groenwold et al. (2010)

and the references therein for details about some other possibilities.

Since the approximations (6) are (diagonal) quadratic, primal problem $\mathcal{P}_{\mathrm{NLP}}$ may trivially be transformed into a sequence of quadratic approximate programs $\mathcal{P}_{\mathrm{PQ}}[k]$, written as

- *Quadratic approximate program* $\mathcal{P}_{\mathrm{PQ}}[k]$

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \bar{f}_0^k(\boldsymbol{x}) = f_0^k + \nabla^T f_0^k \boldsymbol{s} + \frac{1}{2}\boldsymbol{s}^T \boldsymbol{Q}^k \boldsymbol{s} \\ \text{subject to} \quad & \bar{f}_j^k(\boldsymbol{x}) = f_j^k + \nabla^T f_j^k \boldsymbol{s} = 0, && j = 1, 2, \ldots, m, \\ & \check{x}_i \le x_i \le \hat{x}_i, && i = 1, 2, \ldots, n, \end{aligned} \tag{7}$$

with $\boldsymbol{s} = (\boldsymbol{x} - \boldsymbol{x}^k)$ and $\boldsymbol{Q}^k$ the Hessian matrix of the approximate Lagrangian $\mathcal{L}^k$. For details, the reader is referred to Etman et al. (2009, 2012). Since the Lagrangian multipliers $\boldsymbol{\mu}^{k*}$ and $\boldsymbol{\lambda}^{k*}$ at the solution of subproblem $\mathcal{P}_{\mathrm{PQ}}[k]$ are unknown, the multipliers $\boldsymbol{\mu}^k$ and $\boldsymbol{\lambda}^k$ are used to construct the quadratic program. Hence,

$$Q_{ii}^k = c_{2i_0}^k + \sum_{j\in\mathcal{E}} \mu_j^k c_{2i_j}^k + \sum_{j\in\mathcal{I}} \lambda_j^k c_{2i_j}^k, \tag{8}$$

and $Q_{id}^k = 0 \,\forall\, i \ne d$, i and $d = 1, 2, \ldots, n$. For clarity, both the equality and inequality terms are given in (8), since we do not use slack variables for the inequalities in the Gould et al. (2003) solver LSQP, since this would put this specific solver at a disadvantage.

We require the approximate Lagrangian $\mathcal{L}^k$ to be (semi) positive definite. Since $\boldsymbol{Q}^k$ is diagonal, positive definiteness simply requires the individual diagonal elements Q_{ii}^k to be positive. As μ_j^k, c_{2i_0} and c_{2i_j} in principle are unrestricted in sign, we will enforce

$$Q_{ii}^k = \max(\epsilon > 0, Q_{ii}^k), \tag{9}$$

with ϵ prescribed and "small."

3 Alternating directions–type methods

We proceed with a brief outline of the salient features of algorithm SALA; for details, the reader is referred to the cited literature in the mathematical programming community.[2] Here, we follow closely the presentation of Lenoir and Mahey (2007). In essence, algorithm SALA reformulates separable problem (3) with the help of the allocation subspace

$$\mathcal{A} = \left\{ y_{ji} \in \mathcal{R}^{mn} / \sum_{i=1}^{n} y_{ji} = 0,\ j = 1, 2, \ldots, m \right\}. \tag{10}$$

Then, the allocation of resource vectors $y_{ji} = g_{ji}(x_i)$, for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, m$, results in an embedded formulation of problem (3) with a distributed coupling, written as

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \sum_{i=1}^{n} f_i(x_i) \\ \text{subject to} \quad & y_{ji} = g_{ji}(x_i), \\ & y_{ji} \in \mathcal{A}, \\ & l_i \le x_i \le u_i, \end{aligned} \tag{11}$$

for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, m$.

The *approximate* augmented Lagrangian $\mathcal{L}_\rho$ with penalty parameter $\rho > 0$ obtained by associating the multipliers μ_{ji} with the allocation constraints $y_{ji} = \tilde{g}_{ji}(x_i)$ decomposes into the sum

$$\tilde{\mathcal{L}}_\rho(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\mu}) = \sum_{i=1}^{n} \tilde{\mathcal{L}}_{\rho,i}(x_i, y_{ji}, \mu_{ji}), \tag{12}$$

with

$$\begin{aligned} \tilde{\mathcal{L}}_{\rho,i}(x_i, y_{ji}, \mu_{ji}) \ = \ & \tilde{f}_i(x_i) + \sum_{j=1}^{m} \mu_{ji}(\tilde{g}_{ji}(x_i) - y_{ji}) \\ & + \frac{\rho}{2}\sum_{j=1}^{m} (\tilde{g}_{ji}(x_i) - y_{ji})^2. \end{aligned} \tag{13}$$

The stationary point of $\tilde{\mathcal{L}}_{\rho,i}(x_i, y_{ji}, \mu_{ji})$ is obtained via successive minimizations over x_i and y_{ji} in a Gauss–Seidel fashion as to exploit the separability of (11). The minimizations in the x_i yield the n independent subproblems

$$\min_{x_i} \tilde{\mathcal{L}}_{\rho,i}(x_i, y_{ji}^k, \mu_{ji}^k),$$

which can be done in parallel. Since the μ_{ji} are in $\mathcal{A}^\perp$, with $\mathcal{A}$ and $\mathcal{A}^\perp$ mutually orthogonal, we obtain the updates for the y_{ji} as

$$y_{ji}^{l+1} = \tilde{g}_{ji}(x_i^{l+1}) - \frac{1}{n}\left(\tilde{g}_j(x_i^{l+1})\right), \tag{14}$$

again see Lenoir and Mahey, and Boyd for details and proofs.

Subspace $\mathcal{A}^\perp$ has the explicit formulation

$$\mathcal{A}^\perp = \left\{ \mu_{ji} \in \mathcal{R}^{mn} / \mu_{j1} = \mu_{j2} = \ldots = \mu_{jn},\ j = 1, 2, \ldots, m \right\}. \tag{15}$$

[2] An interesting recent application of ADMM in structural optimization may be found in Kanno and Kitayama (2018).

So, at every iteration l, knowledge of the μ_{ji} reduces to the knowledge of its common component $\nu_j = \mu_{j1} = \mu_{j2} = \ldots = \mu_{jn}$, $j = 1, 2, \ldots, m$, and the update step for the ν_j becomes

$$\nu_j^{l+1} = \nu_j^l + \frac{\rho}{n}\left(\tilde{g}_j(x_i^{l+1})\right), \tag{16}$$

again see the cited literature for details. The complete resulting algorithm SALA, without the trivial objective function and constraint scaling given in (21), is listed in Algorithm 1. Again note that not only are the n uncoupled minimizations over the x_i embarrassingly parallel, updating the y_{ji} and ν_j is also embarrassingly parallel.

We impose a subproblem convergence criteria on both the step sizes made by the primal and dual variables, as well as the maximum number of subproblem evaluations.

Algorithm 1 Algorithm SALA for a *given* equality constrained *sub*problem k.

```
1  Initialize: $l = 1, \epsilon > 0, \rho^0 > 0, y_{ji}^0 \in \mathcal{A}, \nu_j^0 = \in \mathcal{A}^\perp$
2  repeat
3      for $i = 1, n$ do
4          $x_i^{l+1} \leftarrow \arg\min_{x_i} \tilde{\mathcal{L}}_{\rho,i}(x_i, y_{ji}^l, \nu_j^l)$
5      end
6      for $j = 1, 2, \ldots, m$ and $i = 1, 2, \ldots, n$ do
7          update $y_{ji}^{l+1}$ using (14)
8      end
9      for $j = 1, 2, \ldots, m$ do
10         update $\nu_j^{l+1}$ using (16)
11     end
12     update $\rho^{l+1}$ using (26)
13     $l \leftarrow l + 1$
14 until $\|\boldsymbol{\nu}^{l+1} - \boldsymbol{\nu}^l\|$ and $\|\boldsymbol{x}^{l+1} - \boldsymbol{x}^l\| \le \epsilon$ or $l \le l_{max}$
15 end
```

3.1 Closed-form expressions for QP-like problems

If the functions g_{ji} are simple enough, the n one-dimensional minimizations over the x_i may even be done in closed-form. For a separable QP-like subproblem, this is indeed the case. Let

$$\tilde{f}(x_i) = a + b_i(x_i - x_i^k) + \frac{1}{2}c_i(x_i - x_i^k)^2, \tag{17}$$

with $\boldsymbol{b}$ and $\boldsymbol{c}$ given n-vectors, and

$$\tilde{g}_j(x_i) = d_{ji} + e_{ji}(x_i - x_i^k), \tag{18}$$

again with $\boldsymbol{d}_j$ and $\boldsymbol{e}_j$ given n-vectors. In order to satisfy (3), we have chosen to distribute the j constraint values equally among each of their n separable functions such that

$$d_{ji} = g_j(\boldsymbol{x}^k)/n. \tag{19}$$

We have now resorted to summation convention—a sum over repeated indices in a term is implied, i.e., we sum over i in (17) and (18) above. Here, the c_i represent curvature information of the objective and the constraint functions (see Etman et al. (2009, 2012)), and we assume *strictly convex* primal approximate subproblems $\mathcal{P}_\text{P}[k]$. Then, the stationary conditions of (13) w.r.t. x_i result in

$$x_i^{l+1} = x_i^k - \left(c_i + \rho e_{ji}^2\right)^{-1}\left(b_i + \nu_j e_{ji} - \rho y_{ji} e_{ji} + \rho d_{ji} e_{ji}\right), \tag{20}$$

and we sum over j. Note that the denominator cannot vanish, since $c_i > 0$ and $\rho > 0$ (although $e_{ji} = 0$ is possible in sparse problems). Line 4 in Algorithm 1 may thus be replaced by the very simple closed-form expression (20). The relation between (6), (17), and (18) is clear; also see Etman et al. (2009, 2012).

As mentioned, suitable separable approximations are briefly presented in Section 2. For the sake of clarity, we here again mention that we use a spherical quadratic approximation (denoted SPH-QDR) and a quadratic approximation to the reciprocal approximation (denoted T2:R), being closely related to the very popular MMA (Svanberg 1987a) approximations. For some examples which are truly separable, we also use exact Hessian information.

3.2 Objective and constraint function scaling

The SALA algorithm is highly sensitive to the scaling in the objective function, constraints, and subproblem. We therefore scale both the objective and constraint functions, such that

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & w_f f(\boldsymbol{x}) \\ \text{subject to} \quad & w_{c_j} g_j(\boldsymbol{x}) = 0, \quad j = 1, 2, \ldots, m, \\ & l_i \le x_i \le u_i, \quad i = 1, 2, \ldots, n, \end{aligned} \tag{21}$$

where the scaling factors, w_f and w_{c_j}, are given by

$$\begin{aligned} w_f &= 1/\max(\|\nabla f(\boldsymbol{x}^{k-1})\|_\infty, 1), \\ w_{c_j} &= 1/\max(\|\nabla c_j(\boldsymbol{x}^{k-1})\|_\infty, 1), \qquad j = 1, 2, \ldots, m, \end{aligned}$$

with $\boldsymbol{x}^k$ denoting the primal values for subproblem k.

3.3 Subproblem scaling

The projected subgradient step given in (16) depends on the step length ρ, with this parameter behaving more like a scaling parameter in the SALA framework than the penalty parameter in classical augmented Lagrangian statements. For example, ρ penalizes the primal coupling constraints, and greater values will accelerate the primal sequence. However, ρ^{-1} penalizes the dual sequence so that a compromise value is expected to be optimal, e.g., see Lenoir and Mahey (2007) and Boyd et al. (2010), who elaborate on optimal scaling strategies in some detail. It is possible to use ρ_i for $i = 1, 2, \ldots, n$ scaling parameters for each of the separable subproblems; but this only leads to increased computational costs (Lenoir and Mahey 2007). Furthermore, if the objective and constraint functions are scaled, the n separable problems have similar orders of magnitude, which further reduces the need for individual separable subproblem scaling parameters. The single parameter update is therefore the preferred choice of subproblem scaling for our numerical testing.

We obtained reasonable numerical results when using a combination of the strategies inspired by Lenoir and Mahey (2007) and Boyd et al. (2010).

From Lenoir and Mahey (2007) , let

$$\beta = \frac{\|\boldsymbol{v}^l - \boldsymbol{v}^{l-1}\|}{\|\boldsymbol{y}^l - \boldsymbol{y}^{l-1}\|}. \tag{22}$$

Then, we update using

$$\rho \leftarrow \rho^{(1-\alpha)} + \beta^{\alpha}, \tag{23}$$

or

$$\rho \leftarrow (1-\alpha)\rho + \alpha\beta, \tag{24}$$

with α small, say 10^{-2} , to prevent oscillatory behavior. We accept that Lenoir and Mahey (2007) prove that

$$\sum_{l=1}^{\infty} \alpha_l = S < \infty \tag{25}$$

is a requirement for theoretical convergence, but for numerical purposes $l \ll \infty$, resulting in small values of α still achieving practical problem convergence.

We now introduce the update strategy inspired by Boyd et al. (2010), coupled with that from (24). The update then becomes

$$\rho \leftarrow \begin{cases} \sqrt{n}\rho & \text{if } \|\boldsymbol{x}^l - \boldsymbol{x}^{l-1}\| > n\|\boldsymbol{v}^l - \boldsymbol{v}^{l-1}\| \\ \rho/\sqrt{n} & \text{if } \|\boldsymbol{v}^l - \boldsymbol{v}^{l-1}\| > n\|\boldsymbol{x}^l - \boldsymbol{x}^{l-1}\| \\ (23) \text{ or } (24) & \text{otherwise.} \end{cases} \tag{26}$$

The update strategy proposed by (23) and (24) takes advantage of second-order problem information (Lenoir and Mahey 2007), which in our numerical testing led to higher levels of accuracy compared with a strategy that purely focused on keeping the primal and dual residual norms within a factor of each other. The downside, however, is the possibility of ρ quickly jumping to unsuitably high values should the resource allocation vector and dual residual norms be of different orders of magnitude. Furthermore, it is difficult to reduce the value of the scaling parameter because of the low values of α that have to be chosen in order to ensure convergence. We combine the two different strategies in the hope of ensuring relatively equal convergence of both the primal and dual sequences, with the advantage of greater levels of accuracy.

4 Numerical experiments

We use the QP form given in (6) in Section 2 to construct an approximate augmented Lagrangian for both the ALGENCAN and SALA solvers. For SALA, the augmented Lagrangian is then decomposed further into the n separable Lagrangian problems given in (13). We reiterate that the approximations used for the Hessian are either the spherical diagonal quadratic approximation (SPH-QDR) or the quadratic approximation to the reciprocal approximation (T2:R). We also compare our algorithms with the state-of-the-art Gould et al. (2003) solver LSQP. The latter solver is Taylor-made for linear or *diagonal* quadratic subproblems.

All numerical results for SALA use the closed-form update of the primal variables given in (20). The step length ρ is updated using (24) and (26) with $\alpha = 10^{-2}$. The maximum number of subproblem evaluations allowed is $l_{\max} = 5 \times 10^4$ with a subproblem convergence tolerance of $\epsilon = 10^{-6}$. For each subproblem k, we initialize at $l = 0$ as follows: for values of $k > 0$, $\rho^0 = 1$, $y_{ji}^0 = y_{ji}^{k-1}$, and $v_j^0 = v_j^{k-1}$. Else, at $k = 0$, $\boldsymbol{y} = \boldsymbol{v} = \boldsymbol{0}$. A maximum of $k = 500$ outer iterations was allowed, after which the algorithm was stopped, and we indicate failure to converge by "—" in the numerical results to follow. The absolute values of the Lagrangian multipliers were bounded to not exceed 10^6, selected rather arbitrarily, to prevent numerical instabilities.

Default settings are used for the ALGENCAN solver, except that we use *epsfeas* = *epsopt* = 10^{-7}, where *epsfeas* and *epsopt* are parameters in ALGENCAN. For Galahad's LSQP solver, default settings were used. We now introduce the absolute maximum constraint violation h, the norm of the KKT conditions $\mathcal{K}$, and the number of function evaluations N_f required for termination. For all the solvers, problem execution was terminated when all of three conditions were met, namely, the Euclidean or 2-norm $\|\boldsymbol{x}^k - \boldsymbol{x}^{k-1}\|_2 \leq 10^{-4}$, $h \leq 10^{-4}$, and $\mathcal{K} \leq 10^{-3}$. We chose convergence tolerances of modest accuracy, as ADMM is usually slow to converge to high accuracy (Boyd et al. 2010). The test problems used are tabulated in Table 1, and we present the results in Table 2.

The results are not overly surprising; sometimes ALGENCAN performs slightly better than SALA, and vice versa. This happens notwithstanding the fact that the two algorithms solve a sequence of identical subproblems, reminiscent of the no-free-lunch (NFL) theorems of optimization (Wolpert and Macready 1997). Arguably, the main reason for this is in all probability that *both* algorithms are known to be sensitive to scaling, albeit that augmented

Lagrangian methods are probably less so than ADMM-type methods. Arguably, Galahad's LSQP solver sets the tone. (Note that $\mathcal{K}$ is smaller for LSQP than for the other solvers; notwithstanding that the same stopping criteria were used.)

The results for Svanberg's snake problem are worthy of special attention: for this problem, none of the algorithms converged. We have on purpose included these results to highlight some limitations of the methods studied. It is pertinent to point out that this problem is very "difficult." The snake problem consists of a very thin feasible slither in n-dimensional space. It is also of note to mention that arguably the most popular algorithms in structural optimization, namely, CONLIN and MMA, also have difficulties solving some of the test problems. CONLIN for example fails when applied to the snake problem, whereas MMA fails for Fleury's weight minimization problem, all in the spirit of NFL.

5 Conclusions and recommendations

We have proposed an iterative separable augmented Lagrangian algorithm (SALA) for optimal structural design. The algorithm solves a sequence of separable quadratic-like programs, able to capture reciprocal- and exponential-like behavior, which is desirable in structural optimization. A salient feature of the algorithm is that the primal and dual variable updates are all updated using closed-form expressions.

For further reading, the reader is referred to the monograph by Boyd et al. (2010) and some of the references mentioned therein, for instance Bertsekas.[3] The first five chapters of the monograph by Boyd et al. give an overview of the main ADMM concepts and methods, and they also treat the special case of the QP and separable objective and constraints. Since algorithms like SALA are known to be very sensitive to scaling, we have proposed a scaling method inspired by the well-known ALGENCAN algorithm. Numerical results for SALA and ALGENCAN suggest that the algorithms perform quite similar. Having said this: "optimal" scaling of the algorithm (and related algorithms) remains an open issue.

Indeed, the sensitivity of SALA to scaling and different parameter settings may well be its biggest drawback. Having said this, even "classical" AL statements are well known to be prone to this, and the same even applies for penalty methods. To take this argument even further, "older" SQP formulations that depended on a merit acceptance function revealed the same difficulties.

The scaling issue posed by SALA is of such complexity that Lenoir and Mahey (2007) dedicated an entire paper to this issue. Interested readers may find an in-depth analysis of the effect of different scaling strategies therein. As mentioned before, scaling is an open-ended problem that will require further investigation but is beyond the scope of this paper.

Although we do not exploit this feature herein, the primal and dual updates in SALA are both embarrassingly parallel, which makes the algorithm suitable for implementation on massively parallel computational devices, including general purpose graphical processor units (GPGPUs). For some problems, a relatively slow ADMM iteration process leads to increased computational effort on the subproblem level, when compared with more traditional SQP methods. We noted this in our numerical testing, as ALGENCAN and LSQP sometimes required less CPU time than a single threaded SALA implementation. However, SALA is theoretically divisible in problem time by at least n, meaning that the advantage over traditional SQP methods should scale with increased problem size. (Problem dimensionality will of course also have to be high enough to offset the overhead computational costs associated with parallel computing.) We hope to demonstrate the possible advantages in the near future.

Finally, while we have used only two approximations herein; many different methods for obtaining the diagonal approximate higher order curvatures may be used. Possibilities include quasi-Cauchy updates (Duysinx et al. 1995), and quadratic approximations to suitable intervening variables (Groenwold et al. 2010; Groenwold and Etman 2010). The last possibility includes the quadratic approximation to the reciprocal and exponential approximations and even the quadratic approximations to the CONLIN and MMA approximations.

6 Replication of results

Our SALA algorithm is part of an optimization suite that is propriety for the time being; we can therefore not share code or code fragments in a meaningful way.

Acknowledgments This work is based on research supported in part by the National Research Foundation of South Africa (Grant Number CPRR160421162639), which is gratefully acknowledged. We would also like to express our sincerest appreciation of the suggestions made by the anonymous reviewers, which have significantly improved our paper.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

[3]This suggestion was proposed by one of the helpful anonymous reviewers.

Appendix A: Tables

Table 1 The test problems

Number	Problem name	Approximation	n	m	References	Notes
1	Svanberg's cantilever	SPH-QDR	5	1	Svanberg (1987b)	
2	Toropov's cantilever	T2:R	1024	1	Groenwold and Etman (2011) and Toropov 2008	1
3	Svanberg's first nonconvex problem	SPH-QDR	200	2	Svanberg (2002)	
4	Svanberg's first nonconvex problem	SPH-QDR	200	2	Svanberg (2002)	
5	12-Corner-polytope problem	SPH-QDR	21	1	Svanberg (1995)	
6	Vanderplaat's cantilever #1	T2:R	200	201	Vanderplaats (2001)	
7	Vanderplaat's cantilever #2	T2:R	200	200	Vanderplaats (2001)	
8	Vanderplaat's cantilever #3	T2:R	200	200	Vanderplaats (2001)	
9	Fleury's weight minimization problem	T2:R	1000	2	Fleury (1979)	
10	Svanberg's snake problem	SPH-QDR	30	41	Svanberg (2007)	
11	Cam-design problem	SPH-QDR	15	49	Dolan et al. (2004)	
12	Svanberg's cantilever	Exact	5	1	Svanberg (1987b)	2
13	HS-43	Exact	4	3	Hock and Schittkowski (1981)	2, 3

Note 1: This is a generalization of Svanberg's cantilever. Note 2: The problem is separable. Note 3: The Hock and Schittkowski test set

Table 2 Numerical results for the test problems using the ALGENCAN and SALA solvers

	ALGENCAN				SALA				LSQP			
PR	f_0^*	h^*	$\mathcal{K}^*$	N_f	f_0^*	h^*	$\mathcal{K}^*$	N_f	f_0^*	h^*	$\mathcal{K}^*$	N_f
1	1.3399564	1.27×10^{-08}	2.89×10^{-05}	13	1.3399564	$0.00\times10^{+00}$	8.43×10^{-06}	20	1.3399563	6.48×10^{-08}	2.89×10^{-05}	13
2	1.3103299	3.04×10^{-07}	3.63×10^{-08}	8	1.3103281	4.46×10^{-06}	3.32×10^{-13}	8	1.3103301	$0.00\times10^{+00}$	7.75×10^{-13}	8
3	51.046453	$0.00\times10^{+00}$	1.02×10^{-02}	100	51.046524	4.31×10^{-04}	1.17×10^{-02}	97	51.046253	$0.00\times10^{+00}$	8.49×10^{-03}	104
4	−148.95382	2.61×10^{-06}	7.02×10^{-03}	164	−148.95388	2.71×10^{-04}	6.08×10^{-03}	169	−148.95382	2.07×10^{-06}	7.02×10^{-03}	164
5	−279.90266	2.01×10^{-07}	8.15×10^{-03}	136	−279.90246	$0.00\times10^{+00}$	7.28×10^{-03}	126	−279.90215	5.67×10^{-08}	5.82×10^{-03}	146
6	63678.094	1.90×10^{-07}	7.49×10^{-03}	9	63678.897	8.07×10^{-05}	3.98×10^{-03}	9	63678.100	3.33×10^{-10}	7.50×10^{-03}	9
7	54176.212	5.53×10^{-09}	2.83×10^{-05}	8	54176.190	4.06×10^{-06}	8.33×10^{-04}	8	54176.212	1.07×10^{-13}	1.98×10^{-05}	8
8	54155.571	3.31×10^{-07}	1.24×10^{-03}	8	54155.570	1.68×10^{-06}	4.67×10^{-02}	8	54155.571	1.92×10^{-08}	1.24×10^{-03}	8
9	950.00005	7.93×10^{-07}	7.75×10^{-05}	22	950.00136	$0.00\times10^{+00}$	2.79×10^{-03}	20	950.00003	1.52×10^{-05}	7.78×10^{-03}	32
10	—	—	—	—	—	—	—	—	—	—	—	—
11	−4.3452690	1.14×10^{-07}	8.65×10^{-05}	5	−4.3815720	3.36×10^{-04}	8.12×10^{-04}	32	−4.3452583	3.55×10^{-15}	1.79×10^{-06}	6
12	1.3399566	$0.00\times10^{+00}$	3.37×10^{-07}	13	1.3399564	3.12×10^{-09}	1.66×10^{-05}	15	1.3399563	3.56×10^{-08}	1.47×10^{-08}	13
13	−44.000002	2.05×10^{-06}	1.10×10^{-05}	6	−44.000003	2.47×10^{-06}	3.04×10^{-05}	8	−44.000003	1.17×10^{-06}	1.15×10^{-05}	6

Superscript * indicates the final values at termination, while "—" indicates that the algorithm failed to terminate

Appendix B: The approximations used

B.1 A spherical diagonal quadratic approximation (SPH-QDR)

To construct a spherical quadratic approximation (Snyman and Hay 2002), we select $c^k_{2i_j} \equiv c^k_{2_j}\ \forall\, i$, which requires the determination of the single unknown $c^k_{2_j}$, to be obtained by (for example) enforcing the condition

$$\tilde{f}_j(\boldsymbol{x}^{k-1}) = f_j(\boldsymbol{x}^{k-1}), \qquad (27)$$

which implies that

$$c^k_{2_j} = \frac{2[f_j(\boldsymbol{x}^{k-1}) - f_j(\boldsymbol{x}^k) - \nabla^T f_j(\boldsymbol{x}^k)(\boldsymbol{x}^{k-1} - \boldsymbol{x}^k)]}{\|\boldsymbol{x}^{k-1} - \boldsymbol{x}^k\|_2^2}. \qquad (28)$$

This results in the approximation proposed by Snyman and Hay (2002). For the first iteration, when no historic information is available, curvatures of unity are assumed. An alternative condition for formulating a spherical quadratic approximation is presented in Wilke et al. (2010).

B.2 The quadratic approximation to the reciprocal approximation (T2:R)

For reciprocal intervening variables, the second-order partial derivatives $c^k_{2i_j}$ are obtained (Groenwold et al. 2010) as

$$c^k_{2i_j} = \frac{\partial^2 \tilde{f}_{\mathrm{R}}}{\partial x_i^2}\left(\boldsymbol{x}^k\right) = \frac{-2}{x_i^k}\left(\frac{\partial f_j}{\partial x_i}\right)^k, \tag{29}$$

where $\tilde{f}_{\mathrm{R}}$ indicates the reciprocal approximation; see also Groenwold and Etman (2010). Of course, (29) is only sensible if $\partial f_j/\partial x_i{}^k < 0$. If not, we choose to use

$$c^k_{2i_j} = \left|\frac{\partial^2 \tilde{f}_{\mathrm{R}}}{\partial x_i^2}\left(\boldsymbol{x}^k\right)\right| = \frac{2}{x_i^k}\left(\frac{\partial f_j}{\partial x_i}\right)^k, \tag{30}$$

which admittedly is very conservative, but this choice has served us well previously. Nevertheless, many other possibilities exist.

References

Birgin EG, Martínez JM (2007) Practical augmented Lagrangian methods for constrained optimization. Fundamentals of algorithms. SIAM, Pennsylvania

Boyd S, Parikh N, Chu E, Peleato B, Eckstein J (2010) Distributed optimization and statistical learning via the alternating direction method of multipliers. Found Trends Mach Learn 3:1–122

Conn AR, Gould NIM, Toint PhL (1992) LANCELOT: a Fortran package for large-scale nonlinear optimization (Release A), volume 17 of springer series in computational mathematics. Springer, Heidelberg

Dolan ED, Moré JJ, Munson TS (2004) Benchmarking optimization software with COPS 3.0. Mathematics and computer science division technical report ANL/MCS-TM-273, Argonne National Laboratory, Illinois

Douglas J, Rachford HH (1956) On the numerical solution of heat conduction problems in two and three space variables. Trans Amer Math Soc 82:421–439

Duysinx P, Zhang WH, Fleury C, Nguyen VH, Haubruge S (1995) A new separable approximation scheme for topological problems and optimization problems characterized by a large number of design variables. In: Ollhoff N, Rozvany GIN (eds) Proc. First World Congress on structural and multidisciplinary optimization, Goslar, pp 1–8

Etman LFP, Groenwold AA, Rooda JE (2009) On diagonal QP subproblems for sequential approximate optimization. In: Proc. Eighth World congress on structural and multidisciplinary optimization. Lisboa. Paper 1065

Etman LFP, Groenwold AA, Rooda JE (2012) First-order sequential convex programming using approximate diagonal QP subproblems. Struct Mult Optim 45:479–488

Fleury C (1979) Structural weight optimization by dual methods of convex programming. Int J Numer Meth Eng 14:1761–1783

Fleury C, Braibant V (1986) Structural optimization: a new dual method using mixed variables. Int J Numer Meth Eng 23:409–428

Gould NIM, Orban D, Toint PhL (2003) GALAHAD, a library of thread-safe Fortran 90 packages for large-scale nonlinear optimization. CM Trans Math Softw 29:353–372

Groenwold AA, Etman LFP (2008) Sequential approximate optimization using dual subproblems based on incomplete series expansions. Struct Multidisc Opt 36:547–570

Groenwold AA, Etman LFP (2010) A quadratic approximation for structural topology optimization. Int J Numer Meth Eng 82:505–524

Groenwold AA, Etman LFP (2011) SAOi: an algorithm for very large scale optimal design. In: Proc. Ninth World congress on structural and multidisciplinary optimization. Shizuoka, Japan. Paper 035

Groenwold AA, Etman LFP, Wood DW (2010) Approximated approximations for SAO. Struct Mult Optim 41:39–56

Hamdi A (2005a) Decomposition for structured convex programs with smooth multiplier methods. Appl Math Comput 169(1):218–241

Hamdi A (2005b) A primal-dual proximal point algorithm for constrained convex programs. Appl Math Comput 162(1):293–303

Hamdi A (2005c) Two-level primal-dual proximal decomposition technique to solve large scale optimization problems. Appl Math Comput 160(3):921–938

Hamdi A, Mahey P (2000) Separable diagonalized multiplier method for decomposing nonlinear programs. Comput Appl Math 19:1–29

Hamdi A, Mahey P, Dussault JP (1997) A new decomposition method in nonconvex programs via separable augmented Lagrangians. In: Gritzman P, Horst R, Sachs E, Tichatschke R (eds) Recent advances in optimization, volume 452 of lecture notes in economics and mathematical systems. Springer

Hestenes MR (1969) Multiplier and gradient methods. J Optim Theory Appl 4:303–320

Hock W, Schittkowski K (1981) Test examples for nonlinear programming codes, volume 187 of lecture notes in economics and mathematical systems. Springer, Berlin

Kanno Y, Kitayama S (2018) Alternating direction method of multipliers as a simple effective heuristic for mixed-integer nonlinear optimization. Struct Multidisc Opt 58:1291–1295

Lenoir A, Mahey P (2007) Global and adaptive scaling in a separable augmented Lagrangian algorithm. Research Report LIMOS RR-07-14. Université Blaise Pascal, Clermont-Ferrand

Lions PL, Mercier B (1984) Splitting algorithms for the sum of two nonlinear operators. SIAM J Control Optim 22:277–293

Nocedal J, Wright SJ (2006) Numerical optimization. Springer series in operations research and financial engineering, 2nd edn. Springer

Powell MJD (1969) A method for nonlinear constraints in optimization. In: Fletcher R (ed) Optimization. Academic Press, New York, pp 283–298

Rockafellar RT (1973) The multiplier method of Hestenes and Powell applied to convex programming. J Optim Theory Appl 12:555–562

Snyman JA, Hay AM (2002) The dynamic-Q optimization method: an alternative to SQP? Comput Math Appl 44:1589–1598

Svanberg K (1987a) The method of moving asymptotes — a new method for structural optimization. Int J Numer Meth Eng 24(2):359–373

Svanberg K (1987b) The method of moving asymptotes - a new method for structural optimization. Int J Numer Meth Eng 24:359–373

Svanberg K (1995) A globally convergent version of MMA without linesearch. In: Rozvany GIN, Olhoff N (eds) Proc. First World congress on structural and multidisciplinary optimization. Goslar, Germany, pp 9–16

Svanberg K (2002) A class of globally convergent optimization methods based on conservative convex separable approximations. SIAM J Optim 12:555–573

Svanberg K (2007) On a globally convergent version of MMA. In: Proc. Seventh World congress on structural and multidisciplinary optimization. Seoul, Paper no. A0052

Toropov VV (2008) Personal discussion

Vanderplaats GN (2001) Numerical optimization techniques for engineering design. Vanderplaats R&D, Inc., Collorado

Wilke DN, Kok S, Groenwold AA (2010) The application of gradient-only optimization methods for problems discretized using non-constant methods. Struct Mult Optim 40:433–451

Wolpert D, Macready W (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1:67–82