Springer Proceedings in Mathematics & Statistics

Mehiddin Al-Baali · Lucio Grandinetti
Anton Purnama *Editors*

Numerical Analysis and Optimization

NAO-IV, Muscat, Oman, January 2017

Springer

Springer Proceedings in Mathematics & Statistics

Volume 235

This book series features volumes composed of selected contributions from workshops and conferences in all areas of current research in mathematics and statistics, including operation research and optimization. In addition to an overall evaluation of the interest, scientific quality, and timeliness of each proposal at the hands of the publisher, individual contributions are all refereed to the high quality standards of leading journals in the field. Thus, this series provides the research community with well-edited, authoritative reports on developments in the most exciting areas of mathematical and statistical research today.

More information about this series at http://www.springer.com/series/10533

Mehiddin Al-Baali · Lucio Grandinetti
Anton Purnama
Editors

Numerical Analysis and Optimization

NAO-IV, Muscat, Oman, January 2017

Springer

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

Anton Purnama
Department of Mathematics
and Statistics
Sultan Qaboos University
Muscat
Oman

Lucio Grandinetti
Dipartimento di Ingegneria Informatica,
Modellistica, Elettronica e Sistemistica
Calabria University
Arcavacata
Italy

ISSN 2194-1009 ISSN 2194-1017 (electronic)
Springer Proceedings in Mathematics & Statistics
ISBN 978-3-319-90025-4 ISBN 978-3-319-90026-1 (eBook)
https://doi.org/10.1007/978-3-319-90026-1

Library of Congress Control Number: 2018938781

Mathematics Subject Classification (2010): 65-XX, 90-XX, 49-XX, 93-XX, 68Nxx, 97Cxx, 97Nxx, 97Pxx, 97Rxx

© Springer International Publishing AG, part of Springer Nature 2018
This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Printed on acid-free paper

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

The 4th International Conference on Numerical Analysis and Optimization



Dedicated to

R. Fletcher and M. J. D. Powell

Preface

This special edited book series of Springer Proceedings in Mathematics and Statistics contains 13 selected keynote papers presented at the *Fourth International Conference on Numerical Analysis and Optimization: Theory, Methods, Applications and Technology Transfer* (NAOIV-2017) held during January 2–5, 2017, at Sultan Qaboos University (SQU), Muscat, Oman. This conference was sponsored by SQU, Oman Mathematics Committee (OMC), COMSTECH (Pakistan), AMPL (USA), German University of Technology (GUtech) in Oman, and the British Council (Oman). Seventeen world-leading researchers gave keynote lectures. In total, thirty-seven international participants contributed talks. After the conference, selected contributions were invited to be submitted as full papers for publication in a special issue of the following international journals: *Optimization Methods and Software*, and *SQU Journal for Science*. More information is available at the Website https://conference.squ.edu.om/Default.aspx?tabid=2157. Thirteen of the keynote papers were selected for this edited proceedings volume each of which was accepted 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. This volume is dedicated to the late Profs. Mike JD Powell and Roger Fletcher (passed away on 2015 and 2016, respectively), who were the pioneers and leading figures in the mathematics of nonlinear optimization.

The NAO conference series held once every 3 years at SQU: the first conference (NAO-2008) held on April 6–8, 2008, the second conference (NAOII-2011) held on January 3–6, 2011, and the third conference (NAOIII-2014) held on January 5–9, 2014. 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 simulate 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 NAOIII-2014 (http://conference.squ.edu.om/Default.aspx?tabid=572) 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 volume 20(2) for the (2015) special issue on *Numerical Analysis and Optimization* (free download from http://web.squ.edu.om/squjs/Archives/Volume20(Part2).htm).

For the past NAOII-2011 (http://conference.squ.edu.om/Portals/16/Conference2011/index.htm) 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 http://web.squ.edu.om/squjs/Archives/Volume17-part(1).htm) and eight papers in the volume 17(2) for the (2012) special issue on *Numerical Analysis* (free download from http://web.squ.edu.om/squjs/Archives/Volume17-part(2).htm).

For the first NAO-2008 (http://conference.squ.edu.om/Portals/16/Conference2008/index.htm) conference, Mike Powell was the first plenary speaker in the NAO series.

Muscat, Oman — Mehiddin Al-Baali
Arcavacata, Italy — Lucio Grandinetti
Muscat, Oman — Anton Purnama

In Memory of Roger Fletcher and Michael Powell

Compilation of Notes from Former Students and Colleagues

Philippe L. Toint
University of Namur
Belgium

During the conference, I first gave a memorial address during the opening session, in the presence of the Omani government delegates, the Faculty Dean, the SQU professors and all scientific delegates. This address was triggered by the disappearance of two major figures in the world community of optimization researchers: Michael J. D. Powell and Roger Fletcher both left us during the years 2015 and 2016, respectively.

Mike Powell was my supervisor during my stay in Cambridge in 1977 for the preparation of my Ph.D. thesis, and a friend and colleague for many years after. His short illness and death were a real shock for me (and many colleagues), since I knew him so well and respected his work so much. He was indeed one of the very early pioneers of numerical optimization in the UK, and his work laid the foundations of several subjects which are today of continued importance, such as quasi-Newton methods, sequential quadratic programming, augmented Lagrangians, trust-regions and interpolation-based derivative-free algorithms. His approach of science and mathematics was at the highest possible standard, always leading to both deep and far-reaching analysis and also to truly practical methods and associated software, ready to be used in many real-world applications (he was especially proud of the fact that his methods were instrumental in the design of the first manned mission to the moon). His support, encouragements and criticisms were crucial in the development of my own vision of optimization and also in the unfolding of my career.

Although I knew Roger Fletcher also since the late 1970s, I only came to closely collaborate with him much later. Together with Powell, he was one of the leading figures in the then nascent field of numerical nonlinear optimization, and his contributions also clearly rank amongst the most influential. His originality and depth, combined with his attention to young students like me, made his presence and suggestions most useful. Several years later, I had a chance to invite him to Belgium

for a conference I was organizing at the time he had started, with S. Leyffer, to work on the idea of filter methods. The discussions we had then on the theoretical underpinning of this very successful approach led to an excellent collaboration of several years and a handful of papers, creating the basis of the now well-known filter convergence theory.

Both Powell and Fletcher were very keen hikers, an activity which I like practicing myself. During my memorial address, I thus had the opportunity to recall some souvenirs of hikes with both of them (together and separately) in the Scottish and Welsh mountains. These were somewhat of an initiation rite for a young researcher and, later, great opportunities to share a common appreciation and nature and landscapes with true friends. I did my best to show that sharing these memories and others was, for me, an important duty of true personal appreciation over many years.

Andrew R. Conn
IBM T. J. Watson Research Center
USA

Michael influenced me a great deal professionally, but here is a non-mathematical influence that (my wife) Barbara thought was more important.

When we stayed with Mike and Catherine, he would bring us up tea to our bedroom every morning and Barbara enjoyed it so much that she coerced me to continue the practice at home.

David M. Gay
AMPL Optimization Inc
USA

In 1977, I was privileged to spend a few weeks visiting the (now defunct) A. E. R. E. Harwell in England. Mike was still there, but was soon to move to Cambridge. In the course of trying to construct an example of Broyden's method never completely converging on a linear problem, in hopes of including such an example in a paper that Bobby Schnabel and I were writing, I was surprised to end up constructing a proof that Broyden's method converges in at most 2n steps on linear problems. This happened a few weeks before I arrived at Harwell. When I told Mike about the result, he did a double take. I have always treasured this memory.

P. S. John Dennis suggested that this result might mean Broyden's method is locally 2n-step Q-quadratically convergent on nonlinear problems, which I was then able to prove.

Mehiddin Al-Baali
Sultan Qaboos University
Oman

Roger Fletcher was amongst the pioneers of numerical optimization. Since he was my Ph.D. supervisor, it is expected that I was influenced by his way of thinking. Indeed, his help and discussions were useful as he accepted a new research idea

only if supported with either proof or motivation. In particular, when I told him after the usual Tuesday afternoon coffee break that "I proved the descent property of the Fletcher-Reeves method when inexact line search is used," he replied "it is interesting". Later on the same day, he found my proof is correct and told me to write up this result immediately (which implies to stop writing my Ph.D. thesis). Then, Roger also expected a proof of the global convergence of the method by extending that of Powell, which he received recently. After checking the paper, he returned the last version to me by adding "by Mehiddin Al-Baali", next to the title of the paper, although he knew it is very interesting result as being "the first practical convergence result" and honestly, it would be my honour to have him as a joint author. He felt I deserve that result alone, and he helped me as being my supervisor.

I used to have useful discussions with Mike Powell whenever I meet him at conferences, including the Dundee meetings. I used to listen to him carefully and learned about his way of analysing mathematical problems, as he was quite accurate with certain expected results. For example, in the Dundee 1987 meeting, he said to me "the self-scaling technique for the quasi-Newton methods is sometimes useful", although this technique was unattractive to many researchers. After a few years, I converted his word "sometimes" to practise by considering scaling only when the value of the scaling parameter is smaller than one. In this way, I was able to show the useful theoretical and numerical features of the self-scaling technique.

Contents

Quasi-Newton Based Preconditioning and Damped Quasi-Newton Schemes for Nonlinear Conjugate Gradient Methods 1
Mehiddin Al-Baali, Andrea Caliciotti, Giovanni Fasano and Massimo Roma

Solving Minimax Problems: Local Smoothing Versus Global Smoothing 23
A. M. Bagirov, N. Sultanova, A. Al Nuaimat and S. Taheri

A Competitive Error in Variables Approach and Algorithms for Finding Positive Definite Solutions of Linear Systems of Matrix Equations 45
Negin Bagherpour and Nezam Mahdavi-Amiri

Sparse Direct Solution on Parallel Computers 67
Iain Duff, Florent Lopez and Stojce Nakov

Revisiting Expression Representations for Nonlinear AMPL Models 99
David M. Gay

On the Consistency of Runge–Kutta Methods Up to Order Three Applied to the Optimal Control of Scalar Conservation Laws 119
Michael Hintermüller and Nikolai Strogies

A Time-Delay Neural Network Model for Unconstrained Nonconvex Optimization 155
Li-Zhi Liao and Yu-Hong Dai

Stabilized Optimization Via an NCL Algorithm 173
Ding Ma, Kenneth L. Judd, Dominique Orban and Michael A. Saunders

Combinatorial Optimization Problems in Engineering Applications 193
Hans D. Mittelmann

Optimal Error Estimates for Semidiscrete Galerkin Approximations to Multi-dimensional Sobolev Equations with Burgers' Type Nonlinearity 209
Ambit K. Pany and Sudeep Kundu

Optimal Power Flow Analysis in Power Dispatch for Distribution Networks 229
Paolo Pisciella, Maria Teresa Vespucci, Giacomo Viganò, Marco Rossi and Diana Moneta

Equilibria and Weighted Complementarity Problems 249
Florian A. Potra

A Mathematical Approach to Living on Sinking Ground 277
Kees Roos

NAOIV-2017 Conference Participants 297

Contributors

A. Al Nuaimat Faculty of Science and Technology, Federation University Australia, Ballarat, VIC, Australia

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

Negin Bagherpour Department of Mathematics, The University of Manchester, Manchester, UK

A. M. Bagirov Faculty of Science and Technology, Federation University Australia, Ballarat, VIC, Australia

Andrea Caliciotti Dipartimento di Ingegneria Informatica, Automatica e Gestionale 'A. Ruberti', SAPIENZA Università di, Roma, Italy

Yu-Hong Dai LSEC, Academy of Mathematics and Systems Sciences, Chinese Academy of Sciences, Beijing, China; School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing, China

Iain Duff STFC-RAL, Didcot, UK

Giovanni Fasano Department of Management, University Ca' Foscari, Venice, Italy

David M. Gay AMPL Optimization Inc., Albuquerque, NM, USA

Michael Hintermüller Weierstrass Institute, Berlin, Germany; Humboldt-Universität zu Berlin, Berlin, Germany

Kenneth L. Judd Hoover Institution, Stanford University, Stanford, CA, USA

Sudeep Kundu Department of Mathematics, Industrial Mathematics Group, IIT Bombay, Powai, Mumbai, India

Li-Zhi Liao Department of Mathematics, Hong Kong Baptist University, Kowloon, Hong Kong

Florent Lopez STFC-RAL, Didcot, UK

Ding Ma Management Science and Engineering, Stanford University, Stanford, CA, USA

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

Hans D. Mittelmann School of Mathematical and Statistical Sciences, Arizona State University, Tempe, AZ, USA

Diana Moneta RSE - Ricerca sul Sistema Energetico S.p.A., Milano, Italy

Stojce Nakov STFC-RAL, Didcot, UK

Dominique Orban GERAD and Department of Mathematics and Industrial Engineering, École Polytechnique, Montréal, QC, Canada

Ambit K. Pany Center of Applied Mathematics, SOA University, Bhubaneswar, India

Paolo Pisciella Department of Industrial Economics and Technology Management, NTNU, 7491 Trondheim, Norway

Florian A. Potra Department of Mathematics and Statistics, University of Maryland Baltimore County, Baltimore, MD, USA

Massimo Roma Dipartimento di Ingegneria Informatica, Automatica e Gestionale 'A. Ruberti', SAPIENZA Università di, Roma, Italy

Kees Roos Delft University of Technology, GA Delft, The Netherlands

Marco Rossi RSE - Ricerca sul Sistema Energetico S.p.A., Milano, Italy

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

Nikolai Strogies Weierstrass Institute, Berlin, Germany

N. Sultanova Faculty of Science and Technology, Federation University Australia, Ballarat, VIC, Australia

S. Taheri Faculty of Science and Technology, Federation University Australia, Ballarat, VIC, Australia

Maria Teresa Vespucci Department of Management, Information and Production Engineering, University of Bergamo, 24044 Dalmine (BG), Italy

Giacomo Viganò RSE - Ricerca sul Sistema Energetico S.p.A., Milano, Italy

Quasi-Newton Based Preconditioning and Damped Quasi-Newton Schemes for Nonlinear Conjugate Gradient Methods



Mehiddin Al-Baali, Andrea Caliciotti, Giovanni Fasano and Massimo Roma

Abstract In this paper, we deal with matrix-free preconditioners for nonlinear conjugate gradient (NCG) methods. In particular, we review proposals based on quasi-Newton updates, and either satisfying the secant equation or a secant-like equation at some of the previous iterates. Conditions are given proving that, in some sense, the proposed preconditioners also approximate the inverse of the Hessian matrix. In particular, the structure of the preconditioners depends both on low-rank updates along with some specific parameters. The low-rank updates are obtained as by-product of NCG iterations. Moreover, we consider the possibility to embed *damped techniques* within a class of preconditioners based on quasi-Newton updates. Damped methods have proved to be effective to enhance the performance of quasi-Newton updates, in those cases where the Wolfe linesearch conditions are hardly fulfilled. The purpose is to extend the idea behind damped methods also to improve NCG schemes, following a novel line of research in the literature. The results, which summarize an extended numerical experience using large-scale CUTEst problems, is reported, showing that these approaches can considerably improve the performance of NCG methods.

Keywords Preconditioning · Large scale unconstrained optimization · Nonlinear conjugate gradient methods · Quasi-Newton methods · Damped techniques Conjugate gradient method

M. Al-Baali
Department of Mathematics and Statistics, Sultan Qaboos University,
Box 36, Muscat 123, Oman
e-mail: albaali@squ.edu.om

A. Caliciotti · M. Roma
Dipartimento di Ingegneria Informatica, Automatica e Gestionale 'A. Ruberti',
SAPIENZA Università di, Roma, Italy
e-mail: caliciotti@dis.uniroma1.it

M. Roma
e-mail: roma@dis.uniroma1.it

G. Fasano (✉)
Department of Management, University Ca' Foscari, Venice, Italy
e-mail: fasano@unive.it

© Springer International Publishing AG, part of Springer Nature 2018

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*,
Springer Proceedings in Mathematics & Statistics 235,
https://doi.org/10.1007/978-3-319-90026-1_1

1 Introduction

Several iterative methods were proposed in the literature, for the solution of the large-scale unconstrained optimization problem $\min_{x\in\mathbb{R}^n} f(x)$, where $f:\mathbb{R}^n\to\mathbb{R}$ (see, e.g., [1–6]). Among them, the nonlinear conjugate gradient (NCG) along with quasi-Newton methods is undoubtedly the most commonly used. Indeed, they both prove to be actually effective in practice and are endowed with a mature theory, including strong convergence properties.

On this purpose, let us first consider a general iterative preconditioned nonlinear conjugate gradient (PNCG) method, which generates a sequence of iterates $\{x_k\}$. Essentially, three choices at current step k strongly affect both the effectiveness and the efficiency of the overall method. The first choice refers to the adopted *linesearch procedure*, along with the selected steplength $\alpha_k > 0$ used to give the next iterate x_{k+1}, being

$$x_{k+1} = x_k + \alpha_k p_k,$$

where p_k is the search direction. The second choice refers to the selection of the parameter β_k, which is responsible for the computation of the next search direction, being

$$p_{k+1} = -g_{k+1} + \beta_k p_k,$$

where $p_1 = -g_1$ and g_k denotes $\nabla f(x_k)$. In the case where the function $f(x)$ is non-quadratic, different expressions for the parameter β_k in the latter formula may yield significantly different (preconditioned) NCG schemes. In particular, among the first classic choices in the literature for the parameter β, we have the proposals by Fletcher and Reeves (FR) [7], Polak and Ribière (PR) [8], Hestenes and Stiefel (HS) [9]. More modern and efficient schemes have also been studied. In particular, we urge to mention the proposals in the seminal papers [10] and [3, 4], since they raised novel ideas which have inspired several advances in the last decade. Recently, Neculai (see [11] and therein references) reported an efficient version of the NCG method, which promises to outperform the proposal in [4]. This gives room to further improvements in the latest literature (see also [5]), where some appealing properties of L-BFGS update are exploited in the context of NCG, with the purpose of improving efficiency. The latter research area has also partially inspired the results reported in the current paper.

The third proper choice for the symmetric positive definite preconditioner $M_{k+1} \in \mathbb{R}^{n\times n}$ often plays a keynote role for the computation of p_{k+1}, being

$$p_{k+1} = -M_{k+1} g_{k+1} + \beta_k p_k,$$

where β_k may depend on M_k and M_{k+1} and $p_1 = -M_1 g_1$. Of course, the latter three choices are not independent. Indeed, an inaccurate linesearch procedure turns to be harmful and may require a large number of function and gradient evaluations. Similarly, a careless choice of the preconditioner risks to possibly destroy both convergence properties and numerical performance of the PNCG. These observations impose a specific attention before selecting a preconditioner.

In the first part of this paper, we review some preconditioners for NCG, which are based on the satisfaction of a *secant-based* equation (see [12–14] for details). Our main purpose here is to show that imposing the satisfaction of the secant condition surely represents an important guideline to gain second-order information about the objective function. However, on highly nonlinear functions, when the distance among the last iterates increases, the satisfaction of the secant equation at any iterate might represent a tight request, which does not necessarily enhance the information on second-order information. On the contrary, in [12] the approximation of an average Hessian matrix is built by using an initial guess suggested by the quadratic case. Then, the initial guess is refined imposing some *secant-like* conditions, which are used to set accordingly some parameters.

We remark that the preconditioners are iteratively constructed and based on satisfying either the secant or a modified secant equation and partially recover the structure of quasi-Newton updates. On the overall, our proposals for preconditioners comply with the next specifications:

- do not rely on the structure of the minimization problem in hand;
- are matrix-free, and hence, they are naturally conceived for large-scale problems;
- are built drawing inspiration from quasi-Newton schemes;
- convey information from previous iterations of the PNCG method.

We urge to recall that the idea of using a quasi-Newton update as a possible preconditioner, within the NCG algorithms, is not new; examples of such an approach can be found for instance in [15, 16] or in the more recent proposal [17]. In particular, the efficient framework in [17] explicitly exploits the relation between the conjugate gradient method and BFGS quasi-Newton approach, in the quadratic case.

In the second part of the paper, we show how to combine damped techniques with preconditioning strategies, as introduced in [18]. Taking inspiration from [19–21], two different damping strategies are proposed. In particular, we focus on the Polak–Ribière (PR) (recently, Polak–Ribière–Polyak (PRP)) method, showing that, under reasonable assumptions, the damped and preconditioned version of this method (denoted by D-PR-PNCG), can be able to efficiently tackle also difficult problems. This is confirmed by the results of an extensive numerical testing reported (see [18] for details).

Under mild assumptions, the proposals in this paper preserve convergence properties for the PNCG method.

As regards the notations, we denote for an n-real vector x, the Euclidean norm by $\|x\|$. Moreover, for a symmetric matrix A, $A \succ 0$ indicates that A is positive definite.

1.1 Preconditioned Nonlinear Conjugate Gradient (PNCG) Method

Here, we first recall a general scheme of PNCG algorithm. In the following scheme, $M_k \in \mathbb{R}^{n\times n}$ denotes a possible positive definite preconditioner at the iteration k.

Preconditioned Nonlinear Conjugate Gradient (PNCG) Scheme

Step 1: Data $x_1 \in \mathbb{R}^n$ and $M_1 \succ 0$. Set $p_1 = -M_1 g_1$ and $k = 1$.

Step 2: Use a linesearch procedure to compute the steplength α_k, which satisfies the *Wolfe conditions*, and set the next iterate as

$$x_{k+1} = x_k + \alpha_k p_k.$$

Step 3: If a *stopping criterion* is satisfied then stop, else compute the coefficient β_k along with the preconditioner $M_{k+1} \succ 0$. Compute a search direction by

$$p_{k+1} = -M_{k+1} g_{k+1} + \beta_k p_k. \tag{1}$$

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

Of course, in case $M_k = I$ for all k, the PNCG scheme reduces to the NCG method. Also, observe that as an alternative, in order to possibly improve the efficiency of NCG by introducing preconditioning strategies, the *Step 3* of PNCG might be replaced by the next one.

Step 3: If a *stopping criterion* is satisfied then stop, else compute the coefficient β_k along with the preconditioner M_{k+1}. If $M_{k+1} \not\succ 0$ or $M_{k+1} g_{k+1} = 0$ then set $M_{k+1} = I$. Compute the search direction

$$p_{k+1} = -M_{k+1} g_{k+1} + \beta_k p_k.$$

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

The steplength α_k and the parameter β_k can be chosen in a variety of ways. In particular, in order to prove global convergence properties, a Wolfe-type linesearch procedure seems mandatory, while to improve the overall efficiency, several values for β_k have appeared in the literature (see also Sect. 1). Here, we neither intend to propose a novel choice of β_k, nor we want to consider any specific linesearch procedure to compute α_k for the PNCG algorithm. In this regard, the Wolfe conditions are well-suited for our purposes, inasmuch as under mild assumptions they guarantee the fulfillment of the usual curvature condition

$$s_k^T y_k > 0,$$

being $s_k = x_{k+1} - x_k$ and $y_k = g_{k+1} - g_k$. On the other hand, we strongly remark the importance of the positive definiteness for preconditioners, in order to prove global convergence results.

2 Quasi-Newton Updates for Preconditioning

In this section, we suitably exploit some quasi-Newton updates in order to build preconditioners. As well known (see, e.g., [1]), when using quasi-Newton methods in place of (1), at iteration k, we generate a search direction of the form

$$p_k = -H_k g_k,$$

where H_k represents an approximation of the inverse Hessian matrix $[\nabla^2 f(x_k)]^{-1}$. Then, as in ***Step 2*** of PNCG, the new iterate x_{k+1} can be obtained according to $x_{k+1} = x_k + \alpha_k p_k$, where α_k as above is a steplength computed by a Wolfe-type procedure. In particular, instead of computing H_k from scratch at each iteration k, quasi-Newton methods update H_k in a simple manner by means of adding a small number of rank one matrices, in order to obtain the new approximation H_{k+1} to be used in the next iteration. Moreover, instead of storing full dense $n \times n$ approximations, they only save a few vectors of length n, which allow to represent the approximations $\{H_k\}$ implicitly.

Among the quasi-Newton schemes, the L-BFGS method is definitely considered one of the most efficient methods, and the amount of storage it requires can be controlled by the user throughout setting the *limited memory* parameter. This method is based on the construction of the approximation of the inverse Hessian matrix, by exploiting curvature information gained only from the most recent iterations. Specifically, H_{k-1} is updated by BFGS at the kth iteration as

$$H_k = V_{k-1}^T H_{k-1} V_{k-1} + \rho_{k-1} s_{k-1} s_{k-1}^T, \tag{2}$$

where

$$\rho_{k-1} = \frac{1}{s_{k-1}^T y_{k-1}}, \qquad V_{k-1} = I - \rho_{k-1} y_{k-1} s_{k-1}^T.$$

In case $f(x)$ is quadratic, i.e., $f(x) = \frac{1}{2} x^T A x + b^T x$, $A \in \mathbb{R}^{n \times n}$, $b \in \mathbb{R}^n$, then we have explicitly $V_{k-1} = I - A s_{k-1} s_{k-1}^T / s_{k-1}^T A s_{k-1}$ and the following lemma holds.

Lemma 1 *Let us consider the quadratic function $f(x) = \frac{1}{2} x^T A x + b^T x$ with $A \succ 0$. Suppose the steplength α_k in **Step 2** of PNCG is computed using an exact linesearch procedure. Given the expression of H_k in (2), along with $H_k \succ 0$ and the positions*

$$\rho_i = \frac{1}{s_i^T y_i}, \quad s_i^T y_i \neq 0, \quad i = 1, \ldots, k,$$

$$V_i = I - \rho_i y_i s_i^T, \qquad i = 1, \ldots, k,$$

then we have

$$H_k = V_{k-1}^T V_{k-2}^T \cdots V_1^T H_k^1 V_1 \cdots V_{k-2} V_{k-1} + \sum_{i=1}^{k-1} \frac{s_i s_i^T}{s_i^T A s_i}, \tag{3}$$

where $H_k^1 \succ 0$ *is given (usually, a multiple of the unit matrix).*

Proof First observe that since $f(x)$ is quadratic, then $y_i = As_i, i = 1, \ldots, k$, and the vectors $s_1, \ldots, s_k$ are mutually conjugate, i.e., $s_i^T A s_j = 0$, for any $1 \le i \ne j \le k$. We prove (3) by complete induction.
When $k = 2$, by (2), we explicitly obtain

$$H_2 = V_1^T H_k^1 V_1 + \rho_1 s_1 s_1^T = V_1^T H_k^1 V_1 + \frac{s_1 s_1^T}{s_1^T A s_1}.$$

Now, assume (3) holds for some $k-1$, and we prove (3) for the index k as follows. Recalling the conjugacy among vectors $\{s_i\}$ yields

$$V_{k-1}^T s_i = \left(I - \frac{s_{k-1} y_{k-1}^T}{s_{k-1}^T A s_{k-1}} \right) s_i = s_i, \qquad i = 1, \ldots, k-2,$$

by (2), we immediately have after some computations

$$\begin{aligned} H_k &= V_{k-1}^T H_{k-1} V_{k-1} + \frac{s_{k-1} s_{k-1}^T}{s_{k-1}^T A s_{k-1}} \\ &= V_{k-1}^T V_{k-2}^T \cdots V_1^T H_k^1 V_1 \cdots V_{k-2} V_{k-1} + \sum_{i=1}^{k-1} \frac{s_i s_i^T}{s_i^T A s_i}. \end{aligned}$$

□

Note that Formula (3) for the quadratic case can suggest iterative updates to generate preconditioners for PNCG. Indeed, drawing inspiration from (3) and [22], in case $f(x)$ is quadratic (i.e., NCG coincides with the conjugate gradient method), we have

$$A^{-1} = \sum_{j=1}^{n} \frac{s_j s_j^T}{s_j^T A s_j}. \tag{4}$$

In view of (4), the rightmost contribution in (3) may represent an approximate inverse of the Hessian matrix A up to the kth iteration. As an extension, we can borrow the last idea also in case $f(x)$ is a general nonlinear function, in order to generate possible preconditioners which approximate the rightmost matrix in (3). In particular, in this regard, we will have to assess a couple of issues:

(a) We have to set a finite number of NCG iterations $m \le n$, which are necessary to build the approximation of the rightmost matrix in (3).

(b) We have to explicitly indicate how to approximately compute the quantities $s_i^T A s_i$, for $i \geq 1$, in (3); indeed, unlike in the quadratic case, when $f(x)$ is a general nonlinear function, the quantity $s_i^T A s_i$ is unavailable at iteration i.

3 Preconditioners Based on the BFGS Update: First Proposal

In this section, we review the preconditioners for PNCG proposed in [12], which exploits the contents of Sect. 2. We now report the general expression of this class of preconditioners.

$$M_{k+1} = \tau_k C_k + \gamma_k v_k v_k^T + \omega_k \sum_{j=k-m}^{k} \frac{s_j s_j^T}{y_j^T s_j}, \tag{5}$$

where $C_k \in \mathbb{R}^{n \times n}$, $v_k \in \mathbb{R}^n$, $\tau_k, \gamma_k, \omega_k \in \mathbb{R}$ and m is positive integer. Here, we consider

$$C_k = \frac{s_k^T y_k}{\|y_k\|^2} I, \qquad \tau_k = \omega_k, \qquad \gamma_k = \frac{2}{s_k^T y_k},$$

$$v_k = s_k - \tau_k C_k y_k - \omega_k \sum_{j=k-m}^{k} \frac{s_j^T y_k}{y_j^T s_j} s_j,$$

$$\omega_k = \frac{\frac{1}{2} s_k^T y_k}{y_k^T C_k y_k + \sum_{j=k-m}^{k} \frac{(s_j^T y_k)^2}{s_j^T y_j}}, \qquad \gamma_k = \frac{2}{s_k^T y_k}$$

and $m \ll n$, $0 \leq m \leq k-1$. For further motivations along with the rationale behind this proposal, we refer to [12]. In the sequel, we report the main theoretical results and a summary of the numerical experience.

Observe that the right-hand side of (5) includes three contributions. More specifically, the rightmost matrix represents an approximate inverse Hessian, as in the guidelines of the conclusions of Sect. 2. In particular, exploiting the mean value theorem, we can write

$$y_j = g_{j+1} - g_j = \int_0^1 \nabla^2 f(x_j + t s_j)^T s_j dt, \qquad j \geq 1,$$

so that assuming $\nabla^2 f(z) = A_j$ constant for $z \in [x_j, x_{j+1}]$, we have

$$y_j^T s_j = \int_0^1 s_j^T \nabla^2 f(x_j + t s_j)^T s_j dt \approx s_j^T A_j s_j, \qquad j \geq 1,$$

showing that the issue (b), at the end of Sect. 2, can be easily treated. Moreover, the integer m in (5) represents a *memory* and guarantees that complying with (a), information from only the lattermost m iterations is collected.

A few comments need also be added, with respect to the role played by the matrix C_k and the parameter τ_k in (5). C_k is chosen similar to the matrix $H_k^1 = \lambda_k I$, where λ_k is the solution of the subproblem

$$\min_{\lambda} \|(\lambda I) y_k - s_k\|^2.$$

In other words, $\lambda_k = y_k^T s_k / \|y_k\|^2$ is a value of the parameter λ which aims at approximately solving the initial secant equation $(\lambda I) y_k = s_k$. As usual, the use of the Wolfe conditions ensures that $\lambda_k > 0$.

On the other hand, the exact role played by the parameter τ_k in (5) is a bit more technical and is in particular related to eigenvalue clustering for the preconditioner M_{k+1}, as highlighted in the next Theorem (see also Proposition 3 in [12]).

Theorem 1 *Let $f(x) = 1/2 x^T A x + b^T x$, with $A \succ 0$, and assume that*

- *$k \geq 2$ iterations of the NCG algorithm are performed.*
- *an exact linesearch procedure is adopted.*
- *M_{k+1} is defined as in (5) with $m \leq n - 2$.*

Then, at least $n - (m + 2)$ eigenvalues of M_{k+1} coincide with τ_k.

As detailed in [12], the next proposition can be proved for the update (5), showing its well-posedness and the satisfaction of some secant-like conditions.

Proposition 1 *Let f be twice continuously differentiable. Suppose that k iterations of NCG are performed, using the strong Wolfe linesearch procedure. Let M_{k+1} be defined as in (5), with $0 \leq m \leq k - 1$, $\tau_k > 0$ and $\gamma_k, \omega_k \geq 0$.*

(i) Let $C_k \in \mathbb{R}^{n \times n}$ be symmetric positive definite, then there exist values of τ_k, γ_k, ω_k such that $M_{k+1} \succ 0$ and the secant equation $M_{k+1} y_k = s_k$ is satisfied.

(ii) Let $f(x) = 1/2 x^T A x + b^T x$, with $A \succ 0$. Suppose $k \geq 2$ iterations of the NCG algorithm are performed, using an exact linesearch. Then, there exist values of τ_k, γ_k, ω_k, and a positive semidefinite matrix C_k, such that $M_{k+1} \succ 0$. Moreover, $M_{k+1} y_k = s_k$ and the modified secant conditions

$$M_{k+1} y_i = \omega_k s_i, \qquad i = k - m, \ldots, k - 1,$$

are satisfied.

Before reporting other proposals for possible preconditioners in PNCG, we highlight the role played by the vector v_k in (5). In particular, the value of v_k is set in such a way that M_{k+1} satisfies the secant equation $M_{k+1} y_k = s_k$ (at iteration k). In this

regard, the computation of vector v_k follows a similar guideline with respect to the idea adopted by SR1 quasi-Newton update (see also [1] for details).

As a preliminary numerical experience which reveals the performance of the proposal M_{k+1} in (5), the preconditioner M_{k+1} has been embedded in PNCG, with $m = \min\{4, k-1\}$ and β_k computed as in the Polak–Ribière (PR) (recently, Polak–Ribière–Polyak (PRP)) formula

$$\beta_k = \frac{\left[g_{k+1} - g_k\right]^T M_{k+1} g_{k+1}}{g_k^T M_k g_k}.$$

In [12], the resulting PR-PNCG has been experienced over a set of 112 large-scale problems of CUTEst collection [23]. This proposal (5) (namely OUR PREC_PR) is compared with the L-BFGS update (setting the memory parameter $m = 4$) (namely PREC-LBFGS_PR), used as a preconditioner, and with the unpreconditioned NCG scheme (namely UNPREC_PR). Results are reported in Figs. 1 and 2, in terms of # iterations and # of function evaluations. Note that the steplength α_k is computed such that the strong Wolfe conditions

$$f_{k+1} \leq f_k + c_1 \alpha_k g_k^T p_k,$$

and

$$|g_{k+1}^T p_k| \leq c_2 |g_k^T p_k|,$$

where $0 < c_1 < 0.5$ and $c_1 < c_2 < 1$, hold (we used as for the code CG+, $c_1 = 0.0001$ and $c_2 = 0.9$). We also remark that in Fig. 1, the original stopping criterion of the code CG+ (see [24]), i.e., $\|g_k\|_\infty \leq 10^{-5}(1 + |f_k|)$, is adopted, while in Fig. 2, the more common criterion from the literature

$$\|g_k\| \leq 10^{-5} \max\{1, \|x_k\|\} \tag{6}$$

is used, showing the effectiveness and efficiency of our first proposal (5).

4 Preconditioners Based on the BFGS Update: Second Proposal

As second proposal for a possible preconditioning strategy, which again exploits the contents in Sect. 2, we have the following update for M_{k+1} in PNCG scheme as proposed in [14].

$$M_{k+1} = \delta_k M_k + \gamma_k v_k v_k^T + \omega_k \frac{p_k p_k^T}{y_k^T p_k}, \qquad \delta_k > 0, \tag{7}$$

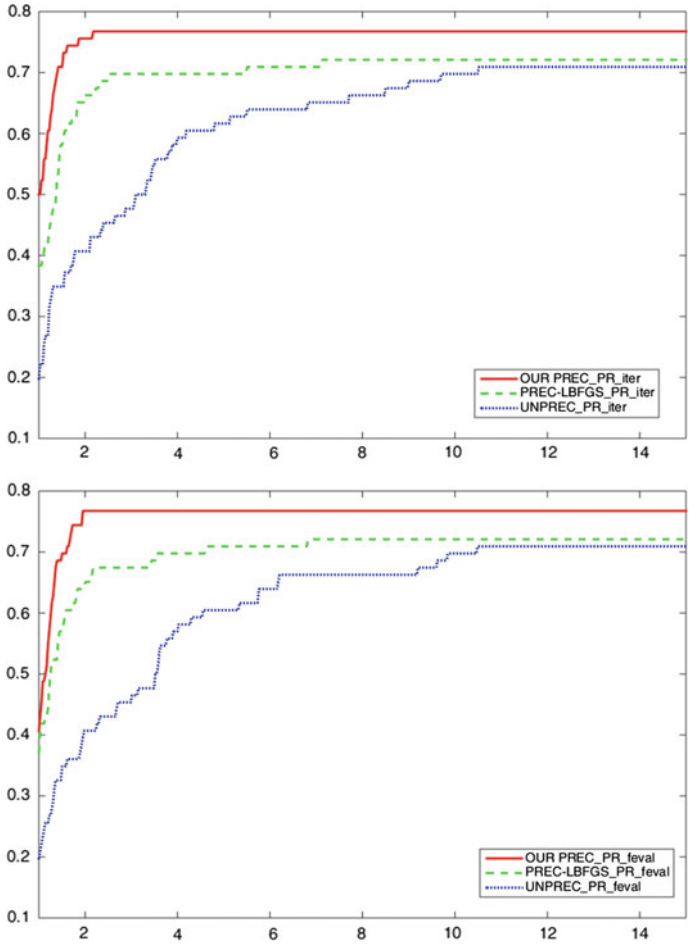


Fig. 1 Performance profiles using the *original* stopping criterion $\|g_k\|_\infty \leq 10^{-5}(1 + |f_k|)$ in the code CG+ [24], adopting PR and with respect to # $iterations$ (up) and # $function\ evaluations$ (down)

with $\gamma_k, \omega_k \in \mathbb{R} \setminus \{0\}$, and where, given M_k and the vector p_k generated by NCG, we have for v_k the expression

$$v_k = \sigma_k \left(s_k - \delta_k M_k y_k - \omega_k p_k\right), \qquad \sigma_k \in \{-1, +1\}.$$

The proposal in (7) follows a different strategy with respect to (5), inasmuch as it more closely attempts to emulate quasi-Newton updates. Indeed, similar to (5) also in (7) M_{k+1} includes three contributions, being the rightmost term $\omega_k p_k p_k^T / y_k^T p_k$ built using information collected at iteration k of the NCG method, and the leftmost term $\delta_k M_k$ being representative of the preconditioner at the previous iteration. Finally, the

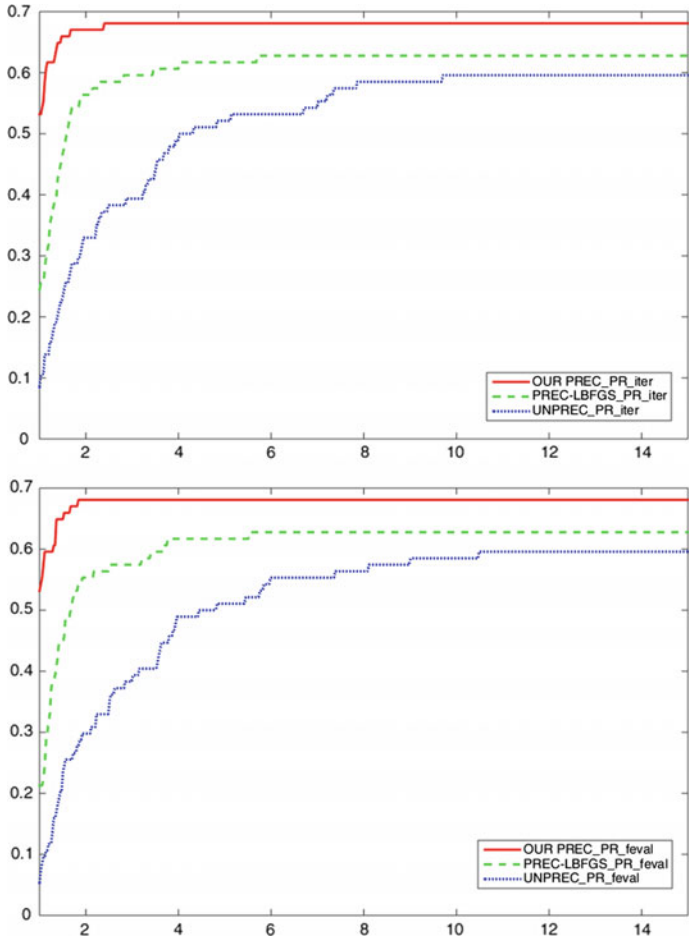


Fig. 2 Profiles using the *novel* stopping criterion (6), adopting PR and with respect to # $iterations$ (up) and # $function\ evaluations$ (down)

term $\gamma_k v_k v_k^T$ in (7) is introduced so that M_{k+1} can explicitly satisfy the secant equation $M_{k+1} y_k = s_k$. The latter considerations confirm that, similar to BFGS update, the dyad $\omega_k p_k p_k^T / y_k^T p_k$ aims at adding the most recent information from NCG to our current preconditioner.

The next couple of theoretical results can also be proved for the proposal (7), confirming to what extent (7) closely resembles quasi-Newton approaches (see [14] for details).

Proposition 2 *Let $f(x) = \frac{1}{2} x^T A x - b^T x$, where A is a symmetric matrix. Suppose k steps of the NCG method are performed, adopting an exact linesearch procedure (which imposes $\nabla f(x_{j+1})^T p_j = 0,\ j = 1, \ldots, k$), in order to detect the stationary point (if any) of the function f. Then, the matrix M_{k+1} in (7) satisfies the modified*

secant equations

$$\begin{cases} M_{k+1} y_j = \delta_j s_j, \qquad \delta_j > 0, \qquad j = 1, \ldots, k-1, \\ M_{k+1} y_k = s_k, \end{cases} \tag{8}$$

provided that the nonzero coefficients γ_j, ω_j, $j = 1, \ldots, k$ *are chosen such that*

$$\begin{cases} \gamma_j = \dfrac{1}{s_j^T y_j - \delta_j y_j^T M_j y_j - \omega_j p_j^T y_j}, \ j = 1, \ldots, k, \\ \omega_j \neq \dfrac{s_j^T y_j - \delta_j y_j^T M_j y_j}{p_j^T y_j}, \qquad\qquad j = 1, \ldots, k. \end{cases} \tag{9}$$

Proposition 2 reveals to what extent the matrix M_{k+1} substantially summarizes some second-order information on the objective function $f(x)$. In particular, by (8), the secant equation at the current iterate x_k is fulfilled, while a weaker condition holds at the previous iterates, being possibly $\delta_j \neq 1$, for $j = 1, \ldots, k-1$. Also, note that the choice of the parameters $\{\delta_j\}$, $\{\gamma_j\}$ and $\{\omega_j\}$ in Proposition 2 does not ensure in general the positive definiteness of M_{k+1}. Indeed, pre-multiplying the second relation in (8) by y_k, we obtain $y_k^T M_{k+1} y_k = y_k^T s_k$, where the right-hand side might be possibly negative, inasmuch as no Wolfe conditions were adopted in Proposition 2 when applying the NCG. On this guideline, the next result helps recover the positive definiteness of the preconditioner M_{k+1} (see [14]).

Proposition 3 *Let* f *be a continuously differentiable function. Suppose that the NCG method is used to minimize the function* f. *Suppose that* $s_k^T y_k > 0$, $M_k \succ 0$, $\varepsilon_k \in (0, 1)$ *and*

$$\begin{aligned} 0 < \delta_k &= (1 - \varepsilon_k) \frac{s_k^T y_k}{y_k^T M_k y_k}, \\ 0 < \omega_k &< \varepsilon_k \alpha_k, \\ 0 < \gamma_k &= \frac{1}{(\varepsilon_k \alpha_k - \omega_k) p_k^T y_k}. \end{aligned}$$

Then conditions (8)–(9) hold and $M_{k+1} \succ 0$ *in (7).*

By Proposition 3, a suitable interval of values for δ_k, γ_k, and ω_k always exists such that (8)–(9) hold and $M_{k+1} \succ 0$, even though an inexact linesearch procedure is adopted (but not necessary the Wolfe linesearch procedure). Moreover, the hypothesis $M_k \succ 0$ might be too restrictive to our purposes, and we can easily prove that what really matters is the weaker condition $y_k^T M_k y_k > 0$ along with the inequality $y_k^T s_k > 0$.

By Proposition 2, we have also a remarkable result in case the objective function $f(x)$ is quadratic. Indeed, after n steps, the matrix M_{n+1} retains information on the

inertia of the Hessian matrix, as in the next corollary (see [14]), where $\lambda_m(\cdot)$ and $\lambda_M(\cdot)$ represent, respectively, the smallest and the largest eigenvalue.

Corollary 1 *Let $f(x) = \frac{1}{2}x^T Ax - b^T x$, where A is symmetric and nonsingular. Suppose that n steps of the CG are performed, in order to detect the stationary point of the function f, and that the vectors $p_1, \ldots, p_n$ are generated.*

(i) If (8)–(9) hold, we have

$$M_{n+1}A = (s_1 \cdots s_n)D(s_1 \cdots s_n)^{-1},$$

with

$$D = diag\{\delta_1, \delta_2, \ldots, \delta_{n-1}, 1\}.$$

(ii) It results

$$\lambda_m(M_{n+1}A) = \lambda_m(D), \qquad \lambda_M(M_{n+1}A) = \lambda_M(D). \tag{10}$$

Several interesting conclusions arise considering the two proposals in Sects. 3 and 4 for M_{k+1}; we urge to carry out the following observations, which are also the result of a deeper investigation not reported here:

- Both the proposals for the preconditioner M_{k+1} are based on the attempt to emulate the BFGS update, in order to possibly benefit from some of its well-known features (i.e., the satisfaction of the secant equation and BFGS attitude to approximate the inverse Hessian in the quadratic case).
- while the scheme in (5) details an update based on $m+1$ pairs (s_j, y_j), $j = k-m, \ldots, k$, provided by the NCG method, the scheme in (7) simply relies on the pair (p_k, y_k) generated at step k of the NCG method.
- the proposal in (7) seems to be endowed with stronger theoretical properties with respect to (5). As also shown in the next sections, the latter fact is also reflected in an appreciable enhancement of numerical performance, over a significant large-scale test set. Indeed, comparing the proposals in Sects. 3 and 4, over the same test set specified in Sect. 3, we obtain the performance profiles in Fig. 3, using (6) for termination which is the same as that used for obtaining Fig. 2.

5 Damped Strategies for NCG Preconditioning

Damped techniques were introduced in the framework of quasi-Newton methods, and their rationale can be summarized as follows. As is well known (see, e.g., [1]), when dealing with the BFGS update, a crucial issue in order to guarantee the positive definiteness of the updated Hessian approximation is the satisfaction of the curvature condition

$$s_k^T y_k > 0. \tag{11}$$

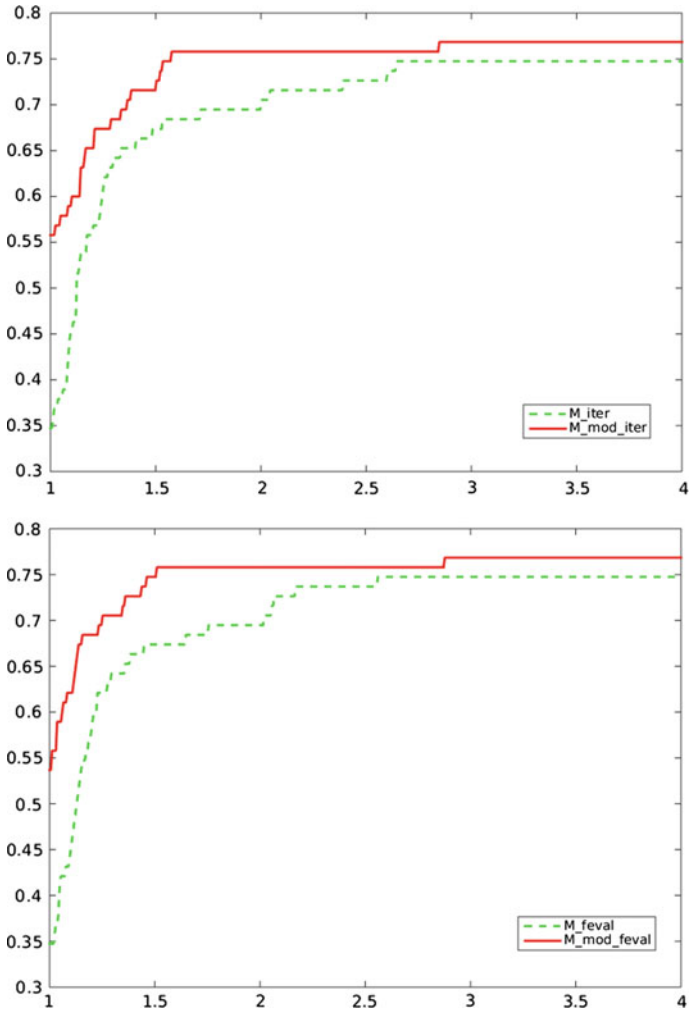


Fig. 3 Comparison between the proposal of preconditioner in (5) (namely M, *dash line*) and the proposal in (7) (namely M_mod, *solid line*), using the stopping criterion (6). Profiles with respect to # *iterations* (up) and # *function evaluations* (down)

In case f is strongly convex, then (11) holds for any pair of points x_k and x_{k+1} (see, e.g. [25]). In case of nonconvex functions, imposing the satisfaction of condition (11) requires a proper choice of the stepsize α_k, from the linesearch procedure adopted. Indeed, in principle, the satisfaction of (11) can always be obtained by a suitable linesearch procedure, provided that the objective function is bounded below. To this aim, as mentioned above, the Wolfe conditions (in practice, the strong Wolfe conditions) are usually adopted, which ensure the fulfillment of condition (11). However, for sufficiently large value of c_2, the value of $s_k^T y_k$ may not be sufficiently positive. In

addition, if only the backtracking linesearch framework is employed, the curvature condition (11) may not hold.

A possible successful strategy to cope with the last issue is to adopt the *damped technique* proposed by Powell in [19], in the context of SQP Lagrangian BFGS methods for constrained optimization and applied for the first time by Al-Baali [26] to unconstrained optimization. In [19], the author proposes to modify the difference of the gradients vector y_k in (11), before performing the BFGS update. Namely, if B_k denotes the current BFGS positive definite Hessian approximation at kth iteration, the following modified (damped) vector is used in place of y_k:

$$\widehat{y}_k = \varphi_k y_k + (1 - \varphi_k) B_k s_k, \tag{12}$$

where φ_k is chosen in $(0, 1]$ such that $s_k^T \widehat{y}_k$ is "sufficiently positive." The latter fact guarantees that the use of the damped vector $\hat{y}_k$ is in principle preferable with respect to y_k. In particular, given $\sigma \in (0, 1]$, the value of the parameter φ_k is often set according with the rule:

$$\varphi_k = \begin{cases} \dfrac{\sigma s_k^T B_k s_k}{s_k^T B_k s_k - s_k^T y_k}, & \text{if} \quad s_k^T y_k < (1 - \sigma) s_k^T B_k s_k, \\ 1, & \text{otherwise}, \end{cases} \tag{13}$$

which for $\sigma = 0.8$ yields that in Sect. 18.3 in [1]. There are several reasons which motivate (13), including the fact that by this choice we have

$$s_k^T \widehat{y}_k = (1 - \sigma) s_k^T B_k s_k, \tag{14}$$

i.e., the quantity $s_k^T \widehat{y}_k$ is sufficiently positive, inasmuch as B_k is positive definite. Al-Baali suggests using the modified damped vector (12) with (13) for unconstrained optimization and extended it to

$$\varphi_k = \begin{cases} \dfrac{\sigma_1 s_k^T B_k s_k}{s_k^T B_k s_k - s_k^T y_k}, & \text{if} \quad s_k^T y_k < (1 - \sigma_1) s_k^T B_k s_k, \\ \dfrac{\sigma_2 s_k^T B_k s_k}{s_k^T B_k s_k - s_k^T y_k}, & \text{if } s_k^T y_k > (1 + \sigma_2) s_k^T B_k s_k, \\ 1, & \text{otherwise}, \end{cases} \tag{15}$$

where $\sigma_1 \in (0, 1]$ and $\sigma_2 \geq 2$. Note that the value $\sigma_2 = \infty$ reduces choice (15) to (13).

In [18], in order to extend the definition of the damped vector $\widehat{y}_k$ in (12), a novel vector $\widehat{y}_k$ is defined as a combination of the original vector y_k and an appropriate vector z_k, namely

$$\widehat{y}_k = \varphi_k y_k + (1 - \varphi_k) z_k, \tag{16}$$

(see also [27]). The vector z_k plays a noteworthy role to ensure that $s_k^T \widehat{y}_k$ is *sufficiently positive*, for suitable values of $\varphi_k \in (0, 1]$. Of course, a key point of this approach is an appropriate choice of z_k. Two choices for z_k have been proposed in [18].

The first proposal corresponds to set $z_k = \eta_k s_k$, where $\eta_k > 0$, based on approximating B_k by $\eta_k I$. This choice originates from the idea of using $z_k = A_{k+1} y_k$ in (16), where A_{k+1} is a positive definitive approximation of the inverse Hessian. In particular, $B_k \approx \eta_k I$ satisfies the *modified secant equation*

$$A_{k+1} y_k = \eta_k s_k.$$

Hence, by using the latter equation, we can set

$$\widehat{y}_k^{(a)} = \varphi_k y_k + (1 - \varphi_k) \eta_k s_k. \tag{17}$$

Interesting properties of (17) are that it does not require the explicit knowledge of the approximate inverse Hessian matrix A_{k+1} and that

$$s_k^T \widehat{y}_k^{(a)} = (1 - \sigma_1) \eta_k \|s_k\|^2 > 0, \tag{18}$$

for appropriate choice of the parameter in (16). This condition may be of great interest if we consider a geometric interpretation of the curvature condition (11). Indeed, since for the vector $\hat{y}_k^{(a)}$ condition (18) is satisfied, it means that $s_k^T \hat{y}_k^{(a)}$ is always *sufficiently positive*. Moreover, it can be easily proved that for proper choices of the parameters η_k and σ, we obtain (as long as (11) holds)

$$s_k^T \widehat{y}_k^{(a)} \geq s_k^T \widehat{y}_k. \tag{19}$$

Furthermore, also in case (11) does not hold, by relation (18), we immediately infer that again (19) holds.

The second proposal corresponds to set in (16) $z_k = -\alpha_k g_k$, so that the novel damped vector becomes

$$\widehat{y}_k^{(b)} = \varphi_k y_k - (1 - \varphi_k) \alpha_k g_k. \tag{20}$$

This choice of z_k comes from the following observation: If $B_k \succ 0$ is an approximation of the Hessian and we consider $-B_k^{-1} g_k$ as search direction, it immediately follows that

$$s_k = x_{k+1} - x_k = -\alpha_k B_k^{-1} g_k,$$

which implies

$$B_k s_k = -\alpha_k g_k.$$

This formula allows to compute the original damped vector (12) without explicitly using the matrix B_k. Indeed, it suffices to replace $B_k s_k$ with $-\alpha_k g_k$ in (12), according with (20).
Similar to the choice $\widehat{y}_k^{(a)}$, also for $\widehat{y}_k^{(b)}$ in (20), we can guarantee that $s_k^T \widehat{y}_k^{(b)}$ is sufficiently positive. In fact, we immediately have from (14)

$$s_k^T \widehat{y}_k^{(b)} = -\alpha_k(1-\sigma_1) s_k^T g_k = -\alpha_k^2(1-\sigma_1) p_k^T g_k > 0,$$

where the last inequality holds because p_k is a descent direction. Several theoretical properties can be proved for the choices (17) and (20) (see also [18]). Some of them are summarized here below, where we assume that the coefficient β_k in PNCG is replaced by the PR–type 'damped coefficient'

$$\hat{\beta}_k^{PR} = \frac{\left(\widehat{y}_k^{(a)}\right)^T M_{k+1} g_{k+1}}{g_k^T M_k g_k}$$

(the resulting PNCG scheme, with $\widehat{y}_k^{(a)}$ in place of y_k will be addressed as D-PR-PNCG).

Assumption 1 (*see* [18])

(a) Given the initial point x_1 and the function $f \in C^1$, the level set $\mathscr{L}_1 = \{x : f(x) \le f_1\}$ is compact.
(b) There exists an open ball $\mathscr{B}_r := \{x : \|x\| < r\}$ containing $\mathscr{L}_1$ where $f(x)$ is continuously differentiable and its gradient $g(x)$ is Lipschitz continuous. In particular, there exists $L > 0$ such that

$$\|g(x) - g(y)\| \le L\|x - y\| \qquad \text{for all} \qquad x, y \in \mathscr{B}_r.$$

(c) There exist $\lambda > 0$ and $\Lambda > 0$ such that the preconditioner $M(x)$, for any $x \in \mathscr{B}_r$, is positive definite with the smallest [largest] eigenvalue $\lambda_m(M(x))$ [$\lambda_M(M(x))$] satisfying

$$0 < \lambda \le \lambda_m(M(x)) \le \lambda_M(M(x)) < \Lambda.$$

Proposition 4 *Let $\{x_k\}$ be an infinite sequence (with $g_k \neq 0$) generated by the D-PR-PNCG method, where the steplength $\alpha_k > 0$ is determined by a linesearch procedure such that, for all k, the following conditions hold:*

(i) $x_k \in \mathscr{L}_1$ *for all* k;
(ii) $\lim_{k\to+\infty} \dfrac{|g_k^T p_k|}{\|p_k\|} = 0$;
(iii) $\lim_{k\to+\infty} \alpha_k \|p_k\| = 0$.

If Assumption 1 holds, then

$$\liminf_{k \to +\infty} \|g_k\| = 0$$

and hence there exists at least a stationary limit point of $\{x_k\}$.

Similar to the proposals in Sects. 3 and 4, we consider now a brief numerical experience on the use of the damped vectors in (17) and (20). A complete study can be found in [18]. Observe that in principle the use of damped techniques fully affects the preconditioning strategies (where y_k is replaced by $\hat{y}_k^{(a)}$ or $\hat{y}_k^{(b)}$), i.e., both the value of β_k along with the preconditioner, and not just the value of β_k. However, our

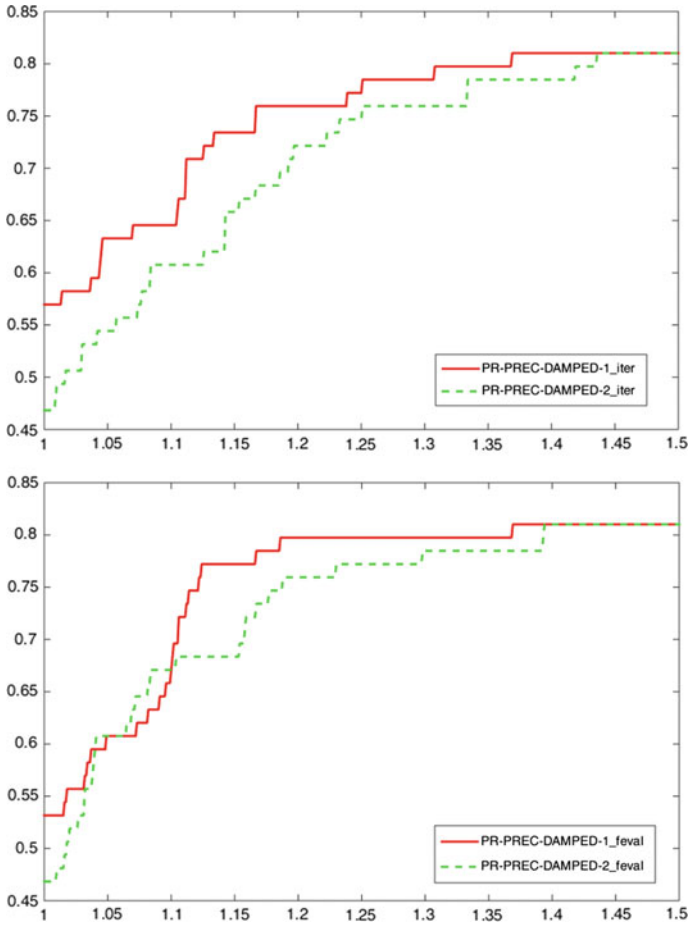


Fig. 4 Comparison between the adoption of the two damped strategies in (17) and in (20). Profiles with respect to # *iterations* (up) and # *function and gradient evaluations* (down)

preliminary aim here is to report a numerical experience with PNCG (and not D-PR-PNCG), i.e., embedding the damped techniques within the preconditioner used in a PNCG scheme, where the standard Polak–Ribière (PR) formula for β_k is used. In particular, the same settings used in Sects. 3 and 4, along with the same test set are considered. We also recall that a standard implementation of the PNCG method in CG+ code was adopted (see [24]), where the preconditioner (5) is included, and the linesearch technique is the same as that in [28]. Finally, the stopping criterion adopted is the standard one in (6). We also recall that in the linesearch procedure adopted in [28] the number of function and gradient evaluations coincide. In Fig. 4, the two damped strategies in (17) (with $\eta_k = 4$ and φ_k chosen as in (13)) and in (20)

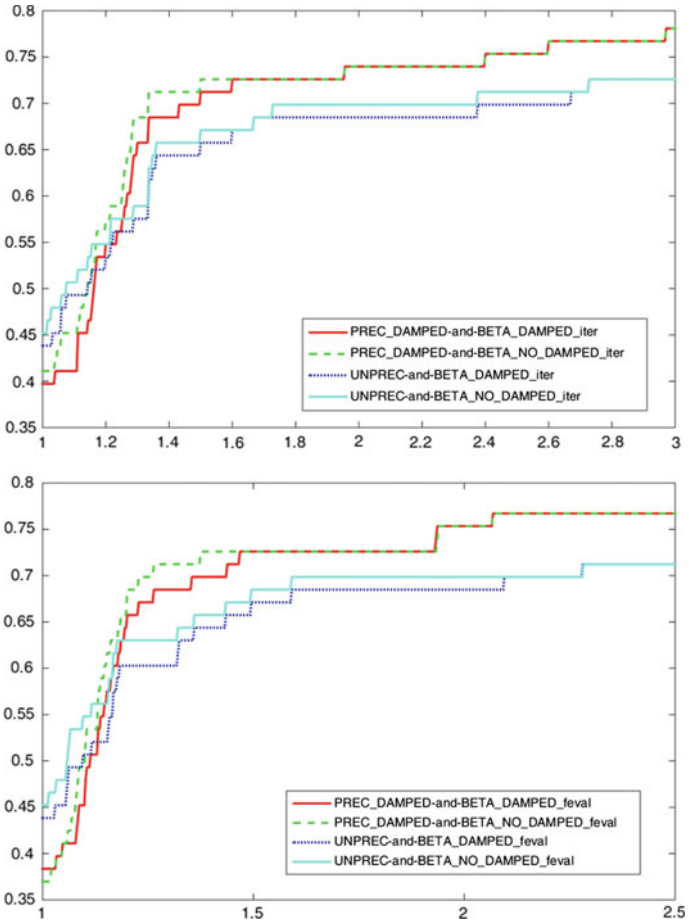


Fig. 5 Comparison between the use $\widehat{\beta}_k^{PR}$ (setting $\widehat{y}_k = \widehat{y}_k^{(a)}$) and β_k^{PR}, in both preconditioned and unpreconditioned cases. Profiles with respect to # *iterations* (up) and # *function and gradient evaluations* (down)

(with φ_k chosen as in (13)) are compared, with respect to both # iterations and # function evaluations. The strategy (17) seems to be somehow preferable to (20).

To complete our analysis, we note that a full information from damped techniques can be used, both affecting the computation of the coefficient β_k and the preconditioner M_{k+1} in PNCG (see [18]). More explicitly, the performances of PNCG vs. D-PR-PNCG (where $\hat{\beta}_k^{PR}$ is used in place of β_k^{PR}) in both the preconditioned and unpreconditioned case are compared. The corresponding results are summarized in Fig. 5 (names of the schemes are self-explanatory). As it can be observed from the profiles, the use of $\hat{\beta}_k^{PR}$ does not yield a noteworthy improvement. Nevertheless, we also observe that the D-PR-PNCG scheme, which also uses $\hat{\beta}_k^{PR}$, reveals to outperform the standard NCG method.

Acknowledgements The research is partially supported by the Italian Flagship Project RITMARE, coordinated by the Italian National Research Council and funded by the Italian Ministry of Education, University and Research.

References

1. Nocedal, J., Wright, S.J.: Numerical Optimization, 2nd edn. Springer, New York (2006)
2. Conn, A.R., Gould, N.I.M., Toint, PhL: Trust region methods. MOS-SIAM Series on Optimization, Philadelphia (2000)
3. Hager, W., Zhang, H.: A survey of nonlinear conjugate gradient methods. Pac. J. Optim. **2**, 35–58 (2006)
4. Hager, W., Zhang, H.: A new conjugate gradient method with guaranteed descent and efficient line search. SIAM J. Optim. **16**, 170–192 (2005)
5. Hager, W., Zhang, H.: The limited memory conjugate gradient method. SIAM J. Optim. **23**, 2150–2168 (2013)
6. Raydan, M.: The Barzilai and Borwein gradient method for large scale unconstrained minimization problems. SIAM J. Optim. **7**, 26–33 (1997)
7. Fletcher, R., Reeves, C.: Function minimization by conjugate gradients. Comput. J. **7**, 149–154 (1964)
8. Polak, E., Ribière, G.: Note sur la convergence de methodes de directions conjuguees. Rev. Fr. d'Informatique Rech. Operationnelle serie rouge **3**(1), 35–43 (1969)
9. Hestenes, M., Stiefel, E.: Methods of conjugate gradients for solving linear systems. J. Res. Natl. Bur. Stand. **49**, 409–436 (1952)
10. Dai, Y., Yuan, Y.: A nonlinear conjugate gradient method with a strong global convergence property. SIAM J. Optim. **10**, 177–182 (1999)
11. Andrei, N.: An adaptive conjugate gradient algorithm for large-scale unconstrained optimization. J. Comput. Appl. Math. **292**, 83–91 (2016)
12. Caliciotti, A., Fasano, G., Roma, M.: Novel preconditioners based on quasi-Newton updates for nonlinear conjugate gradient methods. Optim. Lett. **11**, 835–853 (2017)
13. Caliciotti, A., Fasano, G., Roma, M.: Preconditioning strategies for nonlinear conjugate gradient methods, based on quasi-Newton updates. AIP Conf. Proc. Am. Inst. Phys. [Sergeyev Y.D., Kvasov D.E., Dell'Accio F., Mukhametzhanov M.S. (eds.)] **1776**(090007):1–4 (2016)
14. Caliciotti, A., Fasano, G., Roma, M.: Preconditioned nonlinear Conjugate Gradient methods based on a modified secant equation. Appl. Math. Comput. **318**, 196–214 (2018)
15. Andrei, N.: Scaled memoryless BFGS preconditioned conjugate gradient algorithm for unconstrained optimization. Optim. Methods Softw. **22**, 561–571 (2007)

16. Morales, J.L., Nocedal, J.: Automatic preconditioning by limited memory quasi-Newton updating. SIAM J. Optim. **10**, 1079–1096 (2000)
17. Gratton, S., Sartenaer, A., Tshimanga, J.: On a class of limited memory preconditioners for large scale linear systems with multiple right-hand sides. SIAM J. Optim. **21**, 912–935 (2011)
18. Al-Baali, M., Caliciotti, A., Fasano, G., Roma, M.: Exploiting damped techniques for nonlinear conjugate gradient methods, to appear on Mathematical Methods of Operations Research (2017). https://doi.org/10.1007/s00186-017-0593-1
19. Powell, M.J.D.: Algorithms for nonlinear constraints that use Lagrangian functions. Math. Program. **14**, 224–248 (1978)
20. Al-Baali, M., Grandinetti, L.: On practical modifications of the quasi-Newton BFGS methods. AMO Adv. Model. Optim. **11**, 63–76 (2009)
21. Al-Baali, M., Grandinetti, L., Pisacane, O.: Damped techniques for the limited memory BFGS method for large-scale optimization. J. Optim. Theory Appl. **161**, 688–699 (2014)
22. Fasano, G., Roma, M.: Preconditioning Newton-Krylov methods in nonconvex large scale optimization. Comput. Optim. Appl. **56**, 253–290 (2013)
23. Gould, N.I.M., Orban, D., Toint, PhL: CUTEst: a constrained and unconstrained testing environment with safe threads. Comput. Optim. Appl. **60**, 545–557 (2015)
24. Gilbert, J.C., Nocedal, J.: Global convergence properties of conjugate gradient methods for optimization. SIAM J. Optim. **2**, 21–42 (1992)
25. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, New York (2004)
26. Al-Baali, M.: Damped techniques for enforcing convergence of quasi-Newton methods. Optim. Methods Softw. **29**, 919–936 (2014)
27. Al-Baali, M.: Quasi-Newton algorithms for large-scale nonlinear least-squares. In: Di Pillo, G., Murli, A. (eds.) High Performance Algorithms and Software for Nonlinear Optimization, pp. 1–21. Kluwer Academic, Dordrecht (2003)
28. Moré, J., Thuente, D.: Line search algorithms with guaranteed sufficient decrease. ACM Trans. Math. Softw. (TOMS) **20**, 286–307 (1994)

Solving Minimax Problems: Local Smoothing Versus Global Smoothing



A. M. Bagirov, N. Sultanova, A. Al Nuaimat and S. Taheri

Abstract The aim of this chapter is to compare different smoothing techniques for solving finite minimax problems. We consider the local smoothing technique which approximates the function in some neighborhood of a point of nondifferentiability and also global smoothing techniques such as the exponential and hyperbolic smoothing which approximate the function in the whole domain. Computational results on the collection of academic test problems are used to compare different smoothing techniques. Results show the superiority of the local smoothing technique for convex problems and global smoothing techniques for nonconvex problems.

Keywords Nonsmooth optimization · Nonlinear programming
Smoothing techniques · Minimax problems

1 Introduction

In this chapter, we compare different smoothing techniques to solve the following optimization problem:

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

where

$$f(x) = \max_{i \in I} f_i(x), \quad I = \{1, \ldots, m\} \tag{2}$$

and the functions f_i, $i \in I$ are continuously differentiable. The problem (1) is called the *finite minimax problem*.

The objective function f in Problem (1) is Clarke regular, and its subdifferential at a point $x \in \mathbb{R}^n$ can be expressed as [1, 2]:

A. M. Bagirov (✉) · N. Sultanova · A. Al Nuaimat · S. Taheri
Faculty of Science and Technology, Federation University Australia,
Ballarat, VIC, Australia
e-mail: a.bagirov@federation.edu.au

© Springer International Publishing AG, part of Springer Nature 2018

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*,
Springer Proceedings in Mathematics & Statistics 235,
https://doi.org/10.1007/978-3-319-90026-1_2

$$\partial f(x) = \text{conv}\,\{\nabla f_i(x) : \ i \in R(x)\}\,.$$

Here, "*conv*" is a convex hull of a set, and the set $R(x)$ is defined as:

$$R(x) = \{i \in I : \ f_i(x) = f(x)\}.$$

A point $x \in \mathbb{R}^n$ is called a stationary point for Problem (1) if $0_n \in \partial f(x)$.

The problem (1) can be solved by applying algorithms specifically developed for it (see, for example, [3, 4]) or by applying conventional nonsmooth optimization algorithms such as the bundle and subgradient methods [2, 5–8]. Alternatively, Problem (1) can be reformulated as a nonlinear programming problem, and one can apply powerful nonlinear programming techniques to solve it.

Another approach for solving Problem (1) is to apply smoothing techniques to approximate the objective function using smooth functions. This approximation is controlled by the precision parameter(s). Then, the finite minimax problem is replaced by a sequence of smooth problems. Such an approach allows one to apply powerful smooth optimization algorithms for solving Problem (1).

Smoothing techniques have been widely used to solve nonlinear programming problems via the exact penalty function method [9], the finite maximin problems in nonsmooth optimization [10–12], semi-infinite minimax problems [13, 14], complementarity and variational inequality problems [15–18], and constrained nonlinear least squares problems [19]. Such techniques have also been applied to solve nonsmooth optimization problems in machine learning and regression analysis [20–22].

Smoothing techniques, without any loss of generality, can be divided into two classes: local and global smoothing techniques. Smoothing techniques from the first class try to smooth the objective function in some neighborhood of the so-called kink points (points where the function f is not differentiable). Such techniques were considered, for example, in [23–25].

Smoothing techniques from the second class approximate the objective function f globally, that is in the whole domain of the maximum function. This class includes, in particular, the exponential [11, 26, 27] and hyperbolic smoothing [28–31] techniques. Another global smoothing technique is considered in [32], where the logarithmic barrier function of the epigraph of the maximum function is used to smooth it.

In this chapter, we consider one local smoothing method for solving Problem (1) and compare it with two global smoothing techniques. Contributions of this chapter are: (i) the study of differential properties of the smooth functions approximating the maximum function (2) based on the local smoothing technique; (ii) the design of a minimization algorithm based on such a smoothing technique and the study of its convergence; and (iii) the comparison of the local smoothing technique with the exponential and hyperbolic smoothing methods using results of numerical experiments.

The rest of the chapter is organized as follows. Global smoothing methods are described in Sect. 2. A local smoothing method and its properties are studied in Sect. 3. Section 4 provides the evaluation and comparison of global and local smoothing methods. Section 5 contains some concluding remarks.

2 Global Smoothing Methods

To date, different methods have been developed to smooth nonsmooth functions. Some of these methods were developed specifically to smooth the maximum function (2). There are also some methods for smoothing general convex functions and locally Lipschitz functions.

In the paper [33], the author introduces a technique for smoothing nonsmooth convex functions. The Steklov averaged function, considered in [34], can be applied for smoothing general locally Lipschitz functions. The paper [35] proposes an approach to smooth the function defined in (2).

In this section, we briefly describe the Steklov averaged function, which is used to smooth locally Lipschitz functions, as well the exponential and hyperbolic smoothing techniques for minimax problems. Last two techniques smooth corresponding nonsmooth functions globally. The Steklov averaged function can also be considered as a global smoothing technique as this function is defined using the values of the function f over the whole space $\mathbb{R}^n$; however, the use of mollifiers makes it local smoothing technique.

The Steklov smoothing is the most general and applicable to any locally Lipschitz function and does not require an existence of any structure of a function. Let $f : \mathbb{R}^n \to \mathbb{R}$ be a locally Lipschitz function. Consider the function:

$$f_\alpha(x) = \int_{\mathbb{R}^n} f(x - y)\psi_\alpha(y)dy.$$

The function $\psi_\alpha : \mathbb{R}^n \to \mathbb{R}$, called the *Steklov mollifier*, is defined by

$$\psi_\alpha(y) = \begin{cases} 1/\alpha^n & y \in [-\alpha/2, \alpha/2]^n, \\ 0 & \text{otherwise}, \end{cases}$$

where $\alpha > 0$ is a smoothing parameter. The function f_α is called the *Steklov averaged* function. This function can be rewritten as:

$$f_\alpha(x) = \frac{1}{\alpha^n} \int_{x_1-\alpha/2}^{x_1+\alpha/2} \cdots \int_{x_n-\alpha/2}^{x_n+\alpha/2} f(y)dy_1 \ldots dy_n.$$

Even though the Steklov averaged function is applicable to smooth any locally Lipschitz function, the calculation of its values is not always easy. Therefore, numerical algorithms based on this technique are rather conceptual.

The exponential and hyperbolic smoothing techniques require the structure of a problem to be known, and therefore, they are applicable to a smaller class of problems than the Steklov averaged function. However, algorithms based on these techniques are more efficient and easily implementable.

The exponential smoothing of the function f, defined in (2), is given as [9, 11, 26, 36]:

$$F_p(x) = \frac{1}{p} \log \sum_{i \in I} e^{pf_i(x)}.$$

Here, $p > 0$ is a smoothing parameter. The function F_p can be rewritten as

$$F_p(x) = f(x) + \frac{1}{p} \log \sum_{i \in I} e^{p(f_i(x) - f(x))}.$$

It is well known that [11, 26, 36]

$$f(x) \leq F_p(x) \leq f(x) + \frac{\ln m}{p}, \ x \in \mathbb{R}^n.$$

Different algorithms were developed based on the exponential smoothing technique [11, 26, 27, 36]. These algorithms differ from each other in the way of solving smooth optimization problems, approximating the minimax problem and updating the smoothing parameter.

The hyperbolic smoothing is primarily defined for the function:

$$\theta(x) = \max\{0, x\}, \ x \in \mathbb{R}. \tag{3}$$

This function can be approximated using the following smooth function [29, 31]:

$$\Phi_\tau(x) = \frac{1}{2}\left(x + \sqrt{x^2 + \tau^2}\right),$$

where $\tau > 0$ is a smoothing (or precision) parameter. The function Φ_τ is an increasing C^∞-function and

$$\theta(x) < \Phi_\tau(x) \leq \theta(x) + \tau/2 \ \ \forall x \in \mathbb{R}.$$

The hyperbolic smoothing of the function f, defined in (2), is studied in [28] where this function is reformulated by introducing one additional variable.

To the best of our knowledge, the paper [31] was the first where the hyperbolic smoothing function was considered. In the paper [30], the problem of the optimal covering of plane domains by circles was solved by applying a hyperbolic smoothing technique. Later, this technique was applied to solve the cluster analysis problem using its nonsmooth optimization formulation [29].

3 Local Smoothing Method

In this section, we consider one local smoothing technique, design an algorithm for solving the finite minimax problems based on this technique and study the convergence of this algorithm.

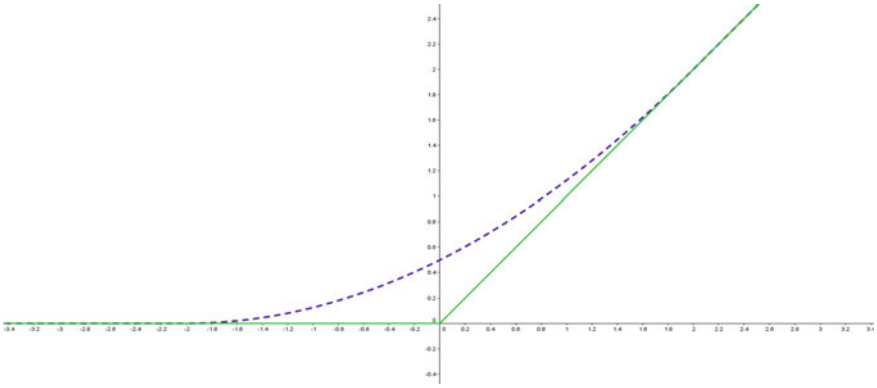


Fig. 1 Local smoothing of the function (3) with $\delta = 2$

Local smoothing techniques were applied to design algorithms for solving the finite minimax problems. The paper [25] introduces a local smoothing function using a rather complex reformulation of the maximum function. Another reformulation of this function is considered in [24], where it is replaced by the sum of more simple maximum functions. Then, the reformulated function is approximated by smooth functions in some neighborhood of kink points.

Consider the function θ defined in (3). This function can be approximated by the following function [23]:

$$\phi_\delta(x) = \begin{cases} 0 & x < -\delta, \\ \dfrac{(x+\delta)^2}{4\delta} & -\delta \le x \le \delta, \\ x & x > \delta, \end{cases} \tag{4}$$

where $\delta > 0$ is a smoothing parameter. The local smoothing function for the function (3) is illustrated in Fig. 1 where the dashed curve shows the smoothing function.

Proposition 1 *The function $\phi_\delta(x)$ has the following properties:*

1. *$\phi_\delta(\cdot)$ is an increasing convex continuously differentiable function;*
2. $\theta(x) < \phi_\delta(x) \le \theta(x) + \dfrac{\delta}{4}\ \forall x \in \mathbb{R}$.

Proof The proof is straightforward. □

3.1 Reformulation of Minimax Problems

The function ϕ_δ can also be applied to smooth the function f, given by (2), after some reformulations. Let $t \in \mathbb{R}$. Consider the following function:

$$F(x,t) = t + \sum_{i \in I} \max\{0,\ f_i(x) - t\}. \tag{5}$$

For a given $(x,t) \in \mathbb{R}^n \times \mathbb{R}$, the index set I is represented as follows:

$$I = J_1 \cup J_2 \cup J_3,$$

where

$$\begin{aligned} J_1 &\equiv J_1(x,t) = \{i \in I : f_i(x) - t < 0\}, \\ J_2 &\equiv J_2(x,t) = \{i \in I : f_i(x) - t = 0\}, \\ J_3 &\equiv J_3(x,t) = \{i \in I : f_i(x) - t > 0\}. \end{aligned}$$

Denote by $\Psi_i(x,t) = \max\{0,\ f_i(x) - t\},\ i \in I$. Then, for the subdifferential of the function Ψ_i at the point (x,t), we have

$$\partial \Psi_i(x,t) = \begin{cases} \{0_{n+1}\}, & i \in J_1, \\ \text{conv}\,\{0_{n+1}, (\nabla f_i(x), -1)\}, & i \in J_2, \\ \{(\nabla f_i(x), -1)\}, & i \in J_3. \end{cases}$$

Since functions $f_i,\ i \in I$ are Clarke regular, it follows that functions Ψ_i are also regular and, therefore, the subdifferential of the function F at the point (x,t) can be expressed as:

$$\partial F(x,t) = \{(0_n, 1)\} + \sum_{i \in J_2} \text{conv}\ \{0_{n+1}, (\nabla f_i(x), -1)\} + \left\{ \sum_{i \in J_3} (\nabla f_i(x), -1) \right\}. \tag{6}$$

The following propositions establish the relationship between functions f and F. Their proofs can be found in [28].

Proposition 2 *Suppose that functions f and F are defined by (2) and (5), respectively. Then,*

$$f(x) = \min_{t \in \mathbb{R}} F(x,t).$$

Proposition 3 *(1) Assume that a point $x^* \in \mathbb{R}^n$ is a stationary point of f. Then, $(x^*, t^*) \in \mathbb{R}^n \times \mathbb{R}$ is a stationary point of the function F where $t^* = f(x^*)$.*

(2) Assume that a point $(x^*, t^*) \in \mathbb{R}^n \times \mathbb{R}$ is a stationary point of the function F. Then, $x^* \in \mathbb{R}^n$ is a stationary point of f.

Proposition 4 *(1) Assume that a point $x^* \in \mathbb{R}^n$ is a local minimizer of f. Then, $(x^*, t^*) \in \mathbb{R}^n \times \mathbb{R}$ is a local minimizer of the function F where $t^* = f(x^*)$.*

(2) Assume that a point $(x^*, t^*) \in \mathbb{R}^n \times \mathbb{R}$ is a local minimizer of the function F. Then, $x^* \in \mathbb{R}^n$ is a local minimizer of f.

3.2 Local Smoothing of the Maximum Function

Now we can apply the local smoothing (4) to the function F. At a point $(x, t) \in \mathbb{R}^n \times \mathbb{R}$, consider the following index sets:

$$I_1 \equiv I_1(x, t, \delta) = \{i \in I : f_i(x) - t < -\delta\},$$
$$I_2 \equiv I_2(x, t, \delta) = \{i \in I : -\delta \le f_i(x) - t \le \delta\},$$
$$I_3 \equiv I_3(x, t, \delta) = \{i \in I : f_i(x) - t > \delta\}.$$

It is clear that $I = I_1 \cup I_2 \cup I_3$. Then, we get the following approximation of the function F:

$$\Phi_\delta(x, t) = t + \sum_{i \in I_2} \frac{(f_i(x) - t + \delta)^2}{4\delta} + \sum_{i \in I_3} (f_i(x) - t)\,. \tag{7}$$

Proposition 5 *For any $x \in \mathbb{R}^n$ and $t \in \mathbb{R}$*

$$0 < \Phi_\delta(x, t) - F(x, t) \le \frac{m\delta}{4}.$$

Proof The result follows from Proposition 1. □

The gradient of the function Φ_δ at $(x, t) \in \mathbb{R}^n \times \mathbb{R}$ is:

$$\nabla \Phi_\delta(x, t) = \left(\nabla_x \Phi_\delta(x, t), \Phi'_{\delta t}(x, t)\right), \tag{8}$$

where

$$\nabla_x \Phi_\delta(x, t) = \sum_{i \in I_2} \frac{(f_i(x) - t + \delta)}{2\delta} \nabla f_i(x) + \sum_{i \in I_3} \nabla f_i(x), \tag{9}$$

$$\Phi'_{\delta t}(x, t) = 1 - \sum_{i \in I_2} \frac{(f_i(x) - t + \delta)}{2\delta} - |I_3|. \tag{10}$$

Proposition 6 *Assume that sequences $\{x_k\}$, $\{t_k\}$ and $\{\delta_k\}$ are given such that $x_k \in \mathbb{R}^n$, $t_k \in \mathbb{R}$, $\delta_k > 0$ and $f(x_k) - t_k \le \delta_k$, $k = 1, 2, \ldots$. Moreover, assume that $x_k \to x$, $t_k \to t$, $\delta_k \downarrow 0$ as $k \to \infty$ and*

$$v = \lim_{k \to \infty} \nabla \Phi_{\delta_k}(x_k, t_k).$$

Then, $v \in \partial F(x, t)$.

Proof Since $t_k \ge f(x_k) - \delta_k$, we have $I_3(x_k, t_k, \delta_k) = \emptyset$ for all $k > 0$. Moreover, this condition and the continuity of functions f_i, $i \in I$ imply that $J_3(x, t) = \emptyset$.

We show that there exists $\bar{k}_0 > 0$ such that

$$J_1(x,t) \subseteq I_1(x_k, t_k, \delta_k). \tag{11}$$

Since $(x_k, t_k) \to (x,t)$ and $\delta_k \downarrow 0$ as $k \to \infty$ for any $\varepsilon > 0$, there exists $\bar{k}_\varepsilon > 0$ such that

$$|f_i(x_k) - f_i(x)| < \varepsilon,\ |t_k - t| < \varepsilon,\ \delta_k < \varepsilon \tag{12}$$

for all $k > \bar{k}_\varepsilon$. Take any $i \in J_1(x,t)$. Then, $f_i(x) - t < 0$. It follows from (12) that

$$f_i(x_k) - t_k + \delta_k < f_i(x) - t + 3\varepsilon.$$

Choose $\varepsilon = -(f_i(x) - t)/6 > 0$. Then, there exists $k_\varepsilon \geq \bar{k}_\varepsilon$ such that

$$f_i(x_k) - t_k + \delta_k < (f_i(x) - t)/2 < 0$$

for all $k > k_\varepsilon$. This means that $f_i(x_k) - t_k < -\delta_k$ for any $k > k_\varepsilon$ and, therefore, $i \in I_1(x_k, t_k, \delta_k)$. Replacing k_ε by $\bar{k}_0$, we get the proof of (11).

Next, we show that there exists $k_0 > 0$ such that

$$I_2(x_k, t_k, \delta_k) \subseteq J_2(x,t) \tag{13}$$

for all $k \geq k_0$. Assume the contrary, that is, for any $k_0 > 0$, there exists $k > k_0$ such that $I_2(x_k, t_k, \delta_k) \nsubseteq J_2(x,t)$. This means that $i \notin J_2(x,t)$ for some $i \in I_2(x_k, t_k, \delta_k)$. If $i \notin J_2(x,t)$, then either $i \in J_1(x,t)$ or $i \in J_3(x,t)$. In the first case, it follows from (11) that there exists $k_1 > 0$ such that $i \in I_1(x_k, t_k, \delta_k)$ for all $k > k_1$. This contradicts the fact that $i \in I_2(x_k, t_k, \delta_k)$ for infinitely many $k > \max\{k_0, k_1\}$. The second case can be considered in a similar way.

In order to get expressions for limits of gradients of the function Φ_δ, notice that $J_3(x,t) = \emptyset$, and therefore, the inequality $f(x) > t$ cannot happen. This means that only the following two cases are possible:

(1) $f(x) < t$;
(2) $f(x) = t$.

Case 1: $J_2(x,t) = \emptyset$. Then, $J_1(x,t) = I$ and according to (11) $I_1(x_k, t_k, \delta_k) = I$ for all $k > \bar{k}_0$. It follows from (8) that

$$\nabla \Phi_{\delta_k}(x_k, t_k) = (0_n, 1)$$

for all $k \geq \bar{k}_0$. Then, in this case, $v = (0_n, 1)$. The expression (6) implies that the subdifferential of the function F at the point (x,t) is:

$$\partial F(x,t) = \{(0_n, 1)\},$$

that is $v \in \partial F(x,t)$.

Case 2: $J_2(x,t) \neq \emptyset$. Define the following two index sets:

$$\bar{I}_1 = \{i \in I : \exists k_0 > 0 \text{ such that } i \in I_1(x_k, t_k, \delta_k)\ \forall k \geq k_0 \text{ and } f_i(x) = t\},$$
$$\bar{I}_2 = \{i \in I : \exists k_1 > 0 \text{ such that } i \in I_2(x_k, t_k, \delta_k)\ \ \forall k \geq k_1\}.$$

It follows from (11) and (13) that $J_2(x,t) = \bar{I}_1 \bigcup \bar{I}_2$. Applying (9) at (x_k, t_k) for $\delta_k > 0$ and sufficiently large $k > 0$, we get:

$$\nabla_x \Phi_{\delta_k}(x_k, t_k) = \sum_{i \in I_2(x_k, t_k, \delta_k)} \left(\frac{f_i(x_k) - t_k + \delta_k}{2\delta_k} \right) \nabla f_i(x_k). \tag{14}$$

If $i \in I_2(x_k, t_k, \delta_k)$, then $-\delta_k \leq f_i(x_k) - t_k \leq \delta_k$ and, therefore:

$$\alpha_{ik} = \frac{f_i(x_k) - t_k + \delta_k}{2\delta_k} \in [0, 1]$$

for all sufficiently large $k > 0$. This means that any limit of the sequence $\{\alpha_{ik}\}$ belongs to the segment $[0, 1]$. Without loss of generality, assume that $\alpha_{ik} \to \bar{\alpha}_i$, $i \in I_2(x_k, t_k, \delta_k)$ as $k \to \infty$. Then, it follows from (14) that

$$\lim_{k \to \infty} \nabla_x \Phi_{\delta_k}(x_k, t_k) = \sum_{i \in \bar{I}_2} \bar{\alpha}_i \nabla f_i(x).$$

In order to get the expression for the second component $\Phi'_{\delta_k t}(x_k, t_k)$ of the gradient $\nabla \Phi_{\delta_k}(x_k, t_k)$, we apply the formula (10) by taking into account that $I_3(x_k, t_k, \delta_k) = \emptyset$:

$$\begin{aligned} \Phi'_{\delta_k t}(x_k, t_k) &= 1 - \sum_{i \in I_2(x_k, t_k, \delta_k)} \frac{f_i(x_k) - t_k + \delta_k}{2\delta_k} \\ &= 1 - \sum_{i \in I_2(x_k, t_k, \delta_k)} \alpha_{ik}. \end{aligned}$$

Then,

$$\lim_{k \to \infty} \Phi'_{\delta_k t}(x_k, t_k) = 1 - \sum_{i \in \bar{I}_2} \bar{\alpha}_i.$$

Thus,

$$v = \left(\sum_{i \in \bar{I}_2} \bar{\alpha}_i \nabla f_i(x), 1 - \sum_{i \in \bar{I}_2} \bar{\alpha}_i \right) = (0_n, 1) + \sum_{i \in \bar{I}_2} \bar{\alpha}_i (\nabla f_i(x), -1).$$

It follows from (6) that the subdifferential of the function F at the point (x, t), when $t = f(x)$, is:

$$\partial F(x,t) = \{(0_n, 1)\} + \sum_{i \in J_2(x,t)} \text{conv}\,\{0_{n+1}, (\nabla f_i(x), -1)\}. \tag{15}$$

The inclusion (13) implies that $\bar{I}_2 \subseteq J_2(x,t)$. Select $\bar{\alpha}_i = 0$ for all $i \in J_2(x,t) \setminus \bar{I}_2$. Therefore, $v \in \partial F(x,t)$. This completes the proof. □

Proposition 7 *For any* $(x,t) \in \mathbb{R}^n \times \mathbb{R}$, *there exists* $\bar{\delta} > 0$ *such that*

$$\nabla \Phi_\delta(x,t) \in \partial F(x,t)$$

for all $\delta \in (0, \bar{\delta})$.

Proof Consider

$$\alpha_i(\delta) = \frac{f_i(x) - t + \delta}{2\delta},\ i \in I.$$

It is clear that $\alpha_i(\delta) \in [0,1]$ for any $i \in I_2(x,t,\delta)$ and, therefore, all limits of the function $\alpha_i(\delta)$ as $\delta \downarrow 0$ belong to the segment $[0,1]$. Then, the gradient of the function $\Phi_\delta(x,t)$ can be written as:

$$\nabla_x \Phi_\delta(x,t) = \sum_{i \in I_2(x,t,\delta)} \alpha_i(\delta) \nabla f_i(x) + \sum_{i \in I_3(x,t,\delta)} \nabla f_i(x), \tag{16}$$

$$\Phi'_{\delta t}(x,t) = 1 - \sum_{i \in I_2(x,t,\delta)} \alpha_i(\delta) - |I_3(x,t,\delta)|. \tag{17}$$

Define the following index sets:

$$\bar{I}_{21} = \{i \in I_2(x,t,\delta) :\ f_i(x) < t\},$$

$$\bar{I}_{22} = \{i \in I_2(x,t,\delta) :\ f_i(x) = t\},$$

$$\bar{I}_{23} = \{i \in I_2(x,t,\delta) :\ f_i(x) > t\}.$$

Let

$$\delta_1 = \min_{i \in \bar{I}_{21}} (t - f_i(x)) > 0 \ \text{ and } \ \delta_2 = \min_{i \in \bar{I}_{23}} (f_i(x) - t) > 0.$$

Consider $\bar{\delta} = \min\{\delta_1, \delta_2\}$. Then, it is obvious that for any $\delta \in (0, \bar{\delta})$, we have:

$$I_1(x,t,\delta) = J_1(x,t),\ I_2(x,t,\delta) = J_2(x,t),\ I_3(x,t,\delta) = J_3(x,t).$$

Comparing (6) with (16) and (17), it is easy to see that $\nabla \Phi_\delta(x,t) \in \partial F(x,t)$ for all $\delta \in (0, \bar{\delta})$. □

Proposition 8 *Suppose that functions f_i, $i \in I$ are continuously differentiable, their gradients ∇f_i are locally Lipschitz on $\mathbb{R}^n$ and $t = f(x)$. Then, the gradient $\nabla \Phi_\delta$ is locally Lipschitz on $\mathbb{R}^n \times \mathbb{R}$ for any fixed $\delta > 0$.*

Proof Let

$$\alpha_i(x) = \max\left\{0, \frac{f_i(x) - f(x) + \delta}{2\delta}\right\}.$$

One can see that $0 \le \alpha_i(x) \le 1/2$ for all $i \in I$ and $x \in \mathbb{R}^n$.

The function α_i is locally Lipschitz since it is represented as a maximum of two locally Lipschitz functions. Take any bounded set $X \subset \mathbb{R}^n$. Let L_i be Lipschitz constants of the functions α_i, $i \in I$ on the set X.

Select arbitrary points $x_1, x_2 \in X$. Define the following index sets:

$$I_{11} = I_1(x_1, f(x_1), \delta) \cap I_1(x_2, f(x_2), \delta),$$

$$I_{12} = I_1(x_1, f(x_1), \delta) \cap I_2(x_2, f(x_2), \delta),$$

$$I_{21} = I_2(x_1, f(x_1), \delta) \cap I_1(x_2, f(x_2), \delta),$$

$$I_{22} = I_2(x_1, f(x_1), \delta) \cap I_2(x_2, f(x_2), \delta).$$

It is clear that $I_{12} \cap I_{21} = \emptyset$ and $I_3(x_1, f(x_1), \delta) = I_3(x_2, f(x_2), \delta) = \emptyset$.

It follows from (9) and (10) that gradients of the function Φ_δ at points $(x_1, f(x_1))$ and $(x_2, f(x_2))$ can be written as:

$$\nabla_x \Phi_\delta(x_1, f(x_1)) = \sum_{i \in I_{21}} \alpha_i(x_1) \nabla f_i(x_1) + \sum_{i \in I_{22}} \alpha_i(x_1) \nabla f_i(x_1),$$

$$\Phi'_{\delta t}(x_1, f(x_1)) = 1 - \sum_{i \in I_{21}} \alpha_i(x_1) - \sum_{i \in I_{22}} \alpha_i(x_1),$$

and

$$\nabla_x \Phi_\delta(x_2, f(x_2)) = \sum_{i \in I_{12}} \alpha_i(x_2) \nabla f_i(x_2) + \sum_{i \in I_{22}} \alpha_i(x_2) \nabla f_i(x_2),$$

$$\Phi'_{\delta t}(x_2, f(x_2)) = 1 - \sum_{i \in I_{12}} \alpha_i(x_2) - \sum_{i \in I_{22}} \alpha_i(x_2).$$

Define $I_0 = I_{12} \cup I_{21}$. It is clear that $\alpha_i(x_1) = 0$ for all $i \in I_{12}$ and $\alpha_i(x_2) = 0$ for all $i \in I_{21}$. Then, the above expressions for gradients can be rewritten as:

$$\nabla_x \Phi_\delta(x_1, f(x_1)) = \sum_{i \in I_0} \alpha_i(x_1) \nabla f_i(x_1) + \sum_{i \in I_{22}} \alpha_i(x_1) \nabla f_i(x_1),$$

$$\Phi'_{\delta t}(x_1, f(x_1)) = 1 - \sum_{i \in I_0} \alpha_i(x_1) - \sum_{i \in I_{22}} \alpha_i(x_1),$$

and

$$\nabla_x \Phi_\delta(x_2, f(x_2)) = \sum_{i \in I_0} \alpha_i(x_2) \nabla f_i(x_2) + \sum_{i \in I_{22}} \alpha_i(x_2) \nabla f_i(x_2),$$

$$\Phi'_{\delta t}(x_2, f(x_2)) = 1 - \sum_{i \in I_0} \alpha_i(x_2) - \sum_{i \in I_{22}} \alpha_i(x_2).$$

Define $\bar{I} = I_0 \cup I_{22}$. Then,

$$\nabla_x \Phi_\delta(x_1, f(x_1)) - \nabla_x \Phi_\delta(x_2, f(x_2)) = \sum_{i \in \bar{I}} [\alpha_i(x_1) - \alpha_i(x_2)] \nabla f_i(x_1) + \sum_{i \in \bar{I}} \alpha_i(x_2) [\nabla f_i(x_1) - \nabla f_i(x_2)], \tag{18}$$

and

$$\Phi'_{\delta t}(x_1, f(x_1)) - \Phi'_{\delta t}(x_2, f(x_2)) = \sum_{i \in \bar{I}} [\alpha_i(x_1) - \alpha_i(x_2)]. \tag{19}$$

Since the functions f_i are continuously differentiable, they are bounded on bounded sets. Then, there exists $M > 0$ such that $\|\nabla f_i(x)\| \leq M$ for all $i \in I$ and $x \in X$. Denote by K_i Lipschitz constants of gradients $\nabla f_i,\ i \in I$ on the set X. It follows from (18) that

$$\|\nabla_x \Phi_\delta(x_1, f(x_1)) - \nabla_x \Phi_\delta(x_2, f(x_2))\| \leq L_0 \|x_1 - x_2\|,$$

where

$$L_0 = \sum_{i \in \bar{I}} (M L_i + K_i/2).$$

Applying (19) for the $(n+1)$-th coordinate of the gradient, we get:

$$|\Phi_{\delta t}(x_1, f(x_1)) - \Phi_{\delta t}(x_2, f(x_2))| \leq \bar{L}_0 \|x_1 - x_2\|,$$

where

$$\bar{L}_0 = \sum_{i \in \bar{I}} L_i.$$

Thus, we have

$$\|\nabla \Phi_\delta(x_1, f(x_1)) - \nabla \Phi_\delta(x_2, f(x_2))\| \le L\|x_1 - x_2\|,$$

where

$$L = (L_0^2 + \bar{L}_0^2)^{1/2}.$$

This completes the proof. □

Remark 1 It should be noted that the Lipschitz constant L depends on δ since Lipschitz constants L_i of the functions $\alpha_i,\ i \in I$ depend on this parameter.

3.3 Minimization Algorithm

This subsection presents an algorithm for solving the finite minimax problem (1). Given a sequence $\{\delta_k\}$ such that $\delta_k > 0$ and $\delta_k \to 0$ as $k \to \infty$, this problem is replaced by the sequence of the following smooth problems:

$$\text{minimize } \Phi_{\delta_k}(x, f(x)). \tag{20}$$

Results from previous two subsections demonstrate that smooth optimization algorithms can be applied to solve Problem (20). We call such algorithms smooth optimization solvers.

Let $\{\varepsilon_k\}$ be a given sequence such that $\varepsilon_k > 0$ and $\varepsilon_k \to 0$ as $k \to \infty$. An algorithm for solving Problem (1) proceeds as follows.

Algorithm 1 An algorithm for solving minimax problems.
Step 1 (Initialization). Select any starting point $x_0 \in \mathbb{R}^n$ and set $t_0 := f(x_0),\ k := 0$.
Step 2 Starting from the point x_k, apply a smooth optimization solver to Problem (20) to find a point $\bar{x}$ such that

$$\|\nabla \Phi_{\delta_k}(\bar{x}, f(\bar{x}))\| < \varepsilon_k. \tag{21}$$

Step 3 Set $x_{k+1} := \bar{x},\ t_{k+1} := f(\bar{x}),\ k := k + 1$ and go to Step 2.

Remark 2 It should be noted that one can choose the smoothing (precision) parameter $\delta > 0$ sufficiently small and solve Problem (20) only once. However, such an approach may make this problem ill-conditioned which will significantly increase computational efforts to solve it. The use of the sequence $\{\delta_k\}$ may help to prevent such a situation.

Remark 3 The choice of sequences $\{\delta_k\}$ and $\{\varepsilon_k\}$ might be crucial for some problems. If δ_k quickly converges to 0, then the ill-conditioned behavior of the problem may gradually increase. In this case, a large number of iterations is required to obtain the condition (21). In order to avoid this, one should ensure that the sequence $\{\delta_k\}$ converges to 0 slower than the sequence $\{\varepsilon_k\}$.

Next, we prove the convergence of Algorithm 1. For the starting point x_0, consider the following set:

$$\mathscr{L}(x_0) = \left\{x \in \mathbb{R}^n : \ f(x) \leq f(x_0)\right\}.$$

Proposition 9 *Assume that the set $\mathcal{L}(x_0)$ is bounded for any starting point $x_0 \in \mathbb{R}^n$. Then, any accumulation point of the sequence $\{x_k\}$ generated by Algorithm 1 is a stationary point of Problem (1).*

Proof It is clear that $x_k \in \mathcal{L}(x_0)$ for all $k \geq 0$. Since the set $\mathcal{L}(x_0)$ is bounded, the sequence $\{x_k\}$ has at least one accumulation point. Assume that x^* is an accumulation point of the sequence $\{x_k\}$, and for the sake of simplicity, assume that $x_k \to x^*$ as $k \to \infty$. It follows from Proposition 6 that $0_{n+1} \in \partial F(x^*, f(x^*))$, that is $(x^*, f(x^*))$ is a stationary point of F. Then, applying Proposition 3, we get that x^* is a stationary point of Problem (1). □

4 Comparison of Smoothing Methods

In this section, we present results on the comparison of algorithms based on the local and global smoothing techniques for solving finite minimax problems. More specifically, we consider Algorithm 1 which uses the local smoothing technique and two other algorithms from [11] and [28] which use the exponential and hyperbolic smoothing techniques, respectively.

An optimization solver in all cases is the quasi-Newton method with BFGS updates. All algorithms were implemented in Fortran 95 and compiled using the compiler *gfortran*. Numerical experiments were carried out in a PC Intel(R) Core(TM) i5-3470S with CPU 2.90GHz and 8GB of RAM.

In the implementation of Algorithm 1, the sequence of the smoothing parameter $\{\delta_k\}$ is chosen as follows: $\delta_{k+1} = 0.2\delta_k$, $k = 1, \ldots, p$, $\delta_1 = 10$. The same sequence of corresponding smoothing parameters was used for the algorithms based on the exponential and hyperbolic smoothing techniques. The sequence $\{\varepsilon_k\}$ is chosen as: $\varepsilon_k \equiv 10^{-4}$ for all $k > 0$ and problems. Both sequences $\{\delta_k\}$ and $\{\varepsilon_k\}$ are the same for all problems.

We say that a problem is solved by a solver if the relative accuracy of 10^{-4} with respect to the function value is achieved, that is a solver finds the value $\bar{f}$ of the objective function f such that

$$\frac{\bar{f} - f_*}{|f_*| + 1} \leq 10^{-4}.$$

Here, f_* is the value of the objective function at a local minimizer. For nonconvex problems, we consider a set of local minimizers to check the ability of algorithms to solve these problems.

It should be noted that the sequence of smoothing parameters can be chosen differently for different problems to significantly accelerate the convergence of algorithms. However, it is not clear how it can be done for specific problems.

Nonsmooth optimization academic test problems from [1, 37] are used to test and to compare the performance of different smoothing techniques. More specifically, we choose problems 1–6, 8–10, 12–18, 20–26, 28–33, 35–37, 56, 58–61, 65, 66 from [1], and also the test problem filter from [37]. All these problems are divided into three different categories: convex small-size problems, nonconvex small-size problems, and large-scale problems. We use ten randomly generated starting points for solving each problem.

The problems 1–6, 8–10, 12–18, 20, 56, 58 are convex. The problems 21–26, 28–33, 35–37, and also the filter problem are nonconvex, and problems 59–61, 65, 66 are large-scale problems. The number of variables in both convex and nonconvex small-size problems ranges from 2 to 50. Most of convex problems are either piecewise linear or piecewise quadratic. Large-scale problems 59, 60, 61 are convex, and problems 65, 66 are nonconvex. The number of variables in these problems is chosen 100, 200, 500, 1000, and 2000.

For small-size problems, we compare the efficiency of algorithms in terms of the number of iterations. For most of these problems, the CPU time used by algorithms is close to 0 and, therefore, we do not include the CPU time. For large-scale problems, we compare the efficiency of algorithms both in terms of the number of iterations and CPU time.

We analyze the results using the performance profiles introduced in [38]. Given a set of solvers S and a set of problems P, one can define performance ratio for each solver as follows:

$$r_{q,s} = \frac{t_{q,s}}{\min\{t_{q,s} : s \in S\}}.$$

Here, $t_{q,s}$ stands for the CPU time (or the number of function evaluations or the number of (sub)gradient evaluations) used by the solver $s \in S$ for solving the problem $q \in P$. Then, a parameter r_M is chosen so that $r_M \geq r_{q,s}$ for all $q \in P$ and $s \in S$. Moreover, $r_{q,s} = r_M$ if and only if the solver s does not solve the problem q. The performance profile $\rho_s(\mu)$ is defined as

$$\rho_s(\mu) = \frac{1}{n_q} \text{size}\left\{q \in P : r_{q,s} \leq \mu\right\},$$

where n_q is the number of problems in P. It is clear that $\mu \in [1, r_M]$.

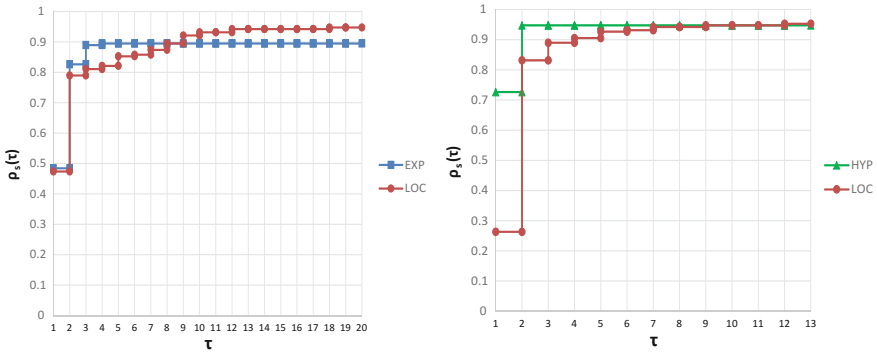
In the performance profiles, the value of $\rho_s(\mu)$ at $\mu = 1$ gives the percentage of test problems for which the corresponding algorithm is the best (it uses least iterations or function calls) and the value of $\rho_s(\mu)$ at the rightmost abscissa gives the percentage of test problems that the corresponding algorithm can solve, that is the robustness of the algorithm (this does not depend on the measured performance). 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. For more information on performance profiles, see [38].

4.1 Results for Convex Problems

Performance profiles for convex problems are presented in Fig. 2. We can see that the performance of the algorithm based on the exponential smoothing technique is slightly better than that of the algorithm based on the local smoothing technique in the sense of efficiency. Furthermore, the algorithm based on the hyperbolic smoothing is significantly more efficient than the algorithm based on the local smoothing technique. In terms of robustness, the algorithm using the local smoothing outperforms both algorithms based on the global smoothing techniques, although the difference between the local smoothing-based algorithm and that of based on the hyperbolic smoothing is not significant.

One explanation to these observations is as follows. The objective functions in most of these test problems are either piecewise linear or piecewise quadratic. The global smoothing of these functions predominantly leads to some kind of quadratic functions in the whole domain, and the quasi-Newton-type methods are efficient to minimize such functions. Therefore, algorithms based on the global smoothing techniques are more efficient than those based on the local smoothing techniques. Since the local smoothing technique approximates the objective function only in some neighborhood of the point of nondifferentiability, the algorithm based on it is more robust than the algorithms based on the global smoothing techniques. In other words, the former algorithm is not as sensitive to the choice of starting points as the latter algorithms.



(a) Local smoothing vs exponential smoothing

(b) Local smoothing vs hyperbolic smoothing

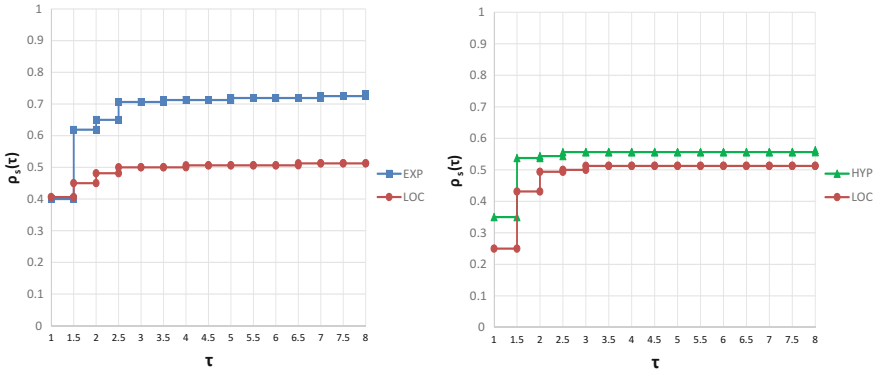
Fig. 2 Comparison of algorithms based on the local and global smoothing using convex problems: number of iterations

4.2 Results for Nonconvex Problems

Figure 3 illustrates the performance profiles of algorithms for nonconvex problems. One can see that in the sense of efficiency the performance of the algorithms based on the exponential and local smoothing techniques is similar, whereas the algorithm based on the hyperbolic smoothing technique demonstrates better performance than that of based on the local smoothing. As far as robustness is concerned, both algorithms using the global smoothing techniques demonstrate better performance than that of using the local smoothing technique. Furthermore, the difference between the algorithms based on the exponential and local smoothing techniques is quite significant (more than 20%).

Note that objective functions are piecewise quadratic only in very few nonconvex small-size test problems and most of them are general nonconvex problems. These problems contain many shallow local minimizers. The use of global smoothing techniques allows one to eliminate most of these local minimizers and to give a good approximation of the objective functions in some neighborhood of deep local minimizers.

On the other side, the use of the local smoothing technique leads to the generation of "false" local minimizers combining several "neighboring" shallow local minimizers, and any algorithm based on this smoothing technique gets stuck in such "local minimizers." This is due to the fact that the local smoothing technique approximates the objective function in some neighborhood of points where the function is nondifferentiable, whereas the global smoothing techniques approximate the objective function in the whole domain. This is the main reason for the success of algorithms based on the global smoothing for solving nonconvex problems in comparison with those based on the local smoothing techniques.



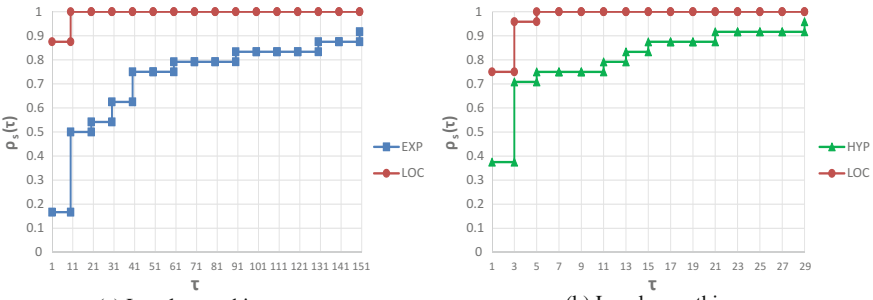
(a) Local smoothing vs exponential smoothing

(b) Local smoothing vs hyperbolic smoothing

Fig. 3 Comparison of algorithms based on the local and global smoothing using nonconvex problems: number of iterations

4.3 *Results for Large-Scale Problems*

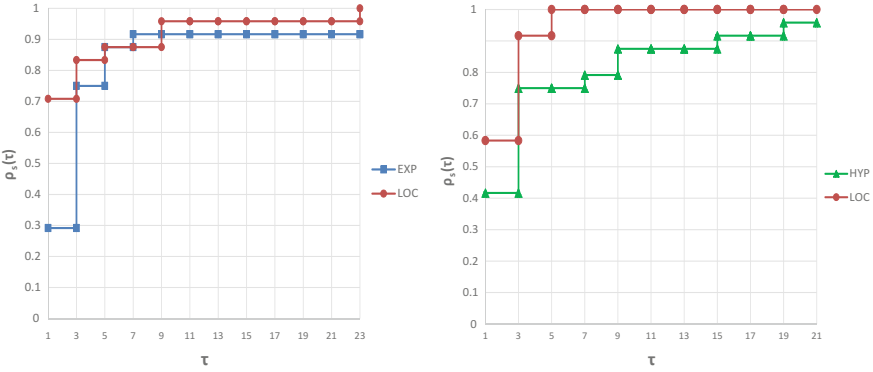
Performance profiles of algorithms using the CPU time and the number of iterations are given in Figs. 4 and 5, respectively. Results presented in these figures demonstrate that the local smoothing-based algorithm outperforms those based on the global smoothing both in the sense of efficiency and robustness. The algorithm based on the local smoothing technique solved all problems with the given accuracy. We can also see that this algorithm is significantly more efficient than that based on the exponential smoothing technique. However, there is no significant difference between the performance of these algorithms in the sense of robustness. The similar picture can be observed when comparing algorithms based on the local and hyperbolic smoothing techniques.



(a) Local smoothing vs exponential smoothing

(b) Local smoothing vs hyperbolic smoothing

Fig. 4 Comparison of algorithms based on the local and global smoothing using large-scale problems: CPU time



(a) Local smoothing vs exponential smoothing

(b) Local smoothing vs hyperbolic smoothing

Fig. 5 Comparison of algorithms based on the local and global smoothing using large-scale problems: number of iterations

We can see that all algorithms are more successful to solve large-scale problems than nonconvex small-size problems. This is due to the fact that three out of five large-scale problems are convex and other two problems have quite simple component functions. In this situation, usually, the algorithm based on the local smoothing technique performs better than those based on the global smoothing techniques.

5 Conclusions

In this chapter, we studied one local smoothing method for solving the finite minimax problems. In order to apply this method, we reformulated the maximum function using one additional variable. Then, this function is approximated by the smooth functions using one smoothing parameter. We established the relationship between gradients of the smooth function and the subdifferential of the reformulated maximum function. An algorithm for solving the finite minimax problems is designed, and its convergence is studied based on these results.

The proposed algorithm is tested and compared with algorithms based on the global smoothing techniques such as the exponential and hyperbolic smoothing methods using nonsmooth optimization academic test problems. These problems include small-size convex and nonconvex as well as large-scale nonsmooth optimization problems. Based on the obtained results, we can draw the following conclusions:

1. For convex small-size problems, the algorithm based on the local smoothing is slightly more robust than algorithms based on the exponential and hyperbolic smoothing methods. However, for these problems, the algorithm based on the hyperbolic smoothing is more efficient than that of based on the local smoothing method.
2. For nonconvex small-size problems, algorithms based on the global smoothing techniques are more robust than the local smoothing-based algorithm. Although the former algorithms are more efficient than the latter algorithm, the difference between the algorithms based on the exponential and local smoothing techniques is not significant.
3. We observed a different picture in the performance of algorithms for solving large-scale problems. The algorithm based on the local smoothing performs significantly better than those based on the global smoothing techniques in the sense of both the number of iterations and used CPU time. One should notice that the most of large-scale problems, used in numerical experiments, are convex.

Note that there exist other more complex local smoothing techniques than the one used in this chapter. However, our numerical experiments show that the use of other local smoothing techniques will not considerably change the main conclusions of this chapter.

References

1. Bagirov, A.M., Karmitsa, N., Mäkelä, M.M.: Introduction to Nonsmooth Optimization: Theory. Practice and Software. Springer, Cham (2014)
2. Mäkelä, M.M., Neittaanmaki, P.: Nonsmooth Optimization. World Scientific, Singapore (1992)
3. Demyanov, V.F., Malozemov, V.N.: Introduction to Minimax. Wiley, New York (1974)
4. Du, D.Z., Pardalos, P.M.: Minimax and Applications. Kluwer Academic Publishers, Dordrecht (1995)
5. Bagirov, A.M., Ganjehlou, A.N.: A quasisecant method for minimizing nonsmooth functions. Optim. Methods Softw. **25**(1), 3–18 (2010)
6. Bagirov, A.M., Karasozen, B., Sezer, M.: Discrete gradient method: derivative-free method for nonsmooth optimization. J. Optim. Theory Appl. **137**, 317–334 (2008)
7. Hiriart-Urruty, J.B., Lemaréchal, C.: Convex Analysis and Minimization Algorithms II. Springer, Berlin (1993)
8. Kiwiel, K.C.: Methods of Descent for Nondifferentiable Optimization. Lecture Notes in Mathematics. Springer, Berlin (1985)
9. Bertsekas, D.P.: Constrained Optimization and Lagrange Multiplier Methods. Academic Press, New York (1982)
10. Li, X.S.: An entropy-based aggregate method for minimax optimization. Eng. Optim. **18**, 277–285 (1992)
11. Xu, S.: Smoothing method for minimax problems. Comput. Optim. Appl. **20**(3), 267–279 (2001)
12. Yang, X.Q.: Smoothing approximations to nonsmooth optimization problems. J. Austral. Math. Soc. Ser. B. **36**, 274–285 (1994)
13. Polak, E., Womersley, R.S., Yin, H.X.: An algorithm based on active sets and smoothing for discretized semi-infinite minimax problems. J. Optim. Theory Appl. **138**, 311–328 (2008)
14. Yin, H.-X.: Error bounds of two smoothing approximations for semi-infinite minimax problems. Acta Math. Applicatae Sinica. **25**(4), 685–696 (2009)
15. Chen, X.: Smoothing methods for complementarity problems and their applications: a survey. J. Oper. Res. Soc. Jpn. **43**(1), 32–47 (2000)
16. Facchinei, F., Jiang, H., Qi, L.: A smoothing method for mathematical programs with equilibrium constraints. Math. Program. **85**, 107–134 (1999)
17. Fukushima, M., Luo, Z.Q., Pang, J.S.: A globally convergent sequential quadratic programming algorithm for mathematical programs with linear complementarity constraints. Comput. Optim. Appl. **10**, 5–34 (1998)
18. Sun, D., Qi, L.: Solving variational inequality problems via smoothing-nonsmooth reformulations. J. Comput. Appl. Math. **129**, 37–62 (2001)
19. Ansari, M.R., 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)
20. Bagirov, A.M., Taheri, S.: DC Programming algorithm for clusterwise linear L_1 regression. J. Oper. Res. Soc. China **5**(2), 233–256 (2017)
21. Bagirov, A.M., Mohebi, E.: An algorithm for clustering using L_1-norm based on hyperbolic smoothing technique. Comput. Intell. **32**(3), 439–457 (2016)
22. Bagirov, A.M., Mohebi, E.: Nonsmooth optimization based algorithms in cluster analysis. In: Celebi, E. (ed.) Partitional Clustering Algorithms, pp. 99–146. Springer, Berlin (2015)
23. Feng, Z.G., Yiu, K.F.C., Teo, K.L.: A smoothing approach for the optimal parameter selection problem with continuous inequality constraint. Optim. Methods Softw. **28**(4), 689–705 (2013)
24. Ye, F., Liu, H., Zhou, Sh, Liu, S.: A smoothing trust-region Newton-CG method for minimax problem. Appl. Math. Comput. **199**, 581–589 (2008)
25. Zang, I.: A smoothing-out technique for min-max optimization. Math. Program. **19**, 61–77 (1980)
26. Polak, E., Royset, J.O., Womersley, R.S.: Algorithms with adaptive smoothing for finite minimax problems. J. Optim. Theory Appl. **119**, 459–484 (2003)

27. Xiao, Y., Yu, B.: A truncated aggregate smoothing Newton method for minimax problems. Appl. Math. Comput. **216**, 1868–1879 (2010)
28. Bagirov, A.M., Al Nuaimat, A., Sultanova, N.: Hyperbolic smoothing function method for minimax problems. Optimization **62**(6), 759–782 (2013)
29. Xavier, A.E.: The hyperbolic smoothing clustering method. Pattern Recog. **43**, 731–737 (2010)
30. Xavier, A.E., Oliveira, A.A.F.D.: Optimal covering of plane domains by circles via hyperbolic smoothing. J. Glob. Optim. **31**(3), 493–504 (2005)
31. Xavier, A.E.: Penalizaćao hiperbólica. I Congresso Latino-Americano de Pesquisa Operacional e Engenharia de Sistemas. 8 a 11 de Novembro, pp. 468–482. Brasil, Rio de Janeiro (1982)
32. Vazquez, F.G., Gunzel, H., Jongen, HTh: On logarithmic smoothing of the maximum function. Ann. Oper. Res. **101**, 209–220 (2001)
33. Nesterov, Yu.: Smooth minimization of nonsmooth functions. Math. Program. **103**(1), 127–152 (2005)
34. Ermoliev, Y.M., Norkin, V.I., Wets, R.J-B.: The minimization of semicontinuous functions: Mollifier subgradients. SIAM J. Control Optim. **33**, 149–167 (1995)
35. Ben-Tal, A., Teboulle, M.: A smoothing technique for nondifferentiable optimization problems. In: Dolecki, S. (ed.) Lecture Notes in Mathematics, vol. 1405, pp. 1–11. Springer, Heidelberg (1989)
36. Peng, J.: A smoothing function and its applications. In: Fukushima, M., Qi, L. (eds.) Reformulation: Nonsmooth, Piecewise Smooth, Semismooth and Smoothing Methods, pp. 293–316. Kluwer, Dordrecht (1998)
37. Lukšan, L., Vlček, J.: Test Problems for Nonsmooth Unconstrained and Linearly Constrained Optimization. In: Technical Report 798, Institute of Computer Science, Academy of Sciences of the Czech Republic, Prague (2000) Available via http://hdl.handle.net/11104/0124190
38. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Program. **91**(2), 201–213 (2002)

A Competitive Error in Variables Approach and Algorithms for Finding Positive Definite Solutions of Linear Systems of Matrix Equations



Negin Bagherpour and Nezam Mahdavi-Amiri

Abstract Here, we refine our recent proposed method for computing a positive definite solution to an overdetermined linear system of equations with multiple right-hand sides. This problem is important in several process control contexts including quadratic models for optimal control. The coefficient and the right-hand side matrices are, respectively, named data and target matrices. In several existing approaches, the data matrix is unrealistically assumed to be error free. We have recently presented an algorithm for solving such a problem considering error in measured data and target matrices. We defined a new error in variables (EIV) error function considering error for the variables, the necessary and sufficient optimality conditions and outlined an algorithm to directly compute a solution minimizing the defined error. Moreover, the algorithm was specialized for a special case when the data and target matrices are rank deficient. Here, after giving a detailed review of the proposed algorithms, we rewrite the algorithms by use of the Householder tridiagonalization instead of spectral decomposition. We then show that in case of full rank data and target matrices, the Householder tridiagonalization is more efficient than the previously considered spectral decomposition. A comparison of our proposed approach and two existing methods are provided. The numerical test results and the associated Dolan-Moré performance profiles show the new approach to be more efficient than the two methods and have comparatively smaller standard deviation of error entries and smaller effective rank as features being sought for control problems.

Keywords Linear systems · Positive definite solution · Error in variables model Householder tridiagonalization

N. Bagherpour
Department of Mathematics, The University of Manchester, Manchester, UK
e-mail: negin.bagherpour@manchester.ac.uk

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

© Springer International Publishing AG, part of Springer Nature 2018

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*,
Springer Proceedings in Mathematics & Statistics 235,
https://doi.org/10.1007/978-3-319-90026-1_3

1 Introduction

The mathematical problem

$$DX \simeq T \tag{1}$$

is considered, where $D, T \in \mathbb{R}^{m\times n}$, with $m \geq n$, are given and a symmetric positive definite matrix $X \in \mathbb{R}^{n\times n}$ is to be computed as a solution of (1). This problem arises in many physical and industrial applications such as estimating the mass inertia matrix in the design of controllers for solid structures and robots; e.g., see [1–3]. Also, modeling a deformable structure leads to such a mathematical problem; e.g., see [4]. We note that in some other applications such as computing the correlation matrix in finance or insurance/reinsurance industries a symmetric positive semi-definite solution of an overdetermined linear system of equations needs to be computed [5]. A number of least squares formulations have been proposed for physical problems, which may be classified as ordinary and error in variables (EIV) models. Single or multiple right hand side least squares problems may also arise. With a single right-hand side, we have an overdetermined linear system of equations $Dx \simeq t$, where $D \in \mathbb{R}^{m\times n}$, $t \in \mathbb{R}^{m\times 1}$, with $m \geq n$, being known and the vector $x \in \mathbb{R}^{n\times 1}$ to be computed. In an ordinary least squares formulation, the error is only associated with t. So, to minimize the corresponding error, the mathematical problem

$$\begin{aligned} &\min \|\Delta t\| \\ &s.t. Dx = t + \Delta t, \end{aligned} \tag{2}$$

is devised. There are a number of methods for solving (2), identified by direct and iterative methods. A desired approach is to use the QR factorization of the matrix D [6]. An iterative method has also been introduced for solving (2) using the GMRES algorithm (see [7]).

In an EIV model, however, errors in both D and t are considered; e.g., see [8]. Total least squares formulation is a well-known EIV model, where the goal is to solve the following mathematical problem (see [9, 10]):

$$\begin{aligned} &\min \|[\Delta D \Delta t]\| \\ &s.t. (D + \Delta D)x = t + \Delta t. \end{aligned} \tag{3}$$

We note that $\|\cdot\|$ in (2) and (3), respectively, denote the vector 2-norm and the matrix Frobenius norm. Both direct [11] and iterative [12] methods have been presented for solving (3). Moreover, the scaled total least squares formulation has been considered to unify both ordinary and total leasts squares formulations; e.g., see [11].

A least squares problem with multiple right-hand side vectors can also be formulated as an overdetermined system of equations $DX \simeq T$, where $D \in \mathbb{R}^{m\times n}$,

$T \in \mathbb{R}^{m\times k}$, with $m \geq n$, being given and the matrix $X \in \mathbb{R}^{n\times k}$ to be computed. With ordinary and total least squares formulations, the respective mathematical problems are

$$\begin{aligned} &\min \|\Delta T\| \\ &s.t. DX = T + \Delta T \\ &X \in \mathbb{R}^{n\times k} \end{aligned} \tag{4}$$

and

$$\begin{aligned} &\min \|[\Delta D, \Delta T]\| \\ &s.t. (D + \Delta D)X = T + \Delta T \\ &X \in \mathbb{R}^{n\times k}. \end{aligned} \tag{5}$$

Common methods for solving (4) are similar to the ones for (2); e.g., see [6, 7]. Connections between ordinary least squares and total least squares formulations have been discussed in [13].

We have recently presented a newly defined EIV model for solving a positive definite linear system [14] with the aim to compute a symmetric positive definite solution $X \in \mathbb{R}^{n\times n}$ to the overdetermined system of equations $DX \simeq T$, where both matrices D and T are associated with errors. Here, we present a brief review of our proposed EIV model and outline a new algorithm to solve the obtained optimization problem. We then present experimental results to show the efficiency of our proposed algorithm in comparison with existing iterative methods to compute proper solutions with lower standard deviation values in lower times.

The rest of our work is organized as follows. In Sect. 2, we review the error function and discuss some of its characteristics. The method presented in [14] for solving the resulting optimization problem in case of a full column rank data matrix is summarized in Sect. 3. Moreover, a new algorithm for solving the same problem using the Householder tridiagonalization is presented in Sect. 3. There, we also review the proposed methods for solving the generated optimization problem in case of rank deficient data matrix. In Sect. 4, a detailed discussion is made on the computational complexity of both methods. Computational results and comparisons with available methods are given in Sect. 5. Section 6 gives our concluding remarks.

2 Problem Formulation

Considering the problem of finding a symmetric and positive definite solution for the overdetermined system of linear equations $DX \simeq T$, in which both D and T assumed to contain error, the values DX and TX^{-1} are predicted values for T and D from the

model $DX \simeq T$; hence, vectors $\Delta T_j = (DX - T)_j$ and $\Delta D_j = (D - TX^{-1})_j$ are the entries of errors in the jth column of T and D, respectively. The value

$$E = \sum_{j=1}^{n} (DX_j - T_j)^T (D_j - (TX^{-1})_j,$$

or equivalently,

$$\begin{aligned} E &= \sum_{j=1}^{n} \sum_{i=1}^{m} (DX - T)_{ij} (D - TX^{-1})_{ij} \\ &= \mathrm{tr}((DX - T)^T (D - TX^{-1})), \end{aligned}$$

seems to be an appropriate measure of error with tr(.) standing for trace of a matrix. Therefore, the problem can be formulated as

$$\min_{X \succ 0} \mathrm{tr}((DX - T)^T (D - TX^{-1})), \tag{6}$$

where X is symmetric and by $X \succ 0$, we mean X is positive definite. Problem (6) proposes a newly defined EIV model for solving a positive definite linear system of equations.

We note that the error E, defined by (6), is

$$E = \|DY - TY^{-T}\|_F^2 = \|(DX - T)Y^{-T}\|_F^2, \tag{7}$$

where $X = YY^T$ and $\|.\|_F$ denotes the Frobenius norm. This new formulation gives a relation between our proposed error and the main system $DX \simeq T$.

An appropriate characteristic of the error formula proposed by (6) is that for a symmetric and positive definite matrix X, the value of E is nonnegative and equals zero if and only if $DX = T$.

3 Mathematical Solution

Here, we consider solving (6) for both a full rank and rank deficient data matrix, D.

3.1 Full Column Rank Data Matrix

We are to develop an algorithm to solve (6) with the assumption that D has full column rank.

Considering the well-known properties of the trace operator [6], we have

$$\mathrm{tr}((DX - T)^T(D - TX^{-1})) = \mathrm{tr}(D^T DX + X^{-1}T^T T) - 2\,\mathrm{tr}(T^T D),$$

and (6) can be written as

$$\min \mathrm{tr}(AX + X^{-1}B), \tag{8}$$

where $A = D^T D$ and $B = T^T T$, and the symmetric and positive definite matrix X is to be computed.

To explain our method for solving (8), we note that $\Phi(X) = \mathrm{tr}(AX + X^{-1}B)$ is strictly convex on the cone of symmetric and positive definite matrices; hence, the first-order optimality conditions are also sufficient for (6).

The KKT conditions for (6) lead to the following nonlinear matrix equation [14]:

$$XAX = B. \tag{9}$$

Note that (9) is an special case of the continuous time Riccati equation (CARE, [15])

$$A^T XE + E^T XA - (E^T XB + S)R^{-1}(B^T XE + S^T) + Q = 0, \tag{10}$$

with $R = 0$, $E = \frac{A}{2}$ and $Q = -B$. There is a MATLAB routine to solve CARE for arbitrary values of A, E, B, S, R, and Q. To use the routine, it is sufficient to type the command

```
X=care(A,B,Q,R,S,E),
```

for the input arguments as in (10). Higham [15] developed an effective method for computing the positive definite solution of the special CARE, with A and B being symmetric and positive definite, using well-known matrix decompositions. His method is cited as the following theorem.

Theorem 1 [15] *Assume $D, T \in \mathbb{R}^{m\times n}$ with $m \geq n$ are known and $rank(D) = rank(T) = n$. Let $D = QR$ be the QR factorization of D. Let $A = D^T D$ and $B = T^T T$. Define the matrix $\tilde{Q} = RBR^T$ and compute its spectral decomposition; that is, $\tilde{Q} = RBR^T = U\tilde{S}^2U^T$. Then, (8) has a unique solution, given by*

$$X^* = R^{-1}U\tilde{S}U^T R^{-T}.$$

Proof See [14].

We are now ready to outline the steps of our proposed algorithm for solving the EIV positive definite linear system. Note that **Algorithm 1** computes the solution of (6) directly.

Algorithm 1 was outlined based on QR factorization for solving (6) in case of a full rank data matrix. In the following, we construct a new algorithm for solving

Algorithm 1 Algorithm 1 - Solving the EIV problem for positive definite linear systems using QR decomposition.

PROCEDURE PDEIV-QR(D, T)
1: Compute the QR decomposition for D and let $D = QR$.
2: Let $\tilde{Q} = RBR^T$, where $B = T^T T$ and compute the spectral decomposition of $\tilde{Q}$, that is, $\tilde{Q} = U\tilde{S}^2U^T$.
3: Set $X^* = R^{-1}U\tilde{S}U^TR^{-T}$.
4: Set $E = \text{tr}((DX^* - T)^T(D - TX^{*-1}))$.

(6) based on tridiagonalization of a symmetric and positive definite matrix using the Householder transformations.

Let $A \in \mathbb{R}^{n\times n}$ be a symmetric and positive definite matrix. Then, there exist Householder transformations H_1, H_2, ... and H_n such that

$$H_nH_{n-1}...H_1A$$

is upper Hessenberg. Furthermore, $H_nH_{n-1}...H_1AH_1^T...H_{n-1}^TH_n^T$ is tridiagonal and we have

$$HAH^T = T,$$

where $H = H_nH_{n-1}...H_1$ is orthonormal and T is tridiagonal [6].

The following theorem outlines our proposed algorithm for solving (6) based on the Householder tridiagonalization.

Theorem 2 *Assume $D, T \in \mathbb{R}^{m\times n}$ with $m \geq n$ are known and $rank(D) = rank(T) = n$. Let $A = D^TD$ and $B = T^TT$. Compute the Householder tridiagonalization for A and set $A = H^TTH$. Now, calculate the Cholesky decomposition of T to obtain $T = LL^T$. Define the matrix $\tilde{Q} = L^THBH^TL$ and compute its spectral decomposition; that is, $\tilde{Q} = L^THBH^TL = US^2U^T$. Then, (6) has a unique solution, given by*

$$X^* = H^TL^{-T}USU^TL^{-1}H.$$

Proof It is sufficient to show that the assigned value to X^* is a symmetric and positive definite matrix and satisfies the KKT conditions $XAX = B$.
While it is clear that X^* is symmetric, we note that for an arbitrary nonzero vector $v \in \mathbb{R}^n$, we have

$$v^TX^{ast}v = v^TH^TL^{-T}USU^TL^{-1}Hv = \|S^{\frac{1}{2}}U^TL^{-1}Hv\|^2 > 0,$$

, and thus, X^* is positive definite. According to the assumptions, we have

$$\begin{aligned} XAX &= H^T L^{-T} U S U^T L^{-1} H H^T L L^T H H^T L^{-T} U S U^T L^{-1} H \\ &= H^T L^{-T} U S^2 U^T L^{-1} H = H^T L^{-T} L^T H B H^T L L^{-1} H = B, \end{aligned}$$

which completes the proof.

We are now ready to present the steps of our proposed algorithm for solving (6) in case of a full column rank data matrix. We later compare the computational complexity of **Algorithms 1** and **2**.

Algorithm 2 Algorithm 2 - Solving the EIV problem for positive definite linear systems using Householder tridiagonalization.

PROCEDURE PDEIV-HH(D, T)
1: Compute the Householder tridiagonalization of $A = D^T D$ and let $A = H^T T H$.
2: Compute the Cholesky decomposition of the tridiagonal matrix T and set $T = LL^T$.
3: Let $\tilde{Q} = L^T H B H^T L$, where $B = T^T T$ and compute the spectral decomposition of $\tilde{Q}$, that is, $\tilde{Q} = U S^2 U^T$.
4: Set $X^* = H^T L^{-T} U \tilde{S} U^T L^{-1} H$.
5: Set $E = \text{tr}((DX^* - T)^T (D - TX^{*-1}))$.

We recently presented an algorithm for solving (6) for the case of the full rank data matrix. The algorithm was constructed based on spectral decomposition which incurs a higher computational complexity. So, the new algorithm using the Householder tridiagonalization is more efficient. In Sect. 5, we discuss the computational complexities in detail.

Next, we extend our proposed method to solve positive definite linear system of equations when the data matrix is rank deficient.

3.2 Rank Deficient Data Matrix

Since the data matrix D is usually produced from experimental measurements, we may have $rank(D) < n$. Here, we extend **Algorithm 1** to solve (6), when $rank(D) = r < n$. It will be shown that, in general, (6) may not have a unique solution. Hence, we also discuss how to find a particular solution of (6) having desirable characteristics for control problems.

We note that a symmetric positive definite matrix X^* is a solution of (6) if and only if

$$X^* A X^* = B. \tag{11}$$

Therefore, below, we discuss how to find a symmetric positive definite matrix X^* satisfying (11).

Let the spectral decomposition of A be $A = U \begin{pmatrix} S^2 & 0 \\ 0 & 0 \end{pmatrix} U^T$, where $S^2 \in \mathbb{R}^{r\times r}$ is a diagonal matrix having the positive eigenvalues of A as its diagonal entries. Since U is orthonormal, (11) can be written as

$$U^T X^* U \begin{pmatrix} S^2 & 0 \\ 0 & 0 \end{pmatrix} U^T X^* U = U^T B U.$$

Letting $\tilde{X} = U^T X U$ and $\tilde{B} = U^T B U$, we have

$$\tilde{X} \begin{pmatrix} S^2 & 0 \\ 0 & 0 \end{pmatrix} \tilde{X} = \tilde{B}. \tag{12}$$

Thus, the matrix $X = U\tilde{X}U^T$ is a solution of (6) if and only if $\tilde{X}$ is symmetric positive definite and satisfies (12). Substituting the block form $\tilde{X} = \begin{pmatrix} \tilde{X}_{rr} & \tilde{X}_{r,n-r} \\ \tilde{X}_{n-r,r} & \tilde{X}_{n-r,n-r} \end{pmatrix}$, where $\tilde{X}_{rr} \in \mathbb{R}^{r\times r}$, $\tilde{X}_{r,n-r} = \tilde{X}_{n-r,r}^T \in \mathbb{R}^{r\times(n-r)}$ and $\tilde{X}_{n-r,n-r} \in \mathbb{R}^{(n-r)\times(n-r)}$, in (12) leads to

$$\tilde{X}_{rr} S^2 \tilde{X}_{rr} = \tilde{B}_{rr}, \tag{13a}$$

$$\tilde{X}_{rr} S^2 \tilde{X}_{r,n-r} = \tilde{B}_{r,n-r}, \tag{13b}$$

$$\tilde{X}_{n-r,r} S^2 \tilde{X}_{r,n-r} = \tilde{B}_{n-r,n-r}. \tag{13c}$$

Let $\bar{D} = S$ and suppose $\bar{T}$ satisfies $\bar{T}^T\bar{T} = \tilde{B}_{rr}$. Consider problem (6) corresponding to the data and target matrices $\bar{D}$ and $\bar{T}$ as follows:

$$\min_{\bar{X}\succ 0} \operatorname{tr}((\bar{D}\bar{X} - \bar{T})^T(\bar{D} - \bar{T}\bar{X}^{-1})). \tag{14}$$

Thus, $\tilde{X}_{rr}$ can be computed using **Algorithm 1** for the input arguments $\bar{D}$ and $\bar{T}$. Substituting the computed $\tilde{X}_{rr}$ in (13b), the linear system of equations

$$\tilde{X}_{rr} S^2 \tilde{X}_{r,n-r} = \tilde{B}_{r,n-r} \tag{15}$$

arises, where $\tilde{X}_{r,n-r} \in \mathbb{R}^{r\times(n-r)}$ is to be computed. Since $\tilde{X}_{rr}$ is positive definite and S^2 is nonsingular, the coefficient matrix of the linear system (15) is nonsingular and $\tilde{X}_{r,n-r}$ can be uniquely computed.

It is clear that since $\tilde{X}$ is symmetric, $\tilde{X}_{n-r,r}$ is the same as $\tilde{X}_{r,n-r}^T$. Now, we check whether the computed $\tilde{X}_{n-r,r}$ and $\tilde{X}_{r,n-r}$ satisfy (13c). Inconsistency of (14) means that there is no symmetric positive definite matrix satisfying (13a)–(13c), and if so, (6) has no solution. Thus, in solving a specific positive definite system with rank deficient

data and target matrices using the presented EIV model, a straightforward method to investigate the existence of solution is to check whether (13c) holds for the given data and target matrices. However, for numerical results, it is necessary to generate meaningful test problems. Hence, in the following two lemmas, we investigate the necessary and sufficient conditions for satisfaction of (13c).

Lemma 1 *Let the spectral decomposition of A be determined as*

$$A = U \begin{pmatrix} S^2 & 0 \\ 0 & 0 \end{pmatrix} U^T,$$

where $S^2 \in \mathbb{R}^{r\times r}$ *and* $rank(A) = rank(B) = r$. *The necessary and sufficient condition for satisfaction of (13c) is that the columns of*

$$BU_r({U_r}^T BU_r)^{-1}{U_r}^T B - B$$

belong to the null space of ${U_{n-r}}^T$.

Proof See [14].

In the following lemma, we give a sufficient condition guaranteeing the existence of a solution for (11). Later in Sect. 6, we use this result to generate consistent test problems.

Lemma 2 *Let the spectral decomposition of B be* $B = V \begin{pmatrix} \sum^2 & 0 \\ 0 & 0 \end{pmatrix} V^T$, *where* $\sum^2 \in \mathbb{R}^{r\times r}$ *and* $rank(A) = rank(B) = r$. *A sufficient condition for satisfaction of (13c) is that*

$$V = U \begin{pmatrix} Q & 0 \\ 0 & P \end{pmatrix}, \tag{16}$$

where $Q \in \mathbb{R}^{r\times r}$ *and* $P \in \mathbb{R}^{(n-r)\times(n-r)}$ *satisfy* $QQ^T = Q^TQ = I$ *and* $PP^T = P^TP = I$.

Proof See [14].

Thus, problem (6) has a solution if and only if the data and target matrices satisfy Lemma 1. In this case, $\tilde{X}_{rr}, \tilde{X}_{r,n-r}$ and its transpose, $\tilde{X}_{n-r,r}$, are, respectively, computed by (13a) and (13b). Hence, the only remaining step is to compute $\tilde{X}_{n-r,n-r}$ so that $\tilde{X}$ is symmetric and positive definite.

We know that $\tilde{X}$ is symmetric positive definite if and only if there exists a nonsingular lower triangular matrix $L \in \mathbb{R}^{n\times n}$ so that

$$\tilde{X} = LL^T, \tag{17}$$

where L is lower triangular and nonsingular. Considering the block forms in (17), we get

$$\tilde{X}_{rr} = L_{rr}{L_{rr}}^T, \tag{18a}$$

$$\tilde{X}_{r,n-r} = L_{rr}{L_{n-r,r}}^T, \tag{18b}$$

$$\tilde{X}_{n-r,r} = L_{n-r,r}{L_{rr}}^T, \tag{18c}$$

$$\tilde{X}_{n-r,n-r} = L_{n-r,r}{L_{n-r,r}}^T + L_{n-r,n-r}{L_{n-r,n-r}}^T. \tag{18d}$$

Therefore, to compute a symmetric positive definite $\tilde{X}$, (18a)–(18d) must be satisfied. Letting $\tilde{X}_{rr} = \tilde{L}\tilde{L}^T$ be the Cholesky decomposition of $\tilde{X}_{rr}$, $L_{rr} = \tilde{L}$ satisfies (18a). Substituting L_{rr} in (18b), ${L_{n-r,r}}^T$ is computed uniquely by solving the resulting linear system. Since (18c) is transpose of (18b), it does not give any additional information. Finally, to compute a matrix $\tilde{X}_{n-r,n-r}$ to satisfy (18d), it is sufficient to substitute an arbitrary lower triangular nonsingular matrix $L_{n-r,n-r}$ in (18d). The resulting $\tilde{X}_{n-r,n-r}$ gives a symmetric positive definite $\tilde{X}$ as follows:

$$\tilde{X} = \begin{pmatrix} \tilde{X}_{rr} & \tilde{X}_{r,n-r} \\ \tilde{X}_{n-r,r} & \tilde{X}_{n-r,n-r} \end{pmatrix}.$$

Now, based on the above discussion, we outline the steps of our algorithm for solving (6) in the case $rank(D) = r < n$.

We note that to solve (6) for a rank defficient data matrix, the matrix $A = D^T D$ must be converted to a block form $\begin{bmatrix} F & 0 \\ 0 & 0 \end{bmatrix}$ where $rank(A) = r$ and $F \in \mathbb{R}^{r\times r}$. For instance, using complete orthogonal decomposition is also possible instead of the spectral decomposition, and, of course, at the expense of a higher computational cost.

We now provide the details of this algorithm.

(Complete Orthogonal Decomposition) [6] Let $A \in \mathbb{R}^{m\times n}$ be an arbitrary matrix with $rank(A) = r$. There exist $R \in \mathbb{R}^{r\times r}$, $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{n\times n}$ so that $R \in \mathbb{R}^{r\times r}$ is upper triangular, $UU^T = U^T U = I$, $VV^T = V^T V = I$ and $A = U \begin{pmatrix} R & 0 \\ 0 & 0 \end{pmatrix} V^T$.

We note that (6) may be solved by use of complete orthogonal decomposition similar to Algorithm 3. Next, we discuss how to find a particular solution of (6) having proper characteristics.

Algorithm 3 Algorithm 3 - Solving the EIV problem for positive definite linear systems with rank deficient data and target matrices using spectral decomposition.

PROCEDURE PDEIV-RD-Spec(D,T,δ)

1: δ as an upper bound for absolute error is taken to be close to the machine (or user's) zero.
2: Let $A = D^T D$ and compute its spectral decomposition:

$$A = U \begin{pmatrix} S^2 & 0 \\ 0 & 0 \end{pmatrix} U^T.$$

3: Let $B = T^T T$ and $\tilde{B} = U^T B U$.
4: Compute $rank(D) = r$ and let

$$\begin{aligned} \tilde{B}_{rr} &= \tilde{B}(1:r, 1:r), \\ \tilde{B}_{r,n-r} &= \tilde{B}(1:r, r+1:n), \\ \tilde{B}_{n-r,n-r} &= \tilde{B}(r+1:n, r+1:n) \end{aligned}$$

5: Let $\bar{D} = S$, assume $\bar{T}$ satisfies $\tilde{B}_{rr} = \bar{T}^T \bar{T}$.
6: Perform **Algorithm 1** with input parameters $D = \bar{D}$ and $T = \bar{T}$, and let $\tilde{X}_{rr} = X^*$.
7: Solve the linear system (13b) to compute $\tilde{X}_{r,n-r}$ and let $\tilde{X}_{n-r,r} = \tilde{X}_{r,n-r}^T$.
8: **if** $\|U_{n-r}{}^T (B U_r (U_r{}^T B U_r)^{-1} U_r{}^T B - B)\| \geq \delta$ **then stop** ((6) has no solution)
9: **else**
10: Let the Cholesky decomposition of $\tilde{X}_{rr}$ be $\tilde{X}_{rr} = \tilde{L}\tilde{L}^T$ and set $L_{rr} = \tilde{L}$.
11: Solve the lower triangular system (18b) to compute $L_{n-r,r}$.
12: Let $L_{n-r,n-r} \in \mathbb{R}^{(n-r)\times(n-r)}$ be an arbitrary nonsingular lower triangular matrix and compute $\tilde{X}_{n-r,n-r}$ using (18d).
13: Let $\tilde{X} = \begin{pmatrix} \tilde{X}_{rr} & \tilde{X}_{r,n-r} \\ \tilde{X}_{n-r,r} & \tilde{X}_{n-r,n-r} \end{pmatrix}$ and $X^* = U \tilde{X} U^T$.
14: Compute $E = \mathrm{tr}((DX^* - T)(D - TX^{*-1}))$.
15: **end if**.

3.3 Particular Solution

As seen before, in case of a rank deficient data matrix, problem (6) might have infinitely many solutions. These solutions are generated by choosing different values for the arbitrary $(n-r) \times (n-r)$ nonsingular lower triangular matrix $L_{n-r,n-r}$. Here, we describe how to find a particular solution X having some desired characteristics for control problems. Effective rank and condition number [14] are the two important items of interest.

To compute a particular solution of (6) with condition number being smaller than a predefined value K and effective rank being as low as possible, it is sufficient to let $\tilde{X}_{n-r,n-r}$ have a spectral decomposition of the form $\tilde{X}_{n-r,n-r} = W \Sigma^2 W^T$, with $\sigma_1^2 = K\lambda_n(F) - \lambda_1(F)$ and σ_i^2, $i = 2, \ldots, n-r$, having small positive values. The value of $K\lambda_n(F) - \lambda_1(F)$ being negative means that the second condition cannot be satisfied for the assumed value of the upper bound, K.

In Sect. 4, we compare the computational complexity of PDEIV-QR by PDEIV-HH and PDEIV-RD-Spec by PDEIV-RD-COD.

4 Computational Complexity

Here, we study the computational complexity of our algorithms for solving the positive definite linear system of equations using our proposed EIV model.

4.1 Full Column Rank Data Matrix

The computational complexity of PDEIV-QR presented in Sect. 3 for the case of full column rank data matrix is the same as solving an $n \times n$ Riccati equation and is not more than $N_{PDEIV-QR} = \frac{19}{3}n^3$; e.g., see [15]. The computational complexity of PDEIV-Spec is given in Table 1; for details on the indicated computational complexities, see [16].

We note that since the expected complexity for computing the spectral decomposition of an $n \times n$ matrix using the divide and conquer algorithm is approximately $4n^3$ [16], PDEIV-HH has $\frac{2}{3}n^3$ less computing cost than the one for the former algorithm based on spectral decomposition [14].

Comparing the resulting complexities of $N_{PDEIV-QR}$ and $N_{PDEIV-HH}$, it can readily be concluded that, independent of the matrix size, the computational complexity of PDEIV-QR is lower than that of PDEIV-HH.

Table 1 Needed computations in PDEIV-HH and the corresponding computational complexities

Computation	Time complexity
$A = D^T D$	$\frac{1}{2}mn^2$
Householder tridiagonalization of $A \in \mathbb{R}^{n\times n}$	$\frac{10}{3}n^3$
$B = T^T T$	$\frac{1}{2}mn^2$
SU^T	n^2
$\tilde{Q} = SU^T BUS$	$\frac{3}{2}n^3$
Spectral decomposition of $\tilde{Q} \in \mathbb{R}^{n\times n}$	$4n^3$
S^{-1}	n
US^{-1}	n^2
$\tilde{U}\tilde{S}\tilde{U}^T$	$\frac{n^3}{2} + n^2$
$X^* = US^{-1}\tilde{U}\tilde{S}\tilde{U}^T S^{-1}U^T$.	$\frac{3}{2}n^3$
Total time complexity	$N_{PDEIV-HH} = mn^2 + \frac{65}{6}n^3$

4.2 Rank Deficient Data Matrix

The computational complexities of PDEIV-RD-Spec and PDEIV-RD-COD presented in Sect. 4 for the case of rank deficient data matrix are, respectively, provided in Tables 2 and 3.

Table 2 Needed computations in PDEIV-RD-Spec and the corresponding computational complexities

Computation	Time complexity
$A = D^T D$	$\frac{1}{2}mn^2$
Spectral decomposition of $A \in \mathbb{R}^{n\times n}$	$4n^3$
$B = T^T T$	$\frac{1}{2}mn^2$
$\tilde{B} = U^T BU$	$\frac{3}{2}n^3$
$N_{PDEIV-QR}$ ($r \times r$ diagonal data matrix)	$4r^3 + 3r^2$
Solving the linear system (13c)	$(\frac{n^3}{3} + n^2)(n - r)$
Spectral decomposition of $B \in \mathbb{R}^{n\times n}$	$2n^3$
Cholesky decomposition of $\tilde{X}_{rr}$	$\frac{r^3}{6}$
Solving the lower triangular system (18b)	$(\frac{n^2}{2})(n - r)$
Computing $\tilde{X}_{n-r,n-r}$ from (18d)	$r(n - r)^2 + \frac{(n-r)^3}{2}$
Total time complexity	$N_{PDEIV-RD-Spec}{}^*$

$*: N_{PDEIV-RD-Spec} = mn^2 + \frac{15}{2}n^3 + \frac{25}{6}r^3 + (\frac{n^3}{3} + n^2)(n - r) + (\frac{n^2}{2})(n - r) + r(n - r)^2 + \frac{(n-r)^3}{2}$

Table 3 Needed computations in PDEIV-RD-COD and the corresponding computational complexities

Computation	Time complexity
Complete orthogonal decomposition of $D \in \mathbb{R}^{m\times n}$	$4mn^2 - \frac{4}{3}n^3$
$A = D^T D = V_r R^T R V_r{}^T$	$nr^2 + \frac{r^3}{2}$
$B = T^T T$	$\frac{1}{2}mn^2$
$\tilde{B} = V^T BV$	$\frac{3}{2}n^3$
$N_{PDEIV-QR}(r \times r)$	$\frac{19}{3}r^3$
Solving the linear system (13c)	$(\frac{n^3}{3} + n^2)(n - r)$
Spectral decomposition of $B \in \mathbb{R}^{n\times n}$	$2n^3$
Cholesky decomposition of $\tilde{X}_{rr}$	$\frac{r^3}{6}$
Solving the lower triangular system (18b)	$(\frac{n^2}{2})(n - r)$
Computing $\tilde{X}_{n-r,n-r}$ from (18d)	$r(n - r)^2 + \frac{(n-r)^3}{2}$
Total time complexity	$N_{PDEIV-RD-COD}{}^*$

$*: N_{PDEIV-RD-COD} = \frac{9}{2}mn^2 + nr^2 + \frac{13}{6}n^3 + 7r^3 + (\frac{n^3}{3} + n^2)(n - r) + (\frac{n^2}{2})(n - r) + r(n - r)^2 + \frac{(n-r)^3}{2}$

Considering the results for $N_{PDEIV-RD-Spec}$ and $N_{PDEIV-RD-COD}$ in Tables 2 and 3, we have

$$\begin{aligned} & N_{PDEIV-RD-Spec} - N_{PDEIV-RD-COD} \\ & = (mn^2 + \frac{15}{2}n^3 + \frac{25}{6}r^3) - (\frac{9}{2}mn^2 + nr^2 + \frac{13}{6}n^3 + \frac{49}{6}r^3) \\ & = -4mn^2 + \frac{16}{3}n^3 - nr^2 - \frac{17}{6}r^3. \end{aligned}$$

We can see that if $4mn^2 + nr^2 + \frac{17}{6}r^3 > \frac{16}{3}n^3$, then PDEIV-RD-Spec has a lower computational complexity; otherwise, the computational complexity of PDEIV-RD-COD is lower.

Thus, based on the above study, the computational complexity of PDEIV-QR is less than that of PDEIV-Spec, for all matrix sizes. But, for the case of rank deficient data matrix, depending on the matrix size and rank, one of the algorithms PDEIV-RD-Spec and PDEIV-RD-COD may have a lower computational complexity.

5 Numerical Results

Here, some numerical results are reported. We made use of MATLAB 2016a on a Windows 10 machine with a 4.3 GHz CPU and a 8 GB RAM. We generated random test problems with random data and target matrices.

In Sect. 5.1, we report the numerical results corresponding to full column rank data matrices. For each one of the five supposed matrix sizes from 100×10 up to 1000×1000, we generated 100 random test problems. To compare the effect of using QR decomposition or Householder tridiagonalization in our proposed approach, we constructed the Dolan-Moré performance profile. The Dolan-Moré performance profile was introduced in [17] to compare the performance of different algorithms on solving a given problem.

It can be concluded from the generated performance profiles in Figs. 1 and 2 that the required time by PDEIV-QR is lower than that of PDEIV-HH.

Also, to compare our proposed approach (PDEIV) with the interior point method (IntP), discussed in [4], and the method proposed by Hu in [1] (HuM), we constructed the corresponding Dolan-Moré performance profiles in Figs. 3 and 4 to confirm that our proposed method generates solutions with lower standard deviation values in lower computing times. We note that according to Figs. 1 and 2, PDEIV-QR outperforms PDEIV-HH; hence, PDEIV-QR having the best results among our proposed algorithms is compared with IntP and HuM. According to a recent report by Gould and Scott [18], if performance profiles are used to compare more than two solvers, we can determine which solver has the best performance, but we cannot necessarily assess the performance of one solver relative to another that is not the best. Hence, to

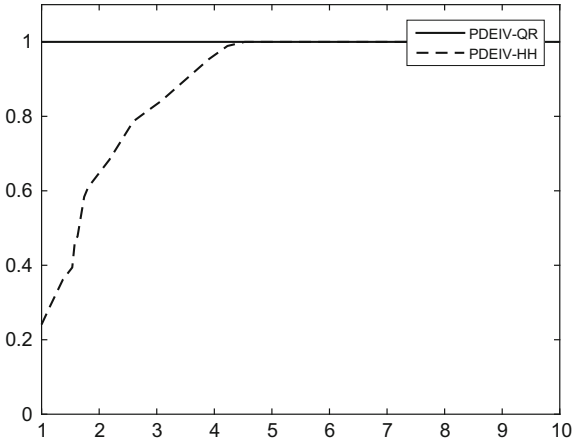


Fig. 1 The Dolan-Moré performance profile (comparing the required times by PDEIV-QR and PDEIV-HH)

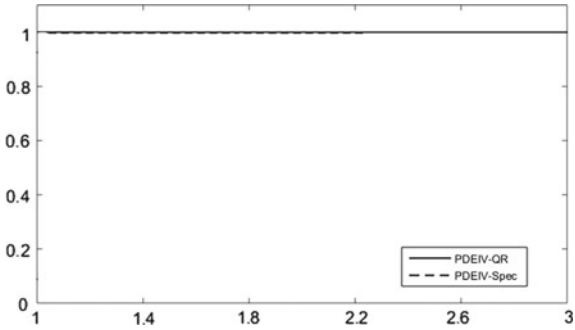


Fig. 2 The Dolan-Moré performance profile (comparing the standard deviation values for PDEIV-QR and PDEIV-HH)

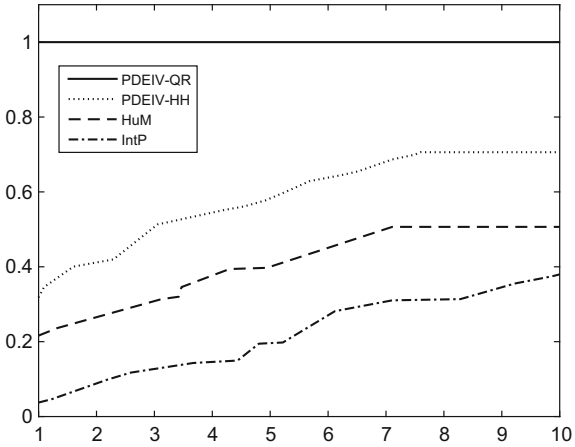


Fig. 3 The Dolan-Moré performance profile (comparing the required time by PDEIV-QR, PDEIV-HH, IntP and HuM)

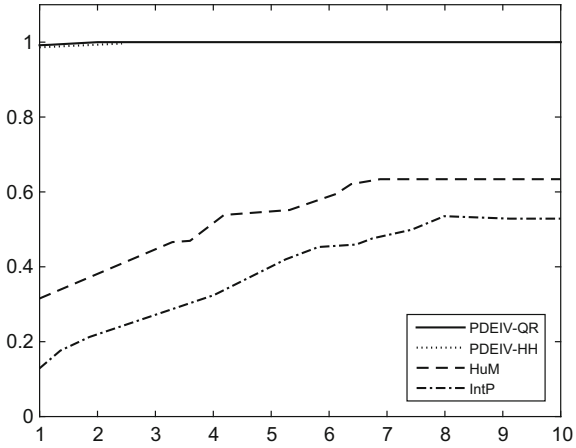


Fig. 4 The Dolan-Moré performance profile (comparing the standard deviation values for PDEIV-QR, PDEIV-HH, IntP and HuM)

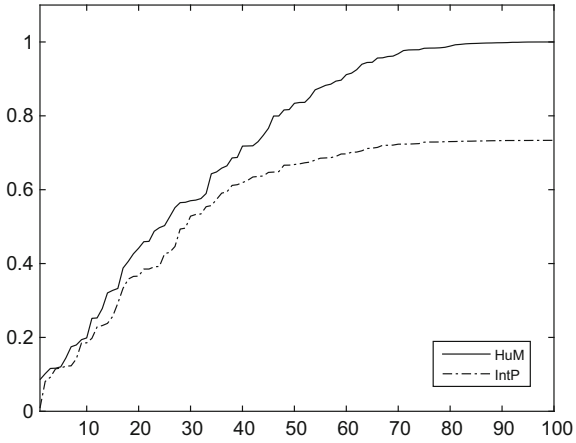


Fig. 5 The Dolan-Moré performance profile (comparing the consumed times for IntP and HuM)

provide a more reliable comparison, we also presented the Dolan-Moré performance profiles for IntP and HuM algorithms separately in Figs. 5 and 6.

These performance profiles show that our proposed approach is able to compute a solution with a smaller value of standard deviation in less computing time.

We also studied the effect of using complete orthogonal decomposition and spectral decomposition in the proposed approach. To compare the efficiency of these decompositions, we constructed the Dolan-Moré performance profiles of required times and standard deviation values for the numerical results produced by PDEIV-RD-Spec and PDEIV-RD-COD in Figs. 7 and 8. The best result obtained by our proposed approach was also compared with the ones obtained by other available methods based on the Dolan-Moré performance profiles as presented in Figs. 9 and 10. Numerical results also confirmed the effectiveness of our proposed algorithms in producing more accurate solutions with lower standard deviation values in lower times.

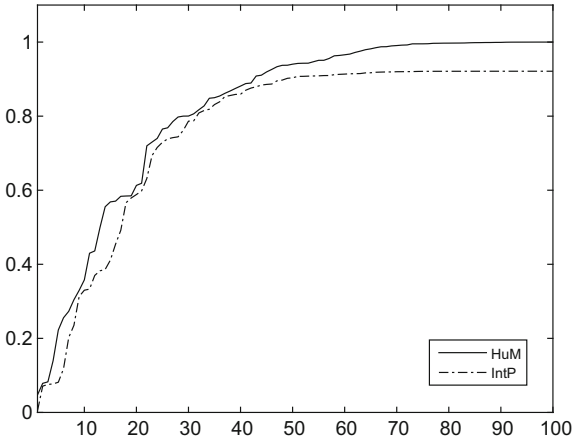


Fig. 6 The Dolan-Moré performance profile (comparing the standard deviation values for IntP and HuM)

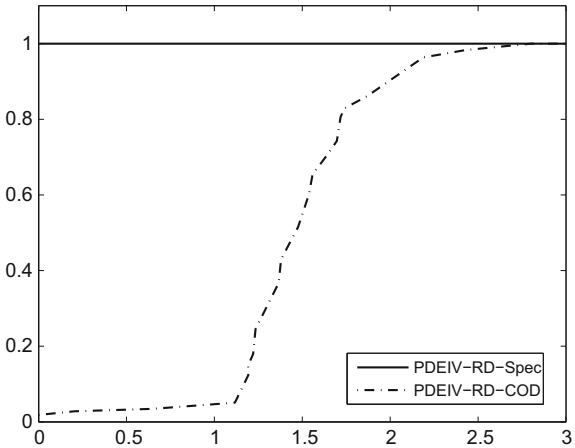


Fig. 7 The Dolan-Moré performance profile (comparing the required times by PDEIV-RD-Spec and PDEIV-RD-COD)

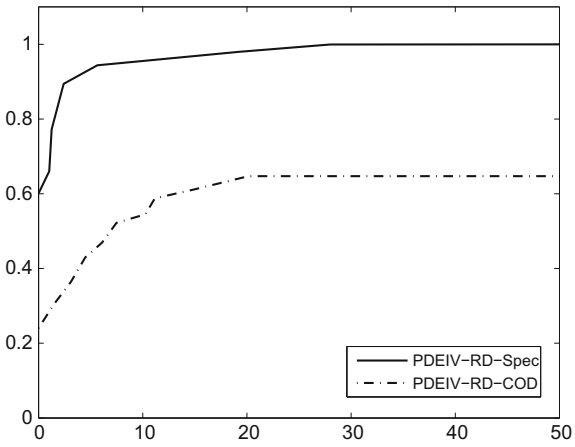


Fig. 8 The Dolan-Moré performance profile (comparing the standard deviation values for PDEIV-RD-Spec and PDEIV-RD-COD)

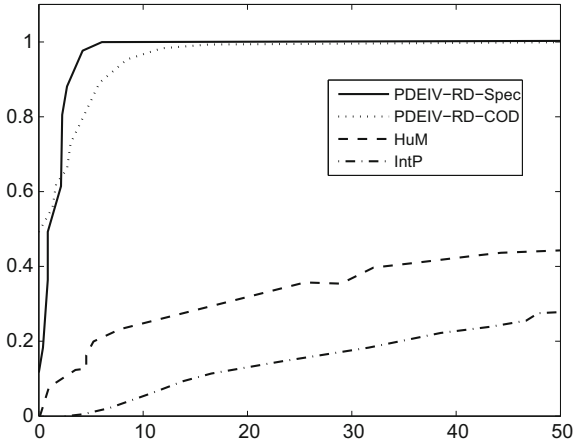


Fig. 9 The Dolan-Moré performance profile (comparing the required times by PDEIV-RD-Spec, IntP and HuM)

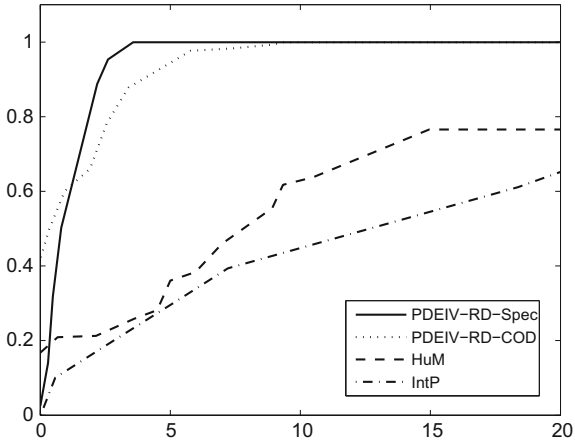


Fig. 10 The Dolan-Moré performance profile (comparing the standard deviation values for PDEIV-RD-Spec, IntP and HuM)

5.1 Full Column Rank Data Matrix

The Dolan-Moré performance profile for the required times by PDEIV-QR and PDEIV-HH given in Fig. 1 confirms that PDEIV-QR computes the solution faster.

However, based on the Dolan-Moré performance profile for the standard deviation values showed in Fig. 1, there is no significant difference between the standard deviation values generated by the two algorithms.

In the following, we compare our proposed approach with the two existing methods, IntP [4] and HuM [1]. The Dolan-Moré performance profiles for the times and standard deviation values shown in Figs. 3 and 4 confirm the efficiency of our proposed approach in computing a solution with smaller value of standard deviation of error in lower time as compared to IntP and HuM. 1000 random test problems with

data and target matrices with sizes less than 1000×500 were generated to construct the Dolan-Moré performance profiles.

To provide a more precise comparison, we now present the Dolan-Moré performance profiles corresponding to IntP and HuM algorithms.

The tolerance for convergence of both existing algorithms IntP and HuM is taken to be 10^{-6} in all implementations.

5.2 Rank Deficient Data Matrix

Here, we report the obtained numerical results, similar to Sect. 5.1, for test problems having rank deficient data matrix.

In Figs. 7 and 8, the Dolan-Moré performance profiles for time and standard deviation values of PDEIV-RD-Spec and PDEIV-RD-COD are shown.

These results show that PDEIV-RD-Spec computes the solution faster, but there is no significant difference between the obtained standard deviations.

To compare our proposed approach with IntP and HuM, the Dolan-Moré profiles corresponding to required times and standard deviations for PDEIV-RD-Spec, as the best-obtained result by our proposed approach, IntP and HuM are represented in Figs. 9 and 10. To construct these profiles, 1000 random test problems with data and target matrices with sizes less than 1000×500 were generated. The Dolan-Moré profiles confirm that PDEIV-RD-Spec and PDEIV-RD-COD compute solutions with smaller values of standard deviations in less times.

Now, we provide a more exact comparison of IntP and HuM algorithms in Figs. 11 and 12.

Considering the numerical results reported in this section, for the data matrix D having full column rank, we observe that required time and standard deviation in error entries by PDEIV-QR is smaller than that of PDEIV-HH. And, if the data matrix is rank deficient, we observe that required times by PDEIV-RD-Spec is smaller than that of PDEIV-RD-COD, required time and standard deviation values for both PDEIV-RD-Spec and PDEIV-RD-COD are considerably less than those of IntP and HuM, and the standard deviation values for PDEIV-RD-Spec is smaller than those of the other three methods.

5.3 A Real Example

Here, we provide a real example which is related to approximating the mass inertia matrix for a hexagonal solid structure [19]. Let

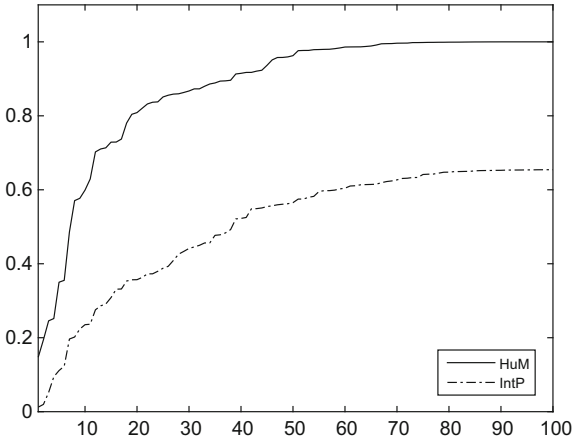


Fig. 11 The Dolan-Moré performance profile (comparing the consumed times for IntP and HuM)

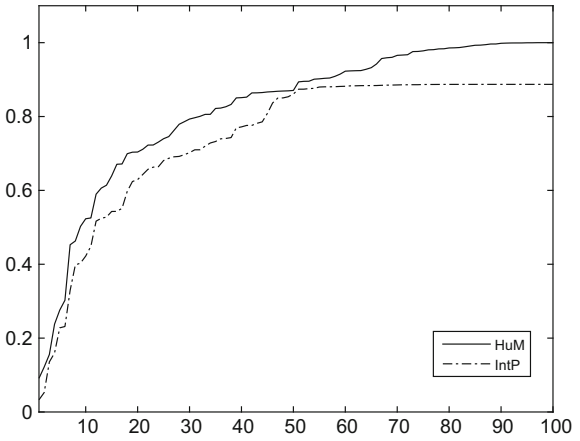


Fig. 12 The Dolan-Moré performance profile (comparing the standard deviation values for IntP and HuM)

$$D = \begin{pmatrix} 1.2354 & 0.9801 & 0.8657 & 0.9635 & 0.8027 & 0.7065 \\ 0.4531 & 1.5644 & 0.8983 & 0.6897 & 0.7125 & 0.9129 \\ 0.5412 & 0.6875 & 1.8435 & 0.8933 & 0.6802 & 0.7983 \\ 0.6432 & 0.7123 & 0.9102 & 1.4655 & 0.7655 & 0.8444 \\ 0.7863 & 0.9834 & 0.4892 & 0.6753 & 1.8765 & 1.1092 \\ 0.5641 & 0.8993 & 0.6738 & 0.7862 & 0.4984 & 1.6936 \end{pmatrix} \tag{19}$$

and

$$T = \begin{pmatrix} 2.8965 & 1,2245 & 1.3766 & 1.2198 & 0.9567 & 0.8342 \\ 0.9871 & 2.2341 & 0.8873 & 0.6981 & 0.7763 & 1.2546 \\ 0.8769 & 1.3126 & 2.3764 & 0.9762 & 1.2189 & 0.8561 \\ 0.9327 & 0.8126 & 1.3879 & 2.8671 & 0.8064 & 0.9439 \\ 0.8321 & 0.9065 & 0.8743 & 0.6576 & 2.6549 & 1.3244 \\ 0.7652 & 0.6983 & 0.9027 & 1.0978 & 1.4653 & 2.7557 \end{pmatrix}. \tag{20}$$

Our proposed Householder approach leads to a symmetric and positive definite solution which minimizes the error value $\mathrm{tr}((DX - T)^T(D - TX^{-1}))$. The standard deviation value for the computed solution is equal to 6.3422×10^{-4} which is considerably smaller than the standard deviation value 2.2189×10^2, obtained by the IntP algorithm.

6 Concluding Remarks

We first reviewed our recently proposed error in variables (EIV) model to solve positive definite linear systems and discussed our established necessary and sufficient conditions of optimality. We also reviewed and refined the algorithms for solving positive definite linear systems, offering three main desirable features. First, consideration of our proposed EIV model in both the data and target matrices admitted a more realistic problem formulation. Second, our proposed algorithm computed the exact solution directly, and, as shown by our numerical results on randomly generated test problems, was faster than two other existing methods, IntP and HuM. Our proposed algorithms were implemented in two variants based on QR and Householder decompositions. According to the reported numerical results, the QR variant outperformed the Householder variant with respect to the required computing time. The generated Dolan-Moré performance profiles also confirmed the efficiency of our proposed approach by computing the solutions faster than other methods. The lower computing time was mainly due to the fact that the produced optimization problem by our newly defined EIV model was not as complicated as the one used by the other methods. Numerical results also showed smaller standard deviation of the errors in the target matrix and lower values of effective rank and condition number, as being desired by control problems.

Acknowledgements The authors thank the Research Council of Sharif University of Technology for supporting this work.

References

1. Hu, H., Olkin, I.: A numerical procedure for finding the positive definite matrix closest to a patterned matrix. Stat. Probab. Lett. **12**, 511–515 (1991)
2. McInroy, J., Hamann, J.C.: Design and control of flexure jointed hexapods. IEEE Trans. Rob. Autom. **16**(4), 372–381 (2000)
3. Poignet, P., Gautier, M.: Comparison of Weighted Least Squares and Extended Kalman Filtering Methods for Dynamic Identification of Robots. In: Proceedings of the IEEE Conference on Robotics and Automation, San Francisco, CA, USA, pp. 3622–3627 (2000)
4. Krislock N. G.: Numerical Solution of Semidefinite Constrained Least Squares Problems, M. Sc. Thesis, University of British Colombia (2003)
5. Rebonato, R., Jãckel, P.: The most general methodology to create a valid correlation matrix for risk management and option pricing purposes. J. Risk **2**, 17–27 (1999)

6. Golub, G.H., Van Loan, C.F.: Matrix Computation, 4th edn. JHU Press, Baltimore (2012)
7. Hayami, K., Yin, J.F., Ito, T.: GMRES method for least squares problems. SIAM. J. Matrix Anal. Appl. **31**(5), 2400–2430 (2010)
8. Cheng, C.L., Kukush, A., Mastronardi, N., Paige, C., Van Huffel, S.: Total least squares and errors-in-variables modeling. Comput. Stat. Data Anal. **52**, 1076–1079 (2007)
9. Golub, G.H., Van Loan, C.F.: An analysis of the total least squares problem. SIAM J. Numer. Anal. **17**, 883–893 (1980)
10. Paige, C.C., Strakoš, Z.: Scaled total least squares fundamentals. Numer. Math. **91**, 117–146 (2000)
11. Van Huffel, S., Vandewalle, J.: The Total Least Squares Problem: Computational Aspects and Analysis. SIAM, Philadelphia (1991)
12. Kang, B., Jung, S., Park, P.: A new iterative method for solving total least squares problem. In: Proceeding of the 8th Asian Control Conference (ASCC), Kaohsiung, Taiwan (2011)
13. Van Huffel, S., Vandewalle, J.: Algebraic connections between the least squares and total least squares problems. Numer. Math. **55**, 431–449 (1989)
14. Bagherpour, N., Mahdavi-Amiri, N.: A new error in variables model for solving positive definite linear system using orthogonal matrix decompositions. Numer. Algorithms **72**(1), 211–241 (2016)
15. Higham, N.J.: Functions of Matrices: Theory and Computation. SIAM, Philadelphia (2008)
16. Demmel, J.W.: Applied Numerical Linear Algebra, 3rd edn. SIAM, Philadelphia (1996)
17. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Program. **91**, 201–213 (2012)
18. Gould, N., Scott, J.: A note on performance profiles for benchmarking software. ACM Trans. Math. Softw. **43**(2), 1–5 (2016)
19. Griffith, D.A., Luhanga, U.: Approximating the inertia of the adjacency matrix of a connected planar graph that is the dual of a geographic surface partitioning. Geogr. Anal. **43**(4), 383–402 (2011)

Sparse Direct Solution on Parallel Computers



Iain Duff, Florent Lopez and Stojce Nakov

Abstract We describe our recent work on designing algorithms and software for solving sparse systems using direct methods on parallel computers. This work has been conducted within an EU Horizon 2020 Project called NLAFET. We first discuss the solution of large sparse symmetric positive definite systems. We use a runtime system to express and execute a DAG-based Cholesky factorization. The runtime system plays the role of a software layer between the application and the architecture and handles the management of task dependencies as well as task scheduling and maintaining data coherency. Although runtime systems are widely used in dense linear algebra, this approach is challenging for sparse algorithms because of the irregularity and variable granularity of the DAGs arising in these systems. We have implemented our software using the OpenMP standard and the runtime systems StarPU and PaRSEC. We compare these implementations to `HSL_MA87`, a state-of-the-art DAG-based solver for positive definite systems. We demonstrate comparable performance on a multicore architecture. We also consider the case when the matrix is symmetric indefinite. For highly unsymmetric systems, we use a completely different approach based on developing a parallel version of a Markowitz threshold ordering. This work is less advanced but we discuss some of the algorithmic challenges involved. Finally, we briefly discuss using a hybrid direct-iterative solver that combines the best of the two approaches and enables the solution of even larger problems in parallel.

Keywords Sparse symmetric systems · Unsymmetric sparse systems
Direct methods · Hybrid methods · Runtime systems · StarPU · OpenMP
PaRSEC · Block cimmino · Markowitz threshold

I. Duff (✉) · F. Lopez · S. Nakov
STFC-RAL, Harwell Campus, Didcot OX11 0QX, UK
e-mail: iain.duff@stfc.ac.uk

F. Lopez
e-mail: florent.lopez@stfc.ac.uk

S. Nakov
e-mail: stojce.nakov@stfc.ac.uk

© Springer International Publishing AG, part of Springer Nature 2018

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*,
Springer Proceedings in Mathematics & Statistics 235,
https://doi.org/10.1007/978-3-319-90026-1_4

1 Introduction

We discuss recent work on the solution of large sparse equations on parallel computers using direct methods in the context of an EU Horizon 2020 Project called NLAFET (Parallel Numerical Linear Algebra for Future Extreme Scale Systems).[1] This is the H2020 FET-HPC Project 671633, and it involves only four partners. The coordinator is Bo Kågström of Umeå University in Sweden, and the other principal investigators are Iain Duff (STFC, UK), Jack Dongarra (University of Manchester, UK) and Laura Grigori (INRIA, Paris). The Project started on 1 November 2015 and will finish on the 31 October 2018.

A major aim of the project is to enable a radical improvement in the performance and scalability of a wide range of real-world applications relying on linear algebra software for future extreme-scale systems.

The key goals are:

- Development of novel architecture-aware algorithms that expose as much parallelism as possible, exploit heterogeneity, avoid communication bottlenecks, respond to escalating fault rates and help meet emerging power constraints.
- Exploration of advanced scheduling strategies and runtime systems, focusing on the extreme scale and strong scalability in multi/many-core and hybrid environments.
- Design and evaluation of novel strategies and software support for both offline and online auto-tuning.
- The results will appear in an open-source NLAFET software library.

1.1 NLAFET Workpackage Overview

The NLAFET Project consists of seven interlocking workpackages as shown in Fig. 1. The three workpackages WP1, WP5 and WP7 concern administration, applications and dissemination while the other four workpackages define our research agenda. We list the workpackages below.

- WP1: Management and coordination.
- WP2: Dense linear systems and eigenvalue problem solvers.
- WP3: Direct solution of sparse linear systems.
- WP4: Communication-optimal algorithms for iterative methods.
- WP5: Challenging applications—a selection.
  Material science, power systems, study of energy solutions and data analysis in astrophysics.
- WP6: Cross-cutting issues.
  Scheduling and runtime systems, auto-tuning, fault tolerance.
- WP7: Dissemination and community outreach.

[1] www.nlafet.eu.

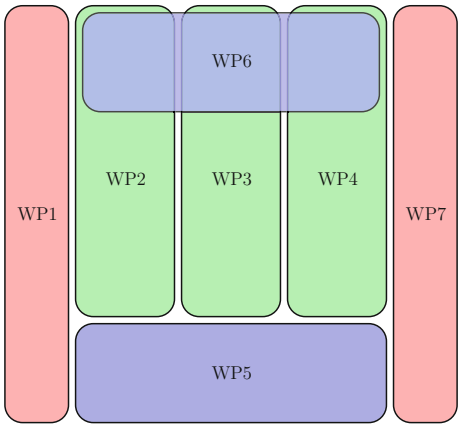


Fig. 1 Overview of NLAFET Project

This paper is primarily concerned with WP3 in which we have four tasks, namely

- T3.1 Lower bounds on communication for sparse matrices.
- T3.2 Direct methods for (near-)symmetric sparse systems.
- T3.3 Direct methods for highly unsymmetric sparse systems.
- T3.4 Hybrid direct-iterative methods.

We will primarily be discussing the work in tasks T3.2 and T3.3 but, before we do so, we will first define what we mean by a direct method.

2 Direct Solution of Sparse Equations

When solving the linear system

$$Ax = b,$$

where the sparse matrix A is of large dimension, typically 10^6 or greater, by direct methods we will consider the factorization:

$$P_r A P_c = LU,$$

where L is a sparse lower triangular matrix and U is a sparse upper triangular matrix and the permutations P_r and P_c are chosen to preserve sparsity and maintain stability. When A is symmetric, then $P_r = P_c^T$, and the factorization can be written as LL^T (Cholesky) or LDL^T; this latter factorization is needed when A is not positive definite.

There are several points to note about sparse direct methods:

- Black box solvers are available.
- Routinely solving problems of order in millions.
- Complexity can be low.
- Storage requirements can be very high. Although almost linear storage in 2D.
- Target is half asymptotic speed of `GEMM`.

We now discuss how we implement this factorization in the case of Task 3.2.

3 Task 3.2 Direct Methods for (Near-)Symmetric Systems

Task 3.2 considers the factorizations LL^T, LDL^T and LU, and all the algorithms will be based on a tree-based representation of the factorization that we describe in Sect. 3.1. We will follow the approach taken by the code `HSL_MA87` [19] for obtaining more fine-grained parallelism by using directed acyclic graphs (DAGs) rather than trees. The use of a DAG as opposed to a tree is illustrated in Sect. 4.4. `HSL_MA87` is based on a low-level synchronization API for handling the parallelism. The main novelty in our approach is to use runtime systems to both express and execute the DAG. We do this working closely with our NLAFET partners in WP6.

A key point about sparse factorization methods is that the kernels involve operations on small dense matrices and we design energy-efficient, low-communication dense kernels for use within our sparse factorizations for both positive definite and indefinite systems.

Within the framework of NLAFET, we are primarily concerned with the runtime systems StarPU and OpenMP (using task features in Version 4.0 or above) both using a sequential task flow (STF) model, and PaRSEC using a parametrized task graph (PTG) model. In all cases, the structure supplied to the runtime system is a DAG.

3.1 Tree-Based Factorization

We feel it is useful to illustrate our approach to sparse factorization using the small 7×7 matrix:

$$
\begin{array}{c|ccccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
1 & \times & \times & \times & \bullet & \bullet & \times & \bullet \\
2 & \times & \times & \times & \bullet & \bullet & \bullet & \times \\
3 & \times & \times & \times & \bullet & \bullet & \bullet & \bullet \\
4 & \bullet & \bullet & \bullet & \times & \times & \times & \bullet \\
5 & \bullet & \bullet & \bullet & \times & \times & \bullet & \bullet \\
6 & \times & \bullet & \bullet & \times & \bullet & \times & \times \\
7 & \bullet & \times & \bullet & \bullet & \bullet & \times & \times
\end{array}
\tag{1}
$$

where the entries marked $\times$ are nonzero and those marked $\bullet$ are zero. Then, if we consider eliminating the first three rows and columns at the first step, the matrix on which we perform the eliminations is

$$\begin{array}{c|ccccc} & 1 & 2 & 3 & 6 & 7 \\ 1 & \times & \times & \times & \times & \\ 2 & \times & \times & \times & \times & \times \\ 3 & \times & \times & \times & \times & \times \\ \\ 6 & \times & \times & \times & \times & \times \\ 7 & & \times & \times & \times & \times \end{array}$$

because the first three entries in rows and columns 4 and 5 are zero.

If we eliminate rows and columns 4 and 5 at step 2, then this can be performed within the matrix:

$$\begin{array}{c|ccc} & 4 & 5 & 6 \\ 4 & \times & \times & \times \\ 5 & \times & \times & \times \\ \\ 6 & \times & \times & \times \end{array}$$

The remaining part of the original matrix can then be factorized, but we need first to update the entries according to the first two pivot steps. Thus, step 3 can be expressed as:

$$\begin{array}{c|cc} & 6 & 7 \\ 6 & \times & \times \\ 7 & \times & \times \end{array} + \begin{array}{c|cc} & 6 & 7 \\ 6 & \times & \times \\ 7 & \times & \times \end{array} + \begin{array}{c|c} & 6 \\ 6 & \times \end{array} \longrightarrow \begin{array}{c|cc} & 6 & 7 \\ 6 & \times & \times \\ 7 & \times & \times \end{array}$$

where the three matrices summed on the left are from the original matrix and from the Schur complements of the factorizations at steps 1 and 2, respectively. The resulting dense matrix on the right is then factorized to complete the factorization of the 7×7 matrix (1).

The factorization can be represented by the tree shown in Fig. 2 where the number in each node corresponds to the pivot step eliminating the variables shown in parenthesis and data must be passed along the tree edges from nodes 1 and 2 to node 3. At each node, a small dense matrix, called a frontal matrix, is partially factorized and the Schur complement is passed for assembly at the parent node of the tree. The reader is referred to [10] for a longer and more detailed discussion of the use of assembly trees in sparse factorization.

This tree representation can be extended to any symmetric matrix, and the sparse factorizations will have similar kernels to those used in the small example. The computation at a node involves dense factorization. Pivots are chosen from the top left block in Fig. 3, but elimination operations are performed on the whole frontal matrix. Rows and columns of the factors can be stored, and the resulting Schur complement (bottom right block) is passed up the tree for future assemblies.

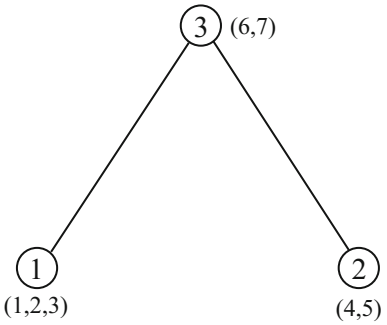


Fig. 2 Assembly tree for 7 × 7 example

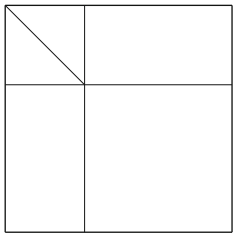


Fig. 3 Generic frontal matrix showing fully summed block

4 Parallelism in Sparse Direct Methods

There are several levels of parallelism in solving sparse systems that we discuss in this section. These levels can all contribute to an efficient parallel solution and can result in potentially high levels of exploitation of extreme-scale computers. It is the intention of our project to investigate the combination of these techniques to obtain good scalability.

We list the levels below and discuss each in the subsections of this section.

- Partitioning.
- Tree-level parallelism.
- Node parallelism (including multithreaded BLAS).
- Inter-node parallelism.

4.1 Partitioning

In many applications, the matrix can be reordered and partitioned so that all the nonzero entries lie in blocks. For example, if the symmetric system is reducible then

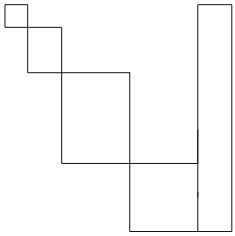


Fig. 4 A singly bordered block diagonal form

the resulting form is block diagonal and the factorization of each block can proceed independently in parallel.

For unsymmetric matrices, reducibility corresponds to a block triangular form and a generalization of this is a bordered block diagonal form, shown in Fig. 4, that we exploit using the methods discussed in Sect. 8.

4.2 Tree-Level Parallelism

We already showed an assembly tree for our 7×7 example (1) in Fig. 2. In the more general case, we can represent the factorization by a similar assembly tree with many more nodes and edges. We show an example of a larger tree in Fig. 5.

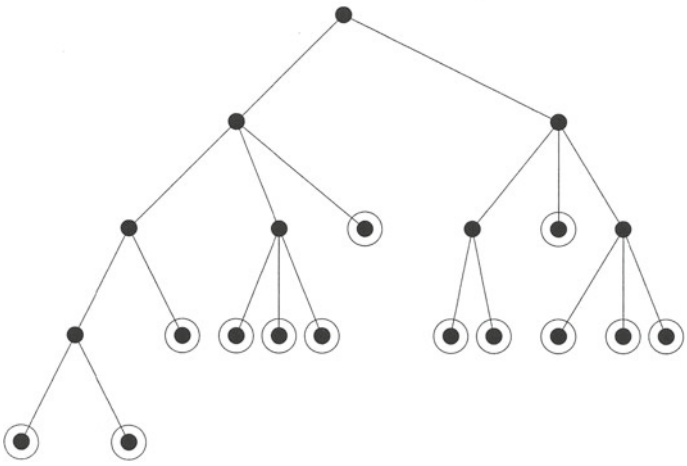


Fig. 5 Start of factorization. Work corresponding to leaf nodes (circled) can proceed immediately and independently

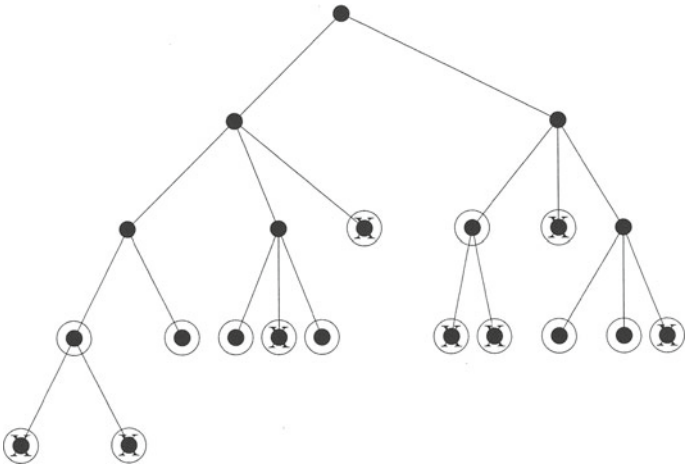


Fig. 6 Part-way through the factorization. When the dense factorization at a node is finished, a cross is superimposed on the circle

As we see in this figure, there is plenty of parallelism available from the tree at the start of the factorization. As the elimination operations at a node complete, other nodes receive the Schur complement information from their children and factorizations can proceed when the data has been received. We show the situation part-way through the factorization in Fig. 6.

4.3 Node Parallelism

Node parallelism comes from the parallel execution of kernels effecting the dense factorization at a node of the tree. Indeed, we can potentially use any tricks developed for parallel dense factorization so that we see immediately the much greater potential for parallelism in the sparse case.

Table 1 Statistics on front sizes in assembly tree. From Duff, Erisman, Reid [10]

Matrix	Tree		Leaf nodes		Top 3 levels		
	Order	Nodes	No.	Av. size	No.	Av. size	% ops
bratu3d	27,792	12,663	11,132	8	296	37	56
cont-300	180,895	90,429	74,673	6	10	846	41
cvxqp3	17,500	8336	6967	4	48	194	70
mario001	38,434	15,480	8520	4	10	131	25
ncvxqp7	87,500	41,714	34,847	4	91	323	61
bmw3_2	227,362	14,095	5758	50	11	1919	44

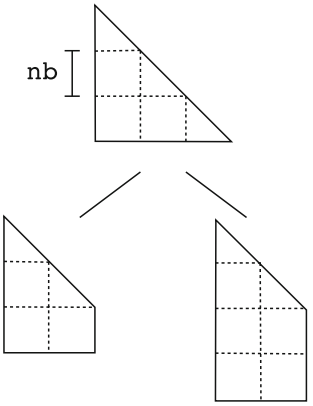


Fig. 7 Part of tree

We show some tree statistics in Table 1 for some medium-sized problems from a range of applications. Although there are many tasks near the leaf nodes, we note that the dimensions of the matrices are small so there is little node parallelism but plenty of tree parallelism. Near the root node, there is not much tree parallelism but the nodes are large, that is the dense matrices at these nodes are of large dimension and so there is plenty of node parallelism. It is this balance that encourages us to believe that good levels of scalability can be obtained and that our target of half the asymptotic speed might be achievable.

4.4 Inter-node Parallelism

The bottleneck in parallel algorithms just based on the assembly tree and employing the parallelism discussed in Sects. 4.2 and 4.3 is that a node has to wait for all its children to complete before it is available to start its own processing. We follow previous work on qr_mumps [1, 2, 6] and `HSL_MA87` [19] to overcome this limitation by dividing the factorization into subblocks (called tiles) as shown in Fig. 7 and then defining the dependency of the resulting task-based approach by using a directed acyclic graph or DAG. We illustrate such a DAG in Fig. 8.

In the DAG shown in Fig. 8, the letters in the circles correspond to kernels involved in the factorization of the nodes of the tree. These kernels are:

- the tasks `f` correspond to the Cholesky factorization of a block corresponding to a tile on the diagonal,
- the tasks `s` represent a triangular solve on a subdiagonal block using a factor computed by a task `f`,
- the tasks `u` perform an update of a block within the node using the blocks created by the kernels `s`, and

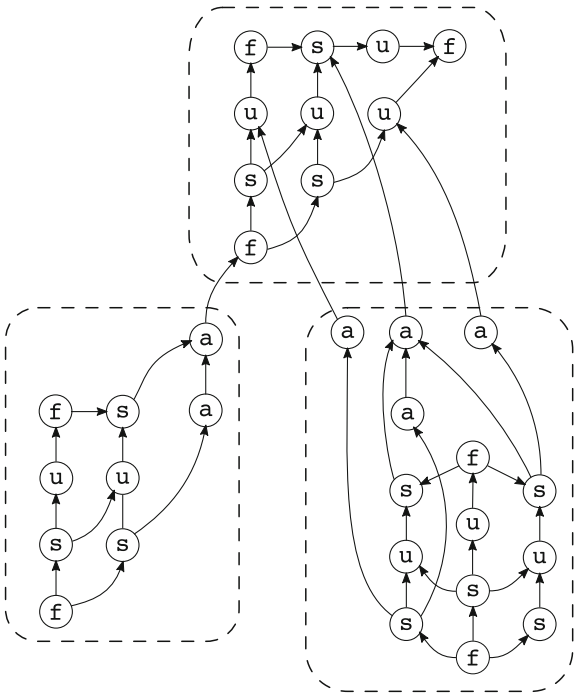


Fig. 8 Directed acyclic graph

- the tasks `a` represent an update between nodes with factorizations at one node updating a block in an ancestor node.

5 Experiments on Symmetric Positive Definite Systems

We run our experiments on a multicore Haswell machine equipped with two Intel(R) Xeon(R) E5-2695 v3 CPUs with fourteen cores each (28 cores in total). Each core is clocked at 2.3 GHz. The asymptotic performance of DGEMM using PLASMA was 768 Gflop/s. We have implemented a sparse factorization routine, SpLLT, exploiting the parallelism described in Sects. 4.2–4.4 and have used both OpenMP and the StarPU runtime system [12] using the STF model. We compare runs of this code with `HSL_MA87` on a range of test problems whose characteristics are given in Table 4 in the Appendix. We see, in Fig. 9, that the performance is generally comparable to the hand-coded HSL code although rather poor performance is seen in a few matrices, for example matrices 15, 19 and 24. A problem with using the runtime systems is that when the tasks are small, the overhead in setting them up in the runtime system predominates and the overhead is particularly high for StarPU. Another reason why

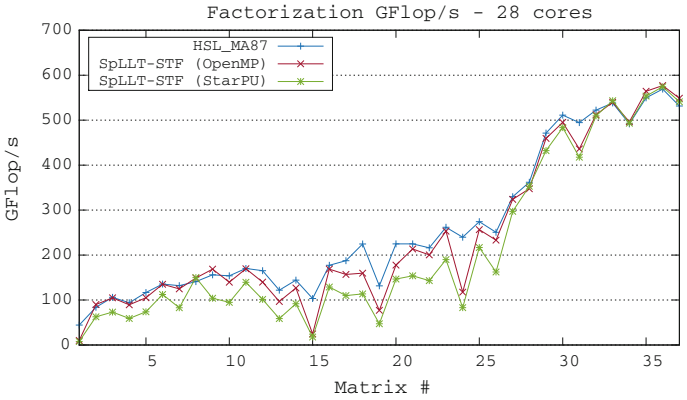


Fig. 9 Performance of sparse Cholesky code SpLLT using STF runtime systems

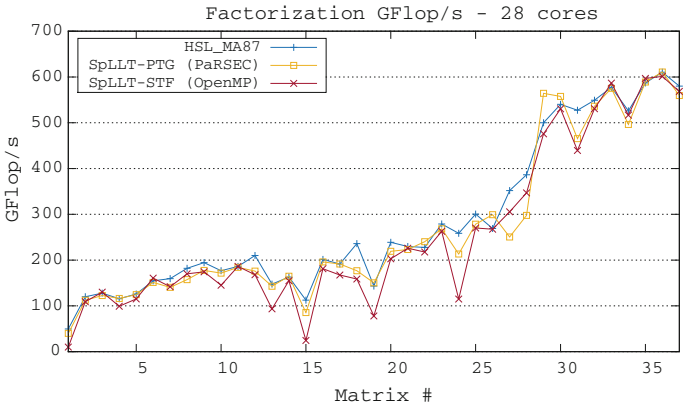


Fig. 10 Performance of sparse Cholesky code SpLLT using PaRSEC

the STF model can show poor performance is that the DAG is traversed sequentially and so some parallelism is missed. To avoid this issue with STF models, we also implemented our algorithm using the PaRSEC runtime system [13] that implements a PTG model. We see in Fig. 10 that PaRSEC performs much better than the STF models on the matrices that we just identified, but the cost of using a runtime system still penalizes cases where there are many small tasks. We can avoid some of this by grouping together into a single task nodes near the leaves of the tree. We call this tree pruning and discuss this in the next section.

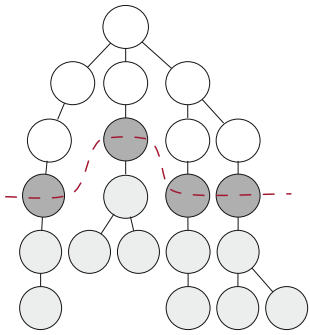


Fig. 11 An illustration of tree pruning

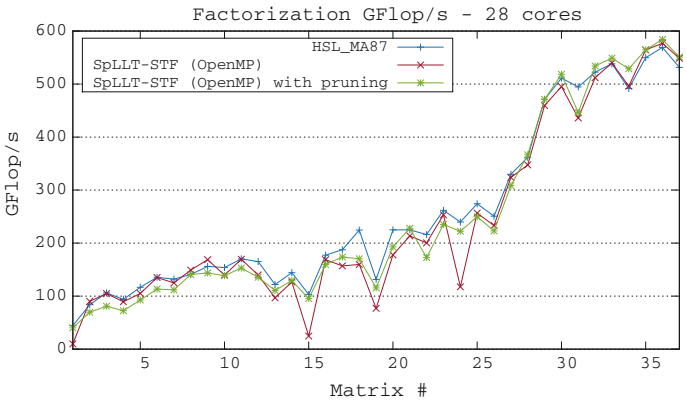


Fig. 12 Effect of tree pruning on OpenMP version

5.1 Tree Pruning Strategy

What we mean by tree pruning is that we do not pursue full-tree parallelism by allocating tasks right down to the level of the leaves but we combine the operations in a subtree as a single task so that the granularity of work in that task is increased. This is illustrated in Fig. 11 where instead of processing the tree right down to the six leaf nodes, we only exploit tree parallelism down to the level of the four heavily shaded nodes and the subtree, for which each is a root, is processed as a single task. This strategy is commonly used when performing a static distribution of work in a multiprocessor environment, for example by MUMPS [3] and, in a shared memory environment, by qr_mumps [6].

We see the effect of doing this pruning in Fig. 12 where the performance on matrices 15, 19 and 24 is clearly much better. We note that the version with pruning is not always better because of the aforementioned loss of parallelism.

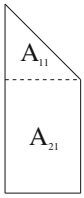


Fig. 13 Frontal matrix showing fully summed columns

6 Symmetric Indefinite Matrices

If the matrix is indefinite, then numerical pivoting is needed. A simple example is the matrix

$$\begin{bmatrix} 0 & \times \\ \times & 0 \end{bmatrix}$$

where it is clear that, whatever, the value of the entries $\times$, a Cholesky factorization will fail because of the zeros on the diagonal. We note that, if $\times$ is nonzero, the matrix is nonsingular. The good news is that we can stably factorize an indefinite matrix using only 1×1 and 2×2 pivots [4, 5].

As is standard in sparse factorization, we use threshold rather than partial pivoting so we want

$$\|Pivot\| \geq u \times \|Largest\ entry\ in\ column\| \tag{2}$$

where u is the threshold parameter ($0 < u \leq 1$) so that a value of 1.0 for the threshold parameter would be equivalent to partial pivoting while relaxing this value gives us more scope to choose pivots on sparsity grounds while still retaining some control over the stability of the factorization. This trade-off is described in detail by [10].

For the symmetric indefinite case, where 2×2 pivots may be needed to preserve stability and symmetry, Duff and Reid [15] recommend, for each 2×2 pivot, the test

$$\left| \begin{bmatrix} a_{kk}^{(k)} & a_{k,k+1}^{(k)} \\ a_{k+1,k}^{(k)} & a_{k+1,k+1}^{(k)} \end{bmatrix}^{-1} \right| \begin{bmatrix} \max\limits_{j \neq k,k+1} |a_{kj}| \\ \max\limits_{j \neq k,k+1} |a_{k+1,j}| \end{bmatrix} \leq \begin{bmatrix} u^{-1} \\ u^{-1} \end{bmatrix}, \tag{3}$$

where $|A|$ is the matrix with each entry the modulus of the corresponding entry of A and $u \leq 0.5$. This corresponds to

$$|a_{kk}^{(k)}|^{-1} \max_i |a_{ik}^{(k)}| \ \leq \ u^{-1} \tag{4}$$

for the 1×1 case.

If we look at the frontal matrix shown in Fig. 13, then pivots can only be chosen from A_{11}. Various pivoting strategies can be adopted, some of which we now discuss.

6.1 Threshold Partial Pivoting

We can use the threshold partial pivoting (TPP) algorithm defined by Eqs. (2)–(4). The problem is that, in Fig. 13, entries in A_{21} may be large enough to prevent a potential pivot from A_{11} satisfying the threshold test (2). In a normal dense matrix factorization, such large entries could be used as the off-diagonal entry in a 2×2 block pivot but that is not possible in this context since we do not have available all the entries in the rows of the matrix corresponding to rows in A_{21} as they need contributions from further up the tree. Pivots (with associated row and column) that cannot be used and are passed to ancestors in the tree are called delayed pivots.

We are thus faced with a choice of just restricting pivoting to A_{11} and hoping that large entries in A_{21} do not cause us problems. This results in an unstable algorithm so normally some precaution is taken to guard against that. Thus, it is common to do some steps of iterative refinement that will either give an accurate answer or will alert the user to the fact that there are problems with the factorization. This is the approach adopted by PARDISO [23].

The alternative is to conduct the elimination column by column and, if a column has too large an entry in the part of the column in A_{21}, to avoid pivoting on that column and to leave it in the uneliminated Schur complement or contribution block for passing to the ancestors in the tree. This does not compromise our threshold pivoting algorithm since the said column will eventually be available for pivoting, certainly at the root node if not before. The problem with this approach is that it does not lend itself to good exploitation of parallelism. We are working a column at a time and need to update later columns before we can test them for pivoting. We must search for the maximum entry in the column that requires access to memory and potentially memory on different processors in a distributed memory environment. However, this is usually the algorithm of choice for serial codes and we can obtain some degree of parallelism by careful implementation. This is the approach adopted by the HSL code `HSL_MA97` [20].

In fact, for many indefinite systems, if the matrix is scaled well beforehand and the threshold parameter is not set too high, there are few instances when a large entry in A_{21} prevents a pivot being chosen [21]. We discuss an algorithm that exploits this in the next section.

6.2 A Posteriori Threshold Pivoting

The intention of a posteriori threshold pivoting (APTP) [18] is to concentrate on obtaining good parallelism in the case when no pivots are delayed but to be able to continue with a stable factorization when there are delayed pivots albeit with a slight loss in efficiency.

In this approach, we first tile the frontal matrix and compute the factors for a block of columns in A_{11} working only within this block. We then use these factors

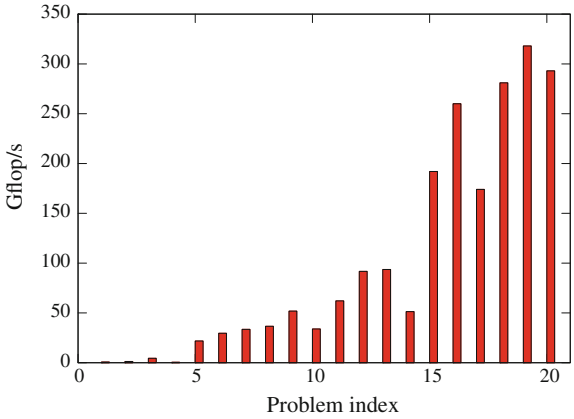


Fig. 14 Performance of SSIDS factorization for easy indefinite matrices on the 28-node Haswell machine

to update in parallel tiles in A_{21} within this block of columns. When doing this, we monitor the size of the entries of L that we are creating and check whether their absolute value is less than u^{-1}. If that is so, then we use the blocks to update the rest of the frontal matrix. If it is not, then we do not trust any columns in the block to the right of the failed entry and we backtrack to the situation at that time, flag the column as being nonpivotal and try to continue the factorization. The main penalty here is that we must also store the unmodified data so that a backtrack can be performed. At worst, the failed columns will correspond to delayed pivots in the TPP algorithm and will be passed with the Schur complement to ancestor nodes.

This algorithm is used in the SSIDS code [18], and the kernel that implements it is used in the NLAFET code SpLDLT. We illustrate its performance on a set of matrices of increasing size in Fig. 14 on our 28-node Haswell machine. The attributes of the test matrices are shown in Table 5 in the Appendix, and they are called *easy* indefinite because they do not have any delayed pivots if the matrices are scaled beforehand. On the Haswell machine, the `GEMM` kernel runs at 768 Gflop/s so that, even with pivoting, we are close to achieving our target of half the asymptotic speed.

6.3 Numerical Pivoting in the Indefinite Case

We have discussed three numerical pivoting options for symmetric indefinite factorizations and have identified codes that implement each of these. In this section, we run some tests on some highly indefinite systems to see how these strategies compare in practice. Some of the results are from the technical report by Hogg [18]. The codes that we are comparing are: SSIDS using APTP, `HSL_MA97` that uses TPP, and PARDISO as implemented in MKL 11.0.3 that factorizes the block A_{11} without any

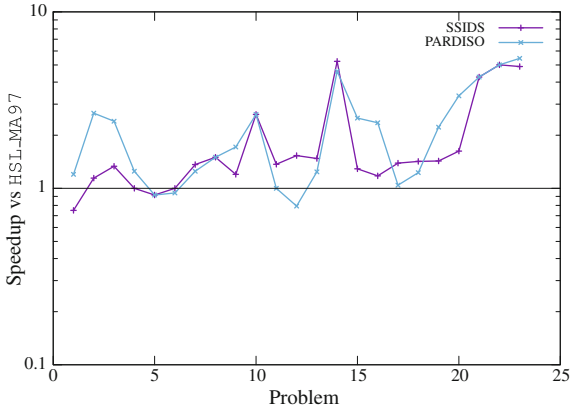


Fig. 15 Comparison of codes on hard indefinite matrices on the 28-core Haswell machine

test against the entries in A_{21}. For these tests, we use the set of indefinite matrices from Hogg and Scott [21] that require substantial numerical pivoting identified by having a large number of delayed pivots. Many of these hard indefinite cases are saddle-point matrices. We list these matrices in Table 6 in the Appendix. We show the comparison in Fig. 15. We see in this figure that both SSIDS and PARDISO obtain much better speedups than the TPP code `HSL_MA97` and they are roughly comparable, with PARDISO having a slight edge over SSIDS.

However, as we mentioned when discussing the different numerical pivoting s-trategies, the one used by PARDISO is unstable and iterative refinement is needed. We examine, in more detail in Table 2, the performance of the three codes on some matrices from the earlier runs. Here we see that while both SSIDS and PARDISO are significantly faster than `HSL_MA97`, SSIDS has a far better backward error than PARDISO, even though iterative refinement is used with PARDISO. Indeed, SSIDS is comparable with the numerical performance of the TPP code as it should be.

In our talks on these approaches, we parodize these codes as: with `HSL_MA97` we have to *pay*, with PARDISO we have to *pray*, while with SSIDS we can *play*.

7 Task 3.3 Direct Methods for Highly Unsymmetric Systems

The algorithms that we will study for highly unsymmetric systems are radically different from those of the previous sections. They are not based on an assembly tree but rather on a right-looking factorization method that uses a Markowitz threshold strategy. Parallelism is obtained largely through the use of extensive blocking.

We define a highly unsymmetric matrix as a matrix whose structure is not well approximated by the structure of $|A| + |A|^T$. Various authors have defined a measure

Table 2 Hard indefinite systems on the 28-core Haswell machine

Matrix	stokes128	cvxqp3	ncvxqp7
Order $\times 10^3$	49.7	17.5	87.5
Entries $\times 10^6$	0.30	0.07	0.31
Factor time			
HSL_MA97	0.15	1.52	8.18
PARDISO	0.12	0.33	1.50
SSIDS V2	0.11	0.29	1.67
Backward error			
HSL_MA97	1.6×10^{-15}	3.1×10^{-11}	4.4×10^{-9}
PARDISO	3.9×10^{-3}	1.1×10^{-6}	1.4×10^{-7}
SSIDS V2	1.4×10^{-15}	2.0×10^{-11}	7.3×10^{-9}

of the asymmetry of a matrix, and here we use that defined in [17] which is the proportion of off-diagonal entries for which there is a corresponding entry in the transpose, viz.

$$si(A) = \frac{number_{i \neq j}\{a_{ij} * a_{ji} \neq 0\}}{nz\{A\}},$$

where si is called the symmetry index and $nz\{A\}$ is the number of off-diagonal entries in the matrix A. A symmetric matrix will thus have a symmetry index of 1.0. Matrices with symmetry indices of less than 0.9 can be considered highly unsymmetric, and these are the main target of this work in Task 3.3. Such matrices are encountered in applications such as chemical engineering, linear programming, economic modelling, power systems and circuit modelling.

Both code and matrices implementing these algorithms can be very unstructured and complicated. We illustrate this in Fig. 16 with a matrix from an econometric model of SE Asia. It is matrix ORANI_678, from the Harwell–Boeing test set and available from the SuiteSparse matrix collection [9].

As an example of how convoluted the coding can be we reproduce the inner-loop from an 1993 HSL code, MA28 (Fig. 17). For many years, this was best code available for this type of matrix and was the code available in Netlib. The innermost loop in the Fortran 77 version of the HSL code MA48 [14] that replaced this was only marginally less ugly.

7.1 Markowitz Threshold Pivoting

Clearly, for any pivot in Gaussian elimination, the maximum fill-in[2] that can occur is the product of the number of other entries in the pivot row with the number of other

[2] An entry that is zero in A but is nonzero in the corresponding entry of the factors is termed fill-in.

entries in the pivot column. Thus, if there are c_j entries in column j and r_i entries in row i, then we define the Markowitz count for a potential pivot in row i, column j as

$$Mark_{ij} = (r_i - 1) \times (c_j - 1). \tag{5}$$

We choose candidate entries with low or minimum Markowitz count to reduce the amount of fill-in. Of course, such a candidate would be unacceptable if its value was zero or very small relative to other entries. We therefore introduce a pivot threshold and, analogously to the case for symmetric matrices in Eq. (2), only consider entries a_{ij} that satisfy

$$|a_{ij}| \geq u * \max_k |a_{kj}|, \quad k = 1, \ldots, m \tag{6}$$

where u is a threshold parameter $0 < u \leq 1.0$. That is to say we only consider entries that are at least u times as large as the largest entry in modulus of all entries in the

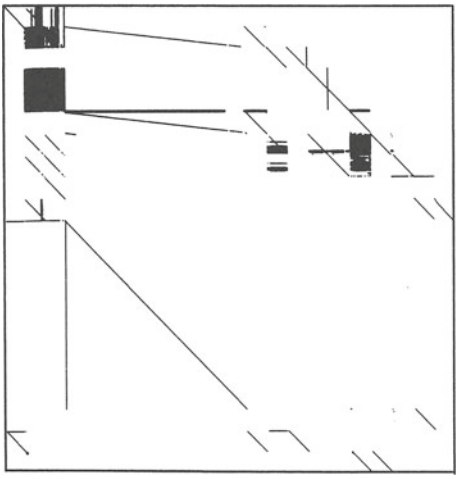


Fig. 16 Matrix from econometric model of SE Asia

```
      DO 590 JJ = J1, J2
         J = ICN (JJ)
         IF (IQ(J). GT.0) GO TO 590
         IOP = IOP + 1
         PIVROW = IJPOS - IQ (J)
         A(JJ) = A(JJ) + AU x A (PIVROW)
         IF (LBIG) BIG = DMAXI (DABS(A(JJ)), BIG)
         IF (DABS(A(JJ)). LT. TOL) IDROP = IDROP + 1
         ICN (PIVROW) = ICN (PIVROW)
  590 CONTINUE
```

Fig. 17 Innermost loop of MA28

column. We call such entries eligible entries. If u were equal to 1.0, then we would be using partial pivoting that is the most common algorithm for dense matrices.

To continue with the factorization, we must first update the remaining matrix using the outer product of the pivot row and column, updating the numerical entries and normally introducing fill-in. This is clearly a right-looking algorithm. For selecting the next pivot, we perform the threshold Markowitz algorithm on this remaining updated matrix of order one less than the previous one, and we continue in this way until all min (m, n) pivots have been chosen although it is advantageous to switch to a dense code when the Schur complement becomes denser [14]. The algorithm is simple but the data structures to implement it efficiently, even in serial mode, are not.

7.2 *Parallel Implementation of Threshold Markowitz Pivoting*

For our parallel implementation, we essentially use the same algorithm, that is a threshold Markowitz algorithm using the same terminology as the previous section. In this implementation, we find a set of independent pivots that can be used and then use these as a block pivot to update the remaining matrix in parallel. We illustrate this in Fig. 18.

When choosing the set of pivots in parallel, we note that the threshold test only needs information from one column and this is our first observation for obtaining significant parallelism. We thus launch our algorithm by scanning columns of the matrix independently. For each column, we compute the largest entry in modulus and choose as a potential pivot in that column an entry that satisfies the threshold test, that is an entry at least u times the maximum just calculated, and is in a row with the least number of entries over all eligible entries in the column. It is possible that all eligible entries are in rows of high count so, although we still flag the entry as a possible pivot, we will not use it unless there are no other suitable pivots.

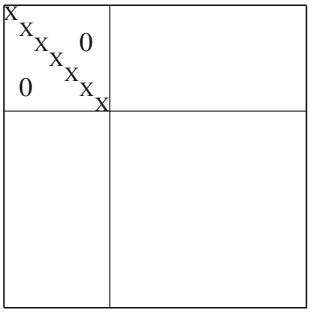


Fig. 18 Block of independent pivots

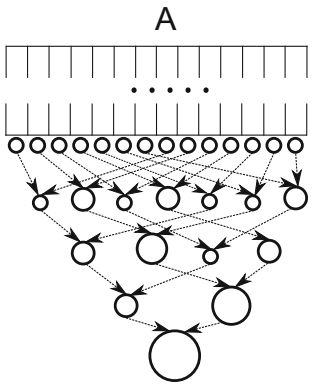


Fig. 19 Combining pivots to get block pivot

Having done this, we then want to construct a set of independent pivots in parallel. We do this by a binary combination illustrated in Fig. 19. We use a parameter to define a block size and then select at random a set of columns of the matrix whose cardinality is the block size. We choose the columns at random because often the structure of the matrix militates against choosing consecutive columns. We then do a cheap scan of columns in the block to identify an independent set. Each block is independent and can be scanned in parallel. If we assume that the column we are seeking to combine with the current block pivot is j_1 and the block pivot is in the set of rows I_2 and columns J_2 and that the potential pivot in column j_1 is in row i_1, then the column is combined with the current block if there are no entries in positions (I_2, j_1) and (i_1, J_2). For checking whether a column yields an independent pivot, we use an integer array of length n that flags whether a row has no entries in all the previously chosen columns. A similar flag is set for the columns so that the test comprises just two lookups followed by an update of the flags. This can be done without having to reset the flag array by incrementing flags at each step and only resetting if integer overflow occurs. We show experiments on the effect of the influence of the block size in the next section.

Having done this pass on all the blocks, we have sets of independent pivots of size up to the block size. We then combine these to get larger sets, continuing to do so in a binary tree fashion as shown in Fig. 19 until we have a single set of independent pivots.

In sequential codes like `MA48` [14], we select the eligible pivot that has the minimum Markowitz count, as defined in Eq. (5). Because we want to get large blocks of independent pivots, we relax this by accepting eligible pivots within a factor of the minimum, that is an entry (i, j) can be chosen as a pivot if its Markowitz cost satisfies the condition

$$Mark_{ij} \leq \alpha_{Mark} \times Best_{Mark} \tag{7}$$

where the Markowitz factor α_{Mark} is greater than or equal to one and $Best_{Mark}$ is the lowest Markowitz count.

Our next step is to perform all the pivot operations for the block of independent pivots in parallel. In effect what we have to do is a parallel sparse matrix by sparse matrix multiply to update the Schur complement. Having done this, we then repeat the parallel pivot selection on the reduced matrix corresponding to the Schur complement that we have just computed.

We terminate the algorithm when either the last few steps of our algorithm ("few" is a parameter that we have set to 5 in our experiments) have failed to obtain a number of independent pivots greater than a preset threshold or the Schur complement reaches a preset density. At that stage, in the present code, we switch to using the `PLASMA` [7] code `GETRF` for parallel dense LU factorization on the remaining Schur complement. We plan in later versions of the code to have a transitional stage where we use a parallel sparse direct code designed for relatively dense sparse matrices, such as the parallel LU factorization that will be developed in Task 3.2.

Because of these various stages, we use four different data structures. The L and U factors continuously grow with the execution of our algorithm without changing the already computed part. For this reason, we use standard CSC storage for the L and U factors. The values in the diagonal matrix D of the LDU factorization are stored in a separate array. On the other hand, the structure of the Schur complement changes because of the update operations. Additionally, at each step we must be able to determine whether a set of pivots are mutually independent. For these reasons, we use a flexible CSC/CSR-based structure. The numerical values are stored in the CSC fashion, while the CSR part stores only the nonzero structure of the matrix. In total, three large arrays are used, one index and one value array for the CSC part and one index array for the CSR part. Having the matrix structure stored by rows and columns provides an efficient way of updating the structures during pivot selection and Schur update. In order to cope with the dynamic nature of the Schur complement, within the CSR/CSC structure extra space is allocated at the end of each row and each column. Each row and column is represented by an offset from the start of the corresponding array, by the number of entries each contains, with the amount of available free space. Additional memory is allocated at the end of each array which is managed by a garbage collector. For each block of free memory, the garbage collector stores the offset from the beginning of the array and its size. When fill-in to a row or a column consumes its available space, it is moved to the next available space provided by the garbage collector. Its old memory is marked as free memory and added to the garbage collector for future reuse. Similarly, space freed when a row or column becomes pivotal is likewise given to the garbage collector.

Table 3 Some highly unsymmetric matrices

Matrix	Order $\times 10^3$	Entries $\times 10^6$	*si*
twotone	120	1.22	0.26
LoadFlow_Newton_0_4	197	3.70	0.46
Jacobian_unbalancedLdf	203	2.76	0.80

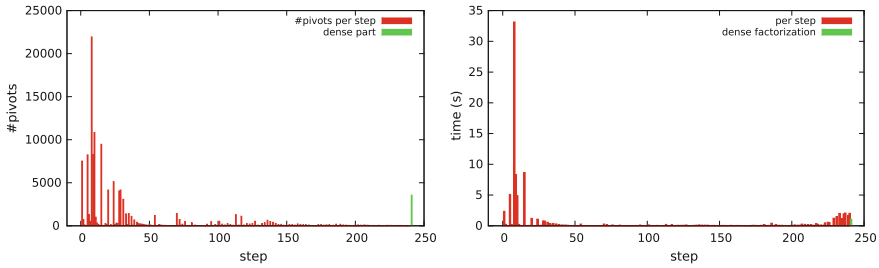


Fig. 20 The number of pivots (left) and the time spent (right) for each step when the matrix twotone is used. Additionally, the number of pivots handled by the dense solver and the time spent in the dense solver are given at the end

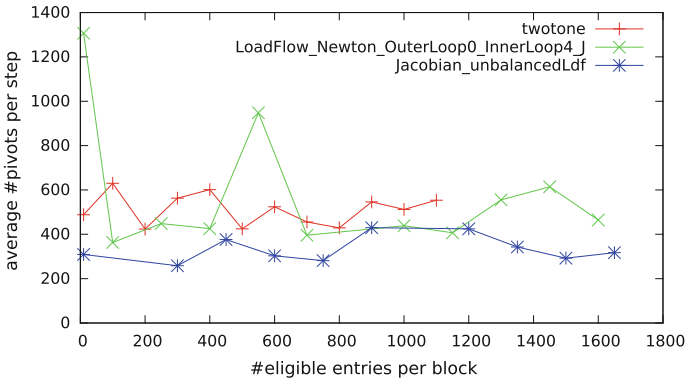


Fig. 21 Impact of the number of candidates per initial pivot set on the average number of pivots per step

7.3 Preliminary Results

In this section, we present results of some experiments with our solver. The tests were performed on the same multicore Haswell machine that we used for the experiments in Sect. 5. All the results presented are sequential (mono-threaded execution). The main attributes of the matrices used in this study are given in Table 3. The matrix twotone is from the SparseSuite set of test matrices, and the other two are from a Power Systems application supplied by Bernd Klöss of DigSILENT GmbH.

As discussed in the previous section, at some point in our algorithm we switch to a dense solver. The reason for this is that once the Schur complement becomes too dense, we get only a few pivots at a time and in addition the operations become more and more expensive. At some point, we get sets of size one only and the execution of each step takes a lot of time. This is considerably improved by switching to the dense solver, and we show the number of pivots selected and the time in Fig. 20.

At the beginning of each step, we need to set up the initial sets of pivots that will be merged later. The impact of the number of candidates per initial pivot set (the block size) on the average number of pivots found per step is given in Fig. 21. We can see that

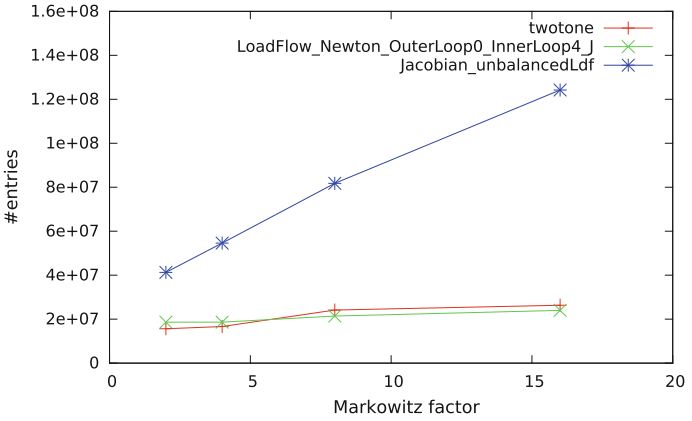


Fig. 22 The impact of the Markowitz factor on the total number of entries in the L and U factors

for all the values of this parameter, our algorithm is able to create large enough sets. One interesting thing to notice is that when the LoadFlow_Newton_0_4 matrix is used, the optimal value is 10. That points to the fact that the largest pivot sets are obtained when we start with a large number of small pivot sets. Since the merge is done using a binary tree and, at each level of the tree, all the merging can be done in parallel, it indicates that our algorithm could potentially be extremely parallel.

Each pivot candidate must satisfy the Markowitz test in Eq. (7). When we relax the constraint on the Markowitz cost, we accept pivots with higher Markowitz cost which will usually introduce more fill-in in the factors. The impact of the Markowitz factor α_{Mark} on the number of entries in the L and U factors is presented in Fig. 22. In this figure, the total number of entries in the L and U factors is shown and includes the dense part for each of them. Thus, as expected when the Markowitz factor is relaxed, the amount of fill-in increases.

8 Task 3.4 Hybrid Direct-Iterative Methods

We have only just commenced work on Task 3.4. Here we consider the use of a hybrid method to solve the system of equations. This enables us to extend the range of direct solvers to larger matrices and to obtain another level of parallelism. The system we want to solve is

$$Ax = b, \tag{8}$$

where A is an $m \times n$ sparse matrix, x is an n-vector and b is an m-vector. In the following, we assume the system is consistent and for simplicity we suppose that A has full rank.

We will study the solution of the system (8) using the block Cimmino method, an iterative method using block-row projections. In this method, the system (8) is subdivided into strips of rows as in the following:

$$\begin{pmatrix} A_1 \\ A_2 \\ \vdots \\ A_p \end{pmatrix} x = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_p \end{pmatrix}. \tag{9}$$

Let $P_{\mathscr{R}(A_i^T)}$ be the projector onto the range of A_i^T and ${A_i}^+$ be the Moore–Penrose pseudo-inverse of the partition A_i. The block Cimmino algorithm then computes a solution iteratively from an initial estimate $x^{(0)}$ according to:

$$u_i = A_i^+ \left(b_i - A_i x^{(k)}\right) \quad i = 1, ..., p \tag{10}$$

$$x^{(k+1)} = x^{(k)} + \omega \sum_{i=1}^{p} u_i, \tag{11}$$

where ω is a real parameter whose value we will shortly show to be immaterial. We note the independence of the set of p equations, which is why the method is so attractive in a parallel environment. The block Cimmino method is described in more detail by Ruiz [22].

Although the matrix in Eq. (8) can be rectangular and the Cimmino method can work on such systems [16], for our main discussion we will assume that A is square and of order n.

With the above notations, the iteration equations are thus:

$$\begin{aligned} x^{(k+1)} &= x^{(k)} + \omega \sum_{i=1}^{p} A_i^+ \left(b_i - A_i x^{(k)}\right) \\ &= \left(I - \omega \sum_{i=1}^{p} A_i^+ A_i\right) x^{(k)} + \omega \sum_{i=1}^{p} A_i^+ b_i \\ &= Q x^{(k)} + \omega \sum_{i=1}^{p} A_i^+ b_i. \end{aligned}$$

The iteration matrix for block Cimmino $H = I - Q$ is then a sum of projectors $H = \omega \sum_{i=1}^{p} \mathscr{P}_{\mathscr{R}(A_i^T)}$. It is thus symmetric and positive definite, and so we can solve

$$Hx = \xi, \tag{12}$$

where $\xi = \omega \sum_{i=1}^{p} A_i^+ b_i$, using conjugate gradient or block conjugate gradient methods. As ω appears on both sides of Eq. (12), we can set it to one.

The starting point for our work on this is the thesis and code of Zenadi [24]. We have been porting his code that uses MPI and multithreading to our machines and have been resolving a few issues. We have been studying the resulting code and have obtained good parallelism for some problems. We show the performance of the code on matrix `cage12` from the SuiteSparse test set in Fig. 23. The machine used for

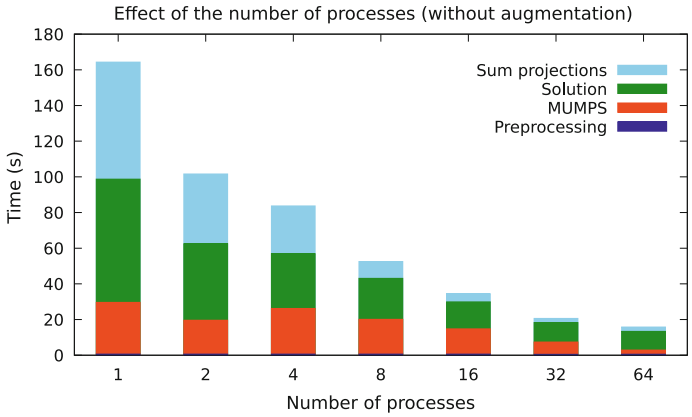


Fig. 23 The speedup of block Cimmino for `cage12` on the scarf machine at RAL

these runs is a local heterogeneous machine called scarf. It has a number of Intel nodes (including some E5-2650, E2660, X5675, X5530 and E5530 nodes).[3] The actual configuration is determined by the batch scheduler at run-time.

The default partitioning method used in the code is PaToH [8]. In the NLAFET project, we are now testing various partitioning approaches since the number of block Cimmino iterations is strongly related to the number of columns in the border of the bordered block diagonal form that we illustrated in Fig. 4. For the matrix `cage12`, we see in Fig. 24 (left) that the number of iterations is not affected by the number of processes and that it does not vary significantly when the number of partitions increases (see Fig. 24 (right)) although, as expected, the number of iterations increases with the number of partitions.

We note that in the code of Zenadi and in the earlier research [11], we have developed an algorithm ABCD that totally avoids the interaction between blocks by augmenting them, so that they are mutually orthogonal. This yields a pseudo-direct method that should converge in one iteration.

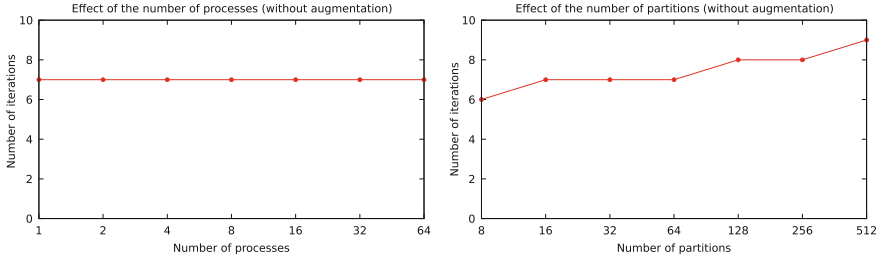


Fig. 24 The effect of the number of processes (left) and number of partitions (right) on the convergence of block Cimmino on a system with matrix `cage12`

[3] See http://www.scarf.rl.ac.uk/hardware.

9 Concluding Remarks

In conclusion, we emphasize that this work is very much still ongoing and we are only just past the halfway point in the project.

What we can say already is that there are lots of parallelism in sparse direct solvers but programming this, while interesting and fun, is extremely tough.

Much of the work that we have described has been done to provide a solid platform for future work in the NLAFET project. We are currently examining in detail the two main approaches for using assembly trees in sparse factorization, namely the supernodal and the multifrontal method. While the data handling is much simpler in multifrontal schemes and exploitation of parallelism should thus be easier, the storage requirements are often higher. Our future work will quantify these issues and give recommendations concerning these two approaches.

One of the reasons why we have used runtime systems is so that our codes can be more easily ported to different architectures and we plan to illustrate this by porting our codes to GPUs and to a heterogeneous system with multicore nodes and GPUs. The authors of the runtime systems are developing versions for distributed memory environments that we look forward to testing.

One approach for solving systems with highly unsymmetric matrices is to permute them to make them less unsymmetric, and this will be used with the LU codes that we will develop as an extension to the tree-based methods for symmetric systems. This approach will then be compared with our threshold Markowitz approach. Finally, the current block Cimmino code uses MUMPS for the direct solution of subproblems. The MUMPS code is arguably one of the best parallel direct solvers, but it is based on MPI. In the context of our hierarchical approach to exploiting parallelism in this hybrid method, we will be targeting a multicore environment at the direct solver level for which our new codes being developed in Task 3.2 might be better suited.

The software that is produced by the NLAFET Project will all be open source and will be available online by the end of the project on 31 October 2018. Some software will be available before then. The GitHub repository for the NLAFET project is: https://github.com/NLAFET, and the software for the SpLLT code will shortly be available in directory SpLLT. The SSIDS code is in the SPRAL package and accessible from the GitHub repository https://github.com/ralna/spral.

Acknowledgements This work is supported by the NLAFET Project funded by the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement 671633. We would like to thank Philippe Gambron for doing the experiments reported in Sect. 8, Jonathan Hogg for his earlier work in the project developing the first versions of some of the kernels used in Sects. 5 and 6, and Tim Davis (Texas A&M) for discussions on parallel Markowitz. We also thank Jennifer Scott and Bo Kågström (Umeå) for their comments on a draft of this manuscript.

Appendix

Test matrices used in experiments.

Table 4 Test matrices and their characteristics. n is the matrix order, $nz(A)$ the number entries in the matrix A, $nz(L)$ the number of entries in the factor L, and *Flops* corresponds to the operation count for the matrix factorization

#	Problem	n (10^3)	nz(A) (10^6)	nz(L) (10^6)	Flops (10^9)	Application/Description
1	Schmid/thermal2	1228	4.9	51.6	14.6	Unstructured thermal FEM
2	Rothberg/gearbox	154	4.6	37.1	20.6	Aircraft flap actuator
3	DNVS/m_t1	97.6	4.9	34.2	21.9	Tubular joint
4	Boeing/pwtk	218	5.9	48.6	22.4	Pressurized wind tunnel
5	Chen/pkustk13	94.9	3.4	30.4	25.9	Machine element
6	GHS_psdef/crankseg_1	52.8	5.3	33.4	32.3	Linear static analysis
7	Rothberg/cfd2	123	1.6	38.3	32.7	CFD pressure matrix
8	DNVS/thread	29.7	2.2	24.1	34.9	Threaded connector
9	DNVS/shipsec8	115	3.4	35.9	38.1	Ship section
10	DNVS/shipsec1	141	4.0	39.4	38.1	Ship section
11	GHS_psdef/crankseg_2	63.8	7.1	43.8	46.7	Linear static analysis
12	DNVS/fcondp2	202	5.7	52.0	48.2	Oil production platform
13	Schenk_AFE/af_shell3	505	9.0	93.6	52.2	Sheet metal forming
14	DNVS/troll	214	6.1	64.2	55.9	Structural analysis
15	AMD/G3_circuit	1586	4.6	97.8	57.0	Circuit simulation
16	GHS_psdef/bmwcra_1	149	5.4	69.8	60.8	Automotive crankshaft
17	DNVS/halfb	225	6.3	65.9	70.4	Half-breadth barge
18	Um/2cubes_sphere	102	0.9	45.0	74.9	Electromagnetics
19	GHS_psdef/ldoor	952	23.7	144.6	78.3	Large door
20	DNVS/ship_003	122	4.1	60.2	81.0	Ship structure
21	DNVS/fullb	199	6.0	74.5	100.2	Full-breadth barge

(continued)

Table 4 (continued)

#	Problem	n (10^3)	nz(A) (10^6)	nz(L) (10^6)	Flops (10^9)	Application/Description
22	GHS_psdef/inline_1	504	18.7	172.9	144.4	Inline skater
23	Chen/pkustk14	152	7.5	106.8	146.4	Tall building
24	GHS_psdef/apache2	715	2.8	134.7	174.3	3D structural problem
25	Koutsovasilis/F1	344	13.6	173.7	218.8	AUDI engine crankshaft
26	Oberwolfach/boneS10	915	28.2	278.0	281.6	Bone micro-FEM
27	ND/nd12k	36.0	7.1	116.5	505.0	3D mesh problem
28	ND/nd24k	72.0	14.4	321.6	2054.4	3D mesh problem
29	Janna/Flan_1565	1565	59.5	1477.9	3859.8	3D mechanical problem
30	Oberwolfach/bone010	987	36.3	1076.4	3876.2	Bone micro-FEM
31	Janna/StocF-1465	1465	11.2	1126.1	4386.6	Underground aquifer
32	GHS_psdef/audikw_1	944	39.3	1242.3	5804.1	Automotive crankshaft
33	Janna/Fault_639	639	14.6	1144.7	8283.9	Gas reservoir
34	Janna/Hook_1498	1498	31.2	1532.9	8891.3	Steel hook
35	Janna/Emilia_923	923	21.0	1729.9	13661.1	Gas reservoir
36	Janna/Geo_1438	1438	32.3	2467.4	18058.1	Underground deformation
37	Janna/Serena	1391	33.0	2761.7	30048.9	Gas reservoir

Table 5 Easy Indefinite. Statistics as reported by the analyse phase of SSIDS with default settings, assuming no delayed pivots

Problem	$n \times 10^3$	$nz(A) \times 10^6$	$nz(L) \times 10^6$	$flops \times 10^9$
Oberwolfach/t2dal	4.26	0.02	0.28	0.02
GHS_indef/dixmaanl	60.00	0.18	1.58	0.05
Oberwolfach/rail_79841	79.84	0.32	4.43	0.33
GHS_indef/dawson5	51.54	0.53	5.69	0.90
Boeing/bcsstk39	46.77	1.07	9.61	2.66
Boeing/pct20stif	52.33	1.38	12.60	5.63
GHS_indef/copter2	55.48	0.41	12.70	6.10
GHS_indef/helm2d03	392.26	1.57	33.00	6.16
Boeing/crystk03	24.70	0.89	10.90	6.26
Oberwolfach/filter3D	106.44	1.41	23.80	8.71
Koutsovasilis/F2	71.50	2.68	23.70	11.30
McRae/ecology1	1000.00	3.00	72.30	18.20
Cunningham/qa8fk	66.13	0.86	26.70	22.10
Oberwolfach/gas_sensor	66.92	0.89	27.00	22.10
Oberwolfach/t3dh	79.17	2.22	50.60	70.10
Lin/Lin	256.00	1.01	126.00	285.00
GHS_indef/sparsine	50.00	0.80	207.00	1390.00
PaRSEC/Ge99H100	112.98	4.28	669.00	7070.00
PaRSEC/Ga10As10H30	113.08	3.11	690.00	7280.00
PaRSEC/Ga19As19H42	133.12	4.51	823.00	9100.00

Table 6 Hard Indefinite. Statistics as reported by the analyse phase of SSIDS with default settings, using matching-based ordering, assuming no delayed pivots

Problem	$n \times 10^3$	$nz(A) \times 10^6$	$nz(L) \times 10^6$	$flops \times 10^9$
TSOPF/TSOPF_FS_b39_c7	28.22	0.37	2.61	0.26
TSOPF/TSOPF_FS_b162_c1	10.80	0.31	1.89	0.36
QY/case39	40.22	0.53	3.87	0.40
TSOPF/TSOPF_FS_b39_c19	76.22	1.00	7.28	0.75
TSOPF/TSOPF_FS_b39_c30	120.22	1.58	11.10	1.10
GHS_indef/cont-201	80.59	0.24	7.12	1.11
GHS_indef/stokes128	49.67	0.30	6.35	1.16
TSOPF/TSOPF_FS_b162_c3	30.80	0.90	6.37	1.41
TSOPF/TSOPF_FS_b162_c4	40.80	1.20	7.32	1.43
GHS_indef/ncvxqp1	12.11	0.04	3.56	2.52
GHS_indef/darcy003	389.87	1.17	23.20	3.01
GHS_indef/cont-300	180.90	0.54	17.20	3.58
GHS_indef/bratu3d	27.79	0.09	7.49	4.72
GHS_indef/cvxqp3	17.50	0.07	6.33	5.27
TSOPF/TSOPF_FS_b300	29.21	2.20	13.40	6.92
TSOPF/TSOPF_FS_b300_c1	29.21	2.20	13.50	7.01
GHS_indef/d_pretok	182.73	0.89	24.80	7.42
GHS_indef/turon_m	189.92	0.91	24.70	7.60
TSOPF/TSOPF_FS_b300_c2	56.81	4.39	27.00	14.10
TSOPF/TSOPF_FS_b300_c3	84.41	6.58	40.50	21.40
GHS_indef/ncvxqp5	62.50	0.24	22.90	24.30
GHS_indef/ncvxqp3	75.00	0.27	39.30	63.70
GHS_indef/ncvxqp7	87.50	0.31	51.00	101.00

References

1. Agullo, E., Buttari, A., Guermouche, A., Lopez, F.: Multifrontal QR factorization for multicore architectures over runtime systems. In: Proceedings of Euro-Par 2013 Parallel Processing, pp. 521–532. Springer, Berlin (2013)
2. Agullo, E., Buttari, A., Guermouche, A., Lopez, F.: Implementing multifrontal sparse solvers for multicore architectures with sequential task flow runtime systems. ACM Trans. Math. Softw. **43**, Article 13, 17p (2016)
3. Amestoy, P.R., Duff, I.S., L'Excellent, J.-Y.: Multifrontal parallel distributed symmetric and unsymmetric solvers. Comput. Methods Appl. Mech. Eng. **184**, 501–520 (2000)
4. Bunch, J.R., Kaufman, L., Parlett, B.N.: Decomposition of a symmetric matrix. Numer. Math. **27**, 95–110 (1976)
5. Bunch, J.R., Parlett, B.N.: Direct methods for solving symmetric indefinite systems of linear equations. SIAM J. Numer. Anal. **8**, 639–655 (1971)
6. Buttari, A.: Fine-grained multithreading for the multifrontal QR factorization of sparse matrices. SIAM J. Sci. Comput. **35**, C323–C345 (2013)
7. Buttari, A., Langou, J., Kurzak, J., Dongarra, J.: A class of parallel tiled linear algebra algorithms for multicore architectures. Parallel Comput. **35**, 38–53 (2009)
8. Çatalyürek, Ü.V., Aykanat, C.: Hypergraph-partitioning-based decomposition for parallel sparse-matrix vector multiplication. IEEE Trans. Parallel Distrib. Syst. **10**, 673–693 (1999)
9. Davis, T.A., Hu, Y.: The university of Florida sparse matrix collection. ACM Trans. Math. Softw. **38**, 1:1–1:25 (2011)
10. Duff, I.S., Erisman, A.M., Reid, J.K.: Direct Methods for Sparse Matrices, 2nd edn. Oxford University Press, Oxford, England (2016)
11. Duff, I.S., Guivarch, R., Ruiz, D., Zenadi, M.: The augmented block Cimmino distributed method. SIAM J. Sci. Comput. **37**, A1248–A1269 (2015)
12. Duff, I.S., Hogg, J., Lopez, F.: Experiments with sparse Cholesky using a sequential task-flow implementation. Technical Report RAL-TR-2016-016, Rutherford Appleton Laboratory, Oxfordshire, England, 2016. NLAFET Working Note 7. Numer. Algeb. Contr. Optim. **8**, 235–258. Submitted to NACO (2018)
13. Duff, I.S., Lopez, F.: Experiments with sparse Cholesky using a parametrized task graph implementation. Technical Report RAL-TR-2017-006, Rutherford Appleton Laboratory, Oxfordshire, England, 2017. NLAFET Working Note 14. In parallel Processing and Applied Mathematics, Springer, pp 197–206. Accepted for presentation at PPAM 2017 (2018)
14. Duff, I.S., Reid, J.K.: The design of MA48, a code for the direct solution of sparse unsymmetric linear systems of equations. ACM Trans. Math. Softw. **22**, 187–226 (1996)
15. Duff, I.S., Reid, J.K.: Exploiting zeros on the diagonal in the direct solution of indefinite sparse symmetric linear systems. ACM Trans. Math. Softw. **22**, 227–257 (1996)
16. Elfving, T.: Block-iterative methods for consistent and inconsistent linear equations. Numer. Math. **35**, 1–12 (1980)
17. Erisman, A.M., Grimes, R.G., Lewis, J.G., Poole Jr., W.G., Simon, H.D.: Evaluation of orderings for unsymmetric sparse matrices. SIAM J. Sci. Stat. Comput. **7**, 600–624 (1987)
18. Hogg, J.: A new sparse LDLT solver using a posteriori threshold pivoting. Technical Report RAL-TR-2016-017, Rutherford Appleton Laboratory, Oxfordshire, England, 2017. NLAFET Working Note 6
19. Hogg, J., Reid, J., Scott, J.: Design of a multicore sparse Cholesky factorization using DAGs. SIAM J. Sci. Comput. **32**, 3627–3649 (2010)
20. Hogg, J., Scott, J.: HSL_MA97: a bit-compatible multifrontal code for sparse symmetric systems. Technical Report RAL-TR-2011-024. Rutherford Appleton Laboratory, Oxfordshire, England (2011)
21. Hogg, J., Scott, J.: A study of pivoting strategies for tough sparse indefinite systems. ACM Trans. Math. Softw. **40**, Article 4, 19p (2013)

22. Ruiz, D.F.: Solution of large sparse unsymmetric linear systems with a block iterative method in a multiprocessor environment. Ph.D. Thesis, Institut National Polytechnique de Toulouse, 1992. CERFACS Technical Report, TH/PA/92/06
23. Schenk, O., Gärtner, K.: On fast factorization pivoting methods for sparse symmetric indefinite systems. Electron. Trans. Numer. Anal. **23**, 158–179 (2006)
24. Zenadi, M.: The solution of large sparse linear systems on parallel computers using a hybrid implementation of the block Cimmino method. Thése de Doctorat, Institut National Polytechnique de Toulouse, Toulouse, France, dcembre 2013

Revisiting Expression Representations for Nonlinear AMPL Models



David M. Gay

Abstract AMPL facilitates stating and solving nonlinear programming problems involving algebraically defined objectives and constraints. For solving such problems, the AMPL/solver interface library provides routines that compute objective functions, constraint residuals, and associated derivatives. Objectives and constraint bodies hitherto have been represented by "executable" expression graphs, in which each node points to its operands and to a function that computes the node's result. Nodes also store partial derivatives for use in computing gradients and Hessians by automatic differentiation. Storing these values makes the graphs nonreentrant. To enable several threads to evaluate the same expression at different points without having separate copies of the expression graphs, such details as variable values and partial derivatives must be stored in thread-specific arrays. We describe and compare some expression-graph representations for use in computing function, gradient, and Hessian values, and for extracting some auxiliary problem information. In particular, we describe some details of an updated AMPL/solver interface library that uses operation lists to represent expressions.

Keywords Expression representation · Automatic differentiation
Modeling language

1 Introduction

The AMPL modeling language [1, 2] facilitates formulating, instantiating, solving, and examining solutions of mathematical programming problems, such as

$$\text{minimize } f(x) \tag{1a}$$

$$\text{s.t. } \ell \le c(x) \le u. \tag{1b}$$

D. M. Gay (✉)
AMPL Optimization Inc., Albuquerque, NM, USA
e-mail: dmg@ampl.com

© Springer International Publishing AG, part of Springer Nature 2018

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*,
Springer Proceedings in Mathematics & Statistics 235,
https://doi.org/10.1007/978-3-319-90026-1_5

with $x \in \mathbb{R}^n$ and $c : \mathbb{R}^n \to \mathbb{R}^m$, possibly with some components of x restricted to integer values. (If there are no constraints, then $m = 0$. When $m > 0$, the ith lower bound ℓ_i can be finite or $-\infty$, i.e., $\ell_i \in \{-\infty\} \cup \mathbb{R}$, and similarly $u_i \in \{+\infty\} \cup \mathbb{R}$, with $\ell_i = u_i \in \mathbb{R}$ if constraint i is an equality constraint.)

While AMPL has "presolve" facilities [3] for simplifying (1), and these simplifications sometimes find solutions, normally AMPL does not solve problems, but relies on separate *solvers* to find solutions. To use such a solver, AMPL writes a representation of the current problem instance to a ".nl file" and invokes the solver as a separate program. The solver typically uses facilities in a special library, the AMPL/solver interface library (ASL), to read the .nl file and acquire various problem details. When f or c is nonlinear, the solver calls ASL routines to compute function values $f(x)$ and $c(x)$ at a specified vector $x \in \mathbb{R}^n$. Most nonlinear solvers also obtain gradient $\nabla f(x)$ and Jacobian $\nabla c(x)$ values from the ASL; some also use the ASL to obtain values of the Hessian of the Lagrangian function,

$$\nabla^2 L(x, y) = \nabla^2 f(x) + \sum_{i=1}^{m} y_i \nabla^2 c_i(x) \tag{2}$$

or Hessian-vector products $\nabla^2 L(x, y) \cdot v$ for specified vectors $v \in \mathbb{R}^n$. This paper explains how the ASL has computed these values and presents an alternative way to compute them. The goal of the present work is to revisit expression representations and evaluations in the ASL with an eye to separating expressions from data so multiple threads can make independent use of the same expressions.

For concreteness, let us consider a tiny problem with $n = 2$ and $m = 1$:

$$\begin{aligned} \text{minimize } & f(x) = (x_1 - 3)^2 + (x_2 + 4)^2 \\ \text{s.t. } & c(x) = x_1 + x_2 = 1. \end{aligned} \tag{3}$$

An AMPL script for stating and solving this problem is

```
var x; var y;
minimize f: (x - 3)^2 + (y + 4)^2;
s.t. c: x + y == 1;
solve;
display x, y;
```

Putting this script into file `tiny.x` and invoking "`ampl tiny.x`", we get

```
MINOS 5.51: optimal solution found.
2 iterations, objective 2
Nonlin evals: obj = 6, grad = 5.
x = 4
y = -3
```

To process the "solve" command above, AMPL writes a .nl file containing

- problem statistics (number of variables, etc.)
- expression graphs for nonlinear parts of objectives and constraints
- linear parts of objectives and constraints
- starting guesses (if specified)
- suffixes, e.g., for a basis (if available).

2 Representations of Expression Graphs

There are various ways to represent expression graphs. The following four ways are roughly equivalent in size and evaluation time.

1. Polish postfix: Operands are pushed onto a stack, and operators remove operands from the top of the stack, compute a result, and push the result onto the stack top. Thus operators follow operands. For a while in the late twentieth century, Hewlett-Packard sold hand calculators that worked this way. For example, on an HP 15C calculator, the computation of $3 \times 4 + 5 = 17$ proceeds as follows:

keystroke	*display*
3	3
Enter	3.00
4	4
$\times$	12.00
5	5
$+$	17.00

Some interpreters for the Pascal programming language, e.g., [4], also used Polish postfix for their "compiled" program representations.

2. Polish prefix: Operators are followed by expressions for their operands. The ASL has long worked this way [5]. For example, the expression graph for (3) could be represented visually by Fig. 1.

Inserting the lines

```
option nl_comments 1;
write gtiny;
```

before the "solve;" line in the "tiny.x" file shown above would cause AMPL to write file "tiny.nl" containing the following lines to represent (3):

```
O0 0        #f
o0          # +
o5          #^
o0          # +
n-3
v0          #x
n2
o5          #^
```

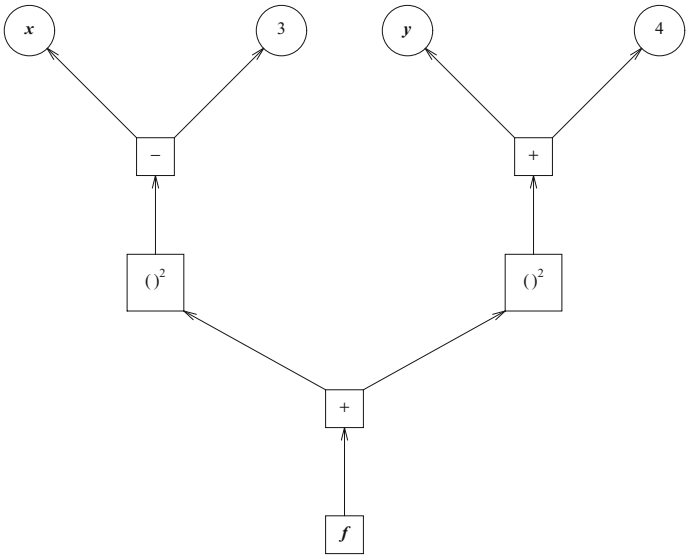


Fig. 1 Expression graph for (3)

```
o0          # +
n4
v1          #y
n2
```

The portion of each line starting with "#" is a comment that is only present when "option nl_comments 1" is in effect. For instance, the lines

```
o0        # +
n4
v1        #y
```

represent $y + 4$, and the lines

```
o5        #^
o0        # +
n4
v1        #y
n2
```

represent $(y + 4)^2$.

3. Executable expression graphs: The ASL has hitherto represented each operation by a data structure that includes pointers to operands and a pointer to a function that carries out the operation. For a binary operation, the data structure has the form shown in Fig. 2 in the C programming language. The "left" and "right" operands are `L` and `R` in Fig. 2, and the `dL` and `dR` fields are for the partial derivatives of the operation's result with respect to the left and right operands. These fields make the code nonreentrant.

Fig. 2 Old ASL binary expression for gradients only

```
struct expr {
        real (*op)(struct expr*);
        int a;
        real dL;
        struct expr *L, *R;
        real dR;
        };
```

(The "a" field is an "adjoint subscript" used when setting up derivative computations. Here and below, we assume "typedef double real;" has appeared.) For example, C source for the "op" function for division, prepared for function and gradient evaluations, is

```
real f_OPDIV(expr *e) {
        real L, R, rv;
        expr *e1 = e->L;
        L = (*e1->op)(e1);
        e1 = e->R;
        if (!(R = (*e1->op)(e1)))
                zero_div(L, "/"); /*no return*/
        rv = L / R;
        if (want_deriv)
                e->dR = -rv * (e->dL = 1. / R);
        return rv;
        }
```

Partial derivatives are not always needed. For instance, partials are not needed in computing the *test* part of an "if *test* then *texpr* else *fexpr*" expression. In the `f_OPDIV` source above, partials are only computed when `want_deriv` is true, in which case the line after "`if (want_deriv)`" computes both `dL` and `dR` values.

4. Operation list: A list, similar in spirit to computer machine instructions, of operators, input operands, and output result locations is sometimes useful. For example, Kearfott's GlobSol solver [6] uses a list of quadruples of integers (called a "codelist"). To convey the general idea, here is a list of instructions for computing (3), using a scratch array `w` initialized with `w[0]` $= x$ and `w[1]` $= y$.

```
w[2] = w[0] - 3;     /* x - 3 */
w[2] = w[2] * w[2];
w[3] = w[1] + 4;     /* y + 4 */
w[3] = w[3] * w[3];
w[2] = w[2] + w[3];
```

The computation ends with $f(x) =$ `w[2]`.

The present work uses a list of tuples of varying lengths to represent operations and operands. For example, a unary operation is represented by three integers, a binary operation by four integers, and a sum of n terms by $n + 3$ integers. A "big

switch", which a good compiler will turn into a jump table, determines the operation carried out:

```
real eval1(int *o, EvalWorkspace *ew) {
       real *w = ew->w;
 top:  switch(*o) {
         case nOPRET:
               return w[o[1]];

         case nOPUMINUS:
               w[o[1]] = -w[o[2]];
               o += 3;  goto top;
         case nOPPLUS:
               w[o[1]] = w[o[2]] + w[o[3]];
               o += 4;  goto top;
         case nOPMINUS:
               w[o[1]] = w[o[2]] - w[o[3]];
               o += 4;  goto top;
         case nOPMULT:
               w[o[1]] = w[o[2]] * w[o[3]];
               o += 4;  goto top;
       ...
```

The above `eval1(...)` routine has a separate work array, `w`, for computed results. The executable expression-graph approach could also be modified to use a separate work array for computed results, but timing results shown below indicate it may be better to use operation lists.

3 Gradient Computations

It is convenient and efficient to use automatic differentiation (AD) to compute gradients, and possibly Hessians (2) or Hessian-vector products, related to (1). AD has been addressed by many papers and books; see, for example, the book [7] by Griewank and Walther, the publications cited therein, and those listed in the Web site http://www.autodiff.org. The basis for AD is simply the chain rule. Suppose t is a variable whose role in (1) can be described as

$$\phi(t) = f(y_1(t), y_2(t), \ldots, y_k(t));$$

t might be a program variable x_i or an intermediate value involved in (1). The notion here is that t is directly involved in the computation of possibly intermediate values $y_1, \ldots, y_k$. The chain rule gives

$$\phi'(t) = \frac{\partial f}{\partial t} = \sum_{i=1}^{k} \frac{\partial f}{\partial y_i} \frac{\partial y_i}{\partial t}.$$

Once we know the *adjoint* $\frac{\partial f}{\partial y}$ of an intermediate variable y, we can add its contribution $\frac{\partial f}{\partial y}\frac{\partial y}{\partial t}$ to the adjoint of each variable t on which y depends directly. So-called backwards AD or reverse AD proceeds by computing adjoints in the reverse order of the operations to compute $f(x)$. Since the ith component of the gradient $\nabla f(x)$ is the adjoint of x_i, reverse AD computes $\nabla f(x)$ in a number of operations proportional to the number needed to compute $f(x)$ itself, which makes reverse AD appealing for computing gradients, at least for computations simple enough that relevant intermediate results can be stored.

As described in [5], the ASL has long used reverse AD to compute gradients by means of "derivative propagation" structures

```
struct derp {
        struct derp *next;
        real *a, *b, *c;
        };
```

in which `a` and `b` point to adjoints and `c` points to a previously computed partial derivative value. The reverse AD computation proceeds via

```
void derprop(derp *d) {
        *d->b = 1.;
        do *d->a += *d->b * *d->c;
                while((d = d->next));
        }
```

Would it be faster to implement reverse AD another way? To see how using integer subscripts rather than pointers would perform, we now consider three ways of computing an inner product, based on the following structures:

```
struct Rpair { double a, b; } *rp;
struct Aoff { double *a, *b; } *p;
struct Ioff { int a, b; } *q;
```

With `Rpair`, the values to be multiplied are in the structure itself. With `Aoff`, pointers to those values are in the structure. With `Ioff`, an auxiliary array, declared by

```
double *v;
```

contains those values. The basic operations for computing an inner product "`dot`" are

```
dot += rp->a * rp->b;       /* Rpair */
dot += *p->a * *p->b;       /* Aoff  */
dot += v[p->a] * v[p->b];   /* Ioff  */
```

Table 1 Relative times for "dot" variants

	32-bit	64-bit
`Rpair`	1.0	1.0
`Aoff` sequential	1.0	1.0
`Ioff` sequential	1.0	1.0
`Aoff` permuted	1.6	1.8
`Ioff` permuted	1.6	1.7

In computing on a laptop computer with an Intel Celeron CPU, sequential memory accesses are faster than randomly permuted ones, as indicated in the Table 1, which presents relative computation times for some inner products, with both 32- and 64-bit addressing. On this machine, at least, there is no disadvantage to using integer subscripts. For separating data fixed by the problem instance from data that depend on current variable values, it is convenient to use integer subscripts into a thread-specific work array, and the above table suggests that doing so may not adversely affect single-thread computations, at least not greatly.

4 Alternative Implementations of `derprop`

For computing gradients in a thread-safe way, i.e., with separate arrays for values that vary with the thread, we could use a computation analogous to the `derprop` routine shown above:

```
struct iderp { int a, b, c; } *d, *de;
...
for(d = ...; d < de; ++d)
        s[d->a] += s[d->b] * w[d->c];
```

However, this has the disadvantage of using a potentially large `s` array, many components of which must be initialized to zero. One possible alternative is to again use a "big switch", such as

```
for(;;)
  switch(*u) {
    case ASL_derp_copy:
         s[u[1]] = s[u[2]];
         u += 3;  break;
    case ASL_derp_add:
         s[u[1]] += s[u[2]];
         u += 3;  break;
    case ASL_derp_copyneg:
         s[u[1]] = -s[u[2]];
```

```
        u += 3;  break;
    case ASL_derp_addneg:
        s[u[1]] -= s[u[2]];
        u += 3;  break;
    case ASL_derp_copymult:
        s[u[1]] = s[u[2]]*w[u[3]];
        u += 4;  break;
    case ASL_derp_addmult:
        s[u[1]] += s[u[2]]*w[u[3]];
        u += 4;  break;
    ...
```

It is simpler to exploit the fact that, aside from "defined variables" (which amount to named common subexpressions and are discussed more in §5 below), each intermediate value is used just once, which allows us to use a loop of the form

```
for(d = ...; d < de; ++d) {
        t = s[d->b] * w[d->c];
        if ((a = d->a) >= a0)
                s[a] = t;
        else
                s[a] += t;
        }
```

In this loop, components `s`[i] with $i <$ `a0` correspond to decision variables x_i or to defined variables; such components must be initialized suitably, e.g., with linear coefficients in the case of decision variables. There is no need to initialize the other components of `s`, and the `s` array can be much shorter. The `w` array contains known constants, variable values, and computed results, including (where appropriate) partial derivatives, with a layout of the form

$$
\begin{array}{rl}
 & const_1 \\
 & \ldots \\
 & const_k \\
w -> & var_1 \\
 & var_2 \\
 & \ldots \\
 & var_n \\
 & result_1 \\
 & \ldots
\end{array}
$$

so that `w[0]` is the value of the first decision variable and negative subscripts are for constants.

5 Funneling Defined Variables

AMPL permits declaring a "variable" whose value is computed from an expression involving other variables. This is logically equivalent to introducing a new variable and an equality constraint that specifies the new variable's value, except that some solvers may arrange for equality constraints only to be satisfied in the limit, whereas defined variables always have exactly their specified value (except for roundoff errors). Some problems that would otherwise involve equality constraints may be stated as unconstrained problems with the help of defined variables, making a wider range of solvers available to solve such problems.

When a defined variable is used in several constraints or objectives, and the expression for the defined variable is sufficiently complicated, it can be worthwhile to precompute the partials of the defined variable with respect to the variables on which it depends. This is sometimes called *funneling* the gradient computation. The current heuristic is to funnel μ adjoint operations to k variables when $\mu > 3k$, since with no funnel we would do at least 2μ adjoint-propagation operations (as the defined variable is shared by at least two constraints or objectives); creating the funnel costs about $\mu + k$ adjoint operations, and applying it twice costs $2k$ more such operations. We only want to do this when it saves time, i.e., when $2\mu > \mu + 3k$.

The MINPACK [8] test problem "Chebyquad", as given in the following AMPL model, provides an example where funnels are worthwhile:

```
# chb50b.mod: MINPACK Chebyquad 50 as
# both objective and constraints
param n > 0 default 50;
var x {j in 1..n} := j/(n+1);
var Tj{j in 1..n} = 2*x[j] - 1;
var T{i in 0..n, j in 1..n} =
        if (i = 0) then 1
        else if (i = 1) then Tj[j]
        else 2 * Tj[j] * T[i-1,j] - T[i-2,j];
minimize ssq: sum{i in 1..n} (
                  (1/n) * sum {j in 1..n} T[i,j]
                  - if (i mod 2 = 0) then 1/(1-i^2)
                  )^2;
s.t. eqn {i in 1..n}:
   (1/n) * sum{j in 1..n} T[i,j] =
         if (i mod 2 = 0) then 1/(1-i^2) else 0;
```

In this model, `T`$[i,j]$ is the ith Chebyshev polynomial evaluated at variable `x`$[j]$, and `Tj` is a helper defined variable.

Table 2 shows evaluation times for the above model with some ASL variants, some without funneling, relative to evaluation time for "old ASL", which is the ASL that has long been available; here and below, unless otherwise noted, timing is on

Table 2 Relative evaluation times for `ch50b` with ASL variants

ASL variant	$f, \nabla f$		$c, \nabla c$	
	32-bit	64-bit	32-bit	64-bit
Old ASL with funnels	1.00	1.00	1.00	1.00
Old ASL without funnels	1.68	1.14	1.62	1.15
Bad ASL with funnels	1.01	0.67	6.47	3.68
New ASL with funnels	0.40	0.28	0.39	0.28
New ASL without funnels	0.45	0.29	0.44	0.29

the same machine used for Table 1. The "bad ASL with funnels" variant was my initial attempt to improve funneling by using data structures that I thought would take less memory. The "new ASL" variant uses an operation list, but the same funneling approach as the old ASL.
Using funnels sometimes gives faster gradient computations. It remains to be seen whether other ways of handling funnels would sometimes give still faster gradient computations.

6 Detecting and Extracting Quadratic Forms

Some solvers give special treatment to quadratic objectives and possibly quadratic constraints. With the old ASL, determining whether an objective or constraint is quadratic and, if so, extracting its Hessian matrix, is done with `qpcheck()` functions that require calling a special `qp_read()` routine to read the .nl file. To compute the values of nonlinear, nonquadratic objectives or constraints, it is then necessary to call a `qp_opify()` routine, and to compute Hessians or Hessian-vector products, it is necessary to completely re-read the .nl file. With the new ASL, the `qpcheck()` routines operate directly on operation lists, carrying out an "evaluation" that computes expression information rather than numeric values, so `qp_opify()` is not needed and Hessians or Hessian-vector products can be computed without re-reading the .nl file. Similar comments apply to the `indicator_constrs()` routine that helps process indicator constraints: The old ASL required calling `qp_read`, and the new ASL does not.

7 Hessians and Hessian-vector Products

For computing Hessians, an approach described by Bruce Christianson [9] works well. Let $p \in \mathbb{R}^n$ be nonzero and consider

$$\phi(\tau) = f(x + \tau p).$$

Then

$$\phi'(\tau) = \nabla f(x + \tau p)^T p. \tag{4}$$

If we compute $\phi'(0)$ by forward AD, then we can use reverse AD to compute $\nabla^2 f(x)\, p$, i.e., a Hessian-vector product. Solvers that use a (possibly preconditioned) nonlinear conjugate-gradient algorithm can use Hessian-vector products directly. For solvers that use explicit (possibly sparse) Hessian matrices, we can use Hessian-vector products to assemble the Hessian matrix.

Griewank and Toint [10] point out that many objectives $f(x)$ have the form

$$f(x) = \sum_{i=1}^{q} f_i(U_i x) \tag{5}$$

in which each matrix $U_i \in \mathbb{R}^{m_i \times n}$ has only a few rows (i.e., $m_i \ll n$). For such a function f,

$$\nabla f(x) = \sum_{i=1}^{q} U_i^T \nabla f_i(U_i x)$$

and

$$\nabla^2 f(x) \;=\; \sum_{i=1}^{q} U_i^T \nabla^2 f_i(U_i x) U_i.$$

In LANCELOT [11] Conn, Gould and Toint exploit "group partially separable" structure:

$$f(x) = \sum_{i=1}^{q} \theta_i\Big(\sum_{j=1}^{k_i} f_{ij}(U_{ij} x)\Big)$$

in which $\theta_i()$ is a unary function. For such f, if $\psi_i(x) = \sum_{j=1}^{k_i} f_{ij}(U_{ij} x)$, then

$$\nabla f(x) \;=\; \sum_{i=1}^{q} \theta_i'(\psi_i(x)) \sum_{j=1}^{k_i} U_{ij}^T \nabla f_{ij}(U_{ij} x)$$

and

$$\begin{aligned}\nabla^2 f(x) = \sum_{i=1}^{q} \{ \quad & \theta_i'(\psi_i(x)) \sum_{j=1}^{k_i} U_{ij}^T \nabla^2 f_{ij}(U_{ij} x) U_{ij} \\ & + \theta_i''(\psi_i(x)) \sum_{j=1}^{k_i} (U_{ij}^T \nabla f_{ij}(U_{ij} x))(U_{ij}^T \nabla f_{ij}(U_{ij} x))^T \}.\end{aligned}$$

By using a suitable expression-graph walk [12], we can find the structure (5) automatically. This graph walk is fairly elaborate, so the new ASL initially keeps the same expression graph representation as the old ASL. After the structure (5) is found, the new ASL does another graph walk to produce the operation lists used for evaluations. With both ASL versions, the net effect is that AMPL users can exploit the structure (5) without even being aware of it.

An example with a rich structure (5) is an empirical energy function for protein folding. In connection with [13], Teresa Head-Gordon provided us with Fortran for the CHARM empirical energy function, and we converted it to an AMPL model, `pfold.mod`, of which the following is an excerpt:

```
# CHARM empirical energy function, derived
# from Fortran supplied by Teresa Head-Gordon.
set D3 circular := 1..3;
set Atoms; var x{i in Atoms, j in D3};
set Bonds;
param ib{Bonds} integer;
param jb{Bonds} integer;
param fcb{Bonds}; param b0{Bonds};

var bond_energy = sum{i in Bonds} fcb[i] *
 (sqrt(sum{j in D3} (x[ib[i],j] - x[jb[i],j]})^2)
   - b0[i])^2;
# ...
minimize energy:   bond_energy + angle_energy
                 + torsion_energy + improper_energy
                 + pair14_energy + pair_energy;
```

The decision variables are the Cartesian coordinates of a collection of atoms. The overall empirical energy is the sum of six energy terms, each a sum of nonlinear expressions involving differences of the coordinates of the atoms involved. Thus, each $U_{ij}x$ is a vector of coordinate differences.

8 Implementations of Hessian-vector Products

When arranging to compute Hessians and Hessian-vector products, the old ASL used `struct expr2`, shown in Fig. 3, which is a more detailed version of the `expr` structure shown in Fig. 1. The `fwd` and `bak` pointers in Fig. 3 enable computing (4), i.e., $\phi'(\tau)$, and applying reverse AD to this computation. In the old ASL, computing (4) used code of the form shown in Fig. 4. This code uses partial derivatives stored in the `expr2` structure shown in Fig. 3, making the code nonreentrant. The new ASL stores thread-dependent quantities in a separate workspace, some of which is composed of structures of the form shown in Fig. 5. In the new ASL, computing (4) uses code of the form shown in Fig. 6. The assignment "`e = e->fwd`" in Fig. 4, to

```
struct expr2 {
   real (*op)(expr2 *);
   int a;        /* adjoint index, then operator class */
   expr2 *fwd, *bak;
   real dO;      /* deriv of op w.r.t. t in x + t*p */
   real aO;      /* adjoint (in Hv computation) of op */
   real adO;     /* adjoint (in Hv computation) of dO */
   real dL;      /* deriv of op w.r.t. left operand */
   expr2 *L, *R; /* left and right operands */
   real dR;      /* deriv of op w.r.t. right operand */
   real dL2;     /* second partial w.r.t. L, L */
   real dLR;     /* second partial w.r.t. L, R */
   real dR2;     /* second partial w.r.t. R, R */
   };
```

Fig. 3 Old ASL binary expression for gradients and Hessians

```
void hv_fwd(expr *e) {
...
for(; e; e = e->fwd) {
    e->aO = e->adO = 0;
    switch(e->a) {
        ...
        case Hv_binaryLR:
            e->dO = e->L->dO*e->dL + e->R->dO*e->dR;
            break;
        case Hv_minusR:
            e->dO = -e->R->dO;
            break;
        ...
        }}}
```

Fig. 4 Old ASL forward computation of ϕ'

```
struct Eresult {
  real O;   /* op value */
  real dO;  /* deriv of op w.r.t. t in x + t*p */
  real aO;  /* adjoint (in Hv computation) of O */
  real adO; /* adjoint (in Hv computation) of dO */
  real dL;  /* deriv of op w.r.t. left operand L */
  real dL2; /* second partial w.r.t. L,L */
  real dR;  /* deriv of op w.r.t. right operand R */
  real dLR; /* second partial w.r.t. L,R */
  real dR2; /* second partial w.r.t. R,R */
  };
```

Fig. 5 New ASL data structure for operation results

```
# denhex.mod: dense Hessian
# expressed without structure
void hv_fwd(int *o, real *w, ...) { ...
for(;;) {
    switch(*o) { ...
        case nOPDIV2:
            r = (Eresult*)(w + o[2]);
            L = (Eresult*)(w + o[3]);
            R = (Eresult*)(w + o[4]);
            r->dO = L->dO*r->dL + R->dO*r->dR;
            o += 5;
            break;
        ... }
        r->aO = r->adO = 0.;
        }}
```

Fig. 6 New ASL forward computation of ϕ'

```
void hv_back(expr *e) { ...
for(; e; e = e->bak) {
    switch(e->a) { ...
        case Hv_binaryLR:
            e1 = e->L;
            e2 = e->R;
            adO = e->adO;
            t1 = adO * e1->dO;
            t2 = adO * e2->dO;
            e1->aO  += e->aO*e->dL + t1*e->dL2 + t2*e->dLR;
            e2->aO  += e->aO*e->dR + t1*e->dLR + t2*e->dR2;
            e1->adO += adO * e->dL;
            e2->adO += adO * e->dR;
            break;
     ... }}}
```

Fig. 7 Old ASL reverse AD of ϕ'

move forward to the next relevant operation, is replaced in Fig. 6 by the assignment "`o += 5`" (i.e., "`o = o + 5`"). The assignment "`r->aO = r->adO = 0.`@" after the `switch` statement in Fig. 6 is an initialization in preparation for the reverse AD on ϕ' that is subsequently carried out.

In the old ASL, applying reverse AD to (4) used code of the form shown in Fig. 7. In the new ASL, the corresponding code has the form shown in Fig. 8. The assignment "`o -= o[1]`" in Fig. 8 corresponds to "`e = e->bak`" in Fig. 7.

9 Comparative Timings

The tables below present some timings of the new ASL relative to the old on some test problems summarized in Table 3. Problems *bearing*, *clnlbeam*, *gasoil*, and *henon80* were provided by Hans Mittelmann [14]; problem *denhex* is from the AMPL model

```
void hv_back(int *o, real *w) { ...
for(;;) {
    switch(o[0]) { ...
        case nOPPOW2: case nOP_atan22:
            r = (Eresult*)(w + o[2]);
            L = (Eresult*)(w + o[3]);
            R = (Eresult*)(w + o[4]);
            L->adO += r->adO * r->dL;
            R->adO += r->adO * r->dR;
            t1 = r->adO * L->dO;  t2 = r->adO * R->dO;
            L->aO  += r->aO*r->dL + t1*r->dL2 + t2*r->dLR;
            R->aO  += r->aO*r->dR + t1*r->dLR + t2*r->dR2;
            break; ...}
     o -= o[1];
     }}
```

Fig. 8 New ASL reverse AD of ϕ'

Table 3 Test problems

Name	*n*	*m*	*Comments*
			n = number of variables
			m = number of constraints
bearing	16000	0	Sparse quadratic objective
ch50	50	0	Many defined variables
ch50b	50	50	Many defined variables
chemeq	38	12	Nonlinear objective, linear constraints
clnlbeam	59999	40000	All nonlinear
denex	500	0	Dense quadratic, all explicit
denop	5000	0	Quadratic, dense due to outer-product
gasoil	32001	31998	Quadratic objective, nonlinear constraints
henon80	21601	161	Linear objective, 80 quadratic constraints
			81 more nonlinear constraints
pfold3	66	0	Many defined variables

```
# denhex.mod: dense Hessian
# expressed without structure
param n integer > 0 default 500;
set I = 1 .. n;
var x{I};
minimize q:   sum{i in I} x[i]^2
            + sum{i in I, j in I} i*j*x[i]*x[j];
```

problem *denhop* is from the AMPL model

```
# denhop.mod: dense, structured Hessian
param n integer > 0 default 5000;
set I = 1 .. n;
var x{I};
minimize q:    sum{i in I} x[i]^2
            + (sum{i in I} i*x[i])^2;
```

problem *pfold3* is the `pfold.mod` mentioned in Sect. 7, together with data connected with [13]; problem *chemeq* is from `chemeq.mod` in [15]; problem *ch50b* corresponds to the `ch50b.mod` shown above, and *ch50* is `ch50b.mod` with just the least-squares objective (no constraints).

Problems *denex* and *denop* have dense Hessians and are for timing Hessian-vector products. The other problems have sparse Hessians, and the timings are for computing them explicitly.

Table 4 shows relative times: new ASL time divided by old ASL time, for computing function and gradient values ("$f, \nabla f$" indicates function and gradient values for the objective, and "c, ∇c" indicates corresponding values for the constraints). The "no Hes." results only involve computing and storing first partial derivatives, whereas the "Hes." results include time for computing (but not using) second partial derivatives. With both old and new ASL, possibly useful partial derivatives are computed during function evaluations and are stored for possible later use in computing gradients and Hessians or Hessian-vector products.

Table 4 Relative function and gradient times: new ASL divided by old, computed on an Intel Celeron CPU with 2048 KB of cache

		32-bit	64-bit	32-bit	64-bit
		no Hes.	no Hes.	Hes.	Hes.
bearing	$f, \nabla f$	0.57	0.41	0.41	0.42
ch50	$f, \nabla f$	0.62	0.52	0.43	0.28
ch50b	$f, \nabla f$	0.92	0.49	0.40	0.28
ch50b	$c, \nabla c$	0.11	0.08	0.40	0.28
chemeq	$f, \nabla f$	0.68	0.67	0.77	0.85
clnlbeam	$f, \nabla f$	0.63	0.55	0.74	0.68
clnlbeam	$c, \nabla c$	0.23	0.23	0.77	0.64
denex	$f, \nabla f$	0.56	0.43	0.15	0.21
denop	$f, \nabla f$	0.53	0.48	0.64	0.54
gasoil	$f, \nabla f$	0.56	0.39	0.28	0.27
gasoil	$c, \nabla c$	0.48	0.36	0.82	0.68
henon80	$c, \nabla c$	0.11	0.12	0.58	0.45
pfold3	$f, \nabla f$	0.74	0.64	0.73	0.74

Table 5 Relative function and gradient times: new ASL divided by old, computed on an Intel Core 2 Quad CPU with 4096 KB of cache

		32-bit no Hes.	64-bit no Hes.	32-bit Hes.	64-bit Hes.
bearing	$f, \nabla f$	0.35	0.29	0.40	0.36
ch50	$f, \nabla f$	0.75	0.73	0.69	0.36
ch50b	$f, \nabla f$	0.89	0.78	0.47	0.29
ch50b	$c, \nabla c$	0.12	0.11	0.48	0.30
chemeq	$f, \nabla f$	0.85	0.84	0.86	0.89
clnlbeam	$f, \nabla f$	0.57	0.46	0.42	0.42
clnlbeam	$c, \nabla c$	0.23	0.20	0.71	0.65
denex	$f, \nabla f$	0.32	0.25	0.30	0.25
denop	$f, \nabla f$	0.58	0.61	0.73	0.74
gasoil	$f, \nabla f$	0.72	0.54	0.31	0.29
gasoil	$c, \nabla c$	0.37	0.32	0.63	0.57
henon80	$c, \nabla c$	0.13	0.10	0.44	0.33
pfold3	$f, \nabla f$	0.78	0.79	0.84	1.07

Cache size and various CPU details may affect the results in Table 4. The Intel Celeron CPU used for Table 4 has 2048 KB of cache. Tables 5 and 6 are similar to Table 4, and were computed using the same binaries (compiled by gcc with –O2), but for different CPUs. The results in Table 5 were computed on an Intel Core 2 Quad CPU with 4096 KB of cache, and those in Table 6 were computed on an Intel Core i7-4700MQ CPU with 6144 KB of cache.

Despite the differences among the CPUs considered here, the general trend is that the new ASL, with its use of operation lists, often runs faster for function and gradient evaluations than does the old ASL. Moreover, the new ASL often takes less memory to represent nonlinear expressions when just one thread is used, as indicated by Table 7, which shows ratios of net memory use for the new ASL relative to the old.

The figures in Table 7 include memory for the operation lists as well as for one thread-specific work array. Each additional thread just needs its own work array, so substantial memory savings are often possible on large problems when multiple threads are used. Allocation of large memory blocks may cause small problems like *chemeq* to take more memory for one thread, but even small problems generally take less additional memory for each additional thread. For example, with *chemeq*, each additional thread requires less than a quarter of the memory needed by the old ASL.

Table 6 Relative function and gradient times: new ASL divided by old, computed on an Intel Core i7-4700MQ CPU with 6144 KB of cache

		32-bit no Hes.	64-bit no Hes.	32-bit Hes.	64-bit Hes.
bearing	$f, \nabla f$	0.44	0.29	0.39	0.29
ch50	$f, \nabla f$	0.58	0.51	0.48	0.19
ch50b	$f, \nabla f$	0.78	0.58	0.38	0.17
ch50b	$c, \nabla c$	0.12	0.10	0.41	0.18
chemeq	$f, \nabla f$	0.63	0.64	0.81	0.89
clnlbeam	$f, \nabla f$	0.55	0.45	0.55	0.41
clnlbeam	$c, \nabla c$	0.25	0.21	0.62	0.53
denex	$f, \nabla f$	0.49	0.32	0.15	0.14
denop	$f, \nabla f$	0.51	0.48	0.66	0.64
gasoil	$f, \nabla f$	0.49	0.43	0.22	0.23
gasoil	$c, \nabla c$	0.37	0.27	0.76	0.62
henon80	$c, \nabla c$	0.12	0.11	0.50	0.36
pfold3	$f, \nabla f$	0.63	0.54	0.71	0.75

Table 7 Relative memory use (one thread): new ASL divided by old

	32-bit no Hes.	64-bit no Hes.	32-bit Hes.	64-bit Hes.
bearing	0.65	0.42	0.68	0.60
ch50	0.75	0.51	0.45	0.33
ch50b	0.85	0.57	0.49	0.36
chemeq	6.65	3.70	3.04	2.49
clnlbeam	0.78	0.58	0.52	0.48
denex	0.68	0.41	0.78	0.58
denop	0.85	0.59	0.90	0.86
gasoil	0.89	0.70	0.90	0.86
henon80	0.69	0.42	0.70	0.61
pfold3	0.89	0.62	0.88	0.82

10 Concluding Remarks

For large problems, using lists of operations rather than executable expression graphs often leads to faster computations and less memory use, particularly when multiple threads are deployed that do independent function (and gradient, etc.) evaluations. Source for the new ASL, using operation lists, appears in [16].

References

1. Fourer, R., Gay, D.M., Kernighan, B.W.: A modeling language for mathematical programming. Manage. Sci. **36**(5), 519–554 (1990)
2. Fourer, R., Gay, D.M., Kernighan, B.W.: AMPL: A Modeling Language for Mathematical Programming. Duxbury Press / Brooks/Cole Publishing Company (2003), second edition
3. Fourer, R., Gay, D.M.: Experience with a primal presolve algorithm. In: Hager, W.W., Hearn, D.W., Pardalos, P.M. (eds.) Large Scale Optimization: State of the Art, pp. 135–154. Kluwer (1994)
4. Turbo Pascal Reference Manual, Borland International (1983)
5. Gay, D.M.: Automatic differentiation of nonlinear AMPL models. In: Griewank, A., Corliss, G.F. (eds.) Automatic Differentiation of Algorithms: Theory, Implementation, and Application. SIAM (1991)
6. Kearfott, R.B.: GlobSol user guide. Opti. Methods Softw. **24**(4–5), 687–708 (2009)
7. Griewank, A., Walther, A.: Evaluating Derivatives. SIAM (2008)
8. Moré, J.J., Garbow, B.S., Hillstrom, K.E.: Guide for MINPACK–1. ANL–80–74, Argonne National Laboratory, Argonne, IL (1980)
9. Christianson, B.: Automatic Hessians by reverse accumulation. IMA J. Numer. Anal. **12**, 135–150 (1992)
10. Griewank, A., Toint, Ph.L.: Optimization of partially separable functions. In: Powell, M.J.D. (ed.) Nonlinear Optimization 1981, pp. 301–312. Academic Press, Cambridge (1982)
11. Conn, A.R., Gould, N.I.M., Toint, Ph.L.: LANCELOT, a Fortran package for large-scale nonlinear optimization (Release A). Springer Series in Computational Mathematics 17 (1992)
12. Gay, D.M.: More AD of nonlinear AMPL models: computing Hessian information and exploiting partial separability. In: Corliss, G.F. (ed.) Computational Differentiation: Applications, Techniques, and Tools. SIAM (1996)
13. Head-Gordon, T., Stillinger, F.H., Gay, D.M., Wright, M.H.: Poly(L-alanine) as a universal reference material for understanding protein energies and structures. Proc. Natl. Acad. Sci. USA **89**, 11513–11517 (1992)
14. Mittelman, Hans. Private communication (2017)
15. http://ampl.com/netlib/ampl/models/nlmodels
16. http://ampl.com/netlib/ampl/solvers2.tgz

On the Consistency of Runge–Kutta Methods Up to Order Three Applied to the Optimal Control of Scalar Conservation Laws



Michael Hintermüller and Nikolai Strogies

Abstract Higher-order Runge–Kutta (RK) time discretization methods for the optimal control of scalar conservation laws are analyzed and numerically tested. The hyperbolic nature of the state system introduces specific requirements on discretization schemes such that the discrete adjoint states associated with the control problem converge as well. Moreover, conditions on the RK coefficients are derived that coincide with those characterizing strong stability preserving Runge–Kutta methods. As a consequence, the optimal order for the adjoint state is limited, e.g., to two even in the case where the conservation law is discretized by a third-order method. Finally, numerical tests for controlling Burgers' equation validate the theoretical results.

Keywords Optimal control · Conservation laws · Discretization methods
RK methods · TVD-RK

1 Introduction

We investigate discretization techniques for problems of optimal control subject to scalar conservation laws in one space dimension which, in conservative form, are given as

$$\begin{aligned} y_t + [f(y)]_x &= 0 \qquad \text{in } \mathcal{Q} := (0, T] \times \mathbb{R}, \\ y(0, x) &= u(x) \quad \text{in } \mathbb{R}. \end{aligned} \tag{1}$$

M. Hintermüller · N. Strogies
Weierstrass Institute, Mohrenstr. 39, 10117 Berlin, Germany
e-mail: michael.hintermueller@wias-berlin.de; hint@math.hu-berlin.de

N. Strogies
e-mail: nikolai.strogies@wias-berlin.de

M. Hintermüller
Humboldt-Universität zu Berlin, Unter den Linden 6, 10099 Berlin, Germany

© Springer International Publishing AG, part of Springer Nature 2018

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*,
Springer Proceedings in Mathematics & Statistics 235,
https://doi.org/10.1007/978-3-319-90026-1_6

Here, $f \in C^2(\mathbb{R})$ is a nonlinear convex flux function that is uniformly convex with $f'' \geq c > 0$. Partial differential equations like (1) might, even for smooth initial data, develop shocks (see, e.g., [8]) and thus require the consideration of weak solutions that satisfy additional conditions, guaranteeing uniqueness of solutions. In case of conservation laws, usually modeling physical processes, the relevant solution is called entropy solution. It is known that the map of control to entropy solution, $u(\cdot) \mapsto y(t, \cdot)$, is usually not differentiable in $L^1(\mathbb{R})$ but *shift-differentiable* in $BV(\mathbb{R})$. This notion of directional differentiability was introduced and discussed for balance laws, inhomogeneous conservation laws with an additional source term, in [9, 32], and extended to strictly hyperbolic systems of balance laws in [4, 11].

In this paper, we consider the model problem

$$\begin{aligned} \text{minimum} \quad & \frac{1}{2}\int_{\mathbb{R}} (y(T,x) - y^d(x))^2 dx + \mathfrak{R}(u) =: \mathcal{J}(y,u) \\ \text{over} \quad & (y,u) \in (L^\infty(\mathcal{Q}) \cap C([0,T]; L^1_{loc}(\mathbb{R}))) \times L^\infty(\mathbb{R}) \\ \text{subject to} \quad & y \text{ solves (1) for } u, \end{aligned} \tag{P}$$

for a desired state $y^d \in PC^1(\mathbb{R})$ with bounded support, the control entering as initial data for (1) and a suitable convex cost functional $\mathfrak{R}(u)$ with an effective domain $dom_{\text{eff}}(\mathfrak{R})$ embedding compactly into $L^1(\mathbb{R})$ and being coercive with respect to $\|u\|_{L^\infty(\mathbb{R})}$. In case of scalar conservation laws, theoretical results for optimality conditions of (P) have been discussed, for example, in [32] in case of an unbounded domain and in [29] for bounded domains with a switching control at the boundary. The main focus of these papers lies on obtaining a representation of the reduced gradient of

$$\frac{1}{2}\int_{\mathbb{R}} (y(T,x) - y^d(x))^2 dx \tag{2}$$

with respect to perturbations in the control. If the conservation law satisfies the weakened one-sided Lipschitz condition (OSL), then the reduced gradient is given by the solution to the adjoint equations (5) below, a linear conservation law with discontinuous coefficients. It is known that such equations, even for Lipschitz continuous terminal conditions, do not admit unique solutions (see, e.g., [6]), and the relevant *reversible solution* is identified utilizing solutions to the sensitivity equation (6) below.

In case of systems of balance laws, theoretical results concerning optimality conditions for a distributed control have been formulated in [10] for an unbounded domain and in [12] for bounded domains in terms of generalized tangent vectors, the first-order variations of the solution.

The numerical treatment of problems of optimal control subject to scalar conservation laws has been studied, for example, in [3]. In case of systems, a first step for function space consistent numerical methods is considered in [22], where a numerical method for the computation of generalized tangent vectors for a system of conservation laws has been introduced.

Addressing (P) numerically is a delicate task since the nonuniqueness of solutions to the adjoint equation requires suitable discretization techniques for both state and adjoint equations, respectively. In [17], numerical results for Burgers' equation have been obtained where the discrete solutions to the adjoint equation can converge to an incorrect solution if the discretization scheme is not chosen properly. In [15, 16], the convergence behavior of a discretization of the primal equation based on the Lax–Friedrichs flux and a mesh-dependent artificial viscosity has been studied. The consistency of the discretized problems with (P) has been proven in [32] for monotone discretization schemes of (1) satisfying certain assumptions. Such discretizations can be interpreted as explicit Euler time discretizations of a system of ordinary differential equations representing a semidiscretization of the conservation law. We will investigate higher-order Runge–Kutta time discretization methods applied to this semidiscretization and derive conditions for the coefficients such that the resulting full discretization of (P) is still consistent.

In the context of optimal control subject to ordinary differential equations, the application of Runge–Kutta (RK) time discretization schemes has been investigated, for example, in [5, 20].

The application of RK schemes to semidiscretizations of conservation laws has been a subject of investigation for a long time (see, e.g., [19, 24]). Given the discretization of a conservation law that is total variation diminishing (TVD) for the basic explicit Euler time stepping, these RK methods are used to construct higher-order approximations of the solutions with respect to time while preserving the overall TVD property of the full discretization scheme.

Notation. Throughout this paper, Δx denotes the width of the spatial discretization, Δt the size of the time step, and $\lambda = \Delta t/\Delta x$. The terminal time is denoted by T, and $N = T/\Delta t$ represents the number of time steps. Moreover, we utilize several function spaces and refer to [1, 14] for details. Besides the standard Lebesgue and Sobolev spaces L^p_{loc} and $W^{k,p}(\Omega)$ with corresponding norms, we consider the space of piecewise continuously differentiable functions with possible discontinuities at finitely many points, the space of all bounded functions that can be approximated pointwise almost everywhere by a sequence of Lipschitz continuous functions $\{w_n\}$ bounded in $C(\mathbb{R}) \cap W^{1,1}_{loc}(\mathbb{R})$, and the space of bounded functions, $PC^1(\Omega)$, $B_{Lip}(\mathbb{R})$, and $B(\mathbb{R})$, respectively, all equipped with the sup norm. Finally, $\mathcal{S}_{\mathcal{M}} := C([0,T]; \mathcal{M}_{loc}(\mathbb{R}) - w(\mathcal{M}_{loc}(\mathbb{R}), C_c(\mathbb{R})))$ denotes the space of continuous functions on $[0,T]$ with values in the local Borel measures on $\mathbb{R}$, $\mathcal{M}(\mathbb{R})$, with the weak topology induced by continuous functions with compact support.

The paper is organized as follows: In Sect. 2, we provide existence results for solutions of the primal and adjoint equations and for (P). Moreover, we introduce the semidiscretization of (1) and recall conditions for the consistency of the fully discretized problems with (P) for time discretizations based on Euler's method. In Sect. 3, we analyze RK schemes and the resulting discretizations of (1) and present conditions for consistency of the resulting discretizations. In Sect. 4, we briefly discuss the fully discretized problems and convergence order of the time discretization, and in Sect. 5, we present numerical results validating our theory of Sect. 3.

2 Preliminaries

In this section, we study appropriate solution concepts for the state equation (1), associated adjoint equations, and we discuss the use of the adjoint when computing the reduced gradient of $\mathcal{J}(y, u)$.

2.1 The State Equation and Its Adjoints

As outlined above, even for smooth initial data solutions to (1) might develop shocks and require to study weak solutions $y \in L^\infty(\mathbb{R}_+ \times \mathbb{R})$ that satisfy the identity

$$\int_{\mathbb{R}\times\mathbb{R}_+} y\phi_t + f(y)\phi_x dxdt + \int_{\mathbb{R}} u(\cdot)\phi(0,\cdot)dx = 0 \text{ for all } \phi \in C_c^\infty(\mathbb{R}_+ \times \mathbb{R}).$$

In general, weak solutions are not unique and the physically relevant solution, referred to as *entropy solution* (see, e.g., [26]), is characterized as follows.

Definition 1 Consider $\eta(y) := |y - k|$ and $q(y) := sign(y - k)(f(y) - f(k))$. A weak solution of (1) is an entropy solution if it satisfies

$$\int_{\mathbb{R}\times\mathbb{R}_+} \eta(y)\phi_t + q(y)\phi_x dxdt + \int_{\mathbb{R}} \eta(u(\cdot))\phi(0,\cdot)dx \geq 0$$

for all $\phi \in C_c^\infty(\mathbb{R} \times \mathbb{R}_+)$, $\phi \geq 0$, and $k \in \mathbb{R}$.

The following result provides the existence of such solutions, provides some properties, and goes back to [32, Proposition 3.4.1].

Proposition 1 *For every $u \in L^\infty(\mathbb{R})$, there exists a unique entropy solution*

$$y \in L^\infty(\mathcal{Q}) \cap C([0, T]; L^1_{loc}(\mathbb{R}))$$

that satisfies $\|y(t)\|_{L^\infty(\mathbb{R})} \leq \|u\|_{L^\infty(\mathbb{R})}$ for all $t \in [0, T]$,

$$\|y^1(t) - y^2(t)\|_{L^1(\mathbb{R})} \leq \|u_0^1 - u_0^2\|_{L^1(\mathbb{R})} \text{ for all } t \in (0, T]$$

with $y^i(t)$ denoting the solution of (1) for initial data $u_0^i \in L^\infty(\mathbb{R})$, $i \in \{1, 2\}$ and, for all controls with $\|u\|_{L^\infty(\mathbb{R})} \leq M_u$, $M_u > 0$, there exists some $C = C(M_u, u, t) > 0$ such that

$$y_x(t) \leq C(M_u, u, t) \tag{3}$$

holds in the sense of distributions.

Consequently, (1) is well defined which allows us to consider (P) next. Note here that y depends on the control u entering as initial data in the state equation. Let the desired state y^d be an element of $PC^1(\mathbb{R})$. If the effective domain of $\mathfrak{R}(\cdot)$, $dom_{\text{eff}}(\mathfrak{R})$, embeds compactly into $L^1(\mathbb{R})$, and $\mathfrak{R}(\cdot)$ is coercive with respect to $\|u\|_{L^\infty(\mathbb{R})}$, then (P) admits a solution (see, e.g., [32, Theorem 2.4.2]).

Proposition 1 allows for considering $y = y(u)$; i.e., the entropy solution of (1) depends on the control. In order to establish a gradient-related descent algorithm for solving (P) iteratively, a gradient representation of the reduced objective in (2) is necessary. In fact, let $u \in PC^1(\mathbb{R})$ and consider fixed positions of discontinuities in the initial data. Then, the reduced objective associated with (P) is given by $\hat{\mathcal{J}}(u) := \mathcal{J}(y(u), u)$ and the application of its gradient to a direction $\delta u \in PC^1(\mathbb{R})$ is given by

$$d_u \mathcal{J}(y(u), u) \cdot \delta u = (p(0, \cdot), \delta u)_{L^2(\mathbb{R})} + d_u \mathfrak{R}(u) \cdot \delta u \tag{4}$$

with $L^2(\mathbb{R})$-scalar product, derivative of the cost term $d_u \mathfrak{R}(u)$ and p denoting the solution to the adjoint equation

$$\begin{aligned} p_t + f'(y(t,x))p_x &= 0 \qquad \text{in } \mathcal{Q}, \\ p(T,x) &= p^T(x) \quad \text{in } \mathbb{R}, \end{aligned} \tag{5}$$

with final time data

$$p^T(x) = \int_0^1 (y(T, x+) + \tau[y(T,x)] - y^d(x))d\tau.$$

Here, $[y(T,x)] := y(T,x-) - y(T,x+)$ denotes a possible jump of the entropy solution to (1) at final time T at x. In general, linear conservation laws with discontinuous coefficients as in (5), even for continuous end data p^T, do not admit unique solutions. While entropy solutions are the physically relevant ones for the nonlinear state equation (1), the proper concept of solutions for (5) is called *reversible solution* and relies on the *one-sided Lipschitz continuity condition*; i.e., there exists an $\alpha \in L^1(0,T)$ with

$$\partial_x f'(y(t,\cdot)) \le \alpha(t). \tag{OSL}$$

Next, we recall the definition of reversible solutions from [6] and [32] for regular and more general final time data p^T, respectively.

Definition 2 In case of regular final time data $p^T \in C^{0,1}_{loc}(\mathbb{R})$, a Lipschitz continuous solution p to (5) is a reversible solution if and only if there exist Lipschitz continuous solutions p_1, p_2 to (5) with $\partial_x p_1 \ge 0$ and $\partial_x p_2 \ge 0$ such that $p = p_1 - p_2$.

In case of $p^T \in B_{Lip}(\mathbb{R})$, they are defined as broad solutions along the generalized backward characteristics.

Broad solutions are solutions to first-order partial differential equations that are, in case of linear conservation laws, constant along the characteristic lines of the

problem; see [12] for the corresponding definitions. In case of nonlinear conservation laws, generalized characteristics have to be considered; see [32] for a discussion. The following result is proven in [32, Corollary 4.2.11].

Proposition 2 *Let f satisfy OSL. Then, for end data $p^T \in B_{Lip}(\mathbb{R})$ there exists a unique reversible solution $p \in B(\mathcal{Q}) \cap C^{0,1}([0,T]; L^1_{loc}(\mathbb{R})) \cap B([0,T]; BV_{loc}(\mathbb{R})) \cap BV_{loc}(\mathcal{Q}^{cl})$ of (5) fulfilling the maximum principle*

$$\|p(t)\|_{B(I)} \leq \|p^T\|_{B(J)}$$

with $I = [z_1, z_2]$ and $J = [z_1 - \|f'(y)\|_{L^\infty}(T-t), z_2 + \|f'(y)\|_{L^\infty}(T-t)]$ for all $z_1, z_2 \in \mathbb{R}$, $z_1 < z_2$ and $t \in [0,T]$.

In case of entropy solutions to (1), OSL of the flux term f is ensured by estimate (3). Moreover, final time data $p^T \in B_{Lip}(\mathbb{R})$ are, for example, elements of $PC^1(\mathbb{R})$.

Reversible solutions are closely related to *duality solutions* (compare Definition 3 below) that characterize directional derivatives of solutions y to the state equation (1) with respect to perturbations in the control u as shown in [7, Theorem 3.1]. In particular, for the perturbed position of a shock in the control u, such sensitivities are measures and defined as weak solutions to

$$\begin{aligned} \mu_t + (f'(y)\mu)_x &= 0 \quad \text{in } \mathcal{Q}, \\ \mu(0) &= \delta u, \end{aligned} \tag{6}$$

for $\delta u \in \mathcal{M}_{loc}(\mathbb{R})$ and y denoting the solution to (1), given the control u (see, e.g., [32, Example 3.1.1]).

Definition 3 Let $f'(y)$ satisfy OSL, and consider $\delta u \in \mathcal{M}_{loc}(\mathbb{R})$. Solutions $\mu \in \mathcal{S}_{\mathcal{M}}$ to (6) are called duality solution if for any $\tau \in (0,T]$, any $p^\tau \in B_{Lip}(\mathbb{R})$ with compact support, and any reversible solution p of

$$\begin{aligned} p_t + f'(y(t,x))p_x &= 0 \qquad \text{in } (0,\tau)\times\mathbb{R}, \\ p(\tau,x) &= p^\tau(x) \quad \text{in } \mathbb{R}, \end{aligned} \tag{7}$$

we have

$$\int_{\mathbb{R}} p^\tau \mu(\tau, dx) = \int_{\mathbb{R}} p(0,x)\delta u(dx)$$

Their existence result follows from [32, Theorem 4.3.7].

Proposition 3 *Let $f'(y)$ satisfy OSL. Then for any $\delta u \in \mathcal{M}_{loc}(\mathbb{R})$, there exists a unique duality solution $\mu \in \mathcal{S}_{\mathcal{M}}$ to (6).*

Moreover, the following relation of reversible and duality solutions has been established in [32, Theorem 4.4.1.]: For every $p^\tau \in B_{Lip}(\mathbb{R})$, $\tau \in (0,T]$, p is a reversible

solution to (7) if it is Borel-measurable and satisfies for all $\sigma \in (0, \tau)$ and all $\delta u \in \mathcal{M}(\mathbb{R})$ the duality relation

$$\int_{\mathbb{R}} p^{\tau} \mu(\tau, dx) = \int_{\mathbb{R}} p(\sigma, x) \delta u(dx), \tag{8}$$

where μ is the duality solution of (6) on $(\sigma, \tau) \times \mathbb{R}$ for the control δu.

2.2 Discrete Schemes

Since state, adjoint, and sensitivity equations, respectively, in general do not admit unique solutions, discretization schemes have to be chosen that approximate the relevant entropy, reversible and duality solution properly. While convergent schemes for (1) are available in terms of monotone schemes, the respective discretization of (5) and (6) has to be derived from such schemes along with properties of reversible and duality solutions.

Monotone schemes operate on cell averages of the solution by accounting for their evolution over time steps. Thus, given a uniform mesh of width Δx on $\mathbb{R}$, the discretization of the initial state is obtained by averaging the function in each interval or cell j of the spatial discretization; i.e.,

$$\mathbf{u_0}_j := T^j(u) = \Delta x^{-1} \int_{x_j}^{x_{j+1}} u(x) dx. \tag{9}$$

with $\Delta x := x_{j+1} - x_j$ for $j \in \mathbb{N}$. A semidiscretization of (1) utilizing the method of lines provides a system of ordinary differential equations (ODEs),

$$\dot{\mathbf{y}}_j = F(\mathbf{y})_j = -\Delta x^{-1} [f^{\Delta}(y_j^n, y_{j+1}^n) - f^{\Delta}(y_{j-1}^n, y_j^n)], \quad \dot{\mathbf{y}}(0) = \mathbf{u}_0. \tag{10}$$

Here, $f^{\Delta} : \mathbb{R}^2 \to \mathbb{R}$ represents a suitable numerical flux function that quantifies the flux between the intervals $j + 1$ and j. It is assumed to be at least Lipschitz continuous and thus guarantees the existence of a unique, Lipschitz continuous solution to (10). Depending on the regularity of the chosen numerical flux and the underlying conservation law, the solution of the semidiscrete system of ODEs might attain even higher regularity with respect to t. In fact, for Burgers' equation and the Engquist–Osher flux (see Sect. 5), the solution to (10) exists and is twice continuously differentiable with respect to time. Using an explicit Euler method with time step size Δt for the time discretization, we obtain

$$y_j^{n+1} = y_j^n - \lambda (f^{\Delta}(y_j^n, y_{j+1}^n) - f^{\Delta}(y_{j-1}^n, y_j^n)), \tag{11}$$

a so-called centered three-point scheme since the average of the state in cell j only depends on the average in the neighboring cells. In more general schemes, the numerical flux function and consequently the average of the state in cell j might depend on cell

averages of the state in further cells, increasing the number of arguments of f^Δ. In the discussion below, this number, compared to (11), will increase but we restrict ourselves to even numbers of arguments for the numerical flux function, i.e., $f^\Delta : \mathbb{R}^{2K} \to \mathbb{R}$ with $K = 1, 2, \ldots$ and formally, f^Δ depending on $(y^n_{j-K+1}, \ldots, y^n_{j+K})$. Thus, the integer K defines the domain of determinacy in that in the general, fully discrete scheme, y^{n+1}_j depends on $y^{n+1}_{j-K}, \ldots, y^{n+1}_{j+K}$; cf. (11) for the case of $K = 1$.

Based on (11), discretization schemes of (5) and (6) can be derived as demonstrated next. Their convergence depends on properties of certain coefficients that are introduced in this process.

Derivatives of entropy solutions to (1) with respect to variations in the initial data are characterized by duality solutions to (6). Thus, if the numerical flux f^Δ is differentiable, a sensitivity scheme can be established that characterizes derivatives of the discrete approximations y^n_j defined by (11) with respect to the discrete initial data $\mathbf{u}$ for $n = 1, \ldots, N$ and $j \in \mathbb{N}$. It is given by

$$\begin{aligned}\mu^{n+1}_j = \mu^n_j - \lambda[&f_2^\Delta(y^n_j, y^n_{j+1})\mu^n_{j+1} + (f_1^\Delta(y^n_j, y^n_{j+1}) \\ &- f_2^\Delta(y^n_{j-1}, y^n_j))\mu^n_j - f_1^\Delta(y^n_{j-1}, y^n_j)\mu^n_{j-1}]\end{aligned} \tag{12}$$

along with the initial data $\mu^0_j = \delta \mathbf{u}_\mathbf{j}$. Here, $f_l^\Delta(v_1, v_2)$ represents the partial derivative of f^Δ with respect to the l-th argument. This scheme can be written explicitly as

$$\mu^{n+1}_j = \sum_{k=-1}^{1} D^n_{j,k}\mu^n_{j+k} \qquad \text{for} \qquad \begin{aligned} D^n_{j,-1} &= \lambda f_1^\Delta(y^n_{j-1}, y^n_j), \\ D^n_{j,0} &= 1 - \lambda(f_1^\Delta(y^n_j, y^n_{j+1}) - f_2^\Delta(y^n_{j-1}, y^n_j)), \\ D^n_{j,1} &= -\lambda f_2^\Delta(y^n_j, y^n_{j+1}). \end{aligned} \tag{13}$$

Note that if these coefficients are nonnegative, (11) forms a monotone discretization scheme since they represent the partial derivatives in, e.g., [31, Definition 5.1]. Similar to the relation of duality and reversible solutions in the continuous setting, the discretization scheme for the adjoint equation has to satisfy the discrete analogue of (8), obtained for $\sigma = t$, $\tau = t + \Delta t$ and rescaling the sum representing the integrals, given by

$$\sum_j p^{n+1}_j \mu^{n+1}_j = \sum_j p^n_j \mu^n_j, \tag{14}$$

for any solution $\{\mu^n_j\}$ of (12) with bounded support, i.e., $\mu^0_j \neq 0$ at finitely many $j \in \mathbb{N}$ only. In case of tracking type objective functionals for desired states with bounded support, the control is assumed to have bounded support as well. Multiplying (13) by p^{n+1}_j, a summation over j and reordering provides

$$\sum_j p^{n+1}_j \mu^{n+1}_j = \sum_j \sum_{k=-1}^{1} D^n_{j-k,k} p^{n+1}_{j-k} \mu^n_j.$$

A substitution into (14) yields the discretization scheme for the adjoint equation, given by

$$p_j^n = p_j^{n+1} + \lambda(D_{j+1,-1}(p_{j+1}^{n+1} - p_j^{n+1}) - D_{j-1,1}(p_j^{n+1} - p_{j-1}^{n+1})), \tag{15}$$

a time discretization for the system of ordinary differential equations

$$\begin{aligned}\dot{\mathbf{p}}_j = \Delta x^{-1}[&f_2(y_{j-1}(t), y_j(t))(p_{j+1}(t) - p_j(t)) \\ &- f_1(y_j(t), y_{j+1}(t))(p_j(t) - p_{j-1}(t))]\end{aligned} \tag{16}$$

utilizing an implicit Euler method along with discrete final time condition $\mathbf{p}(T) = d_{\mathbf{y}(T)}\mathcal{J}(\mathbf{y}(T), \mathbf{u})$. In case of (P), the latter is given as $\mathbf{p}(T)_j = \Delta x(\mathbf{y}(T)_j - T^j(y^d))$.

The following set of coefficients allows for quantifying the difference of neighboring values of the solution to (15) after one time step in the adjoint scheme.

$$\begin{aligned}p_{j+1}^{n+1} - p_j^{n+1} &= \sum_{k=-1}^{1} C_{j,k}^n (p_{j+k+1}^n - p_{j+k}^n), \text{ with} \\ C_{j,-1}^n &= -\lambda f_2^{\Delta}(y_{j-1}^n, y_j^n), \\ C_{j,0}^n &= 1 + \lambda(f_2^{\Delta}(y_j^n, y_{j+1}^n) - f_1^{\Delta}(y_j^n, y_{j+1}^n)), \\ C_{j,1}^n &= \lambda f_1^{\Delta}(y_{j+1}^n, y_{j+2}^n).\end{aligned} \tag{17}$$

Finally, the existence of reversible and duality solutions depends on OSL, respectively, to ensure convergence of their corresponding discrete approximations, the schemes have to be consistent with it. A sufficient condition for OSL consistency that relies on the coefficients

$$l_j^{n+1} := \Delta x^{-1}(y_{j+1}^n - y_j^n), l_{j,K}^{n,+} := \max(0, l_{j-K}^n, \ldots, l_{j+K}^n). \tag{18}$$

was studied in [32, Sect. 6.4.4.] and allows to establish a bound on $\partial_x f'(y)(t, \cdot)$.

By the coefficients introduced above, we are now able to present the following theorem, collecting results from [32, Theorem 6.4.10 and Theorem 6.4.15.]. It characterizes the consistency of nonlinear programs obtained from (P) by discretizing the state system with suitable schemes like (11) and obtaining gradient information of the reduced objective by associated adjoint schemes like (15). The formulation covers centered three-point schemes, i.e., (11), but also holds for larger domains of determinacy (see discussion below (11)).

Theorem 1 ([32]) *For $u \in PC^1(\mathbb{R})$, which provides $y(T, \cdot) \in PC(\mathbb{R})$, and $y^d \in PC^1(\mathbb{R})$, let the flux function $f \in C^2(\mathbb{R})$ satisfy $f'' \geq c > 0$ and consider $K \geq 1$, $K \in \mathbb{N}$ and $M_y > 0$ sufficiently large. Moreover, let the numerical flux f^{Δ} fulfill the following conditions:*

1. $f^\Delta \in C^{1,1}_{loc}(\mathbb{R}^{2K})$ *and the numerical flux is consistent; i.e.,* $f^\Delta(y,\dots,y) = f(y)$*;*
2. *The coefficients* $D^n_{j,l}$ *defined in* (13) *are nonnegative for all* $y^n_j \in [-M_y, M_y]$*,* $j \in \mathbb{N}$*;*
3. *The coefficients* $C^n_{j,l}$ *defined in* (17) *are nonnegative for all* $y^n_j \in [-M_y, M_y]$*,* $j \in \mathbb{N}$*;*
4. *The discrete state* y^n_j *is contained in* $[-M_y, M_y]$ *for* $j \in \mathbb{N}$*,* $n = 1,\dots,N$ *and there exist some* $\nu > 0$ *such that*

$$l^{n+1}_j \le l^{n,+}_{j,K} - \Delta t\, \nu (l^{n,+}_{j,K})^2; \tag{19}$$

5. *The partial derivatives* $f^\Delta_{y_i}$ *are non decreasing on* $[-M_y, M_y]^{2K}$*.*

Then, the solution to the discrete sensitivity equation (12) *converges to the duality solution of* (6) *in* $B([0,T]; \mathcal{M}_{loc}(\mathbb{R}) - w(\mathcal{M}_{loc}(\mathbb{R}), C_c(\mathbb{R})))$ *and the solution to the discrete adjoint equation* (15) *converges to the solution of* (5) *in* $L^r(Q \setminus \bigcup_k D_k)$ *with* $r \in [1,\infty)$ *and* D_k *depending on the shock positions in the terminal condition for the adjoint equation and the height of the corresponding jumps. Moreover,* $\mathbf{p}^0 \to p(0,\cdot)$ *as* $\Delta x \to 0$ *at least in* $L^r(\mathbb{R} \setminus (\bigcup_k D_k \cap \{t = 0\}))$ *where* $\mathbf{p}^0$ *is considered to be piecewise constant on the corresponding intervals of the discretization.*

Convergence of the discrete state y^n_j to y as $\Delta x \to 0$ follows from the theory for monotone discretization schemes for conservation laws which, as discussed above, is ensured by the nonnegativity of the coefficients $D^n_{j,l}$. The monotonicity also determines the constant M_y since, for such schemes, time iterates satisfy the maximum principle (see, e.g., [31, Theorem 13.36]), given as

$$\max_j\{|\mathbf{y}^n_j|\} \le \max_j\{|\mathbf{u}_j|\}, \text{ for all } n = 1,\dots,N.$$

Consequently, the control introduces bounds to the discrete approximation of the solution, $\mathbf{y}^n_j$ with $n = 1,\dots,N$ and $j \in \mathbb{N}$. By the assumed coercivity of $\mathfrak{R}(\cdot)$, the discrete initial condition has to be bounded in $L^\infty(\mathbb{R})$ as $\Delta x \to 0$, allowing to establish a bound on M_y. Condition 2 of Theorem 1 poses explicit conditions on the numerical flux to ensure $D^n_{j,-1} \ge 0$ and $D^n_{j,1} \ge 0$, while $D^n_{j,0} \ge 0$ typically can be ensured by a restriction on Δt and thus on λ; compare (13). In case of $K = 1$, the numerical flux functions (EO) and (LF) (compare Sect. 5) are known to satisfy all conditions of Theorem 1 under certain conditions on Δt (see [32, Chap. 6]).

After introducing the central convergence result for nonlinear programs obtained from (P) by applying suitable discretization techniques, we will now study the impact of higher-order Runge–Kutta (RK) time discretization methods applied to (10). If the solution to the system of ODEs is sufficiently smooth, these schemes are more accurate in that the truncation error depends on some power of the chosen time step Δt. Thus, the term higher order should not be mistaken with the overall truncation error of the discretization strategy for (1) that still heavily relies on the discretization method with respect to the spatial variable. The section closes by introducing RK schemes in the well-known form of Butcher arrays and the Shu–Osher representation.

Definition 4 Consider a uniform time step Δt. In vector notation, an s-stage RK scheme is given by

$$\mathbf{y}^{n,i} = \mathbf{y}^n + \Delta t \sum_{l=1}^{s} a_{il} F(\mathbf{y}^{n,l}), \quad i = 1, \ldots, s,$$
$$\mathbf{y}^{n+1} = \mathbf{y}^n + \Delta t \sum_{i=1}^{s} b_i F(\mathbf{y}^{n,i}).$$

The coefficients $A = (a_{ij})$ and weights $b = (b_j)$ with $1 \leq i, j \leq s$ represent the Butcher array and characterize the method.

Here, we only consider explicit RK methods, thus restricting the coefficients to $a_{ij} = 0$ for $j \geq i$. If the coefficients in A and b satisfy conditions formulated, e.g., in [20, Table 2], the truncation error of the approximation of the ODEs is of higher order.

In numerical methods for conservation laws, a different representation of RK methods is used frequently (see, e.g., [19, 24]).

Definition 5 The Shu–Osher representation of a s-stage RK method is a convex combination of forward Euler steps, parametrized by two sets of coefficients $\{\alpha_{ij}\}$ and $\{\beta_{ij}\}$ for $i = 1, \ldots, s$ and $j = 0, \ldots, s-1$, defined by

$$\mathbf{y}^{(i)} = \sum_{l=0}^{i-1} \alpha_{il} \mathbf{y}^{(l)} - \beta_{il} h F(\mathbf{y}^{(l)}),$$
$$\mathbf{y}^{(0)} = \mathbf{y}^n, \quad \mathbf{y}^{n+1} = \mathbf{y}^{(s)}$$

and satisfying $\alpha_{ij} \geq 0$, $\sum_{j=0}^{i-1} \alpha_{ij} = 1$.

This format allows for studying high-order RK discretizations with respect to time for a given spatial discretization that is TVD. The resulting, so-called, TVD-RK methods preserve the TVD property of the original discretization and have a higher order of accuracy with respect to time in terms of the truncation error. In fact, they are strong stability preserving (SSP) methods and ensure

$$\mathcal{C}(\mathbf{y}(t) + \Delta t F(\mathbf{y}(t))) \leq \mathcal{C}(\mathbf{y}(t)) \text{ for all } \Delta t \leq c_{SSP} \Delta t_E, \tag{20}$$

for arbitrary convex functionals $\mathcal{C} : \mathbb{R}^N \to \mathbb{R}$ including norms and the TV-seminorm, and time steps Δt up to a multiple of the time step Δt_E for the original discretization, employing an explicit Euler method. The scaling factor, $c_{SSP} := \min_{ij} \frac{\alpha_{ij}}{|\beta_{ij}|}$ is called SSP coefficient.

We will derive a close relationship between RK schemes that can be used in the context of Theorem 1 and strong stability preserving RK methods. As a consequence, known restrictions from SSP-RK methods apply in our case as well. In particular, it is known that explicit SPP methods are restricted to order $p \leq 4$ (see [30]). Moreover, we aim for time stepping methods with order matching the number of stages to avoid the computational effort of calculating and storing additional stages. There exist no combination of coefficients in the Butcher tableau of any RK method with the number of stages matching the order such all coefficients in the Shu–Osher representation are nonnegative (see [18]) in case of convergence order $p > 3$. We will see in the next section that we are restricted to RK methods that have a Shu–Osher representation with $\alpha_{ij} \geq 0$, $\beta_{ij} \geq 0$. Consequently, we are restricted to RK schemes of order at most 3.

Equation (11) forms a centered three-point scheme, and Theorem 1 holds with $K = 1$. However, in case of higher-order time stepping methods, this holds for every intermediate step. This increases the domain of determinacy for the full time step which is the number of values in a time slice contributing to values in the following time slice.

3 Consistency of the Runge–Kutta Time Stepping

We are now ready to study optimization RK schemes; i.e., we are interested in RK discretizations of (1) which yield consistent RK schemes for the adjoint (5) with high approximation order. Similar to SSP methods in the context of strong stability preserving methods, we assume that an explicit Euler time stepping for (10) with a properly chosen time step size Δt provides a full discretization of (1) such that the conditions of Theorem 1 are satisfied and identify RK schemes that preserve this properties, thus allowing for an application of the result. Consequently, the basic assumption for the following is given by

(A) The numerical flux f^Δ and Δt are chosen such that conditions 1. to 5. of Theorem 1 are met by (11).

Writing $f(v_1, v_2)$ instead of $f^\Delta(v_1, v_2)$ in a slight misuse of notation, we introduce $F_j^{n,s} : \mathbb{R}^3 \to \mathbb{R}$, its gradient and the vector valued functions $\tilde{F}_j^{n,s} : \mathbb{R}^4 \to \mathbb{R}^3$ and $\hat{F}_j^{n,s} : \mathbb{R}^4 \to \mathbb{R}^3$ utilized for validating the assumptions of Theorem 1 in case of a multistage time discretization of (10):

$$
\begin{aligned}
F_j^{n,s} &:= y_j^{n,s} - \lambda[f(y_j^{n,s}, y_{j+1}^{n,s}) - f(y_{j-1}^{n,s}, y_j^{n,s})],\\
\nabla F_j^{n,s} &:= [-\lambda f_2(y_j^{n,s}, y_{j+1}^{n,s}), 1 - \lambda(f_1(y_j^{n,s}, y_{j+1}^{n,s}) - f_2(y_{j-1}^{n,s}, y_j^{n,s})), \lambda f_1(y_{j-1}^{n,s}, y_j^{n,s})],\\
\tilde{F}_j^{n,s} &:= [\lambda f_1(y_j^{n,s}, y_{j+1}^{n,s}), 1 - \lambda(f_1(y_j^{n,s}, y_{j+1}^{n,s}) - f_2(y_{j-1}^{n,s}, y_j^{n,s})), -\lambda f_2(y_{j-1}^{n,s}, y_j^{n,s})],\\
\hat{F}_j^{n,s} &:= [\lambda f_1(y_{j+1}^{n,s}, y_{j+2}^{n,s}), 1 - \lambda(f_1(y_j^{n,s}, y_{j+1}^{n,s}) - f_2(y_j^{n,s}, y_{j+1}^{n,s})), -\lambda f_2(y_{j-1}^{n,s}, y_j^{n,s})].
\end{aligned}
$$

Here, $n = 0, \ldots, N-1, s$, and $j \in \mathbb{N}$ denote time step, stage of the RK method, and cell number, respectively. By this notation, the general RK scheme with three stages (RK3 scheme) can be rewritten as follows:

$$\begin{aligned} y_j^{n,1} &= y_j^n, \\ y_j^{n,2} &= c_{20} y_j^n + c_{21} F_j^{n,1}, \\ y_j^{n,3} &= c_{30} y_j^n + c_{31} F_j^{n,1} + c_{32} F_j^{n,2}, \\ y_j^{n+1} &= c_{40} y_j^n + c_{41} F_j^{n,1} + c_{42} F_j^{n,2} + c_{43} F_j^{n,3}. \end{aligned} \tag{21}$$

This forms a particular Shu–Osher representation (see Definition 5) of the general Runge–Kutta method in that we assume $\alpha_{ij} = \beta_{ij}$, fixing $c_{SSP} = 1$. In particular, we find the structure of the original Euler time step preserved in that $F_j^{n,i}$, $i = 1, \ldots, 3$ coincides with (11) evaluated at the intermediate steps of the RK scheme and $\nabla F_j^{n,i}$ and $\hat{F}_j^{n,i}$ represent the coefficients from (13) and (17), evaluated at the intermediate steps i of the RK method for $i = 1, \ldots, 3$. The coefficients in (21) are given next.

Definition 6 Given a time discretization of (10) by an RK3 method, the coefficients in (21) are defined by

$$\begin{aligned} c_{21} &= a_{21}, & c_{43} &= b_3, \\ c_{20} &= 1 - a_{21}, & c_{42} &= b_2 - b_3 a_{32}, \\ c_{32} &= a_{32}, & c_{41} &= b_1 - b_3 a_{31} - (b_2 - b_3 a_{32}) a_{21}, \\ c_{31} &= a_{31} - a_{32} a_{21}, & c_{40} &= 1 - \textstyle\sum_{j=1}^{3} c_{4,j}. \\ c_{30} &= 1 - \textstyle\sum_{j=1}^{2} c_{3,j}, \end{aligned}$$

General RK schemes with two stages (RK2 scheme) can be interpreted as RK3 schemes with $a_{3,1} = a_{3,2} = b_3 = 0$ in the corresponding Butcher array. In this case, representation (21) follows by substituting these values into the coefficients of Definition 6. The differentiability of the numerical flux (according to Condition 1 of Theorem 1) allows to obtain the following sensitivity scheme, where $\bar{\mu}_j^{n,s} := [\mu_{j+1}^{n,s}, \mu_j^{n,s}, \mu_{j-1}^{n,s}]$:

$$\begin{aligned} \mu_j^{n,1} &= \mu_j^n, \\ \mu_j^{n,2} &= c_{20} \mu_j^n + c_{21} \nabla F_j^{n,1} \cdot \bar{\mu}_j^{n,1}, \\ \mu_j^{n,3} &= c_{30} \mu_j^n + c_{31} \nabla F_j^{n,1} \cdot \bar{\mu}_j^{n,1} + c_{32} \nabla F_j^{n,2} \cdot \bar{\mu}_j^{n,2}, \\ \mu_j^{n+1} &= c_{40} \mu_j^n + c_{41} \nabla F_j^{n,1} \cdot \bar{\mu}_j^{n,1} + c_{42} \nabla F_j^{n,2} \cdot \bar{\mu}_j^{n,2} + c_{43} \nabla F_j^{n,3} \cdot \bar{\mu}_j^{n,3}. \end{aligned} \tag{22}$$

Assuming $b_j \neq 0$ for $j = 1, \ldots, s$ and given an RK method with coefficients from the corresponding Butcher tableau (see Definition 4), the adjoint scheme will be derived by substituting the intermediate stages into the equation defining μ_j^{n+1} and employing the discrete duality condition (14), followed by collecting suitable terms in the result. To

keep notation short, we restrict ourselves to an RK2 scheme. For more stages, the adjoint scheme is obtained analogously. Any time step is, with $y_j^{n,1} = y_j^n$ and $\mu_j^{n,1} = \mu_j^n$, given as follows:

$$\begin{aligned}\mu_j^{n,2} &= \mu_j^n - a_{21}\lambda(f_2(y_j^n, y_{j+1}^n)\mu_{j+1}^n + (f_1(y_j^n, y_{j+1}^n)\\ &\quad - f_2(y_{j-1}^n, y_j^n))\mu_j^n - f_1(y_{j-1}^n, y_j^n)\mu_{j-1}^n), \qquad (23)\\ \mu_j^{n+1} &= \mu_j^n - b_1\lambda(f_2(y_j^n, y_{j+1}^n)\mu_{j+1}^n + (f_1(y_j^n, y_{j+1}^n)\\ &\quad - f_2(y_{j-1}^n, y_j^n))\mu_j^n - f_1(y_{j-1}^n, y_j^n)\mu_{j-1}^n)\\ &\quad - b_2\lambda(f_2(y_j^{n,2}, y_{j+1}^{n,2})\mu_{j+1}^{n,2} + (f_1(y_j^{n,2}, y_{j+1}^{n,2})\\ &\quad - f_2(y_{j-1}^{n,2}, y_j^{n,2}))\mu_j^{n,2} - f_1(y_{j-1}^{n,2}, y_j^{n,2})\mu_{j-1}^{n,2}). \qquad (24)\end{aligned}$$

Multiplying (24) by p_j^{n+1}, summing over j, and utilizing the bounded support of $\{\mu_i^n\}$, we can reorder the sum and obtain

$$\begin{aligned}&\sum_{j\in\mathbb{N}} \mu_j^{n+1} p_j^{n+1}\\ &= \sum_{j\in\mathbb{N}} \mu_j^n(-b_1\lambda f_2(y_{j-1}^n, y_j^n)p_{j-1}^{n+1} + (1 - b_1\lambda(f_1(y_j^n, y_{j+1}^n) - f_2(y_{j-1}^n, y_j^n)))p_j^{n+1}\\ &\qquad + b_1\lambda f_1(y_j^n, y_{j+1}^n)p_{j+1}^{n+1})\\ &\quad + b_2 \sum_{j\in\mathbb{N}} \mu_j^{n,2}(-\lambda f_2(y_{j-1}^{n,2}, y_j^{n,2})p_{j-1}^{n+1} - \lambda(f_1(y_j^{n,2}, y_{j+1}^{n,2}) - f_2(y_{j-1}^{n,2}, y_j^{n,2}))p_j^{n+1}\\ &\qquad + \lambda f_1(y_j^{n,2}, y_{j+1}^{n,2})p_{j+1}^{n+1}).\end{aligned}$$

Substituting (23) in this expression and further reordering of the sum provides, just as in the case of a single Euler step, an expression for $\sum_{j\in\mathbb{N}} \mu_j^{n+1} p_j^{n+1}$ that depends on μ_i^n and p_i^{n+1} only, providing a formula for p_j^n. Collecting suitable terms, we get

$$\begin{aligned}p_j^{n+1,1} &= p_j^{n+1},\\ p_j^{n+1,2} &= p_j^{n+1} - \lambda a_{12}^{\dagger}(\lambda f_2(y_{j-1}^{n,2}, y_j^{n,2})p_{j-1}^{n+1}\\ &\quad + \lambda(f_1(y_j^{n,2}, y_{j+1}^{n,2}) - f_2(y_{j-1}^{n,2}, y_j^{n,2}))p_j^{n+1} - \lambda f_1(y_j^{n,2}, y_{j+1}^{n,2})p_{j+1}^{n+1}),\\ p_j^n &= p_j^{n+1} - \lambda b_2(f_2(y_{j-1}^{n,2}, y_j^{n,2})p_{j-1}^{n+1,1}\\ &\quad + (f_1(y_j^{n,2}, y_{j+1}^{n,2}) - f_2(y_{j-1}^{n,2}, y_j^{n,2}))p_j^{n+1,1} - f_1(y_j^{n,2}, y_{j+1}^{n,2})p_{j+1}^{n+1,1})\\ &\quad - \lambda b_1(f_2(y_{j-1}^{n,1}, y_j^{n,1})p_{j-1}^{n+1,2}\\ &\quad + (f_1(y_j^{n,1}, y_{j+1}^{n,1}) - f_2(y_{j-1}^{n,1}, y_j^{n,1}))p_j^{n+1,2} - f_1(y_j^{n,1}, y_{j+1}^{n,1})p_{j+1}^{n+1,2}).\end{aligned}$$

Applying this technique to a general RK3 scheme provides the following update rule for the time step $n + 1 \to n$ of the adjoint equation:

$$
\begin{aligned}
p_j^{n+1,1} &= p_j^{n+1},\\
p_j^{n+1,2} &= p_j^{n+1} - \lambda a_{12}^{\dagger}(f_2(y_{j-1}^{n,3}, y_j^{n,3})p_{j-1}^{n+1,1}\\
&\quad + (f_1(y_j^{n,3}, y_{j+1}^{n,3}) - f_2(y_{j-1}^{n,3}, y_j^{n,3}))p_j^{n+1,1} - f_1(y_j^{n,3}, y_{j+1}^{n,3})p_{j+1}^{n+1,1}),\\
p_j^{n+1,3} &= p_j^{n+1} - \lambda a_{13}^{\dagger}(f_2(y_{j-1}^{n,3}, y_j^{n,3})p_{j-1}^{n+1,1}\\
&\quad + (f_1(y_j^{n,3}, y_{j+1}^{n,3}) - f_2(y_{j-1}^{n,3}, y_j^{n,3}))p_j^{n+1,1} - f_1(y_j^{n,3}, y_{j+1}^{n,3})p_{j+1}^{n+1,1})\\
&\quad - \lambda a_{23}^{\dagger}(f_2(y_{j-1}^{n,2}, y_j^{n,2})p_{j-1}^{n+1,2}\\
&\quad + (f_1(y_j^{n,2}, y_{j+1}^{n,2}) - f_2(y_{j-1}^{n,2}, y_j^{n,2}))p_j^{n+1,2} - f_1(y_j^{n,2}, y_{j+1}^{n,2})p_{j+1}^{n+1,2}),\\
p_j^{n} &= p_j^{n+1} - \lambda b_3(f_2(y_{j-1}^{n,3}, y_j^{n,3})p_{j-1}^{n+1,1}\\
&\quad + (f_1(y_j^{n,3}, y_{j+1}^{n,3}) - f_2(y_{j-1}^{n,3}, y_j^{n,3}))p_j^{n+1,1} - f_1(y_j^{n,3}, y_{j+1}^{n,3})p_{j+1}^{n+1,1})\\
&\quad - \lambda b_2(f_2(y_{j-1}^{n,2}, y_j^{n,2})p_{j-1}^{n+1,2}\\
&\quad + (f_1(y_j^{n,2}, y_{j+1}^{n,2}) - f_2(y_{j-1}^{n,2}, y_j^{n,2}))p_j^{n+1,2} - f_1(y_j^{n,2}, y_{j+1}^{n,2})p_{j+1}^{n+1,2})\\
&\quad - \lambda b_1(f_2(y_{j-1}^{n,1}, y_j^{n,1})p_{j-1}^{n+1,3}\\
&\quad + (f_1(y_j^{n,1}, y_{j+1}^{n,1}) - f_2(y_{j-1}^{n,1}, y_j^{n,1}))p_j^{n+1,3} - f_1(y_j^{n,1}, y_{j+1}^{n,1})p_{j+1}^{n+1,3}).
\end{aligned}
\tag{25}
$$

Formally, this corresponds to an implicit multistage method with the following tableau of coefficients:

$$
\frac{A_{RK2}^{\dagger}}{b_{RK2}^{\dagger}} = \begin{array}{cc} 0 & \frac{b_2 a_{21}}{b_1} \\ 0 & 0 \\ \hline b_2 & b_1 \end{array}, \qquad \frac{A_{RK3}^{\dagger}}{b_{RK3}^{\dagger}} = \begin{array}{ccc} 0 & \frac{b_3 a_{32}}{b_2} & \frac{b_3 a_{31}}{b_1} \\ 0 & 0 & \frac{b_2 a_{21}}{b_1} \\ 0 & 0 & 0 \\ \hline b_3 & b_2 & b_1 \end{array}
$$

We observe that the adjoint RK schemes correspond to the methods established in [20] in case of optimal control of ordinary differential equations. Similar to (21), the adjoint scheme can be rewritten in a form that preserves the structure of the single-step adjoint scheme with coefficients $\tilde{c}_{l,k}$.

Proposition 4 *Let (A) be satisfied. For $s \in \{2, 3\}$, consider an s-stage RK scheme for the time discretization of (10). Moreover, let $b_j \neq 0$ hold for $j = 1, \ldots, s$ and the coefficients from the Butcher tableau of the time stepping scheme satisfy the conditions of Table 1. Then, we have $D_{j,l}^n \geq 0$ for $1 \leq n \leq N$, $j \in \mathbb{N}$, $-s \leq l \leq s$ and $s \in \{2, 3\}$.*

Proof The proof establishes conditions in case of RK3 schemes only, but RK2 schemes are treated analogously. Any intermediate stage $i = 1, 2, 3$ and the full time step, $\mu_j^{n+1}, \mu_j^{n,3}, \mu_j^{n,2}, \mu_j^{n,1} = \mu_j^n$, will be expressed in the form

$$
\mu_j^{n,i} = \sum_{l=-1}^{1} \dot{D}_{j,l}^{n,i} \mu_{j+l}^{n,1} + \tilde{D}_{j,l}^{n,i} \mu_{j+l}^{n,2} + \hat{D}_{j,l}^{n,i} \mu_{j+l}^{n,3} \tag{26}
$$

Table 1 Order conditions for RK scheme

Stages	Conditions
2	$a_{21} \geq 0, \quad 1 - a_{21} \geq 0, \quad b_2 \geq 0, \quad b_1 - b_2 a_{21} \geq 0$
3	$a_{32} \geq 0, \quad a_{31} - a_{32}a_{21} \geq 0, \quad 1 - a_{31} - a_{32}(1 - a_{21}) \geq 0, \quad b_3 \geq 0, \quad b_2 - b_3 a_{32} \geq 0, b_3 a_{32} + b_3 a_{31} + b_2 a_{21} - b_3 a_{32} a_{21} \geq 0, \quad b_1 - b_2 a_{21} - b_3(a_{31} - a_{32}a_{21}) \geq 0$

for coefficients $\dot{D}^{n,i}_{j,l}, \tilde{D}^{n,i}_{j,l}, \hat{D}^{n,i}_{j,l} \in \mathbb{R}$ to be determined. For unifying the representation, we utilize the superscript $(n, 4)$ for μ^{n+1}_j and since we consider explicit schemes, it holds that

$$\dot{D}^{n,k}_{j,l} \equiv \tilde{D}^{n,k}_{j,l} \equiv \hat{D}^{n,k}_{j,l} \equiv 0$$

for $k \geq i$.

The proof consists of two parts. Recall that Assumption (A) ensures positivity of the coefficients $D^n_{j,l}$ in (13) only if $\mathbf{y}^n_j \in [-M_y, M_y]$ holds for all $j \in \mathbb{N}$. Thus, these coefficients, evaluated at intermediate steps of the RK method, are not necessarily nonnegative. Consequently, we first assume $D^n_{j,l} \geq 0$ for $y^n_j \in \mathbb{R}$ for all $j \in \mathbb{N}$ and derive conditions on the coefficients in the Butcher array of the RK scheme ensuring positivity of the corresponding coefficients in case of a multistage method. In a second step, we discuss the restriction $y^n_j \in [-M_y, M_y]$ required in Theorem 1.

As outlined above, (22) and the intermediate steps preserve the original single-step structure which, by Assumption (A), ensures positivity of each component in $\nabla F^{n,i}_j$, $i = 1, \ldots, s$. By the assumption on the numerical flux and Δt and considering (22) rewritten in the form of (26), we obtain

$$\dot{D}^{n,4}_{j,l} = c_{40}[0, \ 1, \ 0] + c_{41}\nabla F^{n,1}_j, \quad \tilde{D}^{n,4}_{j,l} = c_{42}\nabla F^{n,2}_j, \quad \hat{D}^{n,4}_{j,l} = c_{43}\nabla F^{n,3}_j.$$

Nonnegativity of these coefficients follows for $c_{4,k} \geq 0$, $k = 0, \ldots, 3$. Similarly, $\dot{D}^{n,3}_{j,l} \geq 0$, $\tilde{D}^{n,3}_{j,l} \geq 0$ if $c_{3,k} \geq 0$, $k = 0, \ldots, 2$ and $\dot{D}^{n,2}_{j,l} \geq 0$ if $c_{20} \geq 0$, $c_{21} \geq 0$. As a consequence, the coefficients of the reduced time step satisfy $D^n_{j,l} \geq 0$ for $1 \leq n \leq N$, $j \in \mathbb{N}$, $-3 \leq l \leq 3$ by consisting of sums and products of nonnegative numbers only.

For the second part of the proof, we observe that the conditions derived in Table 1 exactly match the conditions that characterize strong stability preserving Runge–Kutta methods with full Eulerian time step (see, e.g., [24, Theorem 2]). The latter RK methods ensure the maximum principle, not only for the full time step but also for each intermediate stage as well. Consequently, given $y^n_j \in [-M_y, M_y]$ for all $j \in \mathbb{N}$, we have $y^{n+1}_j \in [-M_y, M_y]$ and in particular $y^{n,i}_j \in [-M_y, M_y]$ for $i = 1, \ldots, s$. Thus, the arguments from Step 1 remain true.

Now, we analyze the coefficients $C^n_{j,l}$ in the representation of $p^n_{j+1} - p^n_j$ and find the following result.

Proposition 5 *Let the assumptions of Proposition 4 be satisfied. Then, the coefficients $C^n_{j,l}$ for $1 \le n \le N$, $j \in \mathbb{N}$ and $-3 \le l \le 3$ are nonnegative if the conditions of Table 1 are satisfied.*

Proof Similar to the discretization scheme for the conservation law and its sensitivity equation, for $\bar{p}^{n+1,s}_j = [p^{n,s}_{j-1}, p^{n+1,s}_j, p^{n+1,s}_{j+1}]$ we find the following representation of the adjoint discretization scheme (25):

$$\begin{aligned}
p^{n+1,1}_j &= p^{n+1}_j,\\
p^{n+1,2}_j &= \tilde{c}_{20} p^{n+1}_j + \tilde{c}_{21} \tilde{F}^{n,3}_j \cdot \bar{p}^{n+1,1}_j,\\
p^{n+1,3}_j &= \tilde{c}_{30} p^{n+1}_j + \tilde{c}_{31} \tilde{F}^{n,3}_j \cdot \bar{p}^{n+1,1}_j + \tilde{c}_{32} \tilde{F}^{n,2}_j \cdot \bar{p}^{n+1,2}_j,\\
p^{n}_j &= \tilde{c}_{40} p^{n+1}_j + \tilde{c}_{41} \tilde{F}^{n,3}_j \cdot \bar{p}^{n+1,1}_j + \tilde{c}_{42} \tilde{F}^{n,2}_j \cdot \bar{p}^{n+1,2}_j + \tilde{c}_{43} \tilde{F}^{n,1}_j \cdot \bar{p}^{n+1,3}_j.
\end{aligned}$$

Abbreviating $\Delta_+ \bar{p}^{n+1,s}_j = [p^{n+1,s}_{j+2} - p^{n+1,s}_{j+1}, p^{n+1,s}_{j+1} - p^{n+1,s}_j, p^{n+1,s}_j - p^{n+1,s}_{j-1}]$, we obtain

$$\begin{aligned}
p^{n+1,1}_{j+1} - p^{n+1,1}_j &= p^{n+1}_{j+1} - p^{n+1}_j,\\
p^{n+1,2}_{j+1} - p^{n+1,2}_j &= \tilde{c}_{20}(p^{n+1}_{j+1} - p^{n+1}_{j+1}) + \tilde{c}_{21} \hat{F}^{n,3}_j \cdot \Delta_+ \bar{p}^{n+1,1}_j,\\
p^{n+1,3}_{j+1} - p^{n+1,3}_j &= \tilde{c}_{30}(p^{n+1}_{j+1} - p^{n+1}_{j+1}) + \tilde{c}_{31} \hat{F}^{n,3}_j \cdot \Delta_+ \bar{p}^{n+1,1}_j + \tilde{c}_{32} \hat{F}^{n,2}_j \cdot \Delta_+ \bar{p}^{n+1,2}_j,\\
p^{n}_{j+1} - p^{n}_j &= \tilde{c}_{40}(p^{n+1}_{j+1} - p^{n+1}_{j+1}) + \tilde{c}_{41} \hat{F}^{n,3}_j \cdot \Delta_+ \bar{p}^{n+1,1}_j + \tilde{c}_{42} \hat{F}^{n,2}_j \cdot \Delta_+ \bar{p}^{n+1,2}_j\\
&\qquad + \tilde{c}_{43} \hat{F}^{n,1}_j \cdot \Delta_+ \bar{p}^{n+1,3}_j.
\end{aligned} \tag{27}$$

As in Proposition 4 and by the assumptions on the numerical flux and Δt, we find the coefficients $C^n_{j,l}$ for $1 \le n \le N$, $j \in \mathbb{N}$ and $-3 \le l \le 3$ to be nonnegative if $\tilde{c}_{j,k} \ge 0$ holds for all $j = 2, \ldots, 4$ and $k = 0, \ldots, 3$.
We first consider the RK2 scheme with $\tilde{c}_{20} = 1 - a^\dagger_{23} = 1 - b_1^{-1} b_2 a_{23}$, $\tilde{c}_{21} = a^\dagger_{23} = b_1^{-1} b_2 a_{21}$ and $b_2 - b_1 a^\dagger_{23} = b_2(1 - a_{21})$. Here, nonnegativity is ensured by the conditions presented in Table 1.
In case of an RK3 scheme, we have $\tilde{c}_{32} = a^\dagger_{23} = b_1^{-2} b_2 a_{21}$, $\tilde{c}_{31} = a^\dagger_{13} - a^\dagger_{23} a^\dagger_{12} = b_1^{-1} b_3 (a_{31} - a_{32} a_{21})$ and $\tilde{c}_{30} = 1 - a^\dagger_{23} - a^\dagger_{13} + a^\dagger_{23} a^\dagger_{12} \ge 0$ which holds if $b_1 - b_2 a_{21} - b_3 a_{31} + b_3 a_{32} a_{21} \ge 0$ is satisfied. Moreover, $\tilde{c}_{40} = b_1 a^\dagger_{23} + b_1 a^\dagger_{13} + b_2 a^\dagger_{12} - b_1 a^\dagger_{12} a^\dagger_{23} \ge 0$ and $\tilde{c}_{41} = b_3 - b_1 a^\dagger_{13} - b_2 a^\dagger_{23} + b_1 a^\dagger_{12} a^\dagger_{23} \ge 0$ if $b_2 a_{21} + b_3 a_{31} + b_3 a_{32} - b_3 a_{32} a_{21} \ge 0$ and $b_3(1 - a_{31} - a_{32} + a_{32} a_{21}) \ge 0$. Finally, we have $\tilde{c}_{42} = b_2 - b_1 a^\dagger_{13} = b_2 - b_3 a_{31}$ and $\tilde{c}_{43} = b_1$. Thus, the conditions of Table 1 imply nonnegativity of the coefficients $\tilde{c}_{j,k}$ and, by the representation (27), of $C^n_{j,l}$ for $n = 1, \ldots, N$, $j \in \mathbb{N}$ and $k = -s, \ldots, s$.

The next result provides conditions ensuring that Assumption 4 of Theorem 1 is satisfied.

Table 2 Conditions for OSL consistency

Stages	Conditions
2	$a_{21} \neq 0, \quad b_2 \neq 0$
3	$a_{21} \neq 0, \quad a_{32} \neq 0, \quad b_3 \neq 0$

Proposition 6 *Let the assumptions of Proposition 4 be satisfied, and let the RK scheme satisfy the conditions of Table 2. Then, there exists a constant $\nu_s > 0$ such that*

$$l_j^{n+1,0} \leq l_{j,s}^{n,0,+} - \Delta t \nu_s (l_{j,s}^{n,0,+})^2$$

is satisfied [cf. (18)]. Here, the constant ν_s depends on the number of stages s.

Proof We start proving the claim for $s = 2$. To establish the existence of $\nu_2 > 0$, we recall (21):

$$\begin{aligned} y_j^{n,2} =& (1 - a_{21}) y_j + a_{21}(y_j - \lambda(f(y_j, y_{j+1}) - f(y_{j-1}, y_j)), \qquad (28) \\ y_j^{n+1} =& b_2 a_{21} y_j + (b_1 - b_2 a_{21})(y_j - \lambda(f(y_j, y_{j+1}) - f(y_{j-1}, y_j)) \\ & + b_2(y_j^{n,2} - \lambda(f(y_j^{n,2}, y_{j+1}^{n,2}) - f(y_{j-1}^{n,2}, y_j^{n,2}))), \qquad (29) \end{aligned}$$

with nonnegative weights by assumption. Since the discretization scheme, utilizing a single Euler step, satisfies (19), substituting (28) into (18) provides

$$l_j^{n,2} \leq (1 - a_{21}) l_{j,0}^{n,0} + a_{21}(l_{j,1}^{n,0,+} - \Delta t \nu (l_{j,1}^{n,0,+})^2) \leq l_{j,1}^{n,0,+} - a_{21} \Delta t \nu (l_{j,1}^{n,0,+})^2 \qquad (30)$$

where we used $0 < a_{21} \leq 1$ and for the ν form the explicit Euler discretization. Similarly, (29) can be estimated by

$$l_j^{n+1,0} \leq b_2 a_{21} l_{j,0}^{n,0} + (b_1 - b_2 a_{21})(l_{j,1}^{n,0,+} - \nu \Delta t (l_{j,1}^{n,0,+})^2) + b_2(l_{j,1}^{n,1,+} - \nu \Delta t (l_{j,1}^{n,1,+})^2) \qquad (31)$$

with

$$\begin{aligned} l_{j,1}^{n,2,+} &= \max(0, l_{j-1}^{n,2}, l_j^{n,2}, l_{j+1}^{n,2}) \\ &\leq \max(0, l_{j+1,1}^{n,0,+} - \Delta t \nu a_{21} (l_{j+1,1}^{n,0,+})^2, l_{j,1}^{n,0,+} \\ &\quad - \Delta t \nu a_{21} (l_{j,1}^{n,0,+})^2, l_{j-1,1}^{n,0,+} - \Delta t \nu a_{21} (l_{j-1,1}^{n,0,+})^2) \end{aligned}$$

where we utilized (30). For $\hat{\nu} = (4\lambda M_y a_{21} \nu \max\{1, \Delta t\})^{-1}$, we find the function $x - \Delta t \lambda \nu \hat{\nu} a_{21} x^2$ to be nonnegative and monotonically increasing for $x \in [0, 2M_y]$. Consequently, we have

$$l_{j,1}^{n,2,+} = \max(l_{j-1}^{n,2}, l_j^{n,2}, l_{j+1}^{n,2}) \le \max(l_{j+1,1}^{n,0,+}, l_{j,1}^{n,0,+}, l_{j-1,1}^{n,0,+}) - \Delta t\,\hat{\nu}(\max(l_{j+1,1}^{n,0,+}, l_{j,1}^{n,0,+}, l_{j-1,1}^{n,0,+}))^2.$$

For (31), by using $\max(l_{j+1,1}^{n,0,+}, l_{j,1}^{n,0,+}, l_{j-1,1}^{n,0,+}) = l_{j,2}^{n,0,+}$ and dropping the negative terms, the following upper bound can be established:

$$l_{j,2}^{n,0,+} - \Delta t\,\hat{\nu} b_2 (l_{j,2}^{n,0,+})^2.$$

Thus, for $\nu_2 = \hat{\nu} b_2$, the RK2 method satisfies the OSL consistency conditions (19) where positivity is ensured by the conditions of Table 2. In case of $s = 3$, we first note that

$$l_{j,1}^{n,3} \le l_{j,2}^{n,0,+} - \Delta t\,\hat{\nu} a_{32} (l_{j,2}^{n,0,+})^2$$

and $a_{32} > 0$ hold true. Now, the same arguments as in the case $s = 2$ provide the existence of $0 < \nu_3 = b_3(4\lambda M_y a_{32}\hat{\nu}\max\{1, \Delta t\})^{-1}$ with

$$l_j^{n+1,0} \le l_{j,3}^{n,0,+} - \Delta t\,\nu_3 (l_{j,3}^{n,0,+})^2,$$

which concludes the proof.

Finally, we discuss Assumption 5 of Theorem 1.

Proposition 7 *Let the assumptions of Proposition 4 be satisfied. Let an RK scheme of order s with $s = 2, 3$ satisfy the conditions of Table 1. Then, the numerical flux*

$$f^{\Delta}(y_{j-s+1}^n, \ldots, y_{j+s}^n) = \sum_{i=1}^{s} b_i f^{\Delta}(y_j^{n,i}, y_{j+1}^{n,i})$$

satisfies assumptions 1 and 5 of Theorem 1, respectively.

Proof By assumption, we have $f^{\Delta}(y, y) = f(y)$ for the original numerical flux. Consequently, one easily finds $y_j^{n,i} = y_j^n = y$ for $i = 1, \ldots, s$ and $y_l = y$ for $l = j - s + 1, \ldots, j + s$. A necessary condition for higher-order time stepping schemes is $\sum b_i = 1$ and consequently $f^{\Delta}(y_{j-s+1}^n, \ldots, y_{j+s}^n) = f(y)$.

The second assertion follows from the corresponding properties of the original numerical flux, representation (21), and the conditions from Table 1 as an application of the chain rule.

Before summarizing the previous results, we revise the assumption $b_j \neq 0$ for $j = 1, \ldots, s$, originally imposed for establishing the discretization scheme of the adjoint. The conditions presented in Table 2 not only enable the results of Proposition 7 but also are necessary for the higher convergence order of the discretization of the state system. These order conditions for RK schemes are summarized, e.g., in [20, Table 2] and given by $b_2 a_{21} = 1/2$ in case of an RK2 scheme and $b_2 a_{21} + b_3(a_{31} + a_{32}) = 1/2$, $b_3 a_{32} a_{21} = 1/6$, and $b_2 a_{21}^2 + b_3(a_{31} + a_{32})^2 = 1/3$ for RK3 methods. Consequently,

RK2 and RK3 schemes of order two and three automatically satisfy the conditions of Table 2 and further have strictly positive coefficients b_2 and b_3. Strict positivity of b_1 now follows from the conditions in Table 1.

Theorem 2 *Let a semidiscretization of the conservation law* (1) *be given such that a time discretization by the explicit Euler method with a suitable time step Δt provides a full discretization* (11) *satisfying (A). Moreover, suppose that an RK scheme for the time discretization of* (10) *is of order $s \leq 3$. Let the RK scheme satisfy the conditions given in Table 1. Then, the resulting full discretization of* (1) *provides a consistent discretization of (P) in the sense of Theorem 1.*

We point out that to consider initial data with upward jumps, in [32] the weakened OSL condition, given as

$$\partial_x f'(y(t,\cdot)) \leq \alpha(t) \text{ for } \alpha \in L^1(\sigma, T) \text{ and all } \sigma \in (0, T),$$

was analyzed as well. In this case, slightly weaker convergence results than provided in Theorem 1 hold. However, under the same conditions as in Theorem 2 for the case of OSL, the application of Runge–Kutta time stepping methods extends to this setting.

The set of RK methods that satisfy the conditions of Tables 1 and 2 is discussed next. In [25], Runge–Kutta methods in terms of the Butcher arrays have been studied in the context of strong stability preserving time stepping. For $s = 2$, Heun's method

$$\begin{array}{cc} 0 & 0 \\ 1 & 0 \\ \hline 1/2 & 1/2 \end{array}, \tag{32}$$

and in case of $s = 3$ the method defined by the array

$$\begin{array}{ccc} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 1/4 & 1/4 & 0 \\ \hline 1/6 & 1/6 & 2/3 \end{array} \tag{33}$$

are the only time stepping schemes satisfying the conditions of Tables 1 and 2. They provide a method that has $c_{SSP} = 1$ in (20), and the associated number of stages matches the order of convergence.

4 Problems of Optimal Control and Order of Convergence for the Adjoint Scheme

The problem of optimal control subject to a system of ordinary differential equations resulting from a semidiscretization of (1) is given by

$$\begin{aligned} &\text{minimize} \quad \tfrac{\Delta x}{2}|\mathbf{y}(T) - \mathbf{y}^{\mathbf{d}}|_{l^2}^2 + \mathfrak{R}(\mathbf{u}) = \mathcal{J}^{\Delta}(\mathbf{y}(T), \mathbf{u}), \\ &\text{over} \quad (\mathbf{y}(T), \mathbf{u}) \in \mathbb{R}^N \times \mathbb{R}^{\tilde{N}}, \\ &\text{subject to} \quad \mathbf{y} \text{ satisfies (1.10)}, \end{aligned} \tag{P_Δ}$$

and can be interpreted as a Mayer problem in the context of optimal control of ODEs where N, $\tilde{N}$ depend on the spatial discretization. First-order necessary optimality conditions for (P_Δ), as given in [13, Theorem 4.2.i], involve the adjoint ODE system

$$\begin{aligned} \dot{\mathbf{p}} &= -[F(\mathbf{y})\mathbf{p}]_{\mathbf{y}} \\ &= \Delta x^{-1}[f_2(y_{j-1}(t), y_j(t))(p_{j+1}(t) - p_j(t)) - f_1(y_j(t), y_{j+1}(t))(p_j(t) - p_{j-1}(t))] \end{aligned}$$

with terminal condition $\mathbf{p}(\mathbf{T}) = \mathcal{J}^{\Delta}_{\mathbf{y}(T)}(\mathbf{y}(T), \mathbf{u})$ as discussed in [20]. Comparing the adjoint system obtained from the optimal control of ordinary differential equations with (16), obtained via sensitivity scheme and discrete duality relation, we find them to be equal.

This equivalence allows for utilizing the order conditions on RK time stepping methods in the context of optimal control of ODEs from [20] also in the context of optimal control for scalar conservation laws. Consulting the conditions formulated in [20, Table 1], Heun's scheme has convergence order 2 with respect to time for both state and adjoint systems. In case of the third-order method, the additional condition $(b_2a_{21} + b_3a_{31})^2/b_1 + (b_3a_{32})^2/b_2 = 1/3$ is not satisfied by the coefficients in (33), and consequently, the approximation of the adjoint is merely of order 2. This observation coincides with the results of [21], where certain numerical flux functions were investigated and conditions on the coefficients of the associated Shu–Osher representation that guarantee total variation stability of the adjoint discretization were derived. In addition, an upper bound for the order of convergence was established, namely imposing the SSP property on the time discretization of the state system, and stability on the discretization of the adjoint system limits the order of convergence for the latter scheme to 2. As outlined before, the consideration of (32) and (33) is sufficient since they represent the only methods with $c_{SSP} = 1$ in (20) and order matching the number of stages.

Let $\mathcal{S}^s : \mathbb{R}^{\tilde{N}} \to \mathbb{R}^N$ denote the solution operator of the underlying full discretization of (1) with s stages. Then, the fully discrete problem is given as

$$\begin{aligned} &\text{minimize } \mathcal{J}^{\Delta}(\mathbf{y}(T), \mathbf{u}), \\ &\quad \text{over } (\mathbf{y}(T), \mathbf{u}) \in \mathbb{R}^N \times \mathbb{R}^{\tilde{N}}, \\ &\text{subject to } \mathbf{y}(T) = \mathcal{S}^s(\mathbf{u}), \end{aligned} \tag{34}$$

where the discrete desired state is obtained by averaging the given function as discussed, in case of the initial datum, in (9).

By design, $\mathcal{S}^s$ is Lipschitz continuous with a constant depending on the Lipschitz constant of the chosen numerical flux, the number of time steps s, and the coefficients of the RK scheme even if these coefficients do not meet the conditions of Table 1. Consequently, following the direct method from the calculus of variations, there exists a solution of (34) independently of the chosen mesh width Δx.

5 Numerical Experiments

We end this paper with a report on numerical experiments associated with (P). In fact, utilizing a Tikhonov-type cost for the control (R) and employing a standard gradient descent scheme with Armijo line search, a numerical study when using Burgers' equation as the underlying state system is conducted.

5.1 Regularization Term

A possible choice for the regularization term is given by $\mathfrak{R}(u) = \|u\|_{BV(\mathbb{R})}$ that ensures boundedness of u in $L^1(\mathbb{R})$ and coercivity with respect to $L^\infty(\mathbb{R})$ due to the continuous embedding $BV(\mathbb{R}) \hookrightarrow L^\infty(\mathbb{R})$. To avoid additional problems due to the nondifferentiability of this choice, our numerical experiments utilize an $H^1(\mathbb{R})$-type cost. However, the assumptions on the cost term ensure a bounded support of the optimal control u^* and the associated solution $y^* = y(u^*)$ of (1), provided the desired state has bounded support as well. Since $u \equiv 0$ is a feasible control with corresponding solution $y^0 := y(0)$ of (1), we obtain

$$\|u^*\|_{L^\infty(\mathbb{R})} \leq \mathfrak{R}(u^*) \leq \mathcal{J}(y^*, u^*) \leq \mathcal{J}(y(0), 0) = \frac{1}{2}\|y^0(T) - y^d\|^2_{L^2(\mathbb{R})} =: M_u.$$

By the maximum principle (see Proposition 1), we further have $\|y^*(t, \cdot)\|_{L^\infty(\mathbb{R})} \leq M_u$ for all $t \in [0, T]$, bounding the overall characteristic speed by

$$\max_{\eta \in [-M_u, M_u]} |f'(\eta)| = M_f.$$

The maximum exists by assumptions on the flux function f and the extreme value theorem. Consequently, if (a, b) with $-\infty < a < b < \infty$ denotes the support of y^d, then $(\tilde{a}, \tilde{b})$ with $\tilde{a} = a - M_f T$ and $\tilde{b} = b + M_f T$ represents the part of the domain of u^* that might influence $y^*(T, \cdot)$ on (a, b) directly. Finally, optimality enforces u^* to tend to zero outside of $(\tilde{a}, \tilde{b})$. As a consequence, we consider

$$\mathfrak{R}(u) = \frac{\alpha}{2}\|u\|^2_{H^1_0(\Omega)}$$

with $\alpha > 0$ and $(\tilde{a}, \tilde{b}) \subset \Omega$ chosen large enough, to allow u^* to tend to zero. Following the discussion above, this restriction of the domain does not change the problem since optimal control and the corresponding state are zero outside Ω and $[0, T] \times \Omega$ anyway. Although this kind of regularization enforces the initial data to be continuous, shock phenomena may still occur in the solution of (1). For example, Burgers' equation with smooth initial data u and some $x \in \mathbb{R}$ with $u'(x) < 0$ develops shocks in finite time $T = -1/\min\{u'(x)\}$; see [27].

5.2 The Algorithm

Since the adjoint equation allows for a compact gradient representation of (2), we utilize a steepest descent method to solve the discretized problem (34) that is given in Algorithm 1.

Algorithm 1 Solution Algorithm

1: Choose $u^{(1)}$, set $k = 1$.
2: **while** stopping criterion not satisfied **do**
3: Solve discretization of primal equation (10) according to (21) to obtain $y^T_{(k)}$
4: Evaluate $\mathcal{J}^\Delta(y_{(k)}, u^{(k)})$
5: Solve adjoint equation (16) according to (25) to obtain $p^0_{(k)}$
6: Compute the update direction $\delta u^{(k)}$
7: Perform line search to obtain $\theta > 0$ with

$$\mathcal{J}^\Delta(y(u^{(k)}_\theta), u^{(k)}_\theta) - \mathcal{J}^\Delta(y(u^{(k)}), u^{(k)}) \leq -\theta\sigma \|\delta u^{(k)}\|^2_{H^1_0(\Omega)} \text{ for } u^{(k)}_\theta := u^{(k)} + \theta \delta u^{(k)} \quad (35)$$

8: Update control, i.e., $u^{(k+)} := u^{(k)}_\theta$, set $k := k + 1$
9: **end while**

Recalling (4), for a control $u^{(k)} \in H^1_0(\Omega)$ the reduced gradient of (P), i.e., $p^0_{(k)} - \alpha \Delta u^{(k)}$, is an element of $H^{-1}(\Omega)$ when considering the function space setting. Thus, the update direction $\delta u^{(k)}$ in $H^1_0(\Omega)$ is given by the Riesz representative (see, e.g., [23]) $\delta u^{(k)} = v - \alpha u^{(k)}$ with $v \in H^1_0(\Omega)$ solving

$$\Delta v = p^0_{(k)} \text{ in } \Omega.$$

The line search is realized by a backtracking method halving the trial step in each iteration until the Armijo condition (35) is satisfied with $\sigma = 10^{-4}$ as suggested in [28].

The algorithm terminates when the relative stopping criterion

$$\|\delta u^{(k)}\|_{H_0^1(\Omega)} \leq \varepsilon_G + \varepsilon_D \|\delta u^{(1)}\|_{H_0^1(\Omega)}$$

is satisfied with $\varepsilon_G = \varepsilon_D = 10^{-6}$.

In all numerical tests, the weight for the regularization term is $\alpha = 10^{-5}$.

5.3 Examples

The numerical tests consider Burgers' equation

$$y_t + \frac{1}{2}[y^2]_x = 0 \tag{36}$$

on the computational domain $(0, 3)$ with homogeneous Dirichlet boundary data. The chosen computational domain is large enough such that the support of the primal and adjoint equations, respectively, is included as discussed above. The control is discretized by standard P_1 conforming finite elements on a mesh of width Δx, and its integral average on the cells is computed exactly for the approximation.

We utilize the following numerical flux functions. First, we use the Engquist–Osher scheme employing the numerical flux

$$f^{EO} : \mathbb{R}^2 \to \mathbb{R}, \quad f^{EO}(v_1, v_2) = \int_{\bar{v}}^{v_1} f'(\xi)^+ d\xi + \int_{\bar{v}}^{v_2} f'(\xi)^- d\xi - f(\bar{v}) \tag{EO}$$

with $\bar{v} \in \mathbb{R}$ arbitrary and $f'(\xi)^+ := \max\{0, f'(\xi)\}$, $f'(\xi)^- := \min\{0, f'(\xi)\}$ denoting positive and negative parts of $f'(\xi)$, respectively. If it exists, $\bar{v}$ is chosen to be the sonic point of the flux function with $f'(\bar{v}) = 0$ to simplify computations. Applied to (36), the solution of the semidiscretization (10) admits a solution that is twice continuously differentiable with respect to time. In case of EO, it was shown in [32] that the assumptions of Theorem 1 are met for time steps Δt such that $\lambda = \Delta t / \Delta x$ satisfies $\lambda \sup_{|y| \leq M_y} |f'(y)| \leq (1 - \rho) 2^{-1}$ and some $\rho \in (0, 1)$.

Second, we also utilize the modified Lax–Friedrichs scheme with numerical flux

$$f^{LF} : \mathbb{R}^2 \to \mathbb{R}, \quad f^{LF}(v_1, v_2) = \frac{1}{2}(f(v_1) + f(v_2)) + \frac{\gamma \Delta x}{2 \Delta t}(v_1 - v_2) \tag{LF}$$

for a parameter $\gamma \in (0, 1)$ (the original Lax–Friedrichs numerical flux is obtained for $\gamma = 1$). For this choice, the solution of (10) is smooth. Again, it was shown in [32] that the assumptions of Theorem 1 are satisfied for a time discretization with an explicit Euler method and time steps such that

$$\lambda \sup_{|y|\le M_y} |f'(y)| \le \min\{(1-\rho)\min\{\gamma, 2(1-\gamma)\}, 1-\gamma\}$$

holds with some $\rho \in (0, 1)$.

Example 1 presents the performance of the algorithm and demonstrates the necessity of employing RK schemes that satisfy the conditions in Table 1 to obtain a consistent discretization of (P) for the same step length as in case of the basic explicit Euler time stepping. We consider the desired, piecewise constant state

$$y^d(x) = \begin{cases} \frac{1}{3}, & x \in [\frac{6}{5}, \frac{5}{3}], \\ -\frac{1}{10}, & x \in (\frac{5}{3}, \frac{7}{4}], \\ 0, & \text{else}, \end{cases}$$

depicted in red in Fig. 2, a spatial discretization of width $\Delta x = 500^{-1}$, $T = 1$, and set

$$\lambda = 1.0.$$

Moreover, we utilize the Engquist–Osher numerical flux, and thus, λ is chosen sufficiently small to satisfy $\lambda \sup_{|y|\le M_y} |f'(y)| < 2^{-1}$ for all initial conditions with $|u| < 1/2$ since $f'(y) = y$ and the maximum principle for entropy solutions (see Proposition 1) ensures $\|y\|_{L^\infty(\mathcal{Q})} \le \|u\|_{L^\infty(\mathbb{R})}$.

For Heun's scheme (32), Algorithm 1 converged within 149.587 iterations for the initialization $u^{(1)} \equiv 0$. The optimal control can be found in Fig. 1, while the corresponding state at $t = T$ is presented in Fig. 2.

Figure 3 displays the final time data for the adjoint state, the difference of desired state, and solution to the state system evaluated at $t = T$, clearly containing discontinuities. This, in particular, requires the adjoint scheme to be TVD-stable to avoid spurious oscillations in the solution. Figure 4 displays this solution to the discretized adjoint equation evaluated at $t = 0$, not showing oscillations.

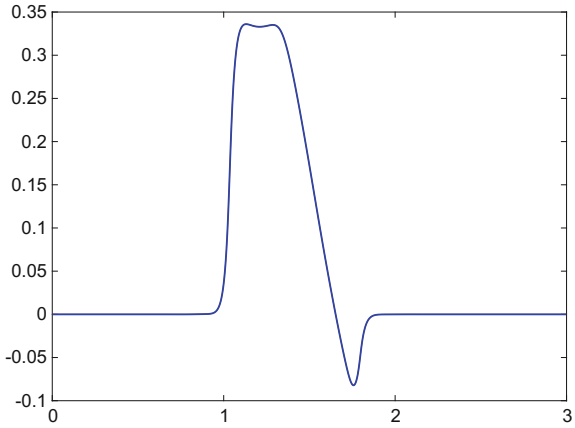


Fig. 1 Example 1. Optimal control

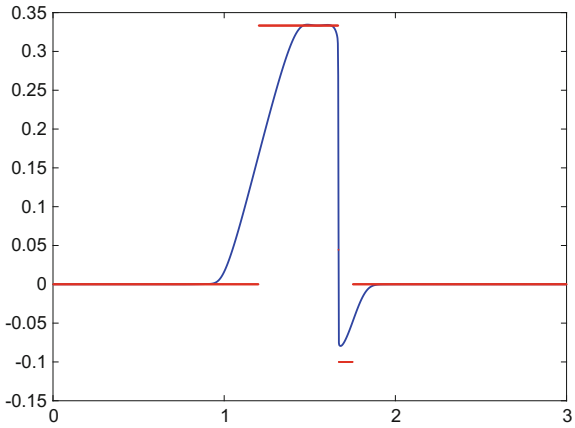


Fig. 2 Example 1. Solution of (1) (blue) versus desired state (red)

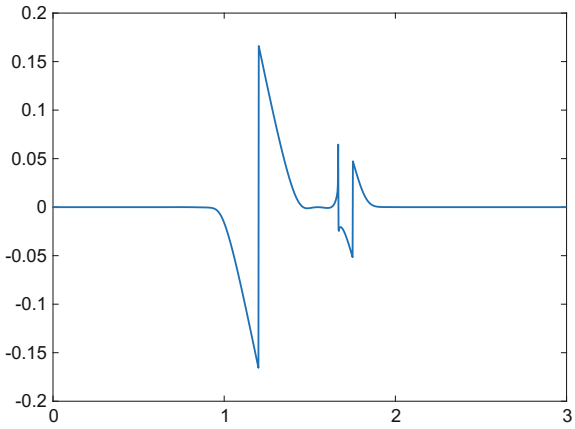


Fig. 3 Example 1. Final time data for the adjoint equation

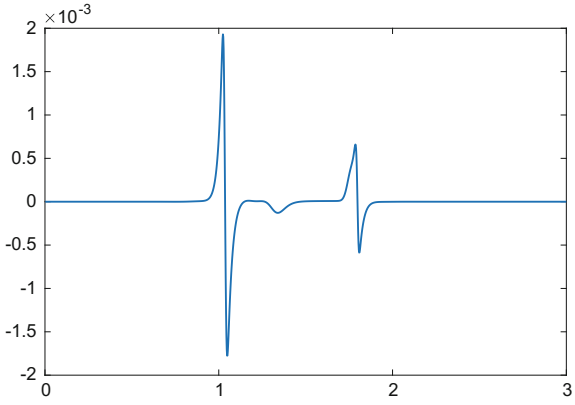


Fig. 4 Example 1. Solution to the adjoint equation in $t = 0$

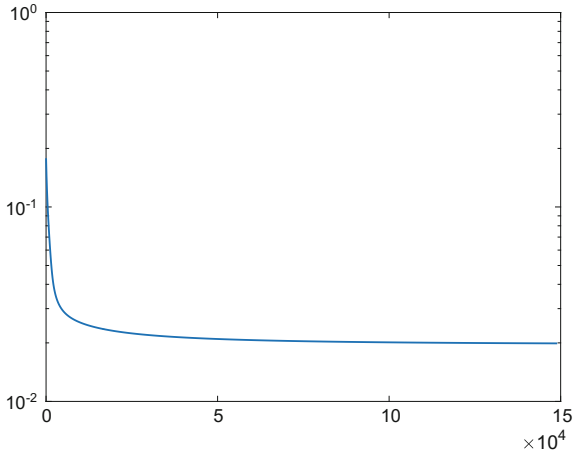


Fig. 5 Example 1. Evolution of the objective value

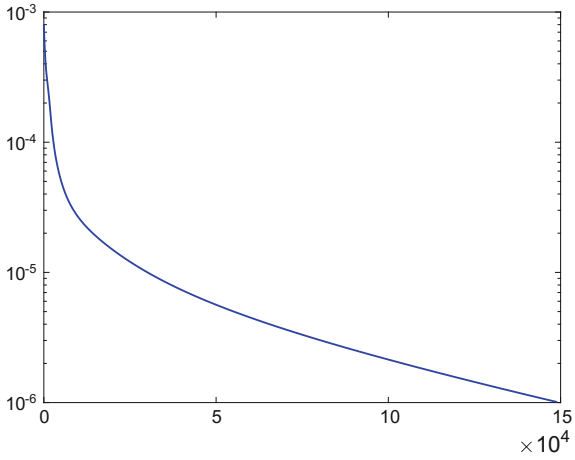


Fig. 6 Example 1. Evolution of the norm of the update direction

To report on the convergence behavior of the algorithm, we show the behavior of the objective value and the $H_0^1(\Omega)$-norm of the update direction in Figs. 5 and 6, respectively.

For the second purpose of this example, we change the initialization of Algorithm 1 to

$$u^{(1)}(x) = \max\{0, \min\{ax, a - ax\}\}, \quad u^{(1)} \in H_0^1(\Omega) \tag{37}$$

with $a = 2.075$ and consider $\Delta x = 1000^{-1}$. Moreover, we verify in every iteration and each time step of steps 3 and 5 of Algorithm 1 that the total variation of state and adjoint at this time is bounded by the total variation if the initial and final time data, respectively, i.e., whether

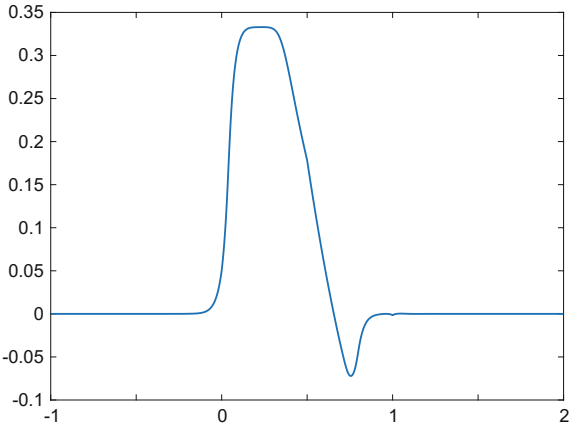


Fig. 7 Example 1. Optimal control for the initial data (37)

$$|\mathbf{y}^i|_{TV} \leq |\mathbf{y}^0|_{TV} \text{ and } |\mathbf{p}^{i-1}|_{TV} \leq |\mathbf{p}^{N_T}|_{TV}$$

hold for $i = 1, \ldots, N_T$ with N_T denoting the number of time steps in the discretization scheme of state and adjoint equations and the total variation seminorm $|\mathbf{v}|_{TF} = \sum_{j=1}^{N-1} |v_{j+1} - v_j|$. If this is not the case, then the algorithm terminates since the corresponding iteration does not satisfy the requirements of Theorem 1. The choice of the initialization $u^{(1)}$ violates $\lambda \sup_{|y| \leq M_y} |f'(y)| \leq (1-\rho)2^{-1}$ for some $\rho \in (0, 1)$, but on the given mesh, Algorithm 1 still converges in case of Heun's scheme (32). Figure 7 displays the corresponding optimal control, while Fig. 8 displays the norm of the update direction along the iterations of the algorithm.

In addition, we consider the RK2 scheme

$$\begin{array}{cc} 0 & 0 \\ \frac{1}{2(1-10^{(-4)})} & 0 \\ \hline 10^{-4} & (1-10^{-4}) \end{array}, \tag{38}$$

clearly violating the conditions of Table 1 as

$$0 \leq b_1 - b_2 a_{21} = 10^{-4} - 1/2.$$

In the same setting, Algorithm 1 terminates in the very first iteration as the computed discrete state fails to satisfy $|y^i|_{TV} \leq |y^0|_{TV}$.

In this example, we reported on the convergence behavior of Algorithm 1 and demonstrated the importance of choosing RK schemes that satisfy the conditions of Table 1. Although the problem with the RK2 scheme (38) can be circumvented by choosing a smaller time step, this is not recommended in the context of optimal control since this increases the numerical effort to solve the problem with respect to computing time along with storage requirements.

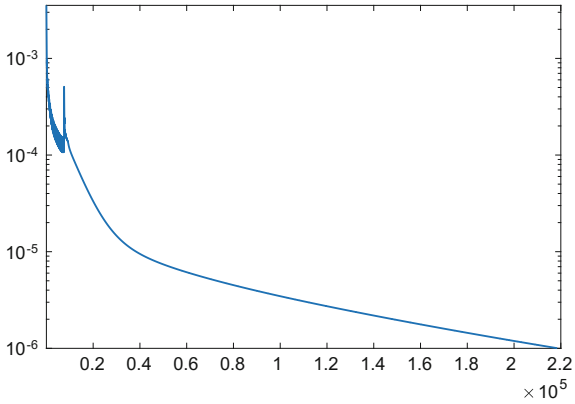


Fig. 8 Example 1. Evolution of the norm of the update direction for the initial data (37)

Example 2 considers a setting, where shocks appear neither in the solution to the state equation nor in the final time data due to the chosen desired state that is taken from [2]. In the latter work, longtime behavior of optimization algorithms for Burgers' equation with respect to several methodologies was studied. We only transform the domain such that the target function is contained in (0, 3). The desired state is given by

$$y^d(x) = \frac{3}{2000}\left(-e^{-(5\sqrt{20}-\phi(x))^2} + e^{-(2\sqrt{20}+\phi(x))^2} + \sqrt{\pi}\phi(x)(erf(5\sqrt{20}-\phi(x)) + erf(2\sqrt{20}+\phi(x)))\right)$$

and it is depicted in Fig. 9. Here, φ represents a linear transformation of arguments from [0, 3] to the interval [−50, 100], a domain in the scale of the original desired state from [2]. To compensate for gradient scaling in the initial state because of the transformed domain, we refrain from longtime behavior analysis and consider $T = 1$ as well as $\Delta x = 5 \times 10^{-2}$. This setting allows for a discussion concerning the approximation order of the solutions to the state and adjoint equations as we can expect them to be regular enough for the Taylor expansion that provides the order of convergence of the time stepping. For both numerical flux functions, we will obtain a reference solution based on the RK3 scheme (33) and compare it with solutions obtained by the explicit Euler and Heun's method. Moreover, we will analyze the order of convergence of the RK methods at the example of the Engquist–Osher numerical flux function. To this end, a further reference solution based on the RK3 scheme and for a time step $\Delta\tilde{t} = 2^{-4}\Delta t$ with $\Delta t = \lambda \Delta x$ is generated. Then, solutions to the problem of optimal control are generated for the RK1, RK2, and RK3 schemes with time step sizes $\Delta\tilde{t} = 2^{-j}\Delta t$, $j = 0, \ldots, 3$ are obtained and compared to the reference solution by providing

$$E_y = \max_{j,n} |y_j^n - \tilde{y}_j^n|, \quad E_p = \max_{j,n} |p_j^n - \tilde{p}_j^n|.$$

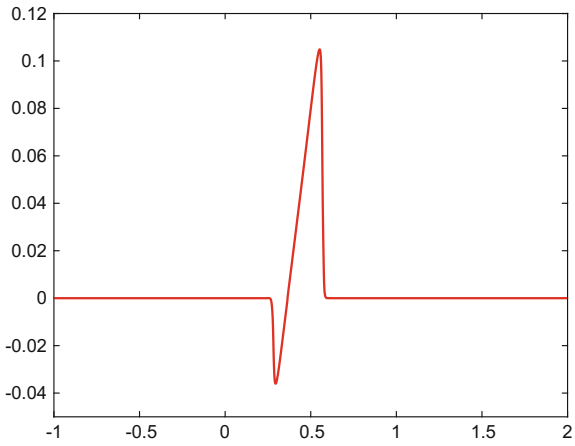


Fig. 9 Example 2. Desired state

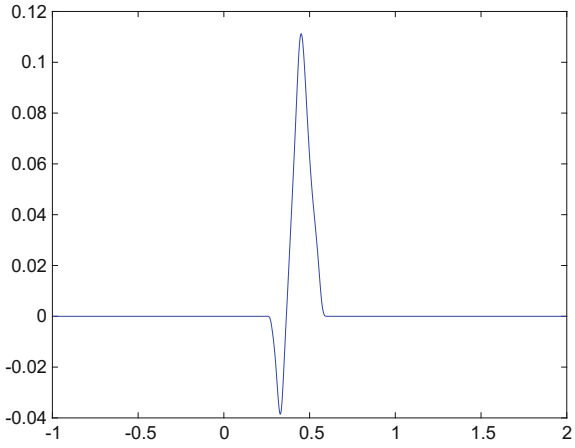


Fig. 10 Example 2. Optimal control for the RK3 scheme

First, we use the numerical flux of the Lax–Friedrichs scheme (LF) with $\gamma = 1/2$ and $\lambda = 2.75$ since in this case the regularity of the solution to (10) with respect to time is sufficient for the application of an RK3 scheme. Concerning the numerical flux functions, this choice of λ satisfies the conditions for the explicit Euler time stepping as formulated in [32]. The solution obtained by (33) provides the reference solution for this test, and the corresponding optimal control is presented in Fig. 10.

Further, we computed the solution for the same parameters utilizing an explicit Euler time discretization and Heun's method (32). In Fig. 11a–c, we present the solutions to the semidiscrete forward problems (10), with the controls obtained by Algorithm 1, evaluated at $t = T$ and the difference to the desired state. In Fig. 11d, the differences of solutions to the lower-order time discretization schemes and the reference solution are plotted at $t = T$.

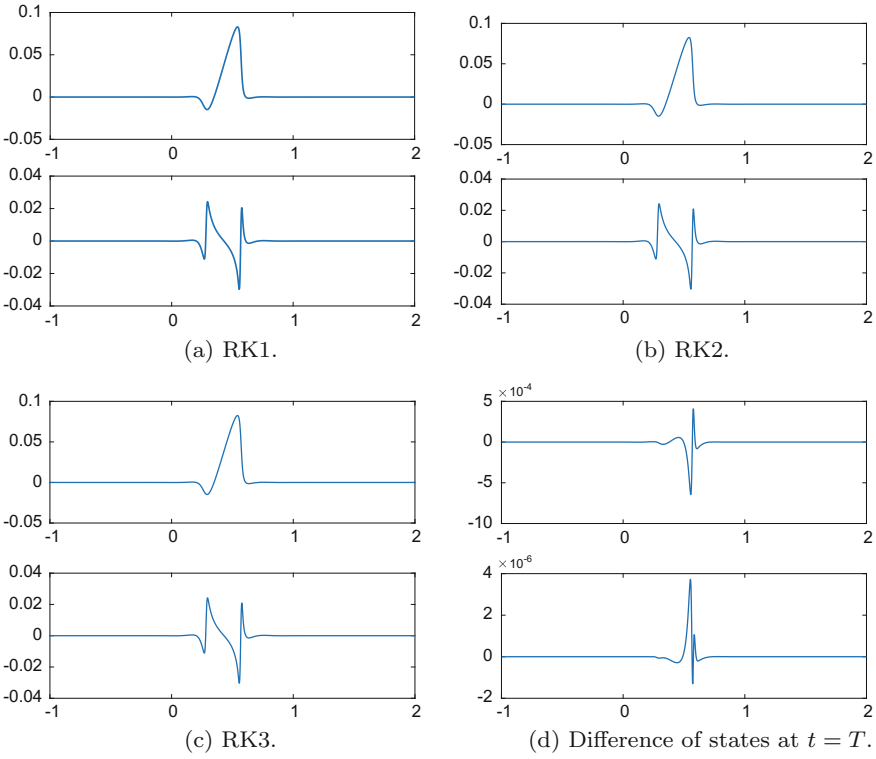
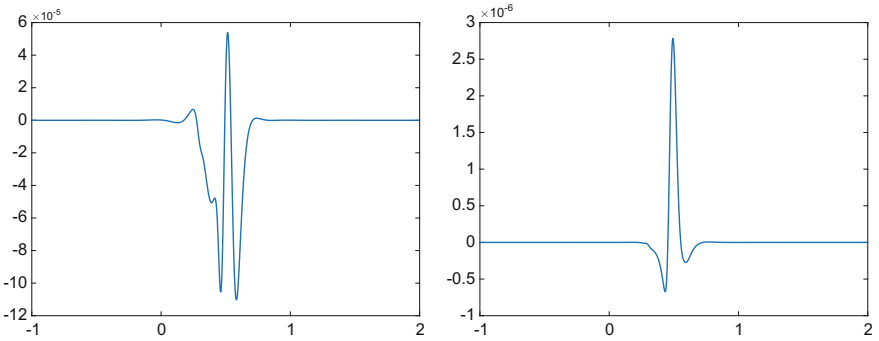


Fig. 11 Example 2. States at $t = T$ with associated control, and differences to desired state

As it can be seen in Fig. 11d, the RK2 scheme approximates the solution obtained from applying the RK3 scheme more accurately than the explicit Euler scheme. In Fig. 12, we have depicted the differences in the optimal control obtained by Algorithm 1 for the lower-order schemes to the reference solutions. Finally, the following tables provide E_y and E_p (Tables 3 and 4).

Again, we observe that the second-order scheme approximates the solution of the RK3 scheme better than the first-order scheme by two orders of magnitude, which, by $\Delta t = 5 \times 10^{-2}/\lambda \approx 2 \times 10^{-2}$, corresponds to the behavior expected for the approximation order for RK schemes.

Next, we test the problem of optimal control with a spatial discretization based on the Engquist–Osher scheme (EO) with $\lambda = 1.0$. The scheme is less diffusive but only provides a solution of (10) which is twice continuously differentiable with respect to time. Again, we obtained the reference solution by the RK3 scheme defined in (33) and depict the corresponding optimal control in Fig. 13.



(a) RK3 vs RK1. (b) RK3 vs RK2.

Fig. 12 Example 2. Difference of the optimal control

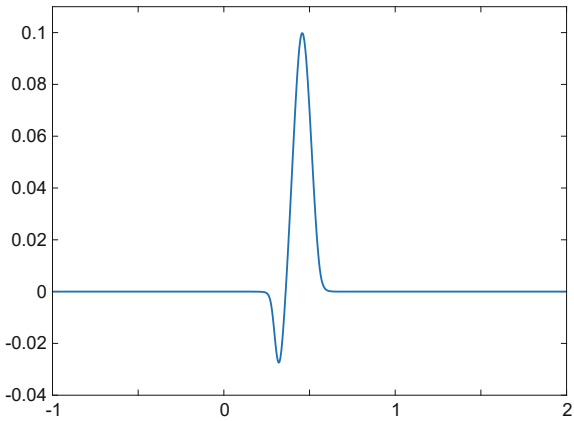
Table 3 L^∞-discrepancy of states to reference state E_y

RK1	RK2
6.4×10^{-4}	1.3×10^{-6}

Table 4 L^∞-discrepancy of adjoint to reference adjoint E_p

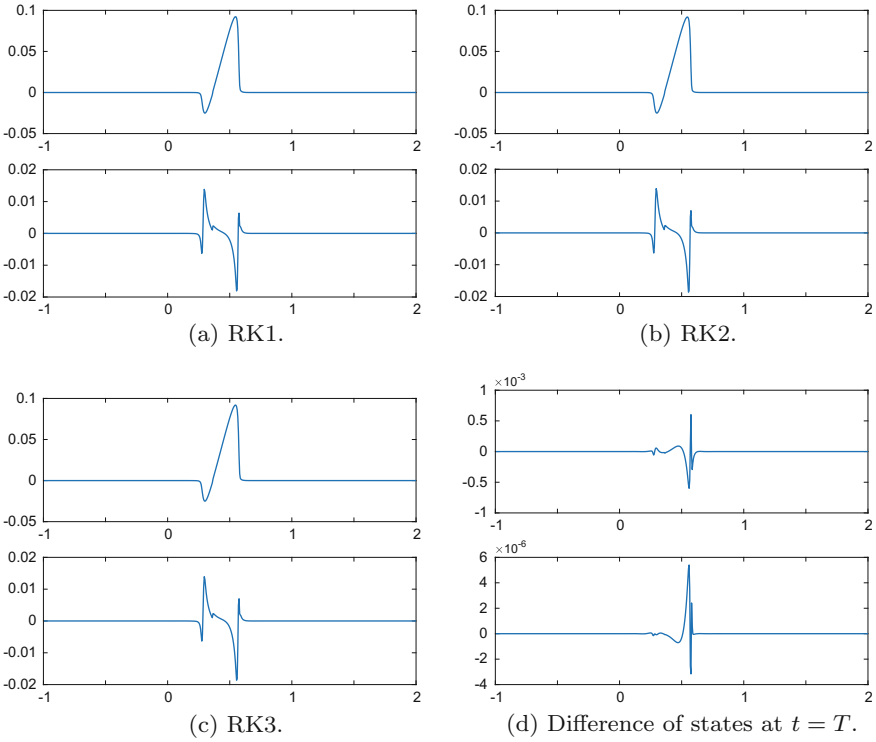
RK1	RK2
1.2×10^{-5}	2.2×10^{-7}

Fig. 13 Example 2. Optimal control for RK3 scheme



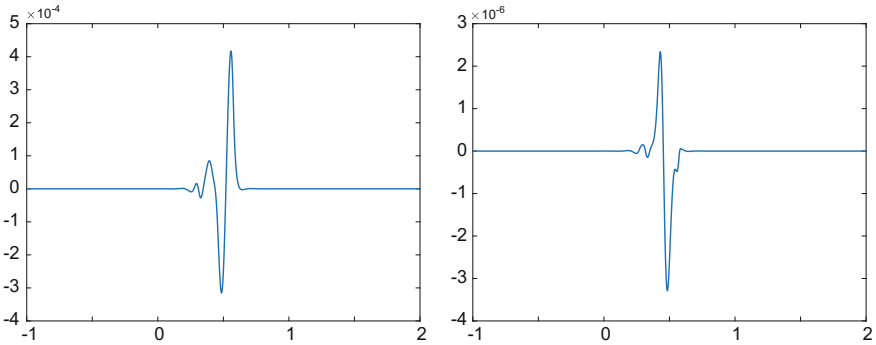
Further, we computed the solution for the same parameters utilizing an explicit Euler time discretization and Heun's method (32). Again, we depict the optimal states evaluated at $t = T$ and the difference to the desired state in Fig. 14a–c, while we find the differences of the optimal states for the RK1 and RK2 schemes at $t = T$ and the reference solution in Fig. 14d.

In Fig. 15, we have depicted the differences in the controls obtained by Algorithm 1 for the lower-order time discretization methods to the reference solutions. Again, we observe a better approximation of the reference solution in case of the RK2 scheme.



(a) RK1. (b) RK2. (c) RK3. (d) Difference of states at $t = T$.

Fig. 14 Example 2. States at $t = T$ and associated control, and differences to desired state



(a) RK3 vs RK1. (b) RK3 vs RK2.

Fig. 15 Example 2. Difference of the optimal control

Table 5 L^∞-discrepancy of states to reference state E_y

2^{-j}	RK1	RK2	RK3
1	6.4×10^{-4}	5.4×10^{-6}	5.1×10^{-8}
1/2	3.2×10^{-4}	1.3×10^{-6}	6.2×10^{-9}
1/4	1.6×10^{-4}	3.3×10^{-7}	7.6×10^{-10}
1/8	8.0×10^{-5}	8.3×10^{-8}	8.4×10^{-11}

Table 6 L^∞-discrepancy of adjoints to reference adjoint E_p

2^{-j}	RK1	RK2	RK3
1	7.0×10^{-5}	4.7×10^{-5}	4.7×10^{-5}
1/2	3.4×10^{-5}	2.2×10^{-5}	2.2×10^{-5}
1/4	1.5×10^{-5}	9.7×10^{-6}	9.7×10^{-6}
1/8	6.2×10^{-6}	3.3×10^{-6}	3.3×10^{-6}

In order to quantify the convergence order, we next generate a new reference solution by utilizing the RK3 scheme and a time step $\Delta\tilde{t} = 2^{-4}\Delta t$. Then, we compare solutions to the problem of optimal control for the RK1, RK2, and RK3 schemes, respectively, and time steps $\Delta\tilde{t} = 2^{-j}\Delta t$ for $j = 0, \ldots, 3$, i.e., we compare the states to the optimal control and the corresponding adjoints. The following tables provide E_y and E_p in case of the Engquist–Osher numerical flux function (Table 5).

This example demonstrated the convergence order of the corresponding discretization schemes. In particular, we observe the order bound for the adjoint in Table 6.

Acknowledgements This work is supported by the German Research Foundation (DFG) within project B02 of CRC TRR 154.

References

1. Adams, R.A., Fournier, J.J.F.: Sobolev Spaces, volume 140 of Pure and Applied Mathematics, 2nd edn. Elsevier/Academic Press, Amsterdam (2003)
2. Allahverdi, N., Pozo, A., Zuazua, E.: Numerical aspects of large-time optimal control of Burgers equation. ESAIM Math. Model. Numer. Anal. **50**(5), 1371–1401 (2016)
3. Banda, M.K., Herty, M.: Adjoint IMEX-based schemes for control problems governed by hyperbolic conservation laws. Comput. Optim. Appl. **51**(2), 909–930 (2012)
4. Bianchini, S.: On the shift differentiability of the flow generated by a hyperbolic system of conservation laws. Discrete Contin. Dynam. Syst. **6**(2), 329–350 (2000)
5. Bonnans, J.F., Laurent-Varin, J.: Computation of order conditions for symplectic partitioned Runge-Kutta schemes with application to optimal control. Numer. Math. **103**(1), 1–10 (2006)
6. Bouchut, F., James, F.: One-dimensional transport equations with discontinuous coefficients. Nonlinear Anal. **32**(7), 891–933 (1998)

7. Bouchut, F., James, F.: Differentiability with respect to initial data for a scalar conservation law. In: Hyperbolic problems: theory, numerics, applications, Vol. I (Zürich, 1998), volume 129 of Internat. Ser. Numer. Math., pp. 113–118. Birkhäuser, Basel (1999)
8. Bressan, A.: Hyperbolic systems of conservation laws, volume 20 of Oxford Lecture Series in Mathematics and its Applications. Oxford University Press, Oxford (2000) (The one-dimensional Cauchy problem)
9. Bressan, A., Guerra, G.: Shift-differentiability of the flow generated by a conservation law. Discrete Contin. Dynam. Syst. **3**(1), 35–58 (1997)
10. Bressan, A., Marson, A.: A maximum principle for optimally controlled systems of conservation laws. Rend. Sem. Mat. Univ. Padova **94**, 79–94 (1995)
11. Bressan, A., Marson, A.: A variational calculus for discontinuous solutions of systems of conservation laws. Comm. Partial Differ. Equ. **20**(9–10), 1491–1552 (1995)
12. Bressan, A., Shen, W.: Optimality conditions for solutions to hyperbolic balance laws. In: Control methods in PDE-dynamical systems, volume 426 of Contemp. Math., pp 129–152. Amer. Math. Soc., Providence, RI (2007)
13. Cesari, L.: Optimization - Theory and Applications, volume 17 of Applications of Mathematics. Springer, New York (1983) (Problems with ordinary differential equations)
14. Dunford, N., Schwartz, J.T.: Linear Operators. Part I. Wiley Classics Library. Wiley, New York (1988). General theory, With the assistance of William G. Bade and Robert G. Bartle, Reprint of the 1958 original, A Wiley-Interscience Publication
15. Giles, M., Ulbrich, S.: Convergence of linearized and adjoint approximations for discontinuous solutions of conservation laws. Part 1: linearized approximations and linearized output functionals. SIAM J. Numer. Anal. **48**(3), 882–904 (2010)
16. Giles, M., Ulbrich, S.: Convergence of linearized and adjoint approximations for discontinuous solutions of conservation laws. Part 2: adjoint approximations and extensions. SIAM J. Numer. Anal. **48**(3), 905–921 (2010)
17. Giles, M.B.: Discrete Adjoint Approximations with Shocks. Hyperbolic Problems: Theory. Numerics, Applications, pp. 185–194. Springer, Berlin (2003)
18. Gottlieb, S., Shu, C.-W.: Total variation diminishing Runge-Kutta schemes. Math. Comp. **67**(221), 73–85 (1998)
19. Gottlieb, S., Shu, C.-W., Tadmor, E.: Strong stability-preserving high-order time discretization methods. SIAM Rev. **43**, (1):89–112 (electronic) (2001)
20. Hager, W.W.: Runge-Kutta methods in optimal control and the transformed adjoint system. Numer. Math. **87**(2), 247–282 (2000)
21. Hajian, S., Hintermüller, M., Ulbrich, S.: Total variation diminishing schemes in optimal control of scalar conservation laws (2017). WIAS-Preprint https://doi.org/10.20347/WIAS.PREPRINT.2383 Accepted in IMA J Numerical Analysis, drx073, https://doi.org/10.1093/imanum/drx073
22. Herty, M., Piccoli, B.: A numerical method for the computation of tangent vectors to 2×2 hyperbolic systems of conservation laws. Commun. Math. Sci. **14**(3), 683–704 (2016)
23. Hinze, M., Pinnau, R., Ulbrich, M., Ulbrich, S.: Optimization with PDE constraints. Mathematical Modelling: Theory and Applications, vol. 23. Springer, New York (2009)
24. Ketcheson, D.I.: Highly efficient strong stability-preserving Runge-Kutta methods with low-storage implementations. SIAM J. Sci. Comput. **30**(4), 2113–2136 (2008)
25. Kraaijevanger, J.F.B.M.: Contractivity of Runge-Kutta methods. BIT **31**(3), 482–528 (1991)
26. Kružkov, S.N.: First order quasilinear equations with several independent variables. Mat. Sb. (N.S.) **81**(123), 228–255 (1970)
27. LeVeque, R.J.: Numerical Methods for Conservation Laws. Lectures in Mathematics ETH Zürich, 2nd edn. Birkhäuser, Basel (1992)
28. Nocedal, J., Wright, S.J.: Numerical Optimization. Springer Series in Operations Research and Financial Engineering, 2nd edn. Springer, New York (2006)
29. Pfaff, S., Ulbrich, S.: Optimal boundary control of nonlinear hyperbolic conservation laws with switched boundary data. SIAM J. Control Optim. **53**(3), 1250–1277 (2015)

30. Ruuth, S.J., Spiteri, R.J.: Two barriers on strong-stability-preserving time discretization methods. In: Proceedings of the Fifth International Conference on Spectral and High Order Methods (ICOSAHOM-01) (Uppsala), volume 17, pp. 211–220 (2002)
31. Toro, E.F.: Riemann Solvers and Numerical Methods for Fluid Dynamics, 3rd edn. Springer, Berlin (2009). A Practical Introduction
32. Ulbrich, S.: Optimal control of nonlinear hyperbolic conservation laws with source terms. Technische Universität München (2001)

A Time-Delay Neural Network Model for Unconstrained Nonconvex Optimization



Li-Zhi Liao and Yu-Hong Dai

Abstract In this paper, we first present some theoretical properties which include the convergence and bounds of both the trajectory and energy function for general neural network models for unconstrained nonconvex optimization problems. Based on this analysis, a novel time-delay neural network model is proposed for unconstrained nonconvex optimization. The simulation results of the new neural network on two examples indicate that the new neural network is quite efficient and outperforms the gradient neural network.

Keywords Unconstrained optimization · Neural network · Time-delay Barzilai-Borwein method

1 Introduction

The continuous trajectory approach for optimization can be classified into two categories. The second category was started in early 1990s and was triggered by the seminal work of Karmarkar's interior point algorithm for linear programming (see

The first author was supported in part by grants from Hong Kong Baptist University (FRG) and General Research Fund (GRF) of Hong Kong. The second author was supported in part by the Chinese NSFC Grant (Nos. 11631013, 11331012 and 71331001) and the Key Project of Chinese National Programs for Fundamental Research and Development Grant (No. 2015CB856002).

L.-Z. Liao
Department of Mathematics, Hong Kong Baptist University,
Kowloon Tong, Kowloon, Hong Kong
e-mail: liliao@hkbu.edu.hk

Y.-H. Dai (✉)
LSEC, Academy of Mathematics and Systems Sciences
Chinese Academy of Sciences, Box 2719, Beijing 100190, China
e-mail: dyh@lsec.cc.ac.cn

Y.-H. Dai
School of Mathematical Sciences, University of Chinese Academy of Sciences,
Beijing 100049, China

© Springer International Publishing AG, part of Springer Nature 2018

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*,
Springer Proceedings in Mathematics & Statistics 235,
https://doi.org/10.1007/978-3-319-90026-1_7

[11]). In particular, the central path in the interior point methods is indeed a continuous trajectory. An early review article on these two categories can be found in [13]. A recent paper on the continuous trajectory on the convex programming can be found in Qian et al. [18]. The second category was pioneered by Hopfield in early 1980s; see Hopfield and his collaborator [7–9]. Since then, a huge number of research articles have been published in the neural network community. Nowadays, neurodynamic optimization is still a very hot research area in neural network. The emerging power of artificial neural networks (we simply call them neural networks throughout this paper) and their tremendous application potentials have attracted significant attention from both industry and academia recently. The research as well as experiment in neural networks have generated many impressive and outstanding results. The application of neural networks to optimization area has also blossomed. An even earlier article by Chua and Lin [6], which solves optimization problems through circuits, shares the similar spirit of neural network.

Generally speaking, neural network provides an alternative and attractive approach for solutions of optimization problems. Many neural network models have been shown to be efficient for some special classes of optimization problems, such as linear programming and quadratic programming (LP/QP) [2, 5, 12, 15, 19, 20, 23–26], linear complementarity problems [22], and assignment problems [22]. Many applications of neural networks in optimization can be found in the book [4].

For nonlinear optimization problems, most of neural network models are established for constrained cases. These models are initially based on either Lagrange method [10, 12, 28, 29] or penalty method [3, 14, 16, 19], and then the gradient method is applied to the resulting unconstrained optimization problems. As a matter of fact, many LP/QP neural networks share the same methodology of removing constraints. Therefore, a good and fast neural network model for solving unconstrained optimization problems becomes not only important but also essential. Unfortunately, the commonly used gradient neural network model could converge very slowly and limit the overall performance. It is the purpose of this paper to investigate the key factors in designing a fast converging model and propose a time-delay neural network model for unconstrained optimization problems. The simulation results show that the new model outperforms the gradient model.

In this paper, we are interested in the neural network models for the following general unconstrained nonconvex optimization problem

$$\min_{x \in R^n} \ f(x) \qquad (P)$$

where $f : R^n \to R^1$ has continuous first-order derivatives. One commonly used neural network model for solving (P) is the gradient model, which has the form

$$\frac{dx(t)}{dt} = -\mu \cdot \nabla f(x(t)), \quad x(0) = x_0, \quad t \geq 0, \tag{1}$$

where $\nabla f(x)$ is the gradient vector of function $f(x)$ and μ (> 0) is a scaling factor. The energy function corresponding to the neural network model (1) is

$$E(x(t)) = f(x(t)). \tag{2}$$

For convenience, we call the neural network representing (1) as **NNI**.

It should be noted that since $f(x)$ is nonlinear and nonconvex, many local minimizers may exist for problem (P). Our study here is focused on the local minimizers of problem (P) (since global minimization normally involves a local minimizer search scheme), but we are not limited to any specific local minimizer or equilibrium point.

The discussion in this paper focuses on two parts. First, under mild assumptions on $f(x)$, we analyze the properties of the following general neural network

$$\frac{dx(t)}{dt} = \mu \cdot d(x(t)), \quad x(0) = x_0, \quad t \geq 0, \tag{3}$$

where $d(x(t))$ is the direction at $x(t)$ and μ is a scalar [in our theoretical analysis, μ is always assumed to be (1)]. We prove that, if $d(x(t))$ maintains the descent property for energy function (2) and is Lipschitz continuous, then every point on the trajectory of the general neural network (3) is not an equilibrium point of $f(x)$ provided that $d(x_0)^T \nabla f(x_0) < 0$. Further, if $d(x(t))$ satisfies the so-called sufficient descent condition (9), then every accumulation point of the trajectory must be an equilibrium point of $f(x)$. We also analyze the asymptotic properties of the general neural network model (3). Our proof is different from the common approach where specific equilibrium point is assumed. Our result, in some sense, is similar to the global convergence result in optimization.

Second, we propose a new efficient first-order neural network model for unconstrained nonconvex optimization. For this purpose, we consider the neural network with two-step time delay. To investigate the behaviors of the new time-delay neural network model and NNI, two optimization test problems are simulated on neural network NNI and the new neural network. The simulation results are very promising and suggest that the new neural network is more efficient than the neural network NNI.

This paper is organized as follows. In the next section, we analyze the stability and asymptotic properties for the general neural network model (3). Section 3 discusses the detailed neural network architecture for the new time-delay neural network model. In Sect. 4, simulation results are reported on two test problems for NNI and the new neural network. Finally, we address some conclusions and future research directions in Sect. 5.

2 Analysis for General Neural Network 3

First, we make the following basic assumptions on the objection function $f(x)$.

Assumption 1 (i) The level set $\mathcal{L}(x_0) = \{x \in R^n \mid f(x) \leq f(x_0)\}$ is bounded for any $x_0 \in R^n$. (ii) For any $x_0 \in R^n$, the gradient $\nabla f(x)$ is Lipschitz continuous in $\mathcal{L}(x_0)$, namely, there exists a constant $L > 0$ such that

$$||\nabla f(x) - \nabla f(y)|| \leq L||x - y||, \qquad \text{for any } x, y \in \mathcal{L}(x_0). \tag{4}$$

From Assumption 1 on $f(x)$, we know by the theory of differential equations that the neural network (3) has a unique solution if $d(x(t))$ is Lipschitz continuous with $x(t)$ and this solution can be extended to $[0, \infty)$. Furthermore, for the neural network (3), we make the following assumptions.

Assumption 2 Let $x(t)$ be a solution of (3). (i) There exists a constant $L_1 > 0$ such that

$$\|d(x(t_1)) - d(x(t_2))\| \leq L_1\|x(t_1) - x(t_2)\|, \qquad \text{for all } t_1, t_2 > 0. \tag{5}$$

(ii) $d(x(t))$ in (3) is a descent direction of $f(x(t))$, i.e.,

$$d(x(t))^T \nabla f(x(t)) < 0, \qquad \text{if } \nabla f(x(t)) \neq 0 \tag{6}$$

and

$$\nabla f(x(t)) = 0 \implies d(x(t)) = 0. \tag{7}$$

Before analyzing the neural network (3), we would like to point out that our coming analysis is also applicable to the neural network with time delay, namely,

$$\frac{dx(t)}{dt} = \mu \cdot d(x(t), x(t - \triangle t), x(t - 2\triangle t), \ldots), \quad x(0) = x_0, \quad t > 0. \tag{8}$$

As a result, we are able to apply the results obtained in this section to the new neural network with time delay proposed in the next section.

Theorem 1 *Suppose that Assumptions 1 and 2 hold. For the neural network (3), if* $\nabla f(x_0) \neq 0$, *we have that*
(a) For any $t > 0$, $\nabla f(x(t)) \neq 0$.
(b) If there exists a constant $\gamma > 0$ *such that*

$$d(x(t))^T \nabla f(x(t)) \leq -\gamma \|\nabla f(x(t))\|^2, \tag{9}$$

then $\lim_{t \to \infty} \nabla f(x(t)) = 0$. *Thus for any sequence* $\{x(t_k)\} \subset \mathcal{L}(x_0)$, *if* $t_k \to \infty$ *and* $\lim_{k \to \infty} x(t_k) = x^*$, *we have* $\nabla f(x^*) = 0$.

Proof (a) From (3) and (6), we have that

$$\frac{df(x(t))}{dt} = [\nabla f(x(t))]^T \frac{dx(t)}{dt} = d(x(t))^T \nabla f(x(t)) \leq 0, \tag{10}$$

which implies that $f(x(t))$ is monotonically decreasing and hence $\{x(t) : t \geq 0\} \subset \mathcal{L}(x_0)$. By Assumptions 1 and 2, there exists a constant $M > 0$ such that for all $t > 0$,

$$\|\nabla f(x(t))\| \leq M \quad \text{and} \quad \|d(x(t))\| \leq M. \tag{11}$$

For any $t > 0$, $c \in (0, \frac{1}{2L_1}]$ and $\tau_1, \tau_2 \in [t, t + c]$, by (5), (3), and (11), we can get

$$\begin{aligned} \|d(x(\tau_1))\| - \|d(x(\tau_2))\| &\leq \|d(x(\tau_1)) - d(x(\tau_2))\| \\ &\leq L_1 \|x(\tau_1) - x(\tau_2)\| \\ &= L_1 \| \int_{\tau_2}^{\tau_1} d(x(t))dt \| \\ &\leq L_1 c \max_{\tau \in [t,t+c]} \|d(x(\tau))\|. \end{aligned} \tag{12}$$

Due to the choices of τ_1 and τ_2, we can show from (12) that

$$\min_{\tau \in [t,t+c]} \|d(x(\tau))\| \geq (1 - L_1 c) \max_{\tau \in [t,t+c]} \|d(x(\tau))\| \tag{13}$$

for any $t > 0$ and $c \in (0, \frac{1}{2L_1}]$. Suppose that $\nabla f(x_0) \neq 0$ but $\nabla f(x(t)) = 0$ for some $t \in (0, \infty)$. Then by (6) and (7), we also have that $d(x_0) \neq 0$ but $d(x(t)) = 0$. However, letting $c = \min\{\frac{t}{2}, \frac{1}{2L_1}\}$, we can get from (13) that

$$0 = \min_{\tau \in [t-c,t]} \|d(x(\tau))\| \geq \frac{1}{2} \max_{\tau \in [t-c,t]} \|d(x(\tau))\| \geq \frac{1}{2} \|d(x(t-c))\|, \tag{14}$$

which implies that $d(x(t - c)) = 0$. Taking $t - c$ as the new t and repeating this procedure recursively yield $d(x_0) = 0$. Thus if $\nabla f(x_0) \neq 0$, then for all $t > 0$, $\nabla f(x(t)) \neq 0$.

(b) We now assume that (9) holds but $\lim_{t \to \infty} \nabla f(x(t)) \neq 0$. Then, there exist a constant $\delta > 0$ and a sequence $\{t_k\}$ satisfying $t_{k+1} \geq t_k + \frac{1}{2(L+L_1)}$ such that

$$\|\nabla f(x(t_k))\| \geq \delta, \qquad \text{for all } k \geq 1. \tag{15}$$

Define $c_k = \frac{|d(x(t_k))^T \nabla f(x(t_k))|}{2(L+L_1)M^2}$. For any $t \in [t_k, t_k + c_k]$, it follows from (4), (5), and (11) that

$$\begin{aligned} & d(x(t))^T \nabla f(x(t)) - d(x(t_k))^T \nabla f(x(t_k)) \\ & \quad \le |d(x(t))^T [\nabla f(x(t)) - \nabla f(x(t_k))]| + |[d(x(t) - d(x(t_k))]^T \nabla f(x(t_k))| \\ & \quad \le (L + L_1) M \|x(t) - x(t_k))\| \\ & \quad \le (L + L_1) M^2 (t - t_k) \\ & \quad \le \frac{1}{2} |d(x(t_k))^T \nabla f(x(t_k))|, \end{aligned} \tag{16}$$

which with (6) gives

$$d(x(t))^T \nabla f(x(t)) \le \frac{1}{2} d(x(t_k))^T \nabla f(x(t_k)), \quad \text{for } t \in [t_k, t_k + c_k]. \tag{17}$$

(11) implies that $t_{k+1} \ge t_k + c_k$. Thus from this, (10), (17), (15), and (9), we obtain

$$\begin{aligned} & f(x(t_{k+1})) - f(x(t_1)) \\ & \quad \le \sum_{i=1}^{k} (f(x(t_i + c_i)) - f(x(t_i))) = \sum_{i=1}^{k} \int_{t_i}^{t_i + c_i} d(x(t))^T \nabla f(x(t)) dt \\ & \quad \le \sum_{i=1}^{k} \frac{1}{2} c_i d(x(t_i))^T \nabla f(x(t_i)) \le -\frac{\gamma^2 \delta^4 k}{4(L + L_1) M^2}. \end{aligned} \tag{18}$$

Letting $k \to \infty$ in (18), we obtain a contradiction with the fact that f is bounded below. The contradiction implies that $\lim_{t \to \infty} \nabla f(x(t)) = 0$. As a result, every accumulation point of $\{x(t) : t \ge 0\}$ is an equilibrium point of $f(x)$. This completes our proof. □

Theorem 1 guarantees that the energy function $f(x(t))$ decreases monotonically following the trajectory generated by the neural network (3) provided that $d(x(t))$ is a descent direction of $f(x)$ at $x(t)$. If (9) holds, then every accumulation point of the trajectory is an equilibrium point of $f(x)$. Similarly to nonlinear optimization, condition (9) could be called as the *sufficient descent condition*. It is obvious that the gradient neural network NNI satisfies the sufficient descent condition. Thus, one direct corollary of Theorem 1 is that every accumulation point of the trajectory generated by the neural network NNI is a stationary point of $f(x)$.

Now, we turn to analyze the asymptotic properties of the general neural network (3). From Theorem 1, we can easily see that if $f(x)$ has continuous nonsingular Hessians in R^n, then the trajectory $\{x(t) : t \ge 0\}$ has exactly one equilibrium point as its accumulation point. In this case, for any x_0, there exists a unique x^* such that

$$\lim_{t \to \infty} x(t) = x^* \quad \text{and} \quad \nabla f(x^*) = 0. \tag{19}$$

Although this may not be true for general functions, we assume that (19) always holds in the following theorem.

To establish the following theorem, we need to assume that

$$\|d(x(t))\| \leq \bar{\gamma}\|\nabla f(x(t))\| \tag{20}$$

holds for some constant $\bar{\gamma} > 0$ and all $t \geq 0$. Here, we should note that condition (20) is not very restrictive. In fact, in most cases, $d(x(t))$ can be written as

$$d(x(t)) = -P(x(t))\nabla f(x(t)), \tag{21}$$

where $P(x(t)) \in R^{n\times n}$ is some matrix. Thus, (20) holds if $P(x(t))$ is continuous in $x(t)$ and $\mathcal{L}(x_0)$ is bounded for any $x_0 \in R^n$.

Theorem 2 *Suppose that Assumptions 1 and 2 hold. Consider the neural network (3), where $d(x(t))$ satisfies (20). If (19) holds, there exist positive constants c_1, c_2, c_3, and η_1 such that for all $t \geq 0$,*

$$\|x(t) - x^*\| \geq \|x(0) - x^*\|e^{-c_1 t}, \tag{22}$$

$$\|\nabla f(x(t))\| \geq \|\nabla f(x(0))\|e^{-c_2 t}, \tag{23}$$

and in the case that (9) holds,

$$f(x(t)) - f(x^*) \geq \eta_1(f(x(0) - f(x^*))e^{-c_3 t}. \tag{24}$$

Proof (20) and (19) imply that $d(x^*) = 0$. Define $\phi(t) = \|x(t) - x^*\|^2$. By (3) and (5), we have that for all $t \geq 0$,

$$\frac{d\phi(t)}{dt} = -2(x(t) - x^*)^T(d(x(t)) - d(x^*)) \geq -2L_1\phi(t). \tag{25}$$

This together with $\phi(t) > 0$ yield

$$\frac{d\phi(t)}{\phi(t)} \geq -2L_1 dt. \tag{26}$$

Integrating (26) from 0 and t, we know that (22) holds with $c_1 = L_1$.

We now prove (23). Similarly to (12) and (13), we can show by (4), (3), and (20) that

$$\min_{\tau\in[t,t+c]} \|\nabla f(x(\tau))\| \geq (1 - L\bar{\gamma}c) \max_{\tau\in[t,t+c]} \|\nabla f(x(\tau))\| \tag{27}$$

holds for any $t > 0$ and $c \in (0, \frac{1}{2L\bar{\gamma}}]$, where $\bar{\gamma}$ is defined in (20). For any $t > 0$, denote $t = kc + \xi$, where k is an integer and $\xi \in [0, c)$. Then by (27),

$$\begin{aligned}\|\nabla f(x(t))\| &\geq (1 - L\bar{\gamma}c)\|\nabla f(x(kc))\| \geq \cdots \\ &\geq (1 - L\bar{\gamma}c)^{k+1}\|\nabla f(x(0))\| \geq (1 - L\bar{\gamma}c)^{\frac{t}{c}+1}\|\nabla f(x(0))\|. \end{aligned}\tag{28}$$

Letting $c \to 0$ in (28), we obtain that (23) holds with $c_2 = L\bar{\gamma}$.

Assume that (9) holds. (20) implies that

$$d(x(t))^T \nabla f(x(t)) \geq -\bar{\gamma} \|\nabla f(x(t))\|^2. \tag{29}$$

From (3), (9), (29), (23), and (19), we can get

$$\begin{aligned} f(x(t)) - f(x^*) &= -\sum_{i=0}^{\infty} \int_{t+i}^{t+i+1} d(x(\tau))^T \nabla f(x(\tau)) d\tau \\ &\geq \gamma \sum_{i=0}^{\infty} \int_{t+i}^{t+i+1} \|\nabla f(x(\tau))\|^2 d\tau \\ &\geq \gamma e^{-2c_2 t} \sum_{i=0}^{\infty} \int_{i}^{i+1} \|\nabla f(x(\tau))\|^2 d\tau \\ &\geq -\frac{\gamma}{\bar{\gamma}} e^{-c_2 t} \sum_{i=0}^{\infty} \int_{i}^{i+1} d(x(\tau))^T \nabla f(x(\tau)) d\tau \\ &= \frac{\gamma}{\bar{\gamma}} e^{-2c_2 t} (f(x(0)) - f(x^*)). \end{aligned} \tag{30}$$

Thus in the case that (9) holds, (24) is true with $\eta_1 = \frac{\gamma}{\bar{\gamma}}$ and $c_3 = 2c_2$. □

The gradient model NNI obviously satisfies (20). Thus by Theorem 2, we know that the gradient norm $\|\nabla f(x(t))\|$ generated by NNI converges to zero at most at some exponential rate.

If $f(x)$ is uniformly convex, namely, there exists a constant $\delta > 0$ such that

$$(\nabla f(x) - \nabla f(y))^T (x - y) \geq \delta \|x - y\|^2, \qquad \text{for all } x, y \in \mathcal{L}(x_0), \tag{31}$$

then, the following Theorem 3 shows that the gradient norm $\|\nabla f(x(t))\|$ generated by the neural network (3) with $d(x(t))$ satisfying (9) converges to zero at some exponential rate exactly.

Theorem 3 *Suppose that Assumptions 1 and 2, and (31) hold. Assume that x^* is the unique minimizer of $f(x)$. If (9) holds, there exist positive constants c_4, c_5, c_6, η_2, and η_3 such that for all $t \geq 0$,*

$$\|x(t) - x^*\| \leq \eta_2 \|x(0) - x^*\| e^{-c_4 t}, \tag{32}$$

$$\|\nabla f(x(t))\| \leq \eta_3 \|\nabla f(x(0)\| e^{-c_5 t}, \tag{33}$$

and

$$f(x(t)) - f(x^*) \leq (f(x(0) - f(x^*)) e^{-c_6 t}. \tag{34}$$

Proof From (31) and Assumptions 1 and 2, it is easy to show that

$$\frac{1}{2}\delta\|x(t)-x^*\|^2 \le f(x(t))-f(x^*) \le \frac{1}{2}L\|x(t)-x^*\|^2 \tag{35}$$

and

$$\delta\|x(t)-x^*\| \le \|\nabla f(x(t))\| \le L\|x(t)-x^*\|. \tag{36}$$

Thus if (34) holds, we see that (32) and (33) hold with $c_4 = c_5 = \frac{c_6}{2}$, $\eta_2 = \sqrt{\frac{L}{\delta}}$ and $\eta_3 = (\frac{L}{\delta})^{3/2}$.

Denote $\psi(t) = f(x(t)) - f(x^*)$. It follows from (3), (9), (36), and (35) that

$$\frac{d\psi(t)}{dt} = d(x(t))^T\nabla f(x(t)) \le -\gamma\|\nabla f(x(t))\|^2 \le -\frac{2\delta^2\gamma}{L}\psi(t). \tag{37}$$

This together with $\psi(t) > 0$ imply

$$\frac{d\psi(t)}{\psi(t)} \le -\frac{2\delta^2\gamma}{L}dt. \tag{38}$$

Thus integrating (38) from 0 to t, we know that (34) holds with $c_6 = \frac{2\delta^2\gamma}{L}$. This completes our proof. □

3 A First-Order Time-Delay Neural Network

As mentioned in Sect. 1, the gradient neural network is commonly used to solve problem (P). However, this neural network sometimes performs very poorly. To improve the performance of this neural network, one way is to set the scalar μ in (1) to be very large. In fact, considering the scalar μ, it is not difficult to see that relation (33) should be changed to

$$\|\nabla f(x(t))\| \le \eta_3\|\nabla f(x(0)\|e^{-\mu c_5 t}. \tag{39}$$

Thus to achieve the same tolerance error, the simulation time t could be reduced if the value of μ is large.

However, the value of μ cannot be arbitrarily large due to the hardware limit. Therefore to reduce the simulation time t, we could consider different $d(x(t))$ in (3) to increase the value of c_5 in (39). For illustration purpose, let us assume that $\mu = 1$ and $f(x)$ is given by

$$f(x) = \frac{1}{2}x^T\Lambda x, \qquad x \in R^n, \tag{40}$$

where Λ is a diagonal matrix whose diagonal elements are $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n > 0$. Given the initial point $x(0) = 1$, one can easily obtain for the neural network NNI

$$\nabla f(x(t)) = (\lambda_1 e^{-\lambda_1 t}, \lambda_2 e^{-\lambda_2 t}, \ldots, \lambda_n e^{-\lambda_n t})^T. \tag{41}$$

Thus in order to satisfy

$$\|\nabla f(x(t))\|_\infty \leq \epsilon, \tag{42}$$

where $\epsilon \ll 1$ is the tolerance error, we must have

$$t \geq -\frac{1}{\lambda_n} \ln \frac{\epsilon}{\lambda_n}. \tag{43}$$

This relation shows that the neural network NNI can be very slow if the least eigenvalue λ_n is very small. At the same time, if Newton's model

$$\frac{dx(t)}{dt} = -[\nabla^2_{xx} f(x(t))]^{-1} \nabla f(x(t)), \qquad t \geq 0 \tag{44}$$

is applicable, we can find

$$\nabla f(x(t)) = (\lambda_1 e^{-t}, \lambda_2 e^{-t}, \ldots, \lambda_n e^{-t})^T, \tag{45}$$

which implies that to satisfy (42), we only need

$$t \geq -\ln \frac{\epsilon}{\lambda_1}. \tag{46}$$

The above relation is clearly much better than (43) if λ_n is very small.

For a general objective function, however, its Hessian may not be available or positive definite. Even the Hessian is positive definite, it may still be very difficult to compute its inverse (see [21]). Therefore, if we want to shorten the simulation time t along the above line, we have to consider to use some matrix, say P, to approximate the inverse of the Hessian, that is, $P \approx [\nabla^2_{xx} f(x(t))]^{-1}$.

To do so, we consider a neural network with time delay. Assume that at time t, the information at times $t - \triangle t$ and $t - 2\triangle t$ is available. Let

$$s = x(t - \triangle t) - x(t - 2\triangle t) \tag{47}$$

and

$$y = \nabla f(x(t - \triangle t)) - \nabla f(x(t - 2\triangle t)). \tag{48}$$

Now we consider the matrix

$$P = I + \frac{ss^T}{s^T y}. \tag{49}$$

It is easy to see that P is positive definite if $s^T y > 0$, and besides the eigenvalue 1, P has an eigenvalue $1 + \frac{s^T s}{s^T y}$. By the mean value theorem, there exists an $\xi \in (0, 1)$ such that

$$\frac{ss^T}{s^T y} = \frac{ss^T}{s^T \nabla^2_{xx} f(x(t - (1 + \xi)\triangle t))s}, \tag{50}$$

then the matrix given by (50) can be regarded as some kind of approximation to the inverse Hessian.

Before adopting the direction $d(x(t)) = -P(x(t))\nabla f(x(t))$ in (3), we should provide some safe measure. It is known that $s^T y > 0$ if $f(x)$ is strictly convex. However, this relation is not true for general objective functions. Thus, we need to check the following condition

$$s^T y \geq \epsilon, \tag{51}$$

where ϵ is a given positive constant. If (51) is not satisfied, the $d(x(t))$ is changed to the negative gradient direction, that is, $d(x(t)) = -\nabla f(x(t))$.

Now, we define the new neural network as neural network **NNII** given by

$$\frac{dx(t)}{dt} = -\mu \begin{cases} (I + \frac{ss^T}{s^T y})\nabla f(x(t)) & \text{if } s^T y \geq \varepsilon, \\ \nabla f(x(t)) & \text{otherwise,} \end{cases} \qquad t \geq 0. \tag{52}$$

In theory, to have a continuous neural network, we may have the following modified neural network

$$\frac{dx(t)}{dt} = -\mu \begin{cases} (I + \frac{ss^T}{s^T y})\nabla f(x(t)) & \text{if } s^T y \geq \varepsilon, \\ (I + \frac{ss^T}{\varepsilon})\nabla f(x(t)) & \text{otherwise,} \end{cases} \qquad t \geq 0. \tag{53}$$

For this modified neural network, we have from Assumption 1 that

$$-\|\nabla f(x(t))\|^2 \geq d(x(t))^T \nabla f(x(t)) \geq -(1 + \frac{4B^2}{\epsilon})\|\nabla f(x(t))\|^2, \tag{54}$$

where B is the bound of the level set $\mathcal{L}(x_0)$ and ϵ is the constant in (51). Thus by Theorem 1, if $\nabla f(x_0) \neq 0$, every accumulation point of the trajectory $\{x(t) : t \geq 0\}$ generated by this modified neural network must be a stationary point of $f(x)$.

Nevertheless, since there is not much difference between (52) and (53), we will implement the simple neural network NNII (52) on circuit. Besides the simple circuit components, we also need

(1) time-delay components to form s in (47) and y in (48);
(2) a simple limiter to check if $s^T y \geq \varepsilon$ (see Fig. 1a);
(3) a comparator to include both cases in (52) (see Fig. 1b);
(4) an activation function to implement $\frac{1}{s^T y}$ (see Fig. 1c).

The circuit symbols for the last three components are shown in Fig. 1.

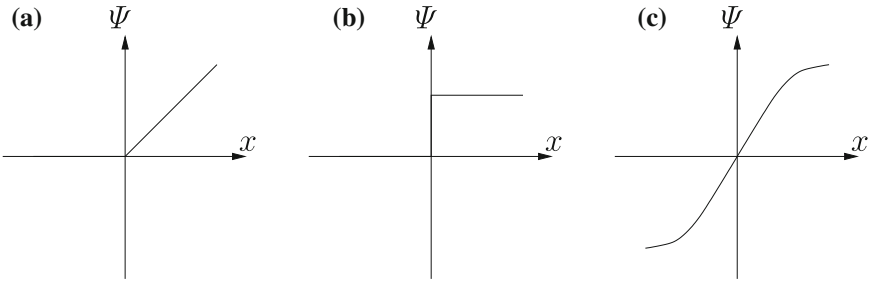


Fig. 1 **a** Simple limiter, **b** Comparator, **c** Activation function

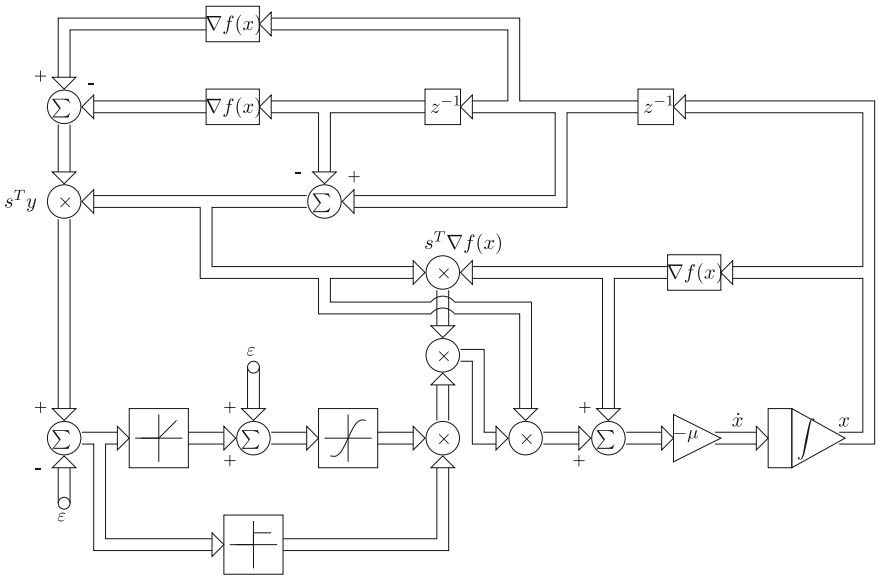


Fig. 2 A block diagram for the time-delay neural network NNII (52)

A complete neural network diagram for NNII is illustrated in Fig. 2.

To end this section, we shall make an explanation for why the proposed extension of the Barzilai–Borwein gradient method [1] to the efficient. Consider again the ordinary gradient neural network (NNI) for minimizing the strictly convex quadratic in (40) with $x(0) = 1$. In this case, the solution is

$$x_i(t) = e^{-\lambda_i t}, \qquad \text{for } i \geq 1. \tag{55}$$

Since the optimal solution is exactly zero, we can see from the above that the components corresponding to those small eigenvalues will converge to zero very slowly. Now, let us consider the unmodified extension of the Barzilai–Borwein gradient neural network

$$\frac{dx(t)}{dt} = -\left(I + \frac{ss^T}{s^T y}\right)\nabla f(x(t)), \quad x(0) = 1, \quad t \geq 0,$$

and consider the special case that s is a multiple of some eigenvector, namely, $s \parallel e_i$ for some i. Then the ith component of the solution corresponding to this Barzilai–Borwein gradient neural network will be

$$x_i(t) = e^{-(1+\lambda_i)t}, \quad t \geq 0.$$

Therefore, we can see that the Barzilai–Borwein extension of the gradient neural network can accelerate the convergence of the components to the small eigenvalues even if $\lambda_i > 0$ is very small. In addition, we can also see that the addition of the identity matrix is also important.

4 Simulation

In this section, the two neural networks NNI and NNII will be simulated on two examples. Our simulations are conducted with MATLAB. The numerical ordinary differential equation solver used in all simulations is ode23s. The parameter μ is fixed at $\mu = 1000$ in all simulation. For NNII, the value of ϵ in (51) is set at $\epsilon = 10^{-12}$. The stopping criterion used in the simulation for both neural networks is

$$\|\nabla f(x(t))\| \leq 10^{-6}. \tag{56}$$

Theorem 1 and the related discussions in the last section guarantee that the stopping condition (56) can be met in a finite time period. We use t_f to denote the final time when (56) is satisfied.

Example 1 A quadratic problem: Suppose that the function $f(x)$ is of the form

$$f(x) = \frac{1}{2}x\left[\begin{pmatrix} 1 & 0 \\ 0 & \lambda \end{pmatrix}\right]x, \quad x \in R^2,$$

where $\lambda > 0$ is a scalar, and that $x(0) = 1$. The simulation results of Example 1 with three different λ values for both neural networks are reported in Table 1.

Example 1 is designed to verify our analysis at the beginning of Sect. 3. From Table 1, we can see that for $\lambda = 1$ and $\lambda = 10^4$, the neural network NNII is at least two times faster than the neural network NNI. For a very small $\lambda = 10^{-4}$, the neural network NNII is about 3000 times faster than the neural network NNI. The simulation results in Table 1 strongly confirm our theoretical analysis in Sect. 3.

Table 1 Simulation results of Example 1

		$\lambda = 0.0001$	$\lambda = 1$	$\lambda = 10{,}000$
NNI	t_f	4.65e + 01	1.49e − 02	1.43e − 02
	$f(x(t_f))$	4.47e − 09	5.91e − 14	1.30e − 13
	$\|\|\nabla f(x(t_f))\|\|_2$	9.46e − 07	3.44e − 07	5.10e − 07
NNII	t_f	1.37e − 02	7.13e − 03	6.87e − 03
	$f(x(t_f))$	1.34e − 09	2.59e − 13	4.33e − 13
	$\|\|\nabla f(x(t_f))\|\|_2$	8.45e − 07	7.20e − 07	9.31e − 07

Table 2 Simulation results of Example 2

	t_f	8.53e + 02
NNI	$f(x(t_f))$	1.56e − 05
	$\|\|\nabla f(x(t_f))\|\|_2$	1.95e − 07
NNII	t_f	3.08e − 02
	$f(x(t_f))$	1.56e − 05
	$\|\|\nabla f(x(t_f))\|\|_2$	4.43e − 07

Example 2 Penalty function (Problem (23) in [17])

$$\begin{cases} f(x) = 10^{-5} \sum_{i=1}^{8} (x_i - 1)^2 + [(\sum_{i=1}^{8} x_i^2) - \frac{1}{4}]^2, \\ x_0 = (1, 2, \ldots, 8)^T, \ f = 1.5557 \ldots 10^{-5} \text{ at } (0.5590 \ldots, \ldots, 0.5590 \ldots)^T. \end{cases}$$

Example 2 is a very nonlinear problem, and the simulation results in Table 2 indicate that the neural network NNII is over 20,000 times faster than the neural network NNI. The evolutions of the energy function $f(x(t))$ for both neural network models are illustrated in Fig. 3 and Fig. 4, respectively.

5 Concluding Remarks

Many existing neural network models are based on the transformation of converting constrained problems into unconstrained problems. Therefore, it is not only very important but also very practical to have an efficient neural network model for unconstrained problems. However, the commonly adopted gradient neural network model for unconstrained problems could be very slow in many circumstances. In this paper, a new time-delay neural network model is proposed for unconstrained problems. The simulation results on the two examples indicate that the new neural network outperforms the gradient neural network and is quite efficient and robust.

Our theoretical analysis guarantees that the new neural network model is globally convergent to an equilibrium point or stationary point. In addition, we also present

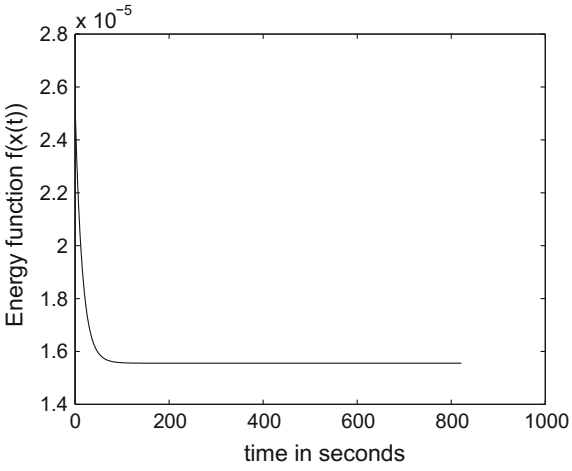


Fig. 3 Evolution of $f(x(t))$ by NNI for Example 2

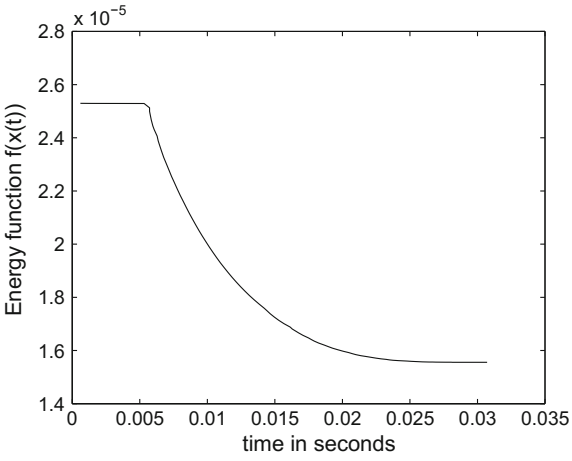


Fig. 4 Evolution of $f(x(t))$ by NNII for Example 2

some convergence and bound results on the trajectory and energy function for general neural network models. Based on our analysis, some new neural network models could be developed to achieve even better results.

This is an early research report when the first author was visiting Lizhi Liao at the beginning of 1998. There have been a lot of research works on neural networks with delays on the first decade of this century; for example, see [27], [30] and the references therein. Nevertheless, we still think that the proposed neural network, (52), is a simple but an efficient neural network with delays. It can be regarded an useful extension of the Barzilai–Borwein gradient method [1] to the neural network.

References

1. Barzilai, J., Borwein, J.M.: Two-point step size gradient methods. IMA J. Numer. Anal. **8**(1), 141–148 (1988)
2. Bouzerdoum, A., Pattison, T.R.: Neural network for quadratic optimization with bound constraints. IEEE Trans. Neural Netw. **4**, 293–304 (1993)
3. Chen, Y.H., Fang, S.C.: Solving convex programming problem with equality constraints by neural networks. Comput. Math. Appl. **36**, 41–68 (1998)
4. Cichocki, A., Unbehauen, R.: Neural Networks for Optimization and Signal Processing. Wiley, Stuttgart (1993)
5. Cichocki, A., Unbehauen, R., Weinzierl, K., Holzel, R.: A new neural network for solving linear programming problems. Eur. J. Oper. Res. **93**, 244–256 (1996)
6. Chua, L.O., Lin, G.-N.: Nonlinear programming without computation. IEEE Trans. Circ. Syst. **31**, 182–188 (1984)
7. Hopfield, J.J.: Neural networks and physical systems with emergent collective computational ability. Proc. Natl. Acad. Sci. **79**, 2554–2558 (1982)
8. Hopfield, J.J.: Neurons with graded response have collective computational properties like those of two-state neurons. Proc. Natl. Acad. Sci. **81**, 3088–3092 (1984)
9. Hopfield, J.J., Tank, D.W.: Neural computation of decisions in optimization problems. Biol. Cybern. **52**, 141–152 (1985)
10. Hou, Z.-G., Wu, C.-P., Bao, P.: A neural network for hierarchical optimization of nonlinear large-scale systems. Int. J. Syst. Sci. **29**(2), 159–166 (1998)
11. Karmarkar, N.: A new polynomial-time algorithm for linear programming. Combinatorica **4**, 373–395 (1984)
12. Kennedy, M.P., Chua, L.O.: Neural networks for nonlinear programming. IEEE Trans. Circ. Syst. **35**, 554–562 (1988)
13. Liao, L.-Z., Qi, H.D., Qi, L.Q.: Neurodynamic optimization. J. Glob. Optim. **28**, 175–195 (2004)
14. Lillo, W.E., Loh, M.-H., Hui, S., Żak, S.H.: On solving constrained optimization problems with neural networks: a penalty method approach. IEEE Trans. Neural Netw. **4**, 931–940 (1993)
15. Maa, C.Y., Shanblatt, M.A.: Linear and quadratic programming neural network analysis. IEEE Trans. Neural Netw. **3**(4), 580–594 (1992)
16. Maa, C.Y., Shanblatt, M.A.: A two-phase optimization neural network. IEEE Trans. Neural Netw. **3**, 1003–1009 (1992)
17. Moré, J.J., Garbow, B.S., Hillstrom, K.E.: Testing unconstrained optimization software. ACM Trans. Math. Softw. **7**, 17–41 (1981)
18. Qian X., Liao L.-Z., Sun J., Zhu H.: The convergent generalized central paths for linearly constrained convex programming. SIAM J. Optim. (2018) (to appear)
19. Rodríguez-Vázquez, A., Domínguez-Castro, R., Rueda, A., Huertas, J.L., Sánchez-Sinencio, E.: Nonlinear switched-capacitor 'neural' networks for optimization problems. IEEE Trans. Circ. Syst. **37**, 384–398 (1990)
20. Tank, D.W., Hopfield, J.J.: Simple 'neural' optimization network: an A/D converter, signal decision circuit, and a linear programming circuit. IEEE Trans. Circ. Syst. **33**, 533–541 (1986)
21. Wang, J.: Recurrent neural networks for computing pseudoinverses of rank-deficient matrices. SIAM J. Sci. Comput. **18**, 1479–1493 (1997)
22. Wang, J., Xia, Y.: Analysis and design of primal-dual assignment networks. IEEE Trans. Neural Netw. **9**, 183–194 (1998)
23. Wu, X.-Y., Xia, Y., Li, J., Chen, W.-K.: A high-performance neural network for solving linear and quadratic programming problems. IEEE Trans. Neural Netw. **7**, 643–651 (1996)
24. Xia, Y.: A new neural network for solving linear and quadratic programming problems. IEEE Trans. Neural Netw. **7**, 1544–1547 (1996)
25. Zak, S.H., Upatising, V., Lillo, W.E., Hui, S.: A dynamical systems approach to solving linear programming problems. In: Elworthy, K.D., Everitt, W.N., Lee, E.B. (eds.) Differential Equations, Dynamical Systems, and Control Science, pp. 913–946. Marcel Dekker, New York (1994)

26. Zak, S.H., Upatising, V., Hui, S.: Solving linear programming problems with neural networks: a comparative study. IEEE Trans. Neural Netw. **6**, 94–104 (1995)
27. Zhang, H., Liu, Z., Huang, G.B., Wang, Z.: Novel weighting-delay-based stability criteria for recurrent neural networks with time-varying delay. Trans. Neural Netw. **21**(1), 91–106 (2010)
28. Zhang, S., Constantinides, A.G.: Lagrange programming neural networks. IEEE Trans. Circ. Syst. 2 Analog Digital Sig. Process. **39**, 441–452 (1992)
29. Zhang, S., Zhu, X., Zou, L.-H.: Second-order neural nets for constrained optimization. IEEE Trans. Neural Netw. **3**, 1021–1024 (1992)
30. Zuo, Z., Yang, C., Wang, Y.: A new method for stability analysis of recurrent neural networks with interval time-varying delay. IEEE Trans. Neural Netw. **21**(2), 339–344 (2010)

Stabilized Optimization Via an NCL Algorithm



Ding Ma, Kenneth L. Judd, Dominique Orban and Michael A. Saunders

Abstract For optimization problems involving many nonlinear inequality constraints, we extend the bound-constrained (BCL) and linearly constrained (LCL) augmented Lagrangian approaches of LANCELOT and MINOS to an algorithm that solves a sequence of nonlinearly constrained augmented Lagrangian subproblems whose nonlinear constraints satisfy the LICQ everywhere. The NCL algorithm is implemented in AMPL and tested on large instances of a tax policy model that could not be solved directly by the state-of-the-art solvers that we tested, because of singularity in the Jacobian of the active constraints. Algorithm NCL with IPOPT as subproblem solver proves to be effective, with IPOPT using second derivatives and successfully warm starting each subproblem.

Keywords Stabilized optimization · LICQ · Augmented Lagrangian · BCL NCL · Interior method · Warm start

D. Ma
Management Science and Engineering,
Stanford University, Stanford, CA, USA
e-mail: dingma@stanford.edu

K. L. Judd
Hoover Institution, Stanford University, Stanford, CA, USA
e-mail: judd@stanford.edu

D. Orban
GERAD and Department of Mathematics and Industrial Engineering,
École Polytechnique, Montréal, QC, Canada
e-mail: dominique.orban@gerad.ca

M. A. Saunders (✉)
Department of Management Science and Engineering,
Stanford University, Stanford, CA, USA
e-mail: saunders@stanford.edu

© Springer International Publishing AG, part of Springer Nature 2018

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*,
Springer Proceedings in Mathematics & Statistics 235,
https://doi.org/10.1007/978-3-319-90026-1_8

1 Introduction

We consider constrained optimization problems of the form

$$\begin{array}{ll} \text{NCO} & \underset{x\in\mathbb{R}^n}{\text{minimize}} \;\; \phi(x) \\ & \text{subject to} \;\; c(x) \geq 0, \quad Ax \geq b, \quad \ell \leq x \leq u, \end{array}$$

where $\phi(x)$ is a smooth nonlinear function, $c(x) \in \mathbb{R}^m$ is a vector of smooth nonlinear functions, and $Ax \geq b$ is a placeholder for a set of linear inequality or equality constraints, with x lying between lower and upper bounds ℓ and u.

In some applications where $m \gg n$, there may be more than n constraints that are essentially active at a solution. The constraints do not satisfy the linear independence constraint qualification (LICQ), and general-purpose solvers are likely to have difficulty converging. Some form of regularization is required. The stabilized SQP methods of Wright [20] and Gill et al. [9, 10] have been developed specifically for such problems. We achieve reliability more simply by adapting the augmented Lagrangian algorithm of the general-purpose optimization solver LANCELOT [4, 5, 15] in the vein of Arreckx and Orban [2] to derive a sequence of regularized subproblems denoted in the next section by NC_k.

2 BCL, LCL, and NCL Methods

The theory for the large-scale solver LANCELOT is best described in terms of the general optimization problem

$$\begin{array}{ll} \text{NECB} & \underset{x\in\mathbb{R}^n}{\text{minimize}} \;\; \phi(x) \\ & \text{subject to} \;\; c(x) = 0, \quad \ell \leq x \leq u \end{array}$$

with *nonlinear equality constraints* and bounds. We let x^* denote a local solution of NECB and (y^*, z^*) denote associated multipliers. LANCELOT treats NECB by solving a sequence of *bound-constrained subproblems* of the form

$$\begin{array}{ll} \text{BC}_k & \underset{x}{\text{minimize}} \;\; L(x, y_k, \rho_k) = \phi(x) - y_k^T c(x) + \frac{1}{2}\rho_k \|c(x)\|^2 \\ & \text{subject to} \;\; \ell \leq x \leq u, \end{array}$$

where y_k is an estimate of the Lagrange multipliers y^* for the equality constraints. This was called a bound-constrained Lagrangian (BCL) method by Friedlander and Saunders [8], in contrast to the linearly constrained Lagrangian methods (LCL) of Robinson [18] and MINOS [16], whose subproblems LC_k contain bounds as in BC_k

and also linearizations of the equality constraints at the current point x_k (including linear constraints).

In order to treat NCO with a sequence of BC_k subproblems, we convert the nonlinear inequality constraints to equalities to obtain

$$\begin{array}{lll} \text{NCO}' & \underset{x,\,s}{\text{minimize}} & \phi(x) \\ & \text{subject to} & c(x) - s = 0, \quad Ax \geq b, \quad \ell \leq x \leq u, \quad s \geq 0 \end{array}$$

with corresponding subproblems (including linear constraints)

$$\begin{array}{lll} \text{BC}_k{}' & \underset{x,\,s}{\text{minimize}} & L(x, y_k, \rho_k) = \phi(x) - y_k^T(c(x) - s) + \frac{1}{2}\rho_k \|c(x) - s\|^2 \\ & \text{subject to} & Ax \geq b, \quad \ell \leq x \leq u, \quad s \geq 0. \end{array}$$

We now introduce variables $r = -(c(x) - s)$ into $\text{BC}_k{}'$ to obtain the *nonlinearly constrained Lagrangian* (NCL) subproblem

$$\begin{array}{lll} \text{NC}_k & \underset{x,\,r}{\text{minimize}} & \phi(x) + y_k^T r + \frac{1}{2}\rho_k \|r\|^2 \\ & \text{subject to} & c(x) + r \geq 0, \quad Ax \geq b, \quad \ell \leq x \leq u, \end{array}$$

in which r serves to make the nonlinear constraints independent. (If NCO includes an equality $c_i(x) = 0$, NC_k would contain $c_i(x) + r_i = 0$.) Assuming existence of finite multipliers and feasibility, for $\rho_k > 0$ and larger than a certain finite value, the NCL subproblems should cause y_k to approach y^* and most of the solution $(x_k^*, r_k^*, y_k^*, z_k^*)$ of NC_k to approach (x^*, y^*, z^*), with r_k^* approaching zero.

Problem NC_k is analogous to Friedlander and Orban's formulation for convex quadratic programs [7, Eq. (3.2)]. See also Arreckx and Orban [2], where the motivation is the same as here, achieving reliability when the nonlinear constraints do not satisfy LICQ.

Note that for general problems NECB, the BCL and LCL subproblems contain linear constraints (bounds only, or linearized constraints and bounds). Our NCL formulation retains nonlinear constraints in the NC_k subproblems, but simplifies them by ensuring that they satisfy LICQ. On large problems, the additional variables $r \in \mathbb{R}^m$ in NC_k may be detrimental to active-set solvers like MINOS or SNOPT [11] because they increase the number of degrees of freedom (superbasic variables). Fortunately, they are easily accommodated by interior methods, as our numerical results show for IPOPT [12, 19]. We expect the same to be true for KNITRO [3, 14]. These solvers are most effective when second derivatives are available, as they are for our AMPL model.

2.1 The BCL Algorithm

The LANCELOT BCL method is summarized in Algorithm BCL. Each subproblem BC_k is solved with a specified optimality tolerance ω_k, generating an iterate x_k^* and the associated Lagrangian gradient $z_k^* \equiv \nabla L(x_k^*, y_k, \rho_k)$. If $\|c(x_k^*)\|$ is sufficiently small, the iteration is regarded as "successful" and an update to y_k is computed from x_k^*. Otherwise, y_k is not altered but ρ_k is increased.

Key properties are that the subproblems are solved inexactly, the penalty parameter is increased only finitely often, and the multiplier estimates y_k need not be assumed bounded. Under certain conditions, all iterations are eventually successful, the ρ_k's remain constant, the iterates converge superlinearly, and the algorithm terminates in a finite number of iterations [4].

Algorithm 1 BCL (Bound-Constrained Lagrangian Method for NECB)

1: **procedure** BCL(x_0, y_0, z_0)
2: Set penalty parameter $\rho_1 > 0$, scale factor $\tau > 1$, and constants $\alpha, \beta > 0$ with $\alpha < 1$.
3: Set positive convergence tolerances $\eta_*, \omega_* \ll 1$ and infeasibility tolerance $\eta_1 > \eta_*$.
4: $k \leftarrow 0$, converged $\leftarrow$ false
5: **repeat**
6: $k \leftarrow k + 1$
7: Choose optimality tolerance $\omega_k > 0$ such that $\lim_{k\to\infty} \omega_k \leq \omega_*$.
8: Find (x_k^*, z_k^*) that solves BC_k to within ω_k.
9: **if** $\|c(x_k^*)\| \leq \max(\eta_*, \eta_k)$ **then**
10: $y_k^* \leftarrow y_k - \rho_k c(x_k^*)$
11: $x_k \leftarrow x_k^*,\ y_k \leftarrow y_k^*,\ z_k \leftarrow z_k^*$ *update solution estimates*
12: **if** (x_k, y_k, z_k) solves NECB to within ω_*, converged $\leftarrow$ true
13: $\rho_{k+1} \leftarrow \rho_k$ *keep* ρ_k
14: $\eta_{k+1} \leftarrow \eta_k/(1 + \rho_{k+1}^{\beta})$ *decrease* η_k
15: **else**
16: $\rho_{k+1} \leftarrow \tau\rho_k$ *increase* ρ_k
17: $\eta_{k+1} \leftarrow \eta_0/(1 + \rho_{k+1}^{\alpha})$ *may increase or decrease* η_k
18: **end if**
19: **until** converged
20: $x^* \leftarrow x_k,\ y^* \leftarrow y_k,\ z^* \leftarrow z_k$
21: **end procedure**

Note that at step 8 of Algorithm BCL, the inexact minimization would typically use the initial guess (x_k^*, z_k^*). However, other initial points are possible. At step 12, we say that (x_k, y_k, z_k) solves NECB to within ω_* if the largest dual infeasibility is smaller than ω_*.

Algorithm 2 NCL (Nonlinearly Constrained Lagrangian Method for NCO)

procedure NCL(x_0, r_0, y_0, z_0)
  Set penalty parameter $\rho_1 > 0$, scale factor $\tau > 1$, and constants $\alpha, \beta > 0$ with $\alpha < 1$.
  Set positive convergence tolerances $\eta_*, \omega_* \ll 1$ and infeasibility tolerance $\eta_1 > \eta_*$.
  $k \leftarrow 0$, converged $\leftarrow$ false
  repeat
    $k \leftarrow k + 1$
    Choose optimality tolerance $\omega_k > 0$ such that $\lim_{k\to\infty} \omega_k \leq \omega_*$.
    Find $(x_k^*, r_k^*, y_k^*, z_k^*)$ that solves NC_k to within ω_k.
    if $\|r_k^*\| \leq \max(\eta_*, \eta_k)$ **then**
      $y_k^* \leftarrow y_k + \rho_k r_k^*$
      $x_k \leftarrow x_k^*,\ r_k \leftarrow r_k^*,\ y_k \leftarrow y_k^*,\ z_k \leftarrow z_k^*$ *update solution estimates*
      if (x_k, y_k, z_k) solves NCO to within ω_*, converged $\leftarrow$ true
      $\rho_{k+1} \leftarrow \rho_k$ *keep* ρ_k
      $\eta_{k+1} \leftarrow \eta_k/(1 + \rho_{k+1}^{\beta})$ *decrease* η_k
    else
      $\rho_{k+1} \leftarrow \tau\rho_k$ *increase* ρ_k
      $\eta_{k+1} \leftarrow \eta_0/(1 + \rho_{k+1}^{\alpha})$ *may increase or decrease* η_k
    end if
  until converged
  $x^* \leftarrow x_k,\ r^* \leftarrow r_k,\ y^* \leftarrow y_k,\ z^* \leftarrow z_k$
end procedure

2.2 The NCL Algorithm

To derive a stabilized algorithm for problem NCO, we modify Algorithm BCL by introducing r and replacing the subproblems BC_k by NC_k. The resulting method is summarized in Algorithm NCL. The update to y_k becomes $y_k^* \leftarrow y_k - \rho_k(c(x_k^*) - s_k^*) = y_k + \rho_k r_k^*$, the value satisfied by an optimal y_k^* for subproblem NC_k. Step 8 of Algorithm NCL would typically use $(x_k^*, r_k^*, y_k^*, z_k^*)$ as initial guess, and that is what we use in our implementation below.

3 An Application: Optimal Tax Policy

Some challenging test cases arise from the tax policy models described in [13]. With $x = (c, y)$, they take the form

$$\begin{array}{lll} \text{TAX} & \underset{c,\,y}{\text{maximize}} & \sum_i \lambda_i U^i(c_i, y_i) \\ & \text{subject to} & U^i(c_i, y_i) - U^i(c_j, y_j) \geq 0 \quad \text{for all } i, j \\ & & \lambda^T(y - c) \geq 0 \\ & & c,\ y \geq 0, \end{array}$$

where c_i and y_i are the consumption and income of taxpayer i, and λ is a vector of positive weights. 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 $T := na \times nb \times nc \times nd \times ne$.

Table 1 summarizes results for a 4D example ($ne = 1$ and $\gamma_1 = 1$). The first term of $U(c, y)$ becomes $\log(c - \alpha)$, the limit as $\gamma \to 1$. Problem NCO and Algorithm NCL were formulated in the AMPL modeling language [6]. The solvers SNOPT [11] and IPOPT [19] were unable to solve NCO itself, but Algorithm NCL was successful with IPOPT solving the subproblems NC_k. We use a default configuration of IPOPT with MUMPS [1] as symmetric indefinite solver to compute search directions. We set the optimality tolerance for IPOPT to $\omega_k = \omega_* = 10^{-6}$ throughout and specified warm starts for $k \geq 2$ using options warm_start_init_point=yes and mu_init=1e-4. These options greatly improved the performance of IPOPT on each subproblem compared to cold starts, for which mu_init=0.1. It is helpful that only the objective function of NC_k changes with k.

Table 1 NCL results on a 4D example with $na, nb, nc, nd = 11, 3, 3, 2$, giving $m = 39006$, $n = 395$. Itns refers to IPOPT's primal-dual interior point method, and time is seconds on an Apple iMac with 2.93 GHz Intel Core i7

k	ρ_k	η_k	$\|r_k^*\|_\infty$	$\phi(x_k^*)$	Itns	Time
1	10^2	10^{-2}	3.1e − 03	−2.1478532e + 01	125	42.8
2	10^2	10^{-3}	1.3e − 03	−2.1277587e + 01	18	6.5
3	10^3	10^{-3}	6.6e − 04	−2.1177152e + 01	27	9.1
4	10^3	10^{-4}	5.5e − 04	−2.1110210e + 01	31	10.8
5	10^4	10^{-4}	2.9e − 04	−2.1066664e + 01	57	24.3
6	10^5	10^{-4}	6.5e − 05	−2.1027152e + 01	75	26.8
7	10^5	10^{-5}	5.2e − 05	−2.1018896e + 01	130	60.9
8	10^6	10^{-5}	9.3e − 06	−2.1015295e + 01	159	81.8
9	10^6	10^{-6}	2.0e − 06	−2.1014808e + 01	139	70.0
10	10^7	10^{-6}	2.1e − 07	−2.1014800e + 01	177	97.6

For this example, problem NCO has $m = 39{,}006$ nonlinear inequality constraints and one linear constraint in $n = 395$ variables $x = (c, y)$, and nonnegativity bounds. Subproblem NC_k has 39,007 constraints and 39,402 variables when r is included. Fortunately, r does not affect the complexity of each IPOPT iteration, but greatly improves stability. In contrast, active-set methods like MINOS and SNOPT are very inefficient on the NC_k subproblems because the large number of inequality constraints leads to thousands of minor iterations, and the presence of r (with no bounds) leads to thousands of superbasic variables. About $3.2n$ constraints were within 10^{-6} of being active.

Table 2 summarizes results for a 5D example. The NC_k subproblems have $m = 32{,}220$ nonlinear constraints and $n = 360$ variables, leading to 32,581 variables including r. Again the options warm_start_init_point=yes and mu_init=1e-4 for $k \geq 2$ led to good performance by IPOPT on each subproblem. About $3n$ constraints were within 10^{-6} of being active.

Table 2 NCL results on a 5D example with $na, nb, nc, nd, ne = 5, 3, 3, 2, 2$, giving $m = 32220$, $n = 360$

k	ρ_k	η_k	$\|r_k^*\|_\infty$	$\phi(x_k^*)$	Itns	Time
1	10^2	10^{-2}	7.0e − 03	−4.2038075e + 02	95	41.1
2	10^2	10^{-3}	4.1e − 03	−4.2002898e + 02	17	7.2
3	10^3	10^{-3}	1.3e − 03	−4.1986069e + 02	20	8.1
4	10^4	10^{-3}	4.4e − 04	−4.1972958e + 02	48	25.0
5	10^4	10^{-4}	2.2e − 04	−4.1968646e + 02	43	20.5
6	10^5	10^{-4}	9.8e − 05	−4.1967560e + 02	64	32.9
7	10^5	10^{-5}	6.6e − 05	−4.1967177e + 02	57	26.8
8	10^6	10^{-5}	4.2e − 06	−4.1967150e + 02	87	46.2
9	10^6	10^{-6}	9.4e − 07	−4.1967138e + 02	96	53.6

Table 3 NCL results on a 5D example with $na, nb, nc, ne, ne = 21, 3, 3, 2, 2$, giving $m = 570780$, $n = 1512$

k	ρ_k	η_k	$\|r_k^*\|_\infty$	$\phi(x_k^*)$	mu_init	Itns	Time
1	10^2	10^{-2}	5.1e − 03	−1.7656816e + 03	10^{-1}	825	7763.3
2	10^2	10^{-3}	2.4e − 03	−1.7648480e + 03	10^{-4}	66	472.8
3	10^3	10^{-3}	1.3e − 03	−1.7644006e + 03	10^{-4}	106	771.3
4	10^4	10^{-3}	3.8e − 04	−1.7639491e + 03	10^{-5}	132	1347.0
5	10^4	10^{-4}	3.2e − 04	−1.7637742e + 03	10^{-5}	229	2450.9
6	10^5	10^{-4}	8.6e − 05	−1.7636804e + 03	10^{-6}	104	1096.9
7	10^5	10^{-5}	4.9e − 05	−1.7636469e + 03	10^{-6}	143	1633.4
8	10^6	10^{-5}	1.5e − 05	−1.7636252e + 03	10^{-7}	71	786.1
9	10^7	10^{-5}	2.8e − 06	−1.7636196e + 03	10^{-7}	67	725.7
10	10^7	10^{-6}	5.1e − 07	−1.7636187e + 03	10^{-8}	18	171.0

For much larger problems of this type, we found that it was helpful to reduce mu_init more often, as illustrated in Table 3. The NC_k subproblems here have $m = 570,780$ nonlinear constraints and $n = 1512$ variables, leading to 572,292 variables including r. Note that the number of NCL iterations is stable ($k \leq 10$), and IPOPT performs well on each subproblem with decreasing mu_init. This time about $6.6n$ constraints were within 10^{-6} of being active.

Note that the LANCELOT approach allows early subproblems to be solved less accurately [4]. It may save time to set $\omega_k = \eta_k$ (say) rather than $\omega_k = \omega_*$ throughout.

4 Conclusions

This work has been illuminating in several ways as we sought to improve our ability to solve examples of problem TAX.

- Small examples of the tax model solve efficiently with MINOS and SNOPT, but eventually fail to converge as the problem size increases.
- IPOPT also solves small examples efficiently, but eventually starts requesting additional memory for the MUMPS sparse linear solver. The solver may freeze, or the iterations may diverge.
- The NC_k subproblems are not suitable for MINOS or SNOPT because of the large number of variables (x, r) and the resulting number of superbasic variables (although warm starts are natural).
- It is often said that interior methods cannot be warm started. Nevertheless, IPOPT has several runtime options that have proved to be extremely helpful for implementing Algorithm NCL. For the results obtained here, it has been sufficient to say that warm starts are wanted for $k > 1$, and that the IPOPT barrier parameter should be initialized at decreasing values for later k (where only the objective of subproblem NC_k changes with k).
- The numerical examples of Sect. 3 had $3n$, $3n$, and $6.6n$ constraints essentially active at the solution, yet were solved successfully. They suggest that the NCL approach with an interior method as subproblem solver can overcome LICQ difficulties on problems that could not be solved directly.

Funding
This work was supported by the National Institute of General Medical Sciences of the National Institutes of Health [award U01GM102098] (DM and MAS) and an NSERC Discovery Grant (DO).

Acknowledgements We are extremely grateful to the developers of AMPL and IPOPT for making the development and evaluation of Algorithm NCL possible. We are especially grateful to Mehiddin Al-Baali and other organizers of the NAO-IV conference *Numerical Analysis and Optimization* at Sultan Qaboos University, Muscat, Oman, which brought the authors and AMPL developers together in January 2017. We also thank the reviewer for final helpful suggestions.

Appendix A AMPL Models, Data, and Scripts

Algorithm NCL has been implemented in the AMPL modeling language [6] and tested on problem TAX. The following sections list each relevant file. The files are available from [17].

A.1 Tax Model

File `pTax5Dncl.mod` codes subproblem NC_k for problem TAX with five parameters $w, \eta, \alpha, \psi, \gamma$, using $\mu := 1/\eta$. Note that for $U(c, y)$ in the objective and constraint functions, the first term $(c - \alpha)^{1-1/\gamma}/(1 - 1/\gamma)$ is replaced by a piecewise-smooth function that is defined for all values of c and α (see [13]).

Primal regularization $\frac{1}{2}\delta\|(c, y)\|^2$ with $\delta = 10^{-8}$ is added to the objective function to promote uniqueness of the minimizer. The vector r is called `R` to avoid a clash with subscript `r`.

```
# pTax5Dncl.mod

# Define parameters for agents (taxpayers)
param na;                  # number of types in wage
param nb;                  # number of types in eta
param nc;                  # number of types in alpha
param nd;                  # number of types in psi
param ne;                  # number of types in gamma
set A := 1..na;            # set of wages
set B := 1..nb;            # set of eta
set C := 1..nc;            # set of alpha
set D := 1..nd;            # set of psi
set E := 1..ne;            # set of gamma
set T = {A,B,C,D,E};       # set of agents

# Define wages for agents (taxpayers)
param wmin;                # minimum wage level
param wmax;                # maximum wage level
param w {A};               # i, wage vector
param mu{B};               # j, mu = 1/eta# mu vector
param mu1{B};              # mu1[j] = mu[j] + 1
param alpha{C};            # k, ak vector for utility
param psi{D};              # g
param gamma{E};            # h
param lambda{A,B,C,D,E}; # distribution density
param epsilon;
param primreg      default 1e-8;   # Small primal
                                     regularization

var y{(i,j,k,g,h) in T} >= 0.1;   # consumption for
                                     tax payer
                                     (i,j,k,g,h)
```

```
var y{(i,j,k,g,h) in T} >= 0.1;  # income  for
                                   tax payer
                                   (i,j,k,g,h)
var R{(i,j,k,g,h) in T, (p,q,r,s,t) in T:
      !(i=p and j=q and k=r and g=s and h=t)}
      >= -1e+20, <= 1e+20;

param kmax       default 20;     # limit on
                                   NCL itns
param rhok       default 1e+2;   # augmented
                                   Lagrangian penalty
                                   parameter
param rhofac     default 10.0;   # increase
                                   factor
param rhomax     default 1e+8;   # biggest rhok
param etak       default 1e-2;   # opttol for
                                   augmented
                                   Lagrangian loop
param etafac     default  0.1;   # reduction factor for
                                    opttol
param etamin     default 1e-8;   # smallest etak
param rmax       default    0;   # max r (for printing)

param rmin       default    0;   # min r (for printing)

param rnorm      default    0;   # ||r||_inf
param rtol       default 1e-6;   # quit if biggest
                                    |r_i| <= rtol

param nT         default    1;   # nT = na*nb
                                   *nc*nd*ne
param m          default    1;   # nT*(nT-1)
                                   = no. of nonlinear
                                    constraints
param n          default    1;   # 2*nT
                                   = no. of nonlinear
                                   variables

param ck{(i,j,k,g,h) in T} default 0;
              # current variable c
param yk{(i,j,k,g,h) in T} default 0;
              # current variable y
param rk{(i,j,k,g,h) in T, (p,q,r,s,t) in T:
              # current variable r = - (c(x) - s)
    !(i=p and j=q and k=r and g=s and h=t)} default 0;
param dk{(i,j,k,g,h) in T, (p,q,r,s,t) in T:
              # current dual variables (y_k)
    !(i=p and j=q and k=r and g=s and h=t)} default 0;

minimize f:
    sum{(i,j,k,g,h) in T}
```

```
    (
       (if c[i,j,k,g,h] - alpha[k] >= epsilon then
           - lambda[i,j,k,g,h] *
                ((c[i,j,k,g,h] - alpha[k])
                 ^(1-1/gamma[h]) / (1-1/gamma[h])
                 - psi[g]*(y[i,j,k,g,h]/w[i])
                 ^mu1[j] / mu1[j])
        else
           - lambda[i,j,k,g,h] *
          (-   0.5/gamma[h] * epsilon^(-1/gamma[h]-1)
           * (c[i,j,k,g,h] - alpha[k])^2
           + ( 1+1/gamma[h])* epsilon^(-1/gamma[h]  )
           * (c[i,j,k,g,h] - alpha[k])
           + (1/(1-1/gamma[h]) - 1 - 0.5/gamma[h])
           * epsilon^(1-1/gamma[h])
                 - psi[g]*(y[i,j,k,g,h]/w[i])
                 ^mu1[j] / mu1[j])
       )
    + 0.5 * primreg * (c[i,j,k,g,h]^2
    + y[i,j,k,g,h]^2)
    )
  + sum{(i,j,k,g,h) in T, (p,q,r,s,t) in
  T: !(i=p and j=q and k=r and g=s and h=t)}
        (dk[i,j,k,g,h,p,q,r,s,t]
         * R[i,j,k,g,h,p,q,r,s,t]
                    + 0.5 * rhok
                    * R[i,j,k,g,h,p,q,r,s,t]^2);

subject to

Incentive{(i,j,k,g,h) in T, (p,q,r,s,t) in T:
           !(i=p and j=q and k=r and g=s and h=t)}:
    (if c[i,j,k,g,h] - alpha[k] >= epsilon then
       (c[i,j,k,g,h] - alpha[k])
       ^(1-1/gamma[h]) / (1-1/gamma[h])
        - psi[g]*(y[i,j,k,g,h]/w[i])
        ^mu1[j] / mu1[j]
     else
        -  0.5/gamma[h] *epsilon^(-1/gamma[h]-1)
        *(c[i,j,k,g,h] - alpha[k])^2
        + (1+1/gamma[h])*epsilon^(-1/gamma[h]  )
        *(c[i,j,k,g,h] - alpha[k])
        + (1/(1-1/gamma[h]) - 1 - 0.5/gamma[h])
        *epsilon^(1-1/gamma[h])
        - psi[g]*(y[i,j,k,g,h]/w[i])^mu1[j]
        / mu1[j]
    )
  - (if c[p,q,r,s,t] - alpha[k] >= epsilon then
       (c[p,q,r,s,t] - alpha[k])^(1-1/gamma[h])
       / (1-1/gamma[h])
        - psi[g]*(y[p,q,r,s,t]/w[i])^mu1[j] / mu1[j]
```

```
    else
       -  0.5/gamma[h] *epsilon^(-1/gamma[h]-1)
       *(c[p,q,r,s,t] - alpha[k])^2
       + (1+1/gamma[h])*epsilon^(-1/gamma[h]  )
       *(c[p,q,r,s,t] - alpha[k])
       + (1/(1-1/gamma[h]) - 1 - 0.5/gamma[h])
       *epsilon^(1-1/gamma[h])
       - psi[g]*(y[p,q,r,s,t]/w[i])^mu1[j] / mu1[j]
   )
 + R[i,j,k,g,h,p,q,r,s,t] >= 0;

Technology:
   sum{(i,j,k,g,h) in T} lambda[i,j,k,g,h]
   *(y[i,j,k,g,h] - c[i,j,k,g,h]) >= 0;
```

A.2 Tax Model Data

File `pTax5Dncl.dat` provides data for a specific problem.

```
# pTax5Dncl.dat

data;

let na := 5;
let nb := 3;
let nc := 3;
let nd := 2;
let ne := 2;

# Set up wage dimension intervals
let wmin := 2;
let wmax := 4;
let {i in A}  w[i]   := wmin + ((wmax-wmin)
/(na-1))*(i-1);

data;

param  mu :=
    1   0.5
    2   1
    3   2 ;

# Define mu1
let {j in B} mu1[j] := mu[j] + 1;

data;

param alpha :=
```

```
     1    0
     2    1
     3    1.5;

param psi :=
     1    1
     2    1.5;

param gamma :=
     1    2
     2    3;

# Set up 5 dimensional distribution
let {(i,j,k,g,h) in T} lambda[i,j,k,g,h] := 1;

# Choose a reasonable epsilon
let epsilon := 0.1;
```

A.3 Initial Values

File `pTax5Dinitial.run` solves a simplified model to compute starting values for Algorithm NCL. The nonlinear inequality constraints are removed, and $y = c$ is enforced. This model solves easily with MINOS or SNOPT on all cases tried. Solution values are output to file `p5Dinitial.dat`.

```
# pTax5Dinitial.run

# Define parameters for agents (taxpayers)
param na := 5;          # number of types in wage
param nb := 3;          # number of types in eta
param nc := 3;          # number of types in alpha
param nd := 2;          # number of types in psi
param ne := 2;          # number of types in gamma
set A := 1..na;         # set of wages
set B := 1..nb;         # set of eta
set C := 1..nc;         # set of alpha
set D := 1..nd;         # set of psi
set E := 1..ne;         # set of gamma
set T = {A,B,C,D,E};    # set of agents

# Define wages for agents (taxpayers)
param  wmin := 2;          # minimum wage level
param  wmax := 4;          # maximum wage level
param  w {i in A} := wmin + ((wmax-wmin)
/(na-1))*(i-1);  # wage vector

# Choose a reasonable epsilon
```

```
param epsilon := 0.1;

# mu vector
param mu {B};               # mu = 1/eta
param mu1{B};               # mu1[j] = mu[j] + 1
param alpha {C};
param gamma {E};
param psi {D};

var c {(i,j,k,g,h) in T} >= 0.1;
var y {(i,j,k,g,h) in T} >= 0.1;

maximize f: sum{(i,j,k,g,h) in T}
   if c[i,j,k,g,h] - alpha[k] >= epsilon then
     (c[i,j,k,g,h] - alpha[k])^(1-1/gamma[h])
      / (1-1/gamma[h])
      -  psi[g] * (y[i,j,k,g,h]/w[i])^mu1[j] / mu1[j]
   else
      -  0.5/gamma[h] *epsilon^(-1/gamma[h]-1)
      *(c[i,j,k,g,h] - alpha[k])^2
      + (1+1/gamma[h])*epsilon^(-1/gamma[h])
      *(c[i,j,k,g,h] - alpha[k])
      + (1/(1-1/gamma[h]) -1 - 0.5/gamma[h])
      *epsilon^(1-1/gamma[h])
      -  psi[g] * (y[i,j,k,g,h]/w[i])^mu1[j] / mu1[j];

subject to
   Budget {(i,j,k,g,h) in T}: y[i,j,k,g,h]
   - c[i,j,k,g,h] = 0;

let {(i,j,k,g,h) in T} y[i,j,k,g,h] := i+1;
let {(i,j,k,g,h) in T} c[i,j,k,g,h] := i+1;

data;

param  mu :=
    1   0.5
    2   1
    3   2 ;

# Define mu1
let {j in B} mu1[j] := mu[j] + 1;

data;

param alpha :=
    1   0
    2   1
    3   1.5;

param psi :=
```

```
74      1    1
75      2    1.5;
76
77  param gamma :=
78      1    2
79      2    3;
80
81  option solver snopt;
82  option show_stats 1;
83
84  option snopt_options  ' \
85     summary_file=6       \
86     print_file=9         \
87     scale=no             \
88     print_level=0        \
89     major_iterations=2000\
90     iterations=50000     \
91     optimality_tol=1e-7  \
92    *penalty=100.0        \
93     superbasics_limit=3000\
94     solution=yes         \
95    *verify_level=3       \
96  ';
97
98
99  display na,nb,nc,nd,ne;
100 solve;
101 display na,nb,nc,nd,ne;
102 display y,c >p5Dinitial.dat;
103 close p5Dinitial.dat;
```

A.4 NCL Implementation

File `pTax5Dnclipopt.run` uses files

```
pTax5Dinitial.run
pTax5Dncl.mod
pTax5Dncl.dat
pTax5Dinitial.dat
```

to implement Algorithm NCL. Subproblems NC_k are solved in a loop until $\|r_k^*\|_\infty \leq$ `rtol = 1e-6`, or η_k has been reduced to parameter `etamin = 1e-8`, or ρ_k has been increased to parameter `rhomax = 1e+8`. The loop variable k is called `K` to avoid a clash with subscript `k` in the model file. The definitions of `etak` and `rhok` inside the loop are simpler than (but similar to) the settings of η_k and ρ_k in Algorithm 2.

Optimality tolerance $\omega_k = \omega_* = 10^{-6}$ is used throughout to ensure that the solution of the final subproblem NC_k will be close to a solution of the original problem if $\|r_k^*\|_\infty$ is small enough for the final k ($\|r_k^*\|_\infty \leq$ `rtol = 1e-6`).

IPOPT is used to solve each subproblem NC_k, with runtime options set to implement increasingly warm starts.

```
# pTax5Dnclipopt.run

reset;   model pTax5Dinitial.run;
reset;   model pTax5Dncl.mod;
data  pTax5Dncl.dat;
data; var include p5Dinitial.dat;

model;
option solver ipopt;
option show_stats 1;

option ipopt_options '\
 dual_inf_tol=1e-6    \
 max_iter=5000        \
';

option opt2 $ipopt_options ' warm_start_init_point
=yes';

# NCL method.
# kmax, rhok, rhofac, rhomax, etak, etafac,
  etamin, rtol
# are defined in the .mod file.

printf "NCLipopt log for pTax5D\n" > 5DNCLipopt.log;
display na, nb, nc, nd, ne, primreg  > 5DNCLipopt.log;
printf "   k      rhok      etak     rnorm
Obj\n" > 5DNCLipopt.log;

for {K in 1..kmax}
{  display na, nb, nc, nd, ne, primreg, K, kmax,
   rhok, etak;
   if K == 2 then {option ipopt_options
   $opt2 ' mu_init=1e-4'};
   if K == 4 then {option ipopt_options
   $opt2 ' mu_init=1e-5'};
   if K == 6 then {option ipopt_options
   $opt2 ' mu_init=1e-6'};
   if K == 8 then {option ipopt_options
   $opt2 ' mu_init=1e-7'};
   if K ==10 then {option ipopt_options
   $opt2 ' mu_init=1e-8'};

```

```
    display $ipopt_options;
    solve;

    let rmax := max({(i,j,k,g,h) in T, (p,q,r,s,t)
    in T:
       !(i=p and j=q and k=r and g=s and h=t)}
       R[i,j,k,g,h,p,q,r,s,t]);
    let rmin := min({(i,j,k,g,h) in T, (p,q,r,s,t)
    in T:
       !(i=p and j=q and k=r and g=s and h=t)}
       R[i,j,k,g,h,p,q,r,s,t]);
    display na, nb, nc, nd, ne, primreg, K, rhok,
    etak, kmax;
    display K, kmax, rmax, rmin;
    let rnorm := max(abs(rmax), abs(rmin));
    # ||r||_inf

    printf "%4i %9.1e %9.1e %9.1e %15.7e\n",
    K, rhok, etak, rnorm, f >> 5DNCLipopt.log;
    close 5DNCLipopt.log;

    if rnorm <= rtol then
    { printf "Stopping: rnorm is small\n";
      display K, rnorm; break; }

    if rnorm <= etak then # update dual estimate dk;
    save new solution
    {let {(i,j,k,g,h) in T, (p,q,r,s,t) in T:
          !(i=p and j=q and k=r and g=s and h=t)}
             dk[i,j,k,g,h,p,q,r,s,t] :=
             dk[i,j,k,g,h,p,q,r,s,t] + rhok
             *R[i,j,k,g,h,p,q,r,s,t];
     let {(i,j,k,g,h) in T} ck[i,j,k,g,h] :=
     c[i,j,k,g,h];
     let {(i,j,k,g,h) in T} yk[i,j,k,g,h] :=
     y[i,j,k,g,h];
     display K, etak;
     if  etak == etamin then { printf "Stopping:
     etak = etamin\n"; break; }
     let etak := max(etak*etafac, etamin);
     display etak;
    }
    else # keep previous solution; increase rhok
    { let {(i,j,k,g,h) in T} c[i,j,k,g,h] :=
      ck[i,j,k,g,h];
      let {(i,j,k,g,h) in T} y[i,j,k,g,h] :=
      yk[i,j,k,g,h];
      display K, rhok;
      if  rhok == rhomax then { printf "Stopping:
      rhok = rhomax\n"; break; }
      let rhok := min(rhok*rhofac, rhomax);
```

```
      display rhok;
    }
}

display c,y;  display na, nb, nc, nd, ne, primreg,
rhok, etak, rnorm;

# Count how many constraint are close to being active.
data;
let nT   := na*nb*nc*nd*ne;   let m := nT*(nT-1);
let n := 2*nT;
let etak := 1.0001e-10;
printf "\n m = %8i\n n = %8i\n", m, n >>
5DNCLipopt.log;
printf "\n Constraints within tol of being
active\n\n" >> 5DNCLipopt.log;
printf "    tol       count    count/n\n\n" >>
5DNCLipopt.log;

for {K in 1..10}
{ let kmax := card{(i,j,k,g,h) in T, (p,q,r,s,t) in T:
                    !(i=p and j=q and k=r and g=s
                    and h=t)
                    and Incentive[i,j,k,g,h,p,q,r,s,t].
                    slack <= etak};
  printf "%9.1e %8i %8.1f\n", etak, kmax,
  kmax/n >> 5DNCLipopt.log;
  let etak := etak*10.0;
}
printf "Created 5DNCLipopt.log\n";
```

References

1. Amestoy, Patrick R., Duff, Iain S., L'Excellent, Jean-Yves, Koster, Jacko: A fully asynchronous multifrontal solver using distributed dynamic scheduling. SIAM J. Matrix Anal. Appl. **23**(1), 15–41 (2001)
2. Arreckx, S., Orban D.: A regularized factorization-free method for equality-constrained optimization. Technical report GERAD G-2016-65. GERAD, Montréal, QC, Canada (2016)
3. Byrd, Richard H., Nocedal, Jorge, Waltz, Richard A.: Knitro: an integrated package for nonlinear optimization. In: Di Pillo, G., Roma, M. (eds.) Large-Scale Nonlinear Optimization, pp. 35–59. Springer, US, Boston, MA (2006)
4. Conn, A.R., Gould, N.I.M., Toint, PhL: A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds. SIAM J. Numer. Anal. **28**, 545–572 (1991)
5. Conn, A.R., Gould, N.I.M., Toint, PhL.: LANCELOT: a fortran package for large-scale nonlinear optimization (release A). Lecture Notes in Computational Mathematics, vol. 17. Springer, Berlin, Heidelberg, New York, London, Paris and Tokyo (1992)
6. Fourer, R., Gay, D.M., Kernighan, B.W.: AMPL: A Modeling Language for Mathematical Programming, 2nd edn. Pacific Grove, Brooks/Cole (2002)

7. Friedlander, M.P., Orban, D.: A primal-dual regularized interior-point method for convex quadratic programs. Math. Prog. Comp. **4**(1), 71–107 (2012)
8. Friedlander, M.P., Saunders, M.A.: A globally convergent linearly constrained Lagrangian method for nonlinear optimization. SIAM J. Optim. **15**(3), 863–897 (2005)
9. Gill, P.E., Kungurtsev, V., Robinson, D.P.: A stabilized SQP method: global convergence. IMA J. Numer. Anal. **37**, 407–443 (2017)
10. Gill, P.E., Kungurtsev, V., Robinson, D.P.: A stabilized SQP method: superlinear convergence. Math. Program. Ser. A **163**, 369–410 (2017)
11. Gill, P.E., Murray, W., Saunders, M.A.: SNOPT: an SQP algorithm for large-scale constrained optimization. SIAM Rev. **47**(1), 99–131 (2005). SIGEST article
12. IPOPT open source NLP solver. https://projects.coin-or.org/Ipopt
13. Judd, K.L., Ma, D., Saunders, M.A., Su, C.-L.: Optimal income taxation with multidimensional taxpayer types. Working paper. Hoover Institution, Stanford University (2017)
14. KNITRO optimization software. https://www.artelys.com/tools/knitro_doc/2_userGuide.html
15. LANCELOT optimization software. http://www.numerical.rl.ac.uk/lancelot/blurb.html
16. Murtagh, B.A., Saunders, M.A.: A projected Lagrangian algorithm and its implementation for sparse nonlinear constraints. Math. Program. Study **16**, 84–117 (1982)
17. NCL: http://stanford.edu/group/SOL/multiscale/models/NCL/
18. Robinson, S.M.: A quadratically-convergent algorithm for general nonlinear programming problems. Math. Program. **3**, 145–156 (1972)
19. Wächter, A., Biegler, L.T.: On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. Math. Program. **106**(1) (2006)
20. Wright, S.J.: Superlinear convergence of a stabilized SQP method to a degenerate solution. Comput. Optim. Appl. **11**, 253–275 (1998)

Combinatorial Optimization Problems in Engineering Applications



Hans D. Mittelmann

Abstract This paper deals with several combinatorial optimization problems. The most challenging such problem is the quadratic assignment problem (QAP). It is considered in both two dimensions (QAP) and three dimensions (Q3AP) and in the context of communication engineering. Semidefinite relaxations are used to derive lower bounds for the optimum while heuristics are applied to either find upper bounds or good feasible solutions. Semidefinite relaxations also yield bounds for questions related to binary and spherical codes including for the kissing number problem. Finally, two combinatorial problems are solved exactly, a Q3AP from communications and a directional sensor location problem.

Keywords Combinatorial optimization · Quadratic assignment problem
Index assignment problem · Modulation diversity · Binary codes
Kissing number · Spherical codes · Sensor location

1 Introduction

As is well-known many engineering applications call for continuous mathematical models such as ordinary or partial differential equations. The problems can be challenging due to their nonlinearity or other features, and sophisticated approaches are required. This work, however, is exclusively addressing problems of importance and timeliness but with a need to solve a discrete optimization problem. The characterizing feature of these problems is that a combinatorial set of feasible solutions exists, such as $n!$ or n^k. While they have finitely many solutions, it is deceptive to regard them as more easily solvable than complex but continuous problems. One obvious reason is that the number of solutions increases very rapidly and enumeration is

H. D. Mittelmann (✉)
School of Mathematical and Statistical Sciences, Arizona State University,
871804, Tempe, AZ 85287-1804, USA
e-mail: mittelmann@asu.edu
http://plato.asu.edu

© Springer International Publishing AG, part of Springer Nature 2018

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*,
Springer Proceedings in Mathematics & Statistics 235,
https://doi.org/10.1007/978-3-319-90026-1_9

out of the question already for moderate size. It is thus the goal to find the optimal solutions without evaluating the bulk of the possible solutions.

We will cover in the following two challenging and previously unsolved problems for which we were able to determine the exact solution and do it with reasonable effort. All other problems will, however, still be out of reach and for them at present only upper and lower bounds on the optimum can be given. If the problems are constrained minimization resp maximization problems, finding lower resp upper bounds to the optimal value is frequently the more challenging task requiring special methods while feasible solutions that are frequently easy to find provide upper resp lower bounds. For this latter task, we will utilize available methods while for the former we have developed algorithms which were successfully applied to engineering applications resulting in the best currently known bounds. Apart from presenting some new and unpublished results, another purpose of the paper is to compile for the first time results for closely related problems.

The by far best-known combinatorial optimization problem is the traveling salesperson problem (TSP) of finding the shortest closed loop that passes through all n given locations exactly once. It has $n!$ possible tours of which in general just one is the optimal one. While $n!$ grows extremely rapidly, it is still possible with advanced methods that have been developed over decades to determine the exact optimum for thousands of locations. This is in sharp contrast to a very closely related problem, the quadratic assignment problem (QAP). In this case in addition to locations there are, in the most basic application, also n facilities given and a prescribed flow between them. The task is to assign the facilities to locations, one by one, such that the overall flow is minimized. Again, there are $n!$ possible assignments but the exact solution is already very challenging for n larger than about 30 and there are unsolved problems of this size [1].

While both the TSP and the QAP are very easy to understand and while they can be seen completely in an abstract framework as a challenging mathematical riddle, they both occur in important applications and need to be solved partly frequently. Every delivery schedule of a shipping company can benefit from TSP solutions, for example. It is less obvious where the QAP has similar applications except in the decision where to put factories, storage facilities etc in order to minimize operational cost. This would likely not have to be solved frequently per day but a few times during the life of a company. These examples do not give a real picture of the prevalence of these problems.

The following sections will cover the determination of lower bounds for generic QAPs with semidefinite optimization methods (SDP), Sect. 2, and the application of these methods to the so-called index assignment problem in communication engineering and to modulation design, Sect. 3. In Sect. 4, a fundamental combinatorial question of relevance in the index assignment problem will be addressed, the question how many binary numbers exist of length n that have a minimal Hamming distance of d. This is of relevance for binary codes. Next, a problem from spherical codes is addressed, namely the kissing number problem. Finally in Sect. 5, two problems will be solved exactly, one is a higher-dimensional QAP while the other deals with sensor location.

2 Semidefinite Relaxations by Matrix Splitting (SDRMS)

As with most important discrete optimization problems also for the QAP, there is an established library of such problems for researchers to apply their algorithms to. The QAPLIB [1] has been established in 1997, and in the mean time only a relatively small set of instances remains unsolved. This is in contrast to other libraries, such as MIPLIB, available at http://miplib.zib.de, for which over time a number of updates have been posted. Still, these unsolved problems present a challenge and they are not of one type but occur in four of the fifteen groups (by author) of QAPLIB. To quote the outcome of the research summarized in the following: Of the 33 presently unsolved problems in QAPLIB, the SDRMS method has produced the best-known lower bounds in 18 cases.

The standard QAP takes the following form

$$\min_{X \in \Pi} \mathrm{Tr}\left(\mathrm{XAX^TB}\right) \tag{1}$$

where $A, B \in \Re^{n \times n}$, and Π is the set of permutation matrices. This problem was first introduced by Koopmans and Beckmann [2] for facility location. The model covers many scenarios arising from various applications such as in chip design [3], image processing [4], and keyboard design [5]. For more applications of QAPs, we refer to the survey paper [6] where many interesting QAPs from numerous fields are listed. In practical applications, one of the matrices is the distance matrix recording mutual distances between locations and the other the flow matrix. We assume that both matrices are symmetric. To compute lower bounds for this and other similar combinatorial optimization problems advantageously, semidefinite relaxations (SDP) have been used. Using the Kronecker product, we have

$$\mathrm{Tr}\left(\mathrm{XAX^TB}\right) = \mathrm{x^T(B \otimes A)x} = \mathrm{Tr}\left(\mathrm{(B \otimes A)xx^T}\right),\ \mathrm{x = vec(X)} \tag{2}$$

where vec(X) is obtained from X by stacking its columns into a vector of order n^2. Many existing SDP relaxations of QAPs are derived by relaxing the rank-1 matrix xx^T to be positive semidefinite with additional constraints on the matrix elements. For convenience, we call such a relaxation the classical SDP relaxation of QAPs. As pointed out in [6], the SDP bounds are tighter compared with bounds based on other relaxations, but usually much more expensive to compute due to the large number $O(n^4)$ of variables and constraints in this classical SDP relaxation.

Our goal is the derivation of SDP-based lower bounding methods requiring only the minimal $O(n^2)$ variables and constraints. In [7, 8] this was accomplished. While in [7], special relaxations for the case of Hamming and Manhattan distance matrices are considered and successfully applied to large problems from QAPLIB in [8] general splittings were given. We outline just the basic approach. The first observation is that with a nonsingular matrix X, the product $Y = XBX^T$ is positive semidefinite when B has this property. Let B be one of the two matrices in (1). We have the freedom to choose the matrix for which a larger lower bound is obtained. Now let B

be split into the difference of two positive semidefinite matrices B^+ and B^-. Such a splitting is always possible but it is not unique. The QAP has then become

$$\min_{X\in\Pi} \mathrm{Tr}\big(\mathrm{A}(\mathrm{Y}^+ - \mathrm{Y}^-)\big) \tag{3}$$

It is necessary to link the auxiliary matrices Y^+ and Y^- to the matrix B and to make sure the matrix X is a relaxed permutation matrix. All this is done in the following basic SDRMS method.

Let e be the all-1 vector and $\min(B)$ the minimum element of B.

$$\min \mathrm{Tr}\big(\mathrm{A}(\mathrm{Y}^+ - \mathrm{Y}^-)\big) \tag{4}$$

subject to

$$Y^+e = XB^+e,\ Y^-e = XB^-e, \tag{5a}$$
$$\mathrm{diag}\,(Y^+) = X\mathrm{diag}\,(B^+),\ Y^+ \geq \min(B^+), \tag{5b}$$
$$\mathrm{diag}\,(Y^-) = X\mathrm{diag}\,(B^-),\ Y^- \geq \min(B^-), \tag{5c}$$
$$Y^+ - XB^+X^T \succeq 0,\ Y^- - XB^-X^T \succeq 0, \tag{5d}$$
$$Xe = X^Te = e,\ X \geq 0 \tag{5e}$$

There are five lines of constraints. The symbol $\succeq$ in the fourth line denotes positive semidefiniteness while the last line defines X as required. This is not a standard SDP problem because the fourth line defines a quadratic and not a linear condition. This can be overcome by taking the matrix square roots R^+ and R^- of the positive semidefinite matrices B^+ and B^-. The fourth line is then replaced by the constraints

$$[I, Z^+T; Z^+, Y^+] \succeq 0, \tag{6a}$$
$$[I, Z^-T; Z^-, Y^-] \succeq 0 \tag{6b}$$

where $Z^+ = XR^+$ and $Z^- = XR^-$.

The resulting linear SDP is advantageously phrased and solved in a modeling language. There are several available packages that have both such a language and have implemented available SDP solvers. Initially, we have used [9] and the built-in solvers SeDuMi and SDPT3. For larger problems, the sizes in QAPLIB go up to 256, this requires both too much memory and too much time. So, we have linked with the package [10] and reduced to acceptable levels. Another issue is the question whether with unavoidable numerical errors the obtained lower bounds are indeed such. For this, we have postprocessed with an interval arithmetic package. These measures were applied in particular for an engineering application to be described next. Here the largest size that had to be considered is 512. For details and results on the QAPLIB problems, we refer to [8].

3 Index Assignment and Modulation Design in Communications Engineering

3.1 Bounds for the Index Assignment Problem

In digital communications, information is coded in binary words of a certain length, transmitted, and decoded on the receiver side. However, this sounds deceptively simple, and the transmission is also prone to errors. Without so-called channel errors, the decoding could proceed in the reverse way to the decoding. But this cannot be assumed and one has to provide the possibility that single and multiple bit errors can be corrected. A binary word representing a piece of information, say, from a digital picture, is not transmitted itself but it is associated with a word in a codebook that must contain sufficiently many such words. This codeword is also not transmitted but its index within the codebook. This index is itself also a binary word.
Depending on how the codewords are arranged physically in the channel, an erroneous transmission will likely have confused a word with one nearby. In all practical so-called modulations, these are words which differ in very few bits from the codeword. One thus has to make sure that codewords that are close to each other are indexed by binary numbers of large Hamming distance, i.e., which differ in many bit positions. It is not surprising the chance for error correction is maximized when a QAP is solved in which the distance matrix is the Hamming distance matrix of the codewords and the flow matrix the matrix of codeword transition probabilities. For one-bit correction, the flow matrix simplifies to the adjacency matrix of the n-dimensional hypercube. For the sake of completeness, we derive the QAP which arises in this context. Details can be found in [11].

A basic element of a signal compression and communication system is the quantizer Q, either scalar or vector. We focus on index assignment of vector quantizers (VQ) for the superior source coding performance of VQ. All our results apply to index assignment of scalar quantizers (SQ) as well, because SQ is just a special case of VQ. A vector quantizer $Q : \Re^d \to \mathbb{C}$ maps a continuous source vector $\mathbf{x} \in \Re^d$ to a codeword $\mathbf{c}_i \in \Re^d$ in the VQ codebook $\mathbb{C} = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_N\}$ by the nearest neighbor rule. The index i rather than the codeword $\mathbf{c}_i$ itself is transmitted via the channel. Upon receiving i correctly, the VQ decoder can reconstruct $\mathbf{x}$ to $\mathbf{c}_i$ by inverse quantizer mapping Q^{-1} (a simple table lookup operation). Typically, the size N of the codebook $\mathbb{C}$ is made an integer power of two, $N = 2^n$ so that the codeword index i is a binary number of n bits. An index assignment of $\mathbb{C}$ is a bijection map $\pi : \mathbb{C} \leftrightarrow \{0, 1\}^n$.

If the transmission is error free, then Q^{-1} is invariant with respect to the index assignment of the N codewords in $\mathcal{C}$, hence the overall system performance is independent of codeword index assignment. However, in the event of a transmission error that an index $\pi(\mathbf{c}_i)$ is received as $\pi(\mathbf{c}_j)$, an input vector $\mathbf{x}$ such that $\mathbf{w}_i = Q(\mathbf{x})$ will be reconstructed as $\mathbf{w}_j$, incurring an extra channel distortion $d(\mathbf{c}_i, \mathbf{c}_j)$ that does depend on the index assignment π. Let $P(j|\imath)$ be the probability of transmitting index $\imath$ but

receiving index $\jmath$, and $P(\mathbf{c}_i)$ be the prior probability of codeword $\mathbf{c}_i \in \mathbb{C}$. Given an index assignment π, the expected channel distortion is

$$\bar{d}_\pi = \sum_{i=1}^{N} P(\mathbf{c}_i) \sum_{j=1}^{N} P(\pi(\mathbf{w}_j)|\pi(\mathbf{c}_i)) d(\mathbf{c}_i, \mathbf{c}_j). \tag{7}$$

Adopting the common probability model of binary symmetric channel (BSC), we have

$$P(\pi(\mathbf{w}_j)|\pi(\mathbf{c}_i)) = (1-p)^{n-h(\pi(\mathbf{w}_j),\pi(\mathbf{c}_i))} p^{h(\pi(\mathbf{w}_j),\pi(\mathbf{c}_i))} \tag{8}$$

where p is the BSC crossover probability, and $h(\cdot,\cdot)$ is the Hamming distance. To minimize the expected BSC channel distortion $\bar{d}_\pi$, one would like to find an optimal index assignment defined by the following objective function

$$\pi_* = \arg\min_{\pi} \sum_{i=1}^{N} P(\mathbf{c}_i) \sum_{j=1}^{N} (1-p)^{n-h(\pi(\mathbf{w}_j),\pi(\mathbf{c}_i))} p^{h(\pi(\mathbf{w}_j),\pi(\mathbf{c}_i)) d(\mathbf{c}_i,\mathbf{c}_j)}. \tag{9}$$

For convenience, we rewrite (7) in a matrix form. Let

$$\mathbf{W} = (P(\mathbf{c}_1), P(\mathbf{c}_2), \ldots, P(\mathbf{c}_N))^T \mathbf{I} \tag{10}$$

be the diagonal matrix consisting of prior probabilities of the VQ codewords, and let

$$\mathbf{B} = \{(1-p)^{n-h(i,j)} p^{h(i,j)}\}_{1\le i\le N, 1\le j\le N} \tag{11}$$

be the symmetric matrix whose elements $B(i, j)$ are the codeword transition probabilities $P(\pi(\mathbf{w}_j)|\pi(\mathbf{c}_i))$ due to BSC bit errors of probability p. Also, denote by $\mathbf{D} = \{d(\mathbf{c}_i, \mathbf{c}_j)\}_{1\le i\le N, 1\le j\le N}$ the symmetric distance matrix between pairs of codewords, and use the $N \times N$ permutation matrix $\mathbf{X}$ to specify π. Now, the expected channel distortion of (7) has the following matrix form

$$\begin{aligned} \bar{d}_\pi &= \sum_{i=1}^{N} P(\mathbf{c}_i) \sum_{j=1}^{N} \{\mathbf{X}\mathbf{B}\mathbf{X}^T\}_{i,j} d(\mathbf{c}_i, \mathbf{c}_j) \\ &= \text{trace}(\mathbf{W}\mathbf{X}\mathbf{B}\mathbf{X}^T\mathbf{D}) \\ &= \text{trace}(\mathbf{D}\mathbf{W}\mathbf{X}\mathbf{B}\mathbf{X}^T) \\ &= \text{trace}(\tilde{\mathbf{D}}\mathbf{X}\mathbf{B}\mathbf{X}^T), \tilde{\mathbf{D}} = \frac{1}{2}(\mathbf{D}\mathbf{W} + \mathbf{D}^T\mathbf{W}^T) \end{aligned} \tag{12}$$

This is the QAP which is solved using our SDRMS method of the previous section. To test this approach, a training set of 18 natural images is used to design

16-dimensional vector quantizers of various fixed integer rates n, i.e., generating codebooks of size 2^n. n was varied from 5 to 9. In order to assess the quality of the computed lower bounds, a heuristic was applied to compute upper bounds. The gaps between the bounds were very satisfactory for engineering standards, in the range of 0.71–1.78 dB for one-bit errors and 0.96–1.82 dB for multiple errors and small transition probability.

3.2 Automatic Repeat Requests and Modulation Diversity

In case even a multiple bit error correction is not possible, repeated transmissions have to be made. This is typically triggered by an automatic repeat request (ARQ). But, in order that the likelihood of a successful decoding is increased, a different index assignment has to be chosen in each of possibly more than one repetition. In our recent work [12, 13], two such scenarios were investigated in detail. The so-called modulation diversity issue was addressed first for a two-way amplify-and-forward relay. To find the optimal modulation design for a given number of repetitions in this case a sequence of standard QAPs as considered in the previous section has to be solved. In fact, with each retransmission a fully general procedure would involve QAPs of dimensions 2, 3, 4, etc with tensors instead of matrices. This would become too costly and is not necessary as our results have shown.

Here the task is not to encode high-resolution pictures but to wirelessly communicate. The length of the codewords does not go to n^9 or n^{10} but a typical size is n^6. The modulation technique is also different. In the previous application, it was phase-shift keying (PSK), here it is quadrature amplitude modulation (QAM). In the former case the symmetries are those induced by the sphere, in the latter those of a square. To explain the need for retransmissions, one considers the case of 16QAM where in the standard numbering according to the Gray mapping, the constellation points 0111 and 1111 are neighbors but they can easily be confused with each other. 0111 is easily distinguishable from 0100, so a rearrangement as in the picture to the right is beneficial, see Fig. 1.

The emphasis in the study was less on assessing the quality of the lower and upper bounds obtained but more on finding feasible solutions, in other words, on near-optimal index assignments and then also to make sure they are not too far from optimal. So, again both methods were applied, an SDP-based lower bounding technique and the best available heuristics for finding upper bounds. For the latter, a robust tabu search method was suitably adapted while our SDP lower bounds were computed as before. As mentioned in [12], the gap between the bounds was typically between 10 and 20%. The industry standard approach to improve modulation diversity is CoRe or constellation rearrangement. Extensive comparisons of this to the method suggested here in [12] show that CoRe is outperformed in all ranges of practical interest. In the more complicated setting of a coordinated MIMO multi-point scenario, a suitable generalized approach showed significantly reduced packet loss and a clear performance gain over simple retransmissions and over a heuristic CoRe

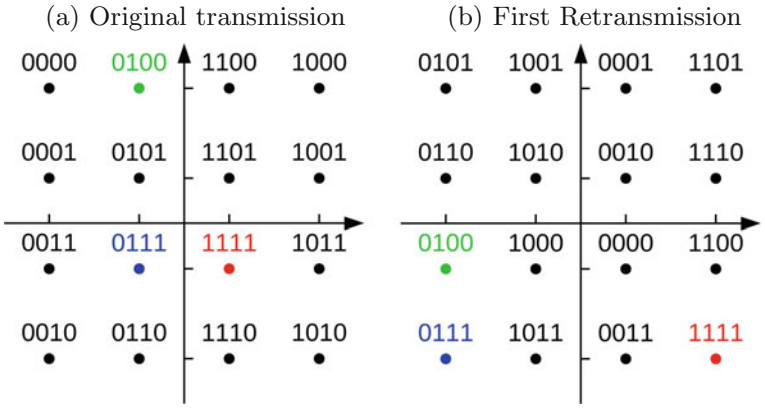


Fig. 1 Constellation rearrangement

from the literature [13]. Due to the MIMO framework, however, two-dimensional QAPs are not sufficient. We need a six-dimensional matrix as in the Lawler form of the QAP.

Let $\mathbf{x}^{(m)}$ be the 3-D permutation matrix in retransmission m, so $\mathbf{x}^{(m)} \in \mathcal{S}$ where

$$\mathcal{S} = \left\{ \mathbf{x} : \sum_{p=1}^{n} x_{pij} = \sum_{i=1}^{n} x_{pij} = \sum_{j=1}^{n} x_{pij} = 1 \right\}. \tag{13}$$

Then the minimization of the so-called bit error rate (BER) can be formulated into a Q3AP as follows:

$$\min_{\mathbf{x}^{(m)} \in \mathcal{S}} \sum_{p=1}^{n} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{q=1}^{n} \sum_{k=1}^{n} \sum_{l=1}^{n} c^{(m)}_{pijqkl} x^{(m)}_{pij} x^{(m)}_{qkl}, \tag{14}$$

Fortunately, it turns out that the matrix c is composed of a two-dimensional (flow) matrix f dependent on the index of the retransmission and a four-dimensional constant (distance) matrix d.

$$\min_{\mathbf{x}^{(m)} \in \mathcal{S}} \sum_{p=1}^{n} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{q=1}^{n} \sum_{k=1}^{n} \sum_{l=1}^{n} f^{(m)}_{pq} d_{ikjl} x^{(m)}_{pij} x^{(m)}_{qkl}, \tag{15}$$

Thus, the effort becomes very reasonable and a sequence of up to $m = 4$ retransmissions is considered with low effort. The upper bounds this time are computed with iterated local search [14].

4 Semidefinite Bounds for Binary and Spherical Codes

4.1 A Fundamental Question from Binary Codes

In the previous section, an underlying principle was to maximize the Hamming distance of certain binary words. The Hamming distance is the number of bit positions in which such words differ and an important fundamental problem is to find the number $A(n, d)$ of binary words of length n having at least Hamming distance d. This is also a very hard problem comparable to the QAP and in general only upper and lower bounds can be obtained. Various methods have been proposed for both purposes. Here, the determination of upper bounds is the more challenging problem, and SDP-based methods had already yielded improved such bounds. However, the SDP problems obtained are huge and SDP solution techniques do not scale as well as in other related optimization tasks such as second-order cone programming (SOCP). Due to their high symmetry, the problems can be reduced to manageable size but it turns out these SDPs are ill-conditioned. With standard double precision computations, insufficient accuracy is obtained which does not allow to make rigorous statements about the value of the bounds or even no statement at all.

It is necessary to do higher precision SDP solves and in particular one solver, SDPA, has been supplied with several such features and this has been even implemented by us for interactive use [15]. Precision higher than quadruple still takes considerable time. It was also a major difficulty to find parameters with which the method did not diverge. It was a major but worthwhile effort to obtain the new upper bounds [16]. They did improve many of the bounds in the regularly updated list [17] and proved even one conjecture, namely that the quadruply shortened binary Golay code is optimal, by pushing the upper bound to the value of the best-known lower bound hereby establishing an exact value, namely 256 for $A(20, 8)$, see Table 1.

For the definition of the SDP whose optimum provides an upper bound on $A(n, d)$, we refer to section 2 of [16]. The quadruple precision solution of the SDPs took up to several months of compute time.

4.2 The Kissing Number Problem

A problem related not to binary codes but to so-called spherical ones [18] is the kissing number problem. Given n-dimensional spheres of equal radius, the question is how many can be placed on the surface of a central one. For $n = 2$ the answer is obviously 6 as an experiment with coins confirms. However, the answer in three dimensions was the subject of a lifelong dispute between Sir Isaac Newton and the scientist David Gregory [19]. It took until 1953 [20] to prove the bound of 12 conjectured by Newton and until 2008 [21] to prove the value of 24 for four dimensions.

We sketch the SDPs that have to be solved to obtain strong upper bounds for the kissing number.

Table 1 Upper and lower bounds for binary codes

n	d	Known lower bound	Known upper bound	New upper bound	$A_4(n, d)$
17	4	2720	3276		3276.800
18	4	5312	6552		6553.600
19	4	10,496	13,104		13,107.200
20	4	20,480	26,168		26,214.400
21	4	36,864	43,688		43,690.667
17	6	256	340		351.506
18	6	512	680	673	673.005
19	6	1024	1280	1237	1237.939
20	6	2048	2372	2279	2279.758
21	6	2560	4096		4096.000
22	6	4096	6941		6943.696
23	6	8192	13,766	13,674	13,674.962
17	8	36	37		38.192
18	8	64	72		72.998
19	8	128	142	135	135.710
20	8	256	274	256	256.000
25	8	4096	5477	5421	5421.499
26	8	4096	9672	9275	9275.544
21	10	42	48	47	47.007
22	10	64	87	84	84.421
23	10	80	150		151.324
24	10	128	280	268	268.812
25	10	192	503	466	466.809
26	10	384	886	836	836.669
27	10	512	1764	1585	1585.071
28	10	1024	3170	2817	2817.313
25	12	52	56	55	55.595
26	12	64	98	96	96.892
27	12	128	169		170.667
28	12	178	288		288.001

The kissing number is the *stability* number of an *infinite* graph. It is bounded by the Lovasz *theta* number. The Lovasz theta number is the solution of an SDP. One considers the graph $\Gamma(S^{n-1}, (0, \pi/3))$ on the vertex set $S^{n-1} = \{x \in \mathbb{R}^n : x \cdot x = 1\}$, edges whose angular distance is less then $\pi/3$, and with inner product greater then $1/2$. The bounds from this basic SDP are then strengthened using *symmetries* and a *Lasserre hierarchy*.

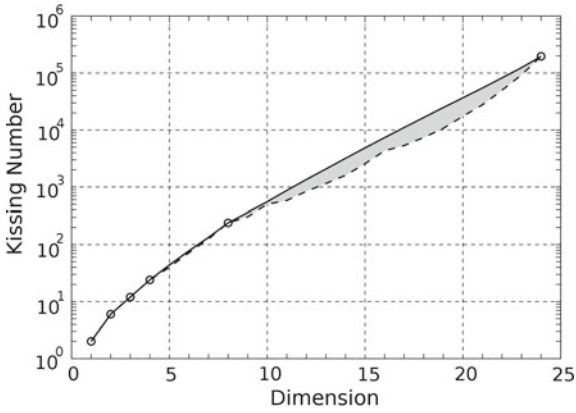


Fig. 2 Upper and lower bounds for the kissing number

$$\vartheta'(\Gamma(S^{n-1},(0,\pi/3))) = \inf\Big\{\lambda : K \in \mathcal{C}(S^{n-1}\times S^{n-1})_{\geq 0},\ K(x,x) = \lambda - 1,\ \text{for all}\, x \in S^{n-1},\ K(x,y) \leq -1,\ \text{for all}\, x,y \in S^{n-1} \ \text{with}\, x\cdot y \leq 1/2\Big\}, \tag{16}$$

$\mathcal{C}(S^{n-1}\times S^{n-1})_{\geq 0}$ is the cone of positive definite *Hilbert-Schmidt kernels*.

Over many years slowly upper and lower bounds had been established, mostly by methods that differed considerably in different dimensions. It is thus remarkable that our approach in [19] did improve on *all* known bounds in dimensions 5 to 23 with the exception of 8 for which an exact value is known, see Fig. 2. Just as in our work on binary codes, also for the kissing number problem we proved a conjecture. The new upper bound of 7355 in dimension 16 shows that a conjecture made in chap. 7, p. 190 of [23], is true, namely that there is no periodic point with a given average theta series.

5 Two Exactly Solved Combinatorial Optimization Problems

5.1 A Three-Dimensional Quadratic Assignment Problem

In this final section, we describe how two problems of combinatorial nature, one of polynomial complexity and one of optimal index assignment, were solved exactly.

In his work, Peter Hahn had addressed many variants of the QAP with a number of different approximation and solution methods. One problem provided to him by the engineering collaborators of [24] was left unsolved. For some time, he made its datafile available on his personal webpage. It is a Q3AP from optimal index

assignment, and the datafile contains the six-dimensional matrix of the form (14) for $n = 16$. Independent of the numerical method used the shear facts that of the 16^{16} elements, over 12 million are nonzero and that their range is 3.6×10^{12} makes this a very challenging problem. To treat it directly as the binary quadratic problem that it is does not permit a solution with reasonable effort. What was exploited is the power available in mixed-integer linear programming (MILP) codes due to very substantial progress both on the software and on the hardware side during recent decades. As a first step, a big-M method is applied to formally rewrite the problem as a MILP.

$$\min \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} w_{ijk} \tag{17}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n} x_{ijk} = 1, \sum_{i=1}^{n}\sum_{k=1}^{n} x_{ijk} = 1, \sum_{j=1}^{n}\sum_{k=1}^{n} x_{ijk} = 1 \tag{18}$$

$$w_{ijk} \geq \sum_{p=1}^{n}\sum_{q=1}^{n}\sum_{r=1}^{n} c_{ijkpqr} x_{pqr} - M(1 - x_{ijk}) \tag{19}$$

$$x_{ijk} \in \{0, 1\}, w_{ijk} \geq 0 \tag{20}$$

What is remarkable about this MILP is that it has of order n^3 variables and constraints. However, its dual bound is much two weak. As the next step, two sets of cutting planes are added. The first has the form

$$w_{ijk} \geq L_{ijk} x_{ijk} \tag{21}$$

Here L_{ijk} is the optimum of the linear assignment problem (3AP)

$$\min \sum_{p=1}^{n}\sum_{q=1}^{n}\sum_{r=1}^{n} c_{ijkpqr} x_{pqr} \tag{22}$$

$$\sum_{p=1}^{n}\sum_{q=1}^{n} x_{pqr} = 1, \forall r \in \{1, \ldots, n\} \tag{23}$$

$$x_{pqr} \in \{0, 1\}, x_{ijk} = 1 \tag{24}$$

It turns out that this 3AP can be solved quite efficiently, but it improves the dual bound only very little. A somewhat stronger improvement is obtained by adding cutting planes of the form

$$w_{ijk} + w_{pqr} \geq T_{ijkpqr}(x_{ijk} + x_{pqr} - 1) \tag{25}$$

where T_{ijkpqr} is the optimal value of the MILP

$$\min(w_{ijk} + w_{pqr}) \tag{26}$$

$$\sum_{s=1}^{n}\sum_{t=1}^{n} x_{pqr} = 1, \forall u \in \{1, \ldots, n\} \tag{27}$$

$$\sum_{s=1}^{n}\sum_{u=1}^{n} x_{pqr} = 1, \forall t \in \{1, \ldots, n\} \tag{28}$$

$$\sum_{t=1}^{n}\sum_{u=1}^{n} x_{pqr} = 1, \forall s \in \{1, \ldots, n\} \tag{29}$$

$$w_{ijk} \geq \sum_{s=1}^{n}\sum_{t=1}^{n}\sum_{u=1}^{n} c_{ijkstu} x_{stu} \tag{30}$$

$$w_{pqr} \geq \sum_{s=1}^{n}\sum_{t=1}^{n}\sum_{u=1}^{n} c_{pqrstu} x_{stu} \tag{31}$$

$$x_{stu} \in \{0, 1\} \tag{32}$$

$$x_{ijk} = 1, x_{pqr} = 1 \tag{33}$$

Again, the solution of these MILPs is not hard but their number is potentially of the order n^6, so as few as possible should be solved. With the basic MILP formulation and two sets of cutting planes, only the minimal groundwork has been done to solve the problem. A key property is its high symmetry [25]. As mentioned before, the modulation PSK introduces the symmetry of the sphere and this is heavily exploited in [26].

A key notion in symmetry is that of an *orbit* and the binary variables of the problem can be partitioned into six orbits. One introduces for an orbit an aggregated variable and with the help of the symbolic package Gecode all possible aggregated solutions can be enumerated, there are 85 of them. This is called orbital shrinking. Still, several other techniques need to be applied such as isomorphism pruning, primal heuristics, parameter tuning, and especially scaling due to the wide range of the matrix elements. For details we refer to [26]. Most of the 85 problems can be solved in minutes, while the 15 hardest ones take about up to half a day. But, all the problems are independent and can be run in parallel, so that the entire problem was solved in less than a day.

5.2 The Directional Sensor Problem

The second problem that was solved exactly and efficiently is one of polynomial complexity but still challenging. It was considered in [27] in a nonconvex formulation which made it difficult to obtain the globally optimal solution. The solution by the best heuristic considered in [27] also was relatively expensive. However, the treatment in [28] confirmed that it could in some cases determine the global optimum. We consider sensors such as cameras and lasers which are in fixed locations and have

a limited field of view (FOV). For simplicity, the setting is two-dimensional and the FOV is a certain angle range. The sensors observe n targets in the plane whose location χ_j (of target j) is not known with precision. It is described instead by a prior distribution $\mathcal{N}(a_j, A_j)$. Sensor i, there are m of them total, has location s_i and can be pointed in one of K possible directions. The problem consists in choosing a direction for each sensor to maximize the expected information gain over a number of possible scenarios resp. assignments. For a justification of this approach, see [28]. If there was only one scenario and no measurement error, we would just direct the sensors to cover as many targets as possible. This corresponds to a set covering problem. For several scenarios and perfect measurement, we would look for the best coverage on average which is still similar to a set covering problem. If target j is in the FOV of sensor i, when it is pointed in direction u_i, we get the measure z_{ij}:

$$z_{ij} = H\chi_j + \eta_{ij} \tag{34}$$

where H is the observation model and η_{ij} is the measurement noise, assumed to be normally distributed according to the distribution $\mathcal{N}(0, R(s_i, u_i, \chi_j))$, with R being the measurement error covariance matrix. No measurement is obtained if the target is not in the FOV of the sensor. For a given scenario, the measurements of all sensors are fused as a posterior distribution in order to obtain a global estimate for each target. This distribution is not Gaussian in general and it cannot easily be computed, instead it is approximated as Gaussian distribution $\mathcal{N}(y_j, P_j)$. Here, the arguments y_j and P_j are computed as

$$P_j = \left(A_j^{-1} + \sum_i H^T (R(s_i, u_i, a_j))^{-1} H\right)^{-1} \tag{35}$$

and

$$y_j = P_j\left(A_j a_j + \sum_i H^T (R(s_i, u_i, a_j))^{-1} z_{ij}\right) \tag{36}$$

with the summations done over the sensors that generated a measurement for target j.

Given a control vector $u = (u_1, \ldots u_n)$, the corresponding objective to be maximized is then

$$E\left[\sum_{j=1}^{n} -\log\left(\frac{\det(P_j(u))}{\det(A_j)}\right)\right] \tag{37}$$

This expectation is approximated by Monte Carlo methods. We generate a number of samples from the joint prior distribution of the target state and compute the average (over the samples) objective values for a given control action.

As given above the directional sensor control problem is nonconvex. Heuristics and rigorous algorithms for this nonlinear integer optimization problem (MINLP) will in general only produce local optima. Global optimization requires either the

use of algorithms that through spatial branch and bound or other techniques can guarantee that or convexification. The latter is possible in this case and was done in [28]. That then allows the use of software which is more efficient since it is not aiming at global optima.

We proceed by reformulating the problem and precomputing some quantities. Given the set S of samples s, we can write the problem as

$$\max \sum_{s \in S} \sum_{j=1}^{n} \left[\log(\det(\overline{P}_{js})) + \log(\det(A_j)) \right] / |S| \tag{38}$$

$$\sum_{k=1}^{m} u_{ik} = 1, \forall i \tag{39}$$

$$\overline{P}_{js} = A_j^{-1} + \sum_{i} \sum_{k} R_{ijks} u_{ik}, \forall j, s \tag{40}$$

$$u_{ik} \in \{0, 1\}, \forall i, k \tag{41}$$

Here $\overline{P}_{js}$ is the inverse of the posterior covariance matrix of target j in scenario s. R_{ijks} is the inverse of the measurement covariance matrix between sensor i pointing in direction k and target j in scenario s if the target is within the FOV of the sensor, or the null matrix otherwise. This matrix is precomputed as is A_j^{-1}. Finally, u_{ik} is a binary variable whose value is one if and only if sensor i is pointing in direction k.

Since the only nonlinearities left are the log det terms in the objective but log det is a concave function of its argument and the variable u_{ik} enters linearly, so the above problem is a nonlinear convex binary optimization problem. A solver such as KNITRO [29] can be used for its global optimization. However, it turns out that the solution with KNITRO is relatively costly for an increasing number of sensors. Therefore, a Bender's decomposition was developed in [28] which reduced the effort considerably, especially if called in a fashion where the best metaheuristic tried in [28] for a comparison of such methods is used together with the best solution found to generate outer approximation cuts for the initial master formulation.

References

1. Burkard, R.E., Cela, E., Karisch, S.E., Ren, F.: QAPLIB-A quadratic assignment problem library. J. Global Optim. **10**, 391–403 (1997)
2. Koopmans, T., Beckmann, M.: Assignment problems and the location of economic activities. Econometrica **25**, 53–76 (1957)
3. Hannan, M., Kurtzberg, J.M.: Placement techniques. In: Breuer, M.A. (ed.) *Design Automation of Digital Systems: Theory and Techniques*, vol. 1, pp. 213–282. Prentice-hall, Englewood Cliffs (1972)
4. Taillard, E.D.: Comparison of iterative searches for the quadratic assingnment problem. Location Sci. **3**, 87–105 (1995)

5. Burkard, R.E., Offermann, J.: Entwurf von Schreibmaschinentastaturen mittels quadratischer Zuordnungsprobleme. Zeitschrift für Oper. Res. **21**, B121–B132 (1977)
6. Loiola, E.M., Maia de Abreu, N.M., Boaventura-Netto, P.O., Hahn, P., Querido, T.: A survey for the quadratic assignment problem. Eur. J. Oper. Res. **176**(2), 657–690 (2007)
7. Mittelmann, H.D., Peng, J.: Estimating bounds for quadratic assignment problems associated with the Hamming and Manhattan distance matrices based on semidefinite programming. SIAM J. Optim. **20**, 3408–3426 (2010)
8. Peng, J., Mittelmann, H.D., Li, X.: A new relaxation framework for quadratic assignment problems based on matrix splitting. Math. Prog. Comp. **2**, 59–77 (2010)
9. Grant, M., Boyd, S.: CVX: Matlab software for disciplined convex programming, version 2.0 beta. available at http://cvxr.com/cvx (2013)
10. Zhao, X., Sun, D., Toh, K.-C.: A Newton-CG augmented Lagrangian method for semidefinite programming. SIAM J. Optim. **20**, 1737–1765 (2010)
11. Wu, X., Mittelmann, H.D., Wang, X., Wang, J.: On computation of performance bounds of optimal index assignment. IEEE Trans. Comm. **59**(12), 3229–3233 (2011)
12. Wu, W., Mittelmann, H.D., Ding, Z.: Modulation design for two-way amplify-and-forward relay HARQ. IEEE Wirel. Commun. Lett. **5**(3), 244–247 (2016)
13. Wu, W., Mittelmann, H.D., Ding, Z.: Modulation design for MIMO-CoMP HARQ. IEEE Commun. Lett. **21**(2), 290–293 (2017)
14. Stützle, T.h.: Iterated local search for the quadratic assignment problem. Eur. J. Oper. Res. **174**(3), 1519–1539 (2006)
15. Yamashita, M., Fujisawa, K., Kojima, M.: SDPA: semidefinite programming algorithm software available at https://sourceforge.net/projects/sdpa, interactive use through https://neos-server.org/neos/solvers/sdp:SDPA
16. Gijswijt, D.C., Mittelmann, H.D., Schrijver, A.: Semidefinite code bounds based on quadruple distances. IEEE Trans. Inf. Theory **58**(5), 2697–2705 (2012)
17. Brouwer, A.: Table of general binary codes, available at https://www.win.tue.nl/~aeb/codes/binary-1.html
18. Wikipedia.: Spherical code, available at https://en.wikipedia.org/wiki/Spherical_code
19. Wikipedia.: Kissing number problem. availabe at https://en.wikipedia.org/wiki/Kissing_number_problem
20. Schütte, K., van der Waerden, B.L.: Das problem der dreizehn Kugeln. Math. Ann. **125**, 325–334 (1953)
21. Musin, O.R.: The kissing number in four dimensions. Ann. Math. **168**, 1–32 (2008)
22. Mittelmann, H.D., Vallentin, F.: High accuracy semidefinite programming bounds for kissing numbers. Exper. Math. **19**, 174–179 (2010)
23. Conway, J.H., Sloane, N.J.A.: *Sphere Packings, Lattices and Groups*, 3rd edn. Springer, Berlin (1999)
24. Hahn, P.M., Kim, B.J., Stützle, T., Kanthak, S., Hightower, W.L., Samra, H., Ding, Z., Guignard, M.: The quadratic three-dimensional assignment problem: exact and approximate solution methods. Eur. J. Oper. Res. **184**(2), 416–428 (2008)
25. Margot, F.: Symmetry in integer linear programming. In: Jünger, M., Liebling, T., Naddef, D., Nemhauser, G., Pulleyblank, W., Reinelt, G., Rinaldi, G., Wolsey, L. (eds.) 50 Years of Integer Programming 1958–2008, pp. 647–686. Springer, Berlin, (2010)
26. Mittelmann, H.D., Salvagnin, D.: On solving a hard quadratic 3-dimensional assignment problem. Math Prog. Comput. **7**(2), 219–234 (2015)
27. Ragi, S., Mittelmann, H.D., Chong, E.K.P.: Directional sensor control: heuristic approaches. IEEE Sens. J. **15**(1), 374–381 (2014)
28. Mittelmann, H.D., Salvagnin, D.: Exact and heuristic approaches for directional sensor control. IEEE Sens. J. **15**(11), 6633–6639 (2015)
29. Byrd, R.H., Nocedal, J., Waltz, R.A.: KNITRO: an integrated package for nonlinear optimization. In: di Pillo, G., Roma, M. (eds.) Large-Scale Nonlinear Optimization, pp. 35–59. Springer, Berlin (2006)

Optimal Error Estimates for Semidiscrete Galerkin Approximations to Multi-dimensional Sobolev Equations with Burgers' Type Nonlinearity



Ambit K. Pany and Sudeep Kundu

Abstract In this paper, a finite element Galerkin method is applied to multi-dimensional Sobolev equations with Burgers' type nonlinearity and zero forcing function. Some a priori estimates for the exact solution, which are valid uniformly in time as $t \to \infty$ and even uniformly in the coefficient of dispersion μ as $\mu \to 0$, are derived. Further, optimal error estimates for semidiscrete Galerkin approximations in $L^\infty(L^2)$ and $L^\infty(H^1)$-norms are established, which again preserve the exponential decay property. Finally, some numerical experiments are conducted which confirm our theoretical findings.

Keywords Multi-dimensional Sobolev equations
Burgers type nonlinearity a priori bounds · Exponential decay property
Semidiscrete Galerkin approximation · Optimal error estimates

1 Introduction

Consider the following multi-dimensional Sobolev equations with Burgers' type nonlinearity:

$$\frac{\partial u}{\partial t} + u(\nabla u \cdot \mathbf{1}) - \mu \Delta u_t - \nu \Delta u = 0, \quad x \in \Omega, \ t > 0, \tag{1}$$

with initial and boundary conditions

$$u(x, 0) = u_0 \quad \text{in } \Omega, \qquad u = 0, \quad \text{on } \partial\Omega, \ t \geq 0, \tag{2}$$

A. K. Pany
Center of Applied Mathematics, SOA University, Bhubaneswar 752030, India
e-mail: ambit.pany@gmail.com

S. Kundu (✉)
Department of Mathematics, Industrial Mathematics Group, IIT Bombay, Powai, Mumbai 400076, India
e-mail: sudeep.kundu85@gmail.com

© Springer International Publishing AG, part of Springer Nature 2018

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*,
Springer Proceedings in Mathematics & Statistics 235,
https://doi.org/10.1007/978-3-319-90026-1_10

where Ω is a bounded convex polygonal or polyhedral domain in $\mathbb{R}^d$, $d = 2, 3$ with boundary $\partial\Omega$, $\mathbf{1} = (1, \ldots, 1)$ has d-components and $u\big(\nabla u \cdot \mathbf{1}\big) = u \sum_{i=1}^{d} u_{x_i}$. Here, ν denotes the coefficient of kinematic viscosity and μ is the coefficient of dispersion.

The mathematical model of vertical non-stationary groundwater flow in porous media with dynamic capillary pressure effect gives rise to the Problem (1)–(2) with μ denoting dynamic capillary coefficient, see [1]. Moreover, a variant of nonlinear Sobolev equation occurs when we replace the nonlinearity $u\big(\nabla u \cdot \mathbf{1}\big)$ by $\nabla \cdot F(u)$ in unsaturated flows in porous media with dynamic capillary pressure saturation relation, see [2, 3]. Regarding existence and uniqueness results, assuming nonlinear function F is smooth and bounded, authors of [1–8] have proved existence and uniqueness results.

Towards finite element methods for Sobolev equations, there are several results for error analysis for the one-dimensional case [9, 10], for the linear multi-dimensional case [11], for the nonlinear multi-dimensional problem [12–15]. In all these papers, finite element error estimates are proved under smooth and bounded nonlinearity.

We observe that the Burgers type of nonlinearity in the Problem (1)–(2) does not fall under the category discussed above by researchers and is quite similar to the nonlinearity in the Navier–Stokes system. Therefore, it needs special care to prove regularity results and carry out error analysis of the finite element approximation of this problem. In this article, we prove results which are valid uniformly in time as $t \to \infty$ and uniformly with respect to μ as $\mu \mapsto 0$. Thereby, it is shown that as $\mu \mapsto 0$, the nonlinear Sobolev equation tends to Burgers equation.

The major results obtained in this paper are summarized as follows:

(i) New regularity results for the solution of (1)–(2) even in $3D$, which are valid uniformly in time are derived. It is further shown that these estimates hold uniformly in μ as $\mu \to 0$, and hence, as $\mu \to 0$ the problem tends to the multi-dimensional Burgers equation.
(ii) Based on the analysis of Bajpai et al. [16], the exponential decay properties for the exact solution are established.
(iii) An optimal error estimate for the semidiscrete Galerkin approximation in the $L^\infty(L^2)$-norm as well as in the $L^\infty(H_0^1)$-norm is derived with error bounds satisfying an exponential decay property.
(iv) Under assumption $\mu = O(h^{2\delta})$, $\delta > 0$ small, it is shown that the error analysis given in (iii)–(iv) yields quasi-optimal estimates.
(v) Numerical experiments are conducted to confirm our theoretical findings.

This paper is organized as follows. In Sect. 2, we discuss the weak formulation and state some basic assumptions. Section 3 is devoted to development of a priori bounds for the exact solutions. In Sect. 4, we describe the semidiscrete Galerkin approximations and derive a priori estimates for the semidiscrete solutions. In Sect. 5, we establish optimal error estimates for the approximation solution. In Sect. 6, results of numerical experiments which confirm our theoretical estimates are established.

2 Preliminaries and Weak Formulation

In this section, $L^2(\Omega)$ is the space of square integrable functions defined in Ω with inner product $(\phi, \psi) = \int_\Omega \phi(x)\psi(x)\,dx$ and norm $\|\phi\| = \left(\int_\Omega |\phi(x)|^2\,dx\right)^{1/2}$. Further, $H^m(\Omega)$ denotes the standard Hilbert Sobolev space of order $m \in \mathbb{N}^+$ with norm $\|\phi\|_m = \left(\sum_{|\alpha|\le m}\int_\Omega |D^\alpha\phi|^2\,dx\right)^{1/2}$. Note that H_0^1 is equipped with a norm $\|\nabla v\| = (\nabla v, \nabla v)^{1/2}$.

For a Banach space X with norm $\|\cdot\|_X$, let $L^p(0, T; X)$ denote the space of measurable X-valued functions ϕ on $(0, T)$ such that $\int_0^T \|\phi(t)\|_X^p dt < \infty$ if $1 \le p < \infty$ and for

$$p = \infty, \quad ess \sup_{0<t<T} \|\phi(t)\|_X < \infty.$$

Throughout this paper, the following assumption is made.
(**A1**). There exists a positive constant M such that the initial approximation solution u_0 satisfies

$$u_0 \in H^2 \cap H_0^1 \text{with } \|u_0\|_2 \le M.$$

Now, we recall the following variant of Poincaré inequalities for our subsequent use: For $v \in H_0^1(\Omega)$, there holds

$$\|v\|^2 \le \lambda_1^{-1}\|\nabla v\|^2, \tag{3}$$

where λ_1^{-1} is the best possible positive constant depending on the domain Ω. Further, when $v \in H_0^1 \cap H^2$,

$$\|\nabla v\|^2 \le \lambda_1^{-1}\|\Delta v\|^2. \tag{4}$$

Now, the weak formulation of (1)–(2) is to seek $u(t) \in H_0^1$ with $u(0) = u_0$ such that for all $t > 0$

$$(u_t, \phi) + \mu(\nabla u_t, \nabla\phi) + \nu(\nabla u, \nabla\phi) + (u(\nabla u \cdot \mathbf{1}), \phi) = 0 \ \forall \phi \in H_0^1. \tag{5}$$

Define the trilinear form $b(\cdot, \cdot, \cdot)$ as

$$b(v, w, \phi) := (v(\nabla w.\mathbf{1}), \phi), \quad v, w, \phi \in H_0^1.$$

Now, it is easy to check that

$$b(w, w, w) = \frac{1}{3}\sum_{j=1}^{d}\int_{\Omega}(w^3)_{x_j}\,dx = -\frac{1}{3}\sum_{j=1}^{d}\int_{\partial\Omega} w^3 \cdot n_j\,d\Gamma = 0 \quad \forall w \in H_0^1. \tag{6}$$

3 A Priori Estimates for the Exact Solution

In this section, some a priori bounds for the solution $u(t)$ of (5) are derived.

Lemma 1 *Let assumptions* (**A1**) *hold true, and let* $0 \le \alpha < \dfrac{\nu\lambda_1}{4\,(1+\mu\lambda_1)}$. *Then, the solution* u *of* (5) *satisfies for all* $t > 0$

$$\left(\|u(t)\|^2 + \mu\|\nabla u(t)\|^2\right) + \beta e^{-2\alpha t}\int_0^t e^{2\alpha s}\|\nabla u(s)\|^2\,ds \le Ce^{-2\alpha t} \tag{7}$$

where $\beta = \nu - 2\alpha(\mu + \lambda_1^{-1}) \ge \nu/2 > 0$.

Proof For $\alpha > 0$ as stated in the Lemma, set $\hat{u}(t) = e^{\alpha t}u(t)$ in (5). Then, choose $\phi = \hat{u}$ in (5) and use (6) in the resulting equation to arrive at

$$\frac{d}{dt}(\|\hat{u}\|^2 + \mu\|\nabla\hat{u}\|^2) + 2\beta\|\nabla\hat{u}\|^2 \le 0. \tag{8}$$

Integrate with respect to time from 0 to t, then multiply by $e^{-2\alpha t}$ and use the assumption (**A2**) to complete the proof of (7).

Remark 1 As a consequence of Lemma 1, we obtain from (8) with $\alpha = 0$, the following estimate

$$\frac{d}{dt}(\|u\|^2 + \mu\|\nabla u\|^2) + \nu\|\nabla u\|^2 \le 0. \tag{9}$$

On integration with respect to time from 0 to t, and using the assumption (**A1**), we obtain

$$\nu\int_0^t \|\nabla u\|^2\,ds \le C. \tag{10}$$

Now, using a priori bounds in Lemma 1 with Bubnov Galerkin method and standard weak compactness arguments, it can be shown that there exists a unique global weak solution u to the Problem (5) for all $t > 0$.

Lemma 2 *Let assumptions* (**A1**) *hold true. Then, for* $0 < \alpha < \dfrac{\nu\lambda_1}{4\,(1+\lambda_1\mu)}$ *and for all* $t > 0$,

$$\|\nabla u(t)\|^2 + \mu\|\Delta u(t)\|^2 + \beta e^{-2\alpha t}\int_0^t e^{2\alpha s}\|\Delta u(s)\|^2\,ds \le Ce^{-2\alpha t}$$

holds, where $\beta = \nu - 2\alpha(\mu + \lambda_1^{-1}) \ge \nu/2 > 0$.

Proof Set $\hat{u} = e^{\alpha t}u$ and use the definition of the Stokes operator Δ to rewrite (5) as

$$(\hat{u})_t - \alpha\,\hat{u} - \mu\,\Delta\hat{u}_t + \mu\alpha\,\Delta\hat{u} - \nu\,\Delta\hat{u} = -e^{-\alpha t}(\hat{u}(\nabla\hat{u}\cdot\mathbf{1})) \quad \forall\phi \in H_0^1. \tag{11}$$

Multiply (11) by $-\Delta\hat{u}$ and integrate over Ω. A use of integration by parts with (3) and $-(\hat{u}_t, \Delta\hat{u}) = \dfrac{1}{2}\dfrac{d}{dt}\|\nabla\hat{u}\|^2$ leads to

$$\frac{1}{2}\frac{d}{dt}(\|\nabla\hat{u}\|^2 + \mu\|\Delta\hat{u}\|^2) + \left(\nu - \alpha(\mu+\lambda_1^{-1})\right)\|\Delta\hat{u}\|^2 = e^{-\alpha t}(\hat{u}(\nabla\hat{u}\cdot\mathbf{1}), \Delta\hat{u}). \tag{12}$$

A use of the Hölder's inequality, estimate the right hand side of (12) as

$$|e^{-\alpha t}(\hat{u}(\nabla\hat{u}\cdot\mathbf{1}), \Delta\hat{u})| \le e^{-\alpha t}\|\hat{u}\|_{L^4}\,\|\nabla\hat{u}\|_{L^4}\,\|\Delta\hat{u}\|. \tag{13}$$

When $d = 2$, a use of Ladyzhenskaya's inequality:

$$\|\hat{u}\|_{L^4} \le C\,\|\hat{u}\|^{\frac{1}{2}}\,\|\nabla\hat{u}\|^{\frac{1}{2}} \quad \text{and} \quad \|\nabla\hat{u}\|_{L^4} \le \|\nabla\hat{u}\|^{\frac{1}{2}}\,\|\Delta\hat{u}\|^{\frac{1}{2}}.$$

in (13) with Young's inequality with $p = 4, q = \frac{4}{3}, \epsilon = \frac{2\nu}{3}$ yields

$$\begin{aligned}|e^{-\alpha t}(\hat{u}(\nabla\hat{u}\cdot\mathbf{1}), \Delta\hat{u})| &\le Ce^{-\alpha t}\|\hat{u}\|^{\frac{1}{2}}\|\nabla\hat{u}\|\|\Delta\hat{u}\|^{\frac{3}{2}} \\ &\le C\left(\frac{1}{\nu}\right)^3 e^{2\alpha t}\|u\|^2\|\nabla u\|^4 + \frac{\nu}{2}\|\Delta\hat{u}\|^2. \end{aligned}\tag{14}$$

When $d = 3$, a use of Ladyzhenskaya's inequality:

$$\|\hat{u}\|_{L^4} \le C\,\|\hat{u}\|^{\frac{1}{4}}\,\|\nabla\hat{u}\|^{\frac{3}{4}} \quad \text{and} \quad \|\nabla\hat{u}\|_{L^4} \le \|\nabla\hat{u}\|^{\frac{1}{4}}\,\|\Delta\hat{u}\|^{\frac{3}{4}}. \tag{15}$$

in (13) with Young's inequality with $p = 8/7, q = 8, \epsilon^p = \frac{4\nu}{7}$ shows

$$\begin{aligned}|e^{-\alpha t}(\hat{u}(\nabla\hat{u}\cdot\mathbf{1}), \Delta\hat{u})| &\le Ce^{-\alpha t}\|\hat{u}\|^{\frac{1}{4}}\|\nabla\hat{u}\|\,\|\Delta\hat{u}\|^{\frac{7}{4}} \\ &\le C\left(\frac{1}{\nu}\right)^7 e^{2\alpha t}\|u\|^2\|\nabla u\|^8 + \frac{\nu}{2}\|\Delta\hat{u}\|^2. \end{aligned}\tag{16}$$

For $d = 2$, substitute (14) in (12) to obtain

$$\|\nabla\hat{u}(t)\|^2 + \mu\|\Delta\hat{u}(t)\|^2 + \beta\int_0^t e^{2\alpha s}\|\Delta u(s)\|^2\,ds \le (\|\nabla u_0\|^2 + \mu\|\Delta u_0\|^2) + C(\nu)\int_0^t \|u(s)\|^2\|\nabla u(s)\|^2\|\nabla\hat{u}(s)\|^2 ds. \tag{17}$$

An application of Gronwall's lemma leads to

$$\|\nabla\hat{u}(t)\|^2 + \mu\|\Delta\hat{u}(t)\|^2 + \beta\int_0^t e^{2\alpha s}\|\Delta u(s)\|^2\,ds \le \{(\|\nabla u(0)\|^2 + \mu\|\Delta u(0)\|^2) + C(\nu)\times exp\left(C(\nu)\int_0^t \|u(s)\|^2\|\nabla u(s)\|^2 ds\right). \tag{18}$$

Apply assumption **(A1)** in (18) to obtain

$$\|\nabla\hat{u}(t)\|^2 + \mu\|\Delta\hat{u}(t)\|^2 + \beta\int_0^t e^{2\alpha s}\|\Delta u(s)\|^2\,ds \le C(\nu)\,exp\left(C(\nu)\int_0^t \|u(s)\|^2\|\nabla u(s)\|^2 ds\right). \tag{19}$$

Multiply (19) by $e^{-2\alpha t}$. Then a use of estimate (10) with bounds of Lemma 1 leads to complete the proof for $d = 2$.
For $d = 3$, substitute (16) in (12) to find at

$$\|\nabla u(t)\|^2 + \mu\|\Delta u(t)\|^2 + \beta e^{-2\alpha t}\int_0^t e^{2\alpha s}\|\Delta u(s)\|^2\,ds \le e^{-2\alpha t}(\|\nabla u_0\|^2 + \mu\|\Delta u_0\|^2) + C(\nu)e^{-2\alpha t}\int_0^t \|u(s)\|^2\,\|\nabla u(s)\|^8\,ds \le C_1 + C_2\int_0^t \|\nabla u(s)\|^8\,ds. \tag{20}$$

Now, under smallness assumption on initial data the boundedness of $\|\nabla u(t)\|$ is proved for all $t > 0$. This completes the rest of the proof.

Remark 2 Using eigenfunction expansion, Galerkin method and passing to the limits with compactness argument of Lions, we can attach a meaning of regularity estimate in Lemmas 1, 2 and in subsequent Lemmas 3–5.

Lemma 3 *Under assumptions* (**A1**)*, there exists a constant* $C = C(\nu, \alpha, \lambda_1, M)$ *such that the following holds true for* $0 < \alpha < \dfrac{\nu\lambda_1}{4\,(1+\lambda_1\mu)}$ *and for all* $t > 0$

$$e^{-2\alpha t}\int_0^t e^{2\alpha s}(\|u_t(s)\|^2 + 2\mu\|\nabla u_t(s)\|^2)\,ds + \nu\|\nabla u(t)\|^2 \le Ce^{-2\alpha t}.$$

Proof Choose $\phi = e^{2\alpha t} u_t$ in (5) to arrive at

$$e^{2\alpha t}(\|u_t\|^2 + \mu\|\nabla u_t\|^2) + \frac{\nu}{2} e^{2\alpha t} \frac{d}{dt}\|\nabla u\|^2 = -e^{2\alpha t} b(u, u, u_t). \tag{21}$$

Use the Ladyzhenskaya's inequality to estimate the nonlinear term on the right-hand side of (21) for $d = 2$, $d = 3$, respectively, as

$$\begin{aligned} |b(u, u, u_t)| \le C\|u\|_{L^4} \|\nabla u\|_{L^4} \|u_t\| &\le C\|u\|^{\frac{1}{2}} \|\nabla u\| \|\Delta u\|^{\frac{1}{2}} \|u_t\| \\ &\le \frac{1}{2}\|u_t\|^2 + C\|\Delta u\|^2 + C\|u\|^2 \|\nabla u\|^4, \end{aligned} \tag{22}$$

and

$$\begin{aligned} |b(u, u, u_t)| \le C\|u\|_{L^4} \|\nabla u\|_{L^4} \|u_t\| &\le C\|u\|^{\frac{1}{4}} \|\nabla u\| \|\Delta u\|^{\frac{3}{4}} \|u_t\| \\ &\le \frac{1}{2}\|u_t\|^2 + C\|\Delta u\|^2 + C\|u\|^2 \|\nabla u\|^8. \end{aligned} \tag{23}$$

Use (22) and (23) in (21) with bound of $\|\nabla u\|$ from Lemma 2, then integrate the resulting inequality with respect to time and apply Young's inequality. Then, multiply the resulting equation by $e^{-2\alpha t}$ to arrive at

$$\begin{aligned} e^{-2\alpha t} \int_0^t e^{2\alpha s}(\|u_t(s)\|^2 + 2\mu\|\nabla u_t(s)\|^2) ds + \nu\|\nabla u(t)\|^2 \\ \le Ce^{-2\alpha t}\|\nabla u_0\|^2 + Ce^{-2\alpha t} \int_0^t e^{2\alpha s} \Big(\|\nabla u(s)\|^2 + \|\Delta u\|^2\Big) ds \\ + e^{-2\alpha t} \int_0^t e^{2\alpha s} \|u(s)\|^2 \|\nabla u(s)\|^2 ds. \end{aligned} \tag{24}$$

A use of Lemmas 1 and 2 leads to the desired result, and this concludes the proof.

Lemma 4 *Let the assumptions* (**A1**) *hold true. Then, there exists a positive constant* $C = C(\nu, \alpha, \lambda_1, M)$ *such that for all* $t > 0$

$$\|u_t(t)\|^2 + \mu\|\nabla u_t(t)\|^2 + \nu e^{-2\alpha t} \int_0^t e^{2\alpha s} \|\nabla u_t(s)\|^2 ds \le Ce^{-2\alpha t}.$$

Proof Differentiate (5) with respect to time and choose $\phi = u_t$ to obtain

$$\frac{1}{2}\frac{d}{dt}(\|u_t\|^2 + \mu\|\nabla u_t\|^2) + \nu\|\nabla u_t\|^2 = -b(u_t, u, u_t) - b(u, u_t, u_t). \tag{25}$$

Apply the Ladyzenskaya's inequality (15) and Young's inequality (with $p = 8$ and $q = 8/7$) to arrive at

$$\begin{aligned} |b(u_t, u, u_t)| &\le C\|u_t\|^{1/4}\|\nabla u\|\|\nabla u_t\|^{7/4} \\ &\le C(\nu)\ \|\nabla u\|^8\ \|u_t\|^2 + \frac{\nu}{4}\ \|\nabla u_t\|^2. \end{aligned} \tag{26}$$

We can bound the other right-hand side term as

$$\begin{aligned} |b(u, u_t, u_t)| &\le C\|\nabla u\|^{1/4}\ \|\nabla u\|^{3/4}\|\nabla u_t\|^{5/4}\|u_t\|^{1/4} \\ &\le \frac{\nu}{4}\ \|\nabla u_t\|^2 + C\Big(\|\nabla u\|^2 + \|u_t\|^2\Big) \end{aligned} \tag{27}$$

Substitute (26) and (27) in (25), multiply by $e^{2\alpha t}$ and integrate from 0 to t with respect to time to obtain

$$\begin{aligned} &\|u_t\|^2 + \mu\|\nabla u_t\|^2 + \nu e^{-2\alpha t}\int_0^t e^{2\alpha s}\|\nabla u_t(s)\|^2\, ds \\ &\qquad \le e^{-2\alpha t}(\|u_t(0)\|^2 + \mu\|\nabla u_t(0)\|^2) \\ &\qquad + Ce^{-2\alpha t}\int_0^t e^{2\alpha s}(\|u_t(s)\|^2 + \|\nabla u\|^2 + \mu\|\nabla u_t(s)\|^2)ds. \end{aligned} \tag{28}$$

From (5), it may be observed that

$$\begin{aligned} \|u_t\|^2 + \mu\|\nabla u_t\|^2 &\le C(\|\Delta u\|^2 + \|u\|^2\|\nabla u\|^4) \\ &\le C(\lambda_1)(\|\Delta u\|^2). \end{aligned} \tag{29}$$

Using (29), we can define (29) at $t = 0$. A use of Lemmas 1, 3 with (**A1**) and (29) in (28) establishes the desired estimates. This completes the rest of the proof.

Lemma 5 *Let assumptions* (**A1**) *hold. Then, there exists a positive constant* $C = C(\nu, \alpha, \lambda_1, M)$ *such that for* $0 < \alpha < \dfrac{\nu\lambda_1}{4\,(1 + \lambda_1\mu)}$ *and for all* $t > 0$

$$\nu\|\Delta u(t)\|^2 + e^{-2\alpha t}\int_0^t e^{2\alpha s}(\|\nabla u_t(s)\|^2 + \mu\|\Delta u_t(s)\|^2)ds \le Ce^{-2\alpha t}. \tag{30}$$

Moreover, the following estimate holds:

$$\mu\|\Delta u_t(t)\| \le Ce^{-2\alpha t}. \tag{31}$$

Proof Rewrite (5) as

$$u_t - \mu\ \Delta u_t - \nu\ \Delta u + u(\nabla u \cdot \mathbf{1}) = 0 \quad \forall \phi \in H_0^1. \tag{32}$$

Form L^2 inner product between (32) and $-e^{2\alpha t}\Delta u_t$ to obtain

$$\frac{\nu}{2}\frac{d}{dt}\|\Delta\hat{u}\|^2 + e^{2\alpha t}\Big(\|\nabla u_t\|^2 + \mu\|\Delta u_t\|^2\Big) = e^{2\alpha t}b(u,u,\Delta u_t) + \nu\,\alpha\|\Delta\hat{u}\|^2$$
$$= I_1 + \nu\,\alpha\|\Delta\hat{u}\|^2. \tag{33}$$

Now, integrate (33) with respect to time from 0 to t and then multiply by $2e^{-2\alpha t}$ to arrive at

$$\nu\|\Delta u\|^2 + 2e^{-2\alpha t}\int_0^t e^{2\alpha s}(\|\nabla u_t\|^2 + \mu\|\Delta u_t\|^2)\,ds \le \nu\, e^{-2\alpha t}\,\|\Delta u_0\|^2$$
$$+ 2e^{-2\alpha t}\int_0^t \Big(I_1(s)\Big)\,ds + 2\nu\,\alpha e^{-2\alpha t}\int_0^t e^{2\alpha s}\|\Delta u(s)\|^2\,ds. \tag{34}$$

For I_2 on the right-hand side of (33), rewrite it as

$$I_1 = e^{2\alpha t}\,b(u,u,\Delta u_t) = \frac{d}{dt}\Big(e^{2\alpha t}\,b(u,u,\Delta u)\Big) - 2\alpha e^{2\alpha t}\,b(u,u,\Delta u)$$
$$- e^{2\alpha t}\,b(u_t,u,\Delta u) - e^{2\alpha t}\,b(u,u_t,\Delta u). \tag{35}$$

Note that an application of the Ladyzhenskaya's inequality (15) with Young's inequality shows that

$$|e^{2\alpha t}\,b(u,u,\Delta u)| \le Ce^{2\alpha t}\,\|u\|^{1/4}\,\|\nabla u\|\,\|\Delta u\|^{7/4}$$
$$\le C(\nu)\,e^{2\alpha t}\,\|u\|^2\,\|\nabla u\|^8 + \frac{\nu}{4}e^{2\alpha t}\|\Delta u\|^2. \tag{36}$$

Using a priori bounds from Lemmas 2 and 4 in (32), we obtain

$$\|\Delta u\| \le \frac{1}{\nu}\Big(\|u_t\| + \|u\|\,\|\nabla u\| + \mu\|\Delta u_t\|\Big) \le C(\nu,\alpha,\lambda_1,M) + \frac{1}{\nu}\mu\|\Delta u_t\|. \tag{37}$$

For the third term on the right-hand side of (35), we again employ Ladyzheskaya's inequality (15) with estimates from Lemmas 2–4, (37) and Young's inequality to obtain

$$e^{2\alpha t}\,b(u_t,u,\Delta u) \le C\,e^{2\alpha t}\,\|u_t\|^{1/4}\,\|\nabla u_t\|^{3/4}\,\|\Delta u\|^{7/4}$$
$$\le C\,e^{2\alpha t}\,\|u_t\|^{1/4}\,\|\nabla u_t\|^{3/4}\,\Big(C(\nu,\alpha,\lambda_1,M) + \mu\,\|\Delta u_t\|\Big)^{7/4}$$
$$\le C(\nu,\alpha,\lambda_1,M)\,e^{2\alpha t}\,\|u_t\|^{1/4}\,\|\nabla u_t\|^{3/4}$$
$$+ C(\nu,\alpha,\lambda_1,M)\,e^{\frac{1}{4}\alpha t}\,\|u_t\|^{1/4}\,\mu^{7/8}\|\nabla u_t\|^{3/4}\,\Big(\sqrt{\mu}\,\|e^{\alpha t}\Delta u_t\|\Big)^{7/4}$$

$$\begin{aligned}
&\leq C(\nu,\alpha,\lambda_1,M)\, e^{2\alpha t}\Big(1+\|\nabla u_t\|^2\Big)\\
&+ C(\nu,\alpha,\lambda_1,M)\, e^{2\alpha t}\,\|u_t\|^2\,\mu^4\,\Big(\mu\,\|\nabla u_t\|^2\Big)^3+\frac{1}{4}e^{2\alpha t}\mu\,\|\Delta u_t\|^2\\
&\leq C(\nu,\alpha,\lambda_1,M)\, e^{2\alpha t}\Big(1+\|\nabla u_t\|^2+\mu^4\,\|u_t\|^2\Big)\\
&+ \quad \frac{1}{4}e^{2\alpha t}\mu\,\|\Delta u_t\|^2. \qquad (38)
\end{aligned}$$

Moreover, for the last term on the right-hand side of (35), a use of the following Agmon inequality (see, [17] which is valid for 3D)

$$\|u\|_{L^\infty}\leq C\|\nabla u\|^{1/2}\,\|\Delta u\|^{1/2}, \qquad (39)$$

with estimates from Lemmas 2–4, (37) and Young's inequality yields

$$\begin{aligned}
e^{2\alpha t}\,b(u,u_t,\Delta u)&\leq Ce^{2\alpha t}\,\|u\|_{L^\infty}\,\|\nabla u_t\|\,\|\Delta u\|\leq C\,e^{2\alpha t}\,\|\nabla u\|^{1/2}\,\|\tilde{\Delta} u\|^{1/2}\,\|\nabla u_t\|\,\|\Delta u\|\\
&\leq C\,e^{2\alpha t}\,\|\nabla u\|^{1/2}\,\|\nabla u_t\|\,\Big(C(\nu,\alpha,\lambda_1,M)+\mu\,\|\Delta u_t\|\Big)^{3/2}\\
&\leq C\,e^{2\alpha t}\,\Big(1+\|\nabla u_t\|^2\Big)+C(\nu,\alpha,\lambda_1,M)\,e^{2\alpha t}\,\|\nabla u_t\|\,\mu^{3/2}\|\Delta u_t\|^{3/2}\\
&\leq C\,e^{2\alpha t}\,\Big(1+\|\nabla u_t\|^2\Big)+C\,\mu\,e^{2\alpha t}\,\Big(\mu\|\nabla u_t\|^2\Big)^2+\frac{1}{4}\mu\|\Delta u_t\|^2. \qquad (40)
\end{aligned}$$

Substituting (38) and (40) in I_2 and integrating with respect to time, use a priori bounds in Lemmas 2–4 to arrive for the second term on the right-hand side of (34) at

$$\begin{aligned}
2e^{-2\alpha t}\int_0^t I_2(s)\,ds&\leq C(\nu,\alpha,\lambda_1,M)\\
&+C\,e^{-2\alpha t}\int_0^t e^{2\alpha s}\Big(1+(1+\mu)\|\nabla u_t\|^2+\|u_t\|^2+\|\Delta u\|^2\Big)\,ds\\
&+\frac{\nu}{2}\|\Delta u(t)\|^2+e^{-2\alpha t}\int_0^t e^{2\alpha s}\Big(\|\nabla u_t\|^2+\mu\|\Delta u_t\|^2\Big)\,ds\\
&\leq C(\nu,\alpha,\lambda_1,M)\;+\frac{\nu}{2}\|\Delta u(t)\|^2\\
&+e^{-2\alpha t}\int_0^t e^{2\alpha s}\Big(\|\nabla u_t\|^2+\mu\|\Delta u_t\|^2\Big)\,ds. \qquad (41)
\end{aligned}$$

Substitute (41) in (34) and use Lemmas 1, 2–4 with assumption **(A1)** and standard kickback argument to arrive at the desired estimate (30). To prove (31), we note from (32) using Lemmas 2–4 with estimate (15) and (30) that

$$\begin{aligned}
\mu\|\Delta u_t(t)\|&\leq \|u_t\|+\nu\|\Delta u\|+\|u(\nabla u\cdot 1)\|\\
&\leq \Big(\|u_t\|+\nu\|\Delta u\|+C\|u\|^{1/4}\,\|\nabla u\|\,\|\Delta u\|^{3/4}\Big)\leq Ce^{-2\alpha t}.
\end{aligned}$$

The main theorem of this section is stated below without proof as its proof follows easily from Lemmas 1, 2–5.

Theorem 1 *Let the assumptions* (**A1**) *hold. Then, there exists a positive constant* $C = C(\nu, \alpha, \lambda_1, M)$ *such that for* $0 \le \alpha < \dfrac{\nu\lambda_1}{2(1+\lambda_1\mu)}$ *the following estimates hold true:*

$$\|u(t)\|_2^2 + \|u_t(t)\|^2 + \mu\|u_t(t)\|_1^2 + e^{-2\alpha t}\int_0^t e^{2\alpha s}(\|u(s)\|_2^2 + \|u_t(s)\|_1^2 + \mu\|u_t(s)\|_2^2)\,ds \le Ce^{-2\alpha t}.$$

4 The Semidiscrete Scheme

With $h > 0$ as a discretization parameter, let V_h, $0 < h < 1$ be finite-dimensional subspaces of H_0^1, and be such that, there exist operators i_h satisfying the following approximation properties:

(**B1**). For each $v \in H_0^1 \cap H^2$, there are approximations $i_h v \in V_h$ such that

$$\|v - i_h v\| + h\|\nabla(v - i_h v)\| \le K_0 h^2 \|v\|_2.$$

For defining the Galerkin approximations, for $v, w, \phi \in H_0^1$, set $a(v, \phi) = (\nabla v, \nabla\phi)$ and $b(v, w, \phi)$ as in Sect. 2. The discrete analogue of the weak formulation (5) is to find $u_h(t) \in V_h$ such that $u_h(0) = u_{0h}$ and for $t > 0$,

$$(u_{ht}, \phi_h) + \mu a(u_{ht}, \phi_h) + \nu a(u_h, \phi_h) + b(u_h, u_h, \phi_h) = 0 \quad \forall \phi_h \in V_h. \tag{42}$$

where $u_{0h} \in V_h$ is a suitable approximation of $u_0 \in H_0^1$ to be defined later.

Since (42) leads to a system of nonlinear ordinary differential equations, an application of Picard's theorem ensures existence of a unique solution u_h for $(0, t_h^*)$ for some $t_h^* > 0$. In order to prove the global existence, we appeal to a continuation argument, provided the discrete solution is bounded for all $t > 0$. Following the argument in the proof of Lemma 1, it is easy to prove the following estimate: for all $t > 0$ and for $0 < \alpha < \dfrac{\nu\lambda_1}{4(1+\mu\lambda_1)}$

$$\|u_h(t)\|^2 + \mu\|\nabla u_h(t)\|^2 + \beta e^{-2\alpha t}\int_0^t e^{2\alpha s}\|\nabla u_h(s)\|^2\,ds \le Ce^{-2\alpha t}, \tag{43}$$

where $\beta = \nu - 2\alpha(\mu + \lambda_1^{-1}) > \nu/2 > 0$. This completes the proof of existence and uniqueness of a global discrete solution for all $t > 0$.

As a consequence of conditions (**B1**), we have the following properties of the L^2 projection $P_h : L^2 \to V_h$. For $\phi \in H_0^1$,

$$\|\phi - P_h\phi\| + h\|\nabla P_h\phi\| \leq Ch\|\nabla\phi\|, \tag{44}$$

and for $\phi \in H_0^1 \cap H^2$,

$$\|\phi - P_h\phi\| + h\|\nabla(\phi - P_h\phi)\| \leq Ch^2\|\Delta\phi\|. \tag{45}$$

We may define the discrete Laplace operator $\Delta_h : V_h \to V_h$ through the bilinear form $a(\cdot, \cdot)$ as

$$a(v_h, \phi_h) = (-\Delta_h v_h, \phi_h) \quad \forall v_h, \phi_h \in V_h. \tag{46}$$

Next in the following Lemma, a priori bounds for the discrete solution u_h of (42), which will be helpful in establishing the error estimates, are stated. The proof can be obtained following the similar steps as in the proofs of Lemmas 1–3.

Lemma 6 *For all $t > 0$, the semi-discrete Galerkin approximation u_h for the approximation solution satisfies*

$$\|u_h(t)\|_1^2 + \mu\|\Delta_h u_h(t)\|^2 + \|\Delta_h u_h(t)\|^2 + e^{-2\alpha t}\int_0^t e^{2\alpha s}(\|\nabla u_h\|^2 + \|\Delta_h u_h\|^2 + \|\nabla u_{ht}\|^2)\,ds \leq Ce^{-2\alpha t}.$$

5 Error Estimates for the Approximation Solution

Set $e = u - u_h$. Then, from (5) and (42), we obtain

$$(e_t, \phi_h) + \mu a(e_t, \phi_h) + \nu a(e, \phi_h) = \Lambda(\phi_h), \tag{47}$$

where $\Lambda(\phi_h) = -b(u, u, \phi_h) + b(u_h, u_h, \phi_h)$. Below, we derive an optimal error estimate of $\|\nabla e(t)\|$, for $t > 0$.

Lemma 7 *Let assumptions* (**A1**) *and* (**B1**) *be satisfied. With $u_{0h} = P_h u_0$, then, there exists a positive constant C depending on λ_1, ν, α and M, such that for fixed $T > 0$ with $t \in (0, T)$ and for $0 \leq \alpha < \dfrac{\nu\lambda_1}{4(1 + \lambda_1\mu)}$, the following estimate holds true*:

$$\|(u - u_h)(t)\|^2 + \mu\|\nabla(u - u_h)(t)\|^2 \leq Ch^2e^{-2\alpha t}.$$

Proof On multiplying (47) by $e^{\alpha t}$ with $\phi_h = P_h\hat{e} = \hat{e} + (P_h\hat{u} - \hat{u})$, it follows that

$$(e^{\alpha t}e_t, \hat{e}) + \mu a(e^{\alpha t}e_t, \hat{e}) + \nu a(\hat{e}, \hat{e}) = e^{\alpha t}\Lambda(P_h\hat{e}) + (e^{\alpha t}e_t, \hat{u} - P_h\hat{u}) + \mu\, a(e^{\alpha t}e_t, \hat{u} - P_h\hat{u}) + \nu a(\hat{e}, \hat{u} - P_h\hat{u}). \tag{48}$$

Using (3) in (48) yields

$$\frac{d}{dt}(\|\hat{e}\|^2 + \mu\|\nabla\hat{e}\|^2) + \left(2\nu - 2\alpha(\mu + {\lambda_1}^{-1})\right)\|\nabla\hat{e}\|^2 \le 2e^{\alpha t}\Lambda(P_h\hat{e}) + \frac{d}{dt}\left(\|\hat{u} - P_h\hat{u}\|^2 + 2\mu\, a(\hat{e}, \hat{u} - P_h\hat{u})\right) - 2\mu\, a(\hat{e}, e^{\alpha t}(u_t - P_h\hat{u}_t)) - 2\alpha\left(\|\hat{u} - P_h\hat{u}\|^2 + \mu\, a(\hat{e}, \hat{u} - P_h\hat{u})\right) + 2\nu a(\hat{e}, \hat{u} - P_h\hat{u}). \tag{49}$$

Apply the Cauchy–Schwarz inequality with Poincaré inequality and Young's inequality to the last three terms on the right-hand side of (49) estimate as

$$|2\alpha\left(\|\hat{u} - P_h\hat{u}\|^2 + \mu\, a(\hat{e}, \hat{u} - P_h\hat{u})\right) + 2\nu a(\hat{e}, \hat{u} - P_h\hat{u}) + 2\mu\, a(\hat{e}, e^{\alpha t}(u_t - P_hu_t)|$$
$$\le C(\alpha, \lambda_1, \nu)\left(\|\nabla(\hat{u} - P_h\hat{u})\|^2 + \mu^2\|e^{\alpha t}\nabla(u_t - P_hu_t)\|^2 + \frac{\nu}{2}\|\nabla\hat{e}\|^2.\right. \tag{50}$$

Use the property of P_h to estimate the first term on the right-hand side of (49) as

$$2e^{\alpha t}\Lambda(P_h\hat{e}) = -2e^{-\alpha t}\left(b(\hat{e}, \hat{e}, \hat{u} - P_h\hat{u}) + b(\hat{e}, \hat{u}, P_h\hat{e}) + b(\hat{u}, \hat{e}, P_h\hat{e})\right). \tag{51}$$

Then, using the generalized Hölder's inequality, Agmon's inequality (39), Young's inequality, the Sobolev embedding theorem and (44), we arrive at

$$\begin{aligned} 2e^{\alpha t}|\Lambda(P_h\hat{e})| &\le 2e^{-\alpha t}\left(\|\hat{u}\|_{L^\infty}\|\nabla\hat{e}\|\|P_h\hat{e}\| + \|\nabla\hat{e}\|\|\Delta\hat{u}\|\|P_h\hat{e}\| + \|\nabla\hat{e}\|\,\|\nabla\hat{e}\|\|\nabla(\hat{u} - P_h\hat{u})\|\right) \\ &\le 2e^{-\alpha t}\left(\left(\|\nabla\hat{u}\|^{\frac{1}{2}}\|\Delta\hat{u}\|^{\frac{1}{2}} + \|\Delta\hat{u}\|\right)\|\hat{e}\|\|\nabla\hat{e}\| + (\|\nabla\hat{u}\| + \|\nabla\hat{u}_h\|)\|\nabla\hat{e}\|\|\nabla(\hat{u} - P_h\hat{u})\|\right) \\ &\le C(\nu)e^{-2\alpha t}\left(\left(\|\nabla\hat{u}\|\|\Delta\hat{u}\| + \|\Delta\hat{u}\|^2\right)\|\hat{e}\|^2 + \|\nabla(\hat{u} - P_h\hat{u})\|^2\right) + \frac{\nu}{2}\|\nabla\hat{e}\|^2. \end{aligned} \tag{52}$$

Integrating (49) with respect to time from 0 to t, substitute bounds (50) and (52) to arrive at

$$\|\hat{e}(t)\|^2 + \mu\|\nabla\hat{e}(t)\|^2 + \beta\int_0^t\|\nabla\hat{e}\|^2ds \le C(\|e(0)\|^2 + \|\nabla e(0)\|^2) + C(\alpha, \nu, \lambda_1, M)\left(\|\nabla(\hat{u} - P_h\hat{u})\|^2 + \int_0^t(\|\nabla(\hat{u} - P_h\hat{u})\|^2 + \mu^2\|\nabla(\hat{u}_t - P_h\hat{u}_t)\|^2)ds\right) + C\int_0^t\left(\|\nabla u\|\|\Delta u\| + \|\Delta u\|^2\right)\|\hat{e}\|^2\,ds. \tag{53}$$

Using (45) and (**B1**) in (53) yields

$$\|\hat{e}(t)\|^2 + \mu||\nabla\hat{e}(t)\|^2 + \beta\int_0^t \|\nabla\hat{e}\|^2 ds$$
$$\leq Ch^2\Big(\|u_0\|_2^2 + \|\hat{u}\|_2^2 + \int_0^t (\|\hat{u}\|_2^2 + \|\hat{u}_t\|_2^2)ds\Big)$$
$$+ C\int_0^t \big(\|\nabla u\|\|\Delta u\| + \|\Delta u\|^2\big)\big(\|\hat{e}\|^2 + \mu\|\nabla\hat{e}\|^2\big)ds.$$

From the a priori bounds in Theorem 2 and use of the Gronwall lemma, we arrive at

$$\|\hat{e}(t)\|^2 + \mu\|\nabla\hat{e}(t)\|^2$$
$$+ \beta\int_0^t \|\nabla\hat{e}\|^2\, ds \leq C(\nu,\alpha,\lambda_1,M)h^2\exp\Big(\int_0^t (\|\Delta u\|^2 + \|\nabla u\|\|\Delta u\|)\, ds\Big).$$

Using a priori bounds given in Lemma 2 yields

$$\int_0^t \big(\|\nabla u\|\, \|\tilde{\Delta u}\| + \|\Delta u\|^2\big)ds \leq C(1 - e^{-2\alpha t}) < C < \infty, \tag{54}$$

and hence, we arrive at

$$\|(u - u_h)(t)\|^2 + \mu\|\nabla(u - u_h)(t)\|^2 \leq Ch^2 e^{-2\alpha t}.$$

This concludes the proof.

Note that the Lemma 7 provides a suboptimal error estimates in $L^\infty(L^2)$-norm. Therefore, in the remaining part of this section, we derive an optimal error estimate in $L^\infty(L^2)$-norm.
Lets define Ritz projection $R_h : H_0^1 \to V_h$ by

$$(\nabla(u - R_h u), \nabla\chi) = 0 \quad \forall\chi \in V_h. \tag{55}$$

It is easy to cheek that

$$\|u - R_h u\| + h\|\nabla(u - R_h u)\| \leq Ch^2\|\Delta u\| \quad \forall u \in H_0^1 \cap H^2. \tag{56}$$

Now we split $\mathbf{e}$ as

$$\mathbf{e} := u - u_h = (u - R_h u) + (R_h u - u_h) = \eta + \xi. \tag{57}$$

Using (57) with (55) in (47) , we obtain

$$\begin{aligned}(\xi_t, \phi_h) + \mu a(\xi_t, \phi_h) + (\eta_t, \phi_h) + \mu a(\eta_t, \phi_h) + \nu a(\xi, \phi_h) \\ &= -((u(\nabla u.\mathbf{1}) - u_h(\nabla u_h.\mathbf{1})), \phi_h) \\ &= -\frac{1}{2}((u^2 - u_h^2), \nabla\phi_h.\mathbf{1}) \\ &= -\frac{1}{2}((\xi + \eta)(u + u_h), \nabla\phi_h.\mathbf{1}) \\ &= \Lambda(\phi_h). \end{aligned} \tag{58}$$

Lemma 8 *Let assumptions* (**A1**) *and* (**B1**) *hold. Then, there exists a positive constant* $C = C(\nu, \alpha, \lambda_1, M)$ *such that for* $0 < \alpha < \dfrac{\nu\lambda_1}{4\,(1 + \lambda_1\mu)}$ *and for all* $t > 0$

$$\|\xi\|^2 + \mu\|\nabla\xi\|^2 + \beta e^{-2\alpha t}\int_0^t e^{2\alpha s}\|\nabla\xi\|^2\,ds \le C\,\frac{h^4}{\mu}e^{-2\alpha t}.$$

Proof On multiplying (59) by $e^{\alpha t}$ with $\phi_h = \hat{\xi}$, it follows that

$$\begin{aligned}\frac{d}{dt}(\|\hat{\xi}\|^2 + \mu\|\nabla\hat{\xi}\|^2) + (2\nu - 2\alpha(\mu + \lambda^{-1}))\|\nabla\hat{\xi}\|^2 \\ &= 2e^{\alpha t}\Lambda(\hat{\xi}) + 2(\hat{\eta}_t, \hat{\xi}) + 2\mu(\nabla\hat{\eta}_t, \nabla\hat{\xi}) \\ &\quad - \alpha((\hat{\eta}, \hat{\xi}) + \mu(\nabla\hat{\eta}, \nabla\hat{\xi})). \end{aligned} \tag{59}$$

Apply the Cauchy–Schwarz inequality with Poincaré inequality and Young's inequality to the last four terms on the right-hand side of (59) estimate as

$$\begin{aligned}|2(\hat{\eta}_t, \hat{\xi}) + 2\mu(\nabla\hat{\eta}_t, \nabla\hat{\xi}) + \alpha((\hat{\eta}, \hat{\xi}) + \mu(\nabla\hat{\eta}, \nabla\hat{\xi}))| \\ &\le C(\alpha, \lambda, \nu)\Big(\|\hat{\eta}_t\|^2 + \mu^2\|\nabla\hat{\eta}_t\|^2 \\ &\quad + \|\hat{\eta}\|^2 + \mu^2\|\nabla\hat{\eta}\|\Big) + \frac{\nu}{4}\|\nabla\hat{\xi}\|^2. \end{aligned} \tag{60}$$

Using the generalized Hölder's inequality, Agmon's inequality (39), and Young's inequality with the Sobolev embedding theorem, the first term on the right-hand side of (59) is estimated as

$$\begin{aligned}2e^{\alpha t}|\Lambda(\hat{\xi})| \le C(\alpha, \nu)\Big(\|\hat{\xi}\|^2\|\Delta u\|^2 + \|\hat{\xi}\|^2\|\Delta u_h\|^2 + \|\eta\|^2\|\Delta\hat{u}\|^2 + \|\eta\|^2\|\Delta\hat{u}_h\|^2\Big) \\ + \frac{\nu}{4}\|\hat{\xi}\|^2 \end{aligned} \tag{61}$$

Integrating (59) with respect to time from 0 to t, bounds (60) and (61) are substituted to arrive at

$$\begin{aligned}\|\hat{\xi}\|^2+\mu\|\nabla\hat{\xi}\|^2+\beta\int_0^t\|\nabla\hat{\xi}\|^2ds &\le e^{2\alpha t}(\|\xi(0)\|^2+\mu\|\nabla\xi(0)\|^2))\\&+C(\alpha,\lambda,\nu)\int_0^t\Big(\|\hat{\eta}_t\|^2+\mu^2\|\nabla\hat{\eta}_t\|^2+\|\hat{\eta}\|^2+\mu^2\|\nabla\hat{\eta}\|^2\Big)ds\\&+C(\alpha,\nu)\int_0^t\Big(\|\hat{\xi}\|^2\|\Delta u\|^2+\|\hat{\xi}\|^2\|\Delta u_h\|^2\Big)ds\\&+\int_0^t\Big(\|\eta\|^2\|\Delta\hat{u}\|^2+\|\eta\|^2\|\Delta\hat{u}_h\|^2\Big)ds. \qquad (62)\end{aligned}$$

Using (55), (56) and a priori bounds given in Lemma 2 in (62) yields

$$\begin{aligned}\|\hat{\xi}\|^2+\mu\|\nabla\hat{\xi}\|^2+\beta\int_0^t\|\nabla\hat{\xi}\|^2ds &\le e^{2\alpha t}(\|\xi(0)\|^2+\mu\|\nabla\xi(0)\|^2))\\&+C(\alpha,\lambda,\nu)h^4\int_0^t\Big(\|\Delta\hat{u}_t\|^2+\mu^2\|\Delta\hat{u}_t\|^2+\|\Delta\hat{u}\|^2+\nu^2\|\Delta\hat{u}\|^2\Big)ds\\&+C(\alpha,\nu)\int_0^t\Big((\|\Delta u\|^2+\|\Delta u_h\|^2)\|\hat{\xi}\|^2\Big)ds\\&+\int_0^t\Big(\|\eta\|^2\|\Delta\hat{u}\|^2+\|\eta\|^2\|\Delta\hat{u}_h\|^2\Big)ds. \qquad (63)\end{aligned}$$

Now using bounds from Lemmas 1, 2 , 5 and Lemma 6 in (63), we obtain

$$\begin{aligned}\|\hat{\xi}\|^2+\mu\|\nabla\hat{\xi}\|^2+\beta\int_0^t\|\nabla\hat{\xi}\|^2ds &\le e^{2\alpha t}(\|\xi(0)\|^2+\mu\|\nabla\xi(0)\|^2))\\&+C(\alpha,\nu)\int_0^t\Big((\|\Delta u\|^2+\|\Delta u_h\|^2)(\|\hat{\xi}\|^2+\mu\|\nabla\hat{\xi}\|\Big)ds\\&+C(\alpha,\nu,\lambda,M)\,\frac{h^4}{\mu}. \qquad (64)\end{aligned}$$

From the a priori bounds in Theorem 2 and use of the Gronwall lemma, we arrive at

$$\begin{aligned}\|\hat{\xi}(t)\|^2+\mu\|\nabla\hat{\xi}(t)\|^2+\beta\int_0^t\|\nabla\hat{\xi}\|^2\,ds&\\\le C(\nu,\alpha,\lambda_1,M)\,\frac{h^4}{\mu}\,\exp\Big(\int_0^t(\|\Delta u\|^2+\|\Delta u_h\|^2)\,ds\Big).&\end{aligned}$$

A use of a priori bounds given in Lemmas 2 and 6 yields

$$\int_0^t\big(\|\Delta u\|^2+\|\Delta u_h\|^2\big)ds\le C(1-e^{-2\alpha t})\le C<\infty, \qquad (65)$$

and hence, we arrive at

$$\|\xi\|^2 + \mu\|\nabla\xi\|^2 \leq C\,\frac{h^4}{\mu}\,e^{-\alpha t}.$$

This concludes the proof.

we derive the main Theorem 2 of this section.

Theorem 2 *Let the assumptions* (**A1**) *and* (**B1**) *be satisfied. Further, let the discrete initial approximation solution be* $u_{0h} = P_h u_0$. *Then, there exists a positive constant* C *depending on* $\nu,\ \lambda_1,\ \alpha$ *and* M *such that, for all* $t \in (0, T]$ *and for* $0 \leq \alpha < \dfrac{\nu\lambda_1}{4\left(1+\lambda_1\mu\right)}$, *the following estimate holds*:

$$\|(u-u_h)(t)\| + h\|\nabla(u-u_h)(t)\| \leq C\,\frac{h^2}{\sqrt{\mu}}e^{-2\alpha t}. \tag{66}$$

Note that with $\mu = O(h^{2\delta}),\ \delta > 0$ small,

$$\|(u-u_h)(t)\| + h\|\nabla(u-u_h)(t)\| \leq C\,h^{2-\delta}\,e^{-2\alpha t}.$$

6 Numerical Experiments

In this section, we discuss some numerical experiments related to exponential decay property and order of convergence results for the Problem (1)–(2). For time discretization, we use backward Euler method.

Example 1 Here, we take $\mu = 1$, $\nu = 1$ in $I = (0, 1) \times (0, 1)$ with initial condition as $u_0 = 2x(x-1)y(y-1)$ (exact solution as $t = 0$). Time interval is taken as $t = [0, 10]$.

From Fig. 1, we can easily visualize that solution decays exponentially as time increases in L^2-norm.

Example 2 For the Problem (1)–(2), we take $\mu = 1$, $\nu = 1$ in $I = (0, 1) \times (0, 1)$ which satisfy the homogeneous Dirichlet boundary condition. Here, we take initial condition as $u_0 = x(x-1)y(y-1)$ (exact solution as $t = 0$). Since exact solution is unknown, so we take refined mesh solution with piecewise $\mathbb{P}1$ polynomial as exact solution. Also, we can take refined mesh solution with piecewise $\mathbb{P}2$ polynomial as exact solution.

From the Table 1, the convergence rates are given for $T = 1$. Note that $\|u(t_n) - u_h^n\|_1$ is of order one which satisfies our theoretical result. From the Table spsreftable:1, it is also observed that the convergence rate for L^2-error is of order two. So we establish optimal order of convergence both theoretically and numerically.

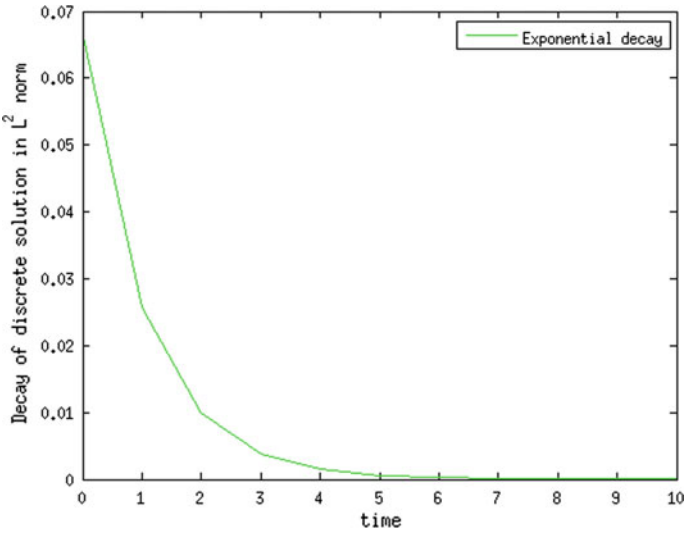


Fig. 1 The L^2 norm of the approximate solution for Example 1 at t=0,1,2,…showing exponential decay

Table 1 Errors and convergence rate at time $T = 1$

h	$\|u(t_n) - u_h^n\|_{L^2}$	Conv. rate	$\|u(t_n) - u_h^n\|_{H^1}$	Conv. rate
1/2	0.0059599		0.0420676	
1/4	0.0016971	1.812	0.022789	0.8843
1/8	0.00043828	1.953	0.011399	0.9994
1/16	0.00011031	1.9902	0.005552	1.037
1/32	0.000027487	2.0048	0.0026178	1.084
1/64	0.00000676	2.023	0.001152	1.184

Acknowledgements Both authors thanks Professor Amiya K. Pani for his valuable suggestion.

References

1. Cuesta, C.M., Pop, I.S.: Numerical schemes for a pseudo-parabolic Burgers equation: discontinuous data and long-time behaviour. J. Comp. Appl. Math. **224**, 268–283 (2009)
2. Fan, Y., Pop, I.S.: A class of pseudo-parabolic equations: existence, uniqueness of weak solutions, and error estimates for the Euler-implicit discretization. Math. Methods Appl. Sci. **34**, 2329–2339 (2011)
3. Mikelic, A.: A global existence result for the equations describing unsaturated flow in porous media with dynamic capillary pressure. J. Differ. Equ. **248**, 1561–1577 (2010)
4. Cao, X., Pop, I.S.: Uniqueness of weak solutions for a pseudo-parabolic equation modeling two phase flow in porous media. Appl. Math. Lett. **46**, 25–30 (2015)

5. Showalter, R.E.: A nonlinear Parabolic-Sobolev equation. J. Math. Anal. Appl. **50**, 183–190 (1975)
6. Showalter, R.E.: The Sobolev equation I. Appl. Anal. **5**, 15–22 (1975)
7. Showalter, R.E.: The Sobolev equation II. Appl. Anal. **5**, 81–99 (1975)
8. Showalter, R.E.: Sobolev equations for nonlinear dispersive systems. Appl. Anal. **7**, 297–308 (1975)
9. Arnold, D.N., Douglas Jr., J., Thomeé, V.: Superconvergence of a finite element approximation to the solution of a Sobolev equation in a single space variable. Math. Comp. **36**, 53–63 (1981)
10. Nakao, M.T.: Error estimates of a Galerkin method for some nonlinear Sobolev equations in one space dimension. Numer. Math. **47**, 139–157 (1985)
11. Liu, T., Lin, Y.-P., Rao, M., Cannon, J.R.: Finite element methods for Sobolev equations. J. Comp. Math. **20**, 627–642 (2002)
12. Ewing, R.E.: Numerical solution of Sobolev partial differential equations. SIAM J. Numer. Anal. **12**, 345–363 (1975)
13. Ewing, R.E.: Time-stepping Galerkin methods for nonlinear Sobolev partial differential equations. SIAM J. Numer. Anal. **15**, 1125–1150 (1978)
14. Lin, Y-P.: Galerkin methods for nonlinear Sobolev equations. Aequationes Math. **40**, 54–66 (1990)
15. Lin, Y.-P., Zhang, T.: Finite element methods for nonlinear Sobolev equations with nonlinear boundary condition. J. Math. Anal. Appl. **165**, 180–191 (1992)
16. Bajpai, S., Nataraj, N., Pani, A.K., Damazio, P., Yuan, J.Y.: Semidiscrete Galerkin method for equations of motion arising in Kelvin-Voigt model of viscoelastic fluid flow. Numer. Methods PDE **29**, 857–883 (2013)
17. Foias, C., Manley, O., Rosa, R., Temam, R.: Navier-Stokes equations and turbulence. Cambridge University Press, Cambridge (2001)

Optimal Power Flow Analysis in Power Dispatch for Distribution Networks



Paolo Pisciella, Maria Teresa Vespucci, Giacomo Viganò, Marco Rossi and Diana Moneta

Abstract This paper presents two applications of Optimal Power Flow analysis for active and reactive power redispatch in medium-voltage distribution networks and shows how this tool can be used to efficiently manage the selection and operation of network resources as well as the definition of a market interface with the transmission network. The description of the frameworks is complemented by the analysis of a case study for the optimal selection and operation of available devices.

Keywords Optimal power redispatch · Active distribution network Integration of renewable resources · Distributed generation

1 From Centralized to Distributed Generation

Distributed generation (DG) employs small-scale technologies to produce electricity in the proximity of the consumption areas [1]. DG technologies are normally constituted of modular and renewable energy generators. These generators offer a better

P. Pisciella (✉)
Department of Industrial Economics and Technology Management, NTNU, Alfred Getz Vei 3, 7491 Trondheim, Norway
e-mail: paolo.pisciella@ntnu.no

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. Viganò · M. Rossi · D. Moneta
RSE - Ricerca sul Sistema Energetico S.p.A., via Rubattino 54, 20134 Milano, Italy
e-mail: giacomo.vigano@rse-web.it

M. Rossi
e-mail: marco.rossi@rse-web.it

D. Moneta
e-mail: diana.moneta@rse-web.it

© Springer International Publishing AG, part of Springer Nature 2018

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*, Springer Proceedings in Mathematics & Statistics 235, https://doi.org/10.1007/978-3-319-90026-1_11

exploitation of renewable resources, reducing the environmental impact compared to the usage of traditional large-scale generators. In contrast with the established paradigm of using few large-scale plants, located away from centres, and long transmission lines to bring power where load is located, DG utilizes a large number of small plants with capacities ranging from a fraction of a kilowatt [kW] to about 10 megawatt [MW], whereas conventional power plant can reach capacities exceeding 1000 MW [3].

In the current model, large power plants utilize combustion (coal, oil and natural gas) or nuclear reactions to generate large amounts of power that require to be transmitted from production plants to final consumers, often over long distances. This creates disadvantages, mainly due to inefficiencies and power losses over the lengthy transmission lines, possible criticalities in integrating power lines into the landscape and security-related issues. In conclusion, conventional generation usually determines high level of CO_2 and pollution emissions and issues related to supply security. Utilization of distributed energy can mitigate the negative impact of these issues, since DG is often produced by small modular energy conversion units, such as solar panels or small wind generators. So the European Union is moving towards the adoption of generation based on renewable energy sources (RES) as a strategy for increasing competitiveness and sustainability and fostering fuel independence. Finally, the market liberalization has opened the possibility for DG to compete more efficiently with the conventional units, increasing its affordability.

The two different approaches are displayed in Fig. 1.

Power plants connected to the distribution network are normally site specific: depending on the peculiarities of the installation site, one might have wind turbines,

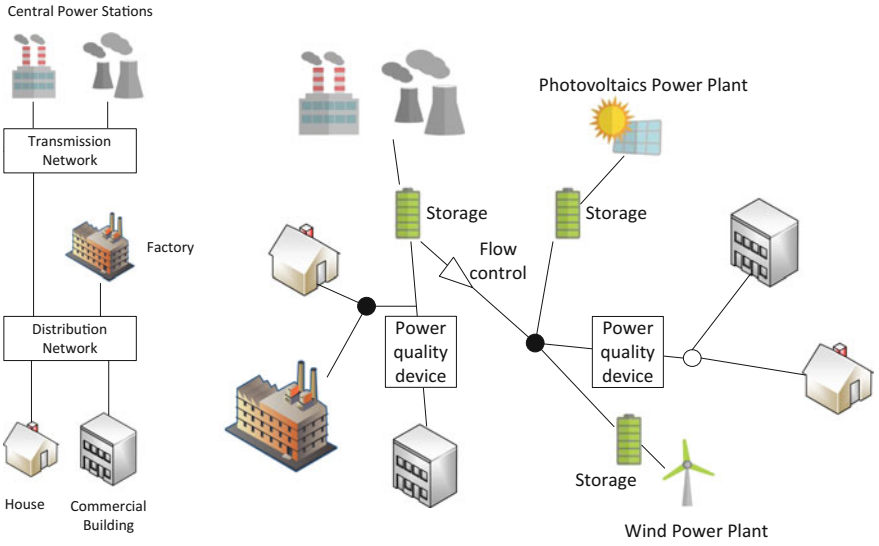


Fig. 1 Different power system paradigms with centralized power plants (**a**) and distributed generation (**b**) (*Source* Introduction to Distributed Generation. European Commission Research and Innovation)

cogenerators, biomass energy production, geothermal energy production, photovoltaic systems and hydro-thermal plants. Energy consumers may also install similar technology on an individual basis [2]. Therefore, DG technologies can both sell their energy as market participants or they can operate energy production for immediate consumption.

It is also a shared opinion that the distributed process involved in the DG can improve the efficiency of electric power production and distribution through the reduction of the transmission distances, which might save potential losses up to about 9% of the produced power. These losses are mainly due to the growing levels of congestion [23]. Moreover, power quality is often affected by sudden changes in voltages or electrical flows that result from different causes, such as poor switching operations in the network, interruptions and network disturbances from loads and DGs. Costs related to the management of this transmission grid are included in the final consumers' bill. Therefore, the use of on-site generated power is expected to bring the end-user to obtain a higher level of power quality at lower costs. In addition, the end-users will take the new role of prosumers, by simultaneously being consumers and producers of electric power, enabling in this way the possibility to sell the excess production to the grid. Producing directly for the end-users also allows to reduce the demand during peak times and minimize the congestion of power on the network.

Another advantage of the DG technologies takes the form of improved reliability for industries that require uninterrupted services. It is estimated that only in the USA, the costs related to power outages and quality disturbances amount to 119 billion USD per year [12], bringing the average per hour cost of a power outage to 6,480,000 USD [20]. Moreover, DG technology improves the security of the grid. The decentralization of power production helps reducing the negative effects of interruptions of services by insulating the grid from failures if a line or a power plant goes down. In case of sensitive customers (hospitals, schools), DGs can supply the loads in case of outages of the public grid.

Finally, one of the most important effects of distributed generation is related to the environmental safeguard. Large power plants are responsible for large levels of pollution, while a widespread use of RES-based DG technology would reduce CO_2 emissions by a substantial extent [14]. Under an economical viewpoint, DG can support the nations to increase their diversity of energy sources, often free from consumption of fossil fuels. The result is an improved insulation of the economies from price shocks, interruptions and fuel shortages [11].

The cost reduction and the increased availability of distributed power resources are boosting the shift towards a distributed generation paradigm. Nonetheless, this transition is also fostered by the ability to overcome the constraints related to an expansion of the power generation and transmission system. The reduced size of the generation units, the lower capital requirements and the short time needed for their construction establish a particularly good ground for the development of DG systems [21].

These power technologies are today ready to compete with the cost and performance of central power plants. This because today's technology allows these resources to work in a coordinated manner either on-site or remotely. This synchro-

nization of distributed technologies places DG to operate in an integrated energy system alongside with conventional power plants.

2 The Bottleneck of Transmission Networks

Even in the most developed countries, environmental and regulatory barriers often hinder the possibility of expanding the power transmission network. Even when these constraints can be overcome, the high investment levels and the long planning process make it economically unsuitable. Network constraints constitute a big reason that advocates in favour of the development of distributed power generation. This is even clearer in developing countries, where a less developed transmission and distribution system makes it necessary to call for a DG paradigm for meeting energy needs. The large amount of capital needed to expand and/or strengthen the transmission network is testified by the investments that have been allocated in projects around the world. As a mention, in 2012 US electric utilities invested approximately 20 billion USD on transmission alone. Also, Europe needs to increase the transmission capacity in many regions, but difficulties in regulations and high investment costs are often leading to delays or cancellations of projects [5]. Nevertheless, in 2006, the association of European Transmission System Operators indicated that in some countries not a single overhead power line exceeding five kilometres has been built in the last 10 years. The International Energy Agency (IEA) has estimated that 187 billion USD of transmission investments is required in Europe through 2030 [15]. In developing countries such as China and India, the main investments are focused on the expansion of the transmission and distribution infrastructure. Here, distributed power can provide power to remote areas, currently not reached by the transmission network. Developing countries also face the problem of electricity theft, which cannot be measured directly, but only estimated. This issues can be solved or reduced through the use of DG, locating the production near the end-users. It is important to understand that DG must be supported by an appropriate distribution network, which must also take care of the integration and coordination of the DG resources. In fact, the widespread installation of generation units might lead to higher levels of active and reactive power losses or voltage deviations due to imbalances between production from renewable sources and load [24]. These voltage fluctuation are subject to continuous variations due to the non-programmable change of output over time of the renewable generators. Voltage fluctuations are, in turn, directly linked to the power quality delivered to customers. Different DG power plant types may have different impacts on distribution networks. For example, the power injected into the distribution network from a photovoltaic system might have a different impact to that of a wind turbine [25].

The new paradigm introduced with smart grids involves the exploitation of well-established technologies and concepts by pooling and coordinating the available resources. This allows for an aggregated response to the increased demand and requirements regarding quality and sustainability of the energy sector [6]. The fast

processing and exchange of information between available resources bring even more incentive in developing a common framework to manage the network operations in coordination. This integration of network resources allows for the creation of a more robust and reliable power system which is less hindered by human limitations.

In order to fully exploit this integration, it is important that the system is endowed with some degree of intelligence so as to be able to provide responses to sudden changes in the network structure. In other words, it is important to give the system the possibility to automatically balance the generation-load profile, congestion on lines or shift in voltages. In this case, the system's intelligence can be represented by the response of computer programs that employ optimization techniques and support decision-making.

This chapter will describe the usage of Optimal Power Flow (OPF) analysis for ensuring efficiency in dispatch of power in distribution networks with distributed generation and storage devices. It is assumed that the Distribution System Operator (DSO) has the possibility of dispatching resources connected to the distribution network in order to guarantee the proper level of quality of supply; in addition, distributed resources can offer their flexibility to the Transmission System Operator (TSO) through the ancillary service market. The OPF tool will provide its utility in two applications: the efficient choice of devices for power redispatch and the construction of equivalent capabilities to provide bids from the distribution network to the transmission network.

3 A Procedure for the Optimal Management of Medium-Voltage AC Networks with Distributed Generation and Storage Devices

In this section, we describe a two-step solution method for the problem of determining the best control action to take when imbalance between load and generation occurs in a medium-voltage distribution network. Due to the partial unpredictability of load and generation stemming from renewable sources, distribution networks are subject to continuous imbalances. The Distribution System Operator (DSO) is the agent in charge of rebalancing the supply by using the available resources in the grid. The framework defines the DSO as facing two layers of decisions, with the first layer consisting in the selection of the most suitable devices to be used in the power redispatch and the second layer consisting in determining the optimal usage of the selected resources to perform the power redispatch. The problem is modelled as a Mixed Integer Nonlinear Problem (MINLP) where integrality is due to the binary variables for modelling fixed costs related to the selected devices and nonlinearity pertains to the constraints typical of the Optimal Power Flow problem in alternating current. The time period, or planning horizon, under consideration is discretized in time units (e.g. one day divided in 24 h or in 96 quarters of hour), and intertemporal energy balance constraints are introduced to model storage units. Distribution networks with a large number of nodes and lines give rise to large

dimensional MINLP models, which in turn require large computational effort for their solution. Bosisio et al. [4] propose to decouple the determination of the devices to be used and the AC power redispatch to account for security constraints (current in lines) and delivered power quality (voltages) for every considered time unit. In this approach, an approximate solution of the MINLP problem is obtained by means of a two-step procedure, with the first step based on a Mixed Integer Linear Program and the second step taking into account the nonlinear OPF constraints. In this approach, the DSO is assumed to have knowledge on the planned output of active and reactive power by the controllable generators and on the forecast for load and power output of non-controllable generators. The possible imbalances between load and generation are solved by employing an OPF tool providing the least cost solution for the power redispatch by determining the optimal contribution of internal regulation resources (i.e. directly operated by the DSO), such as On-Load Tap Changers (OLTC) and storage units, and external regulation resources, such as active and reactive power injection/absorption from controllable resources, which are required to modify their production plans in exchange for an economic compensation (for both positive and negative variations of power generation). Costs may be defined either as market prices for the usage of controllable resources or as values for accounting deterioration of internal resources, such as OLTC or storage devices.

The MILP model proposed in [4] for the first step of the procedure is as follows:

Sets

$\mathcal{N}$	set of nodes, indexed by i
$\mathcal{L}$	set of lines, indexed by l
$\mathcal{G}$	set of power generators, indexed by g
$\mathcal{B}$	set of storage devices, indexed by b
$\mathcal{T}$	set of time units, indexed by t, in which the time horizon is divided
$\mathcal{G}^{ND} \subseteq \mathcal{G}$	set of non-dispatchable generators
$\mathcal{G}^{DS} \subseteq \mathcal{G}$	set of dispatchable generators with interruptible production
$\mathcal{G}^{D} \subseteq \mathcal{G}$	set of dispatchable generators with non-interruptible production

Parameters

dt	[h]	duration of time unit
C_g^S	[€]	fixed cost for interrupting production of generator g
C_g^{DF}	[€]	fixed cost for modifying scheduled production of generator g
C_g^{DU}	[€/MWh]	unitary cost for increasing production of generator g
C_g^{DD}	[€/MWh]	unitary cost for decreasing production of generator g
C_b^{in}	[€/MWh]	unitary cost for charging storage device b
C_b^{out}	[€/MWh]	unitary cost for discharging storage device b
$\hat{p}_{g,t}$	[MW]	scheduled power output of generator $g \in \mathcal{G}^{DS} \bigcup \mathcal{G}^{D}$ at time t
$P_g^{\max}$	[MW]	maximum power output of dispatchable generator g
$P_g^{\min}$	[MW]	minimum power output of dispatchable generator g
$p_{g,t}^F$	[MW]	power output of non-dispatchable generator g at time t

e_b^{iniz}	[MWh]	energy in storage device b at the beginning of $t = 1$
e_b^{fin}	[MWh]	energy required in storage unit b at the end of time horizon
$e_b^{\max}$	[MWh]	capacity of storage device b
$p_b^{\max}$	[MW]	maximum charge/discharge of storage device b
η_b^h	[−]	energy loss coefficient of storage device b
η_b^{in}	[−]	charge loss coefficient of storage device b
η_b^{out}	[−]	discharge loss coefficient of storage device b
$L_{i,t}^F$	[MW]	load at node i at time t
$l_{i,t}^F$	[−]	loss rate at node i at time t
$\sigma_{i,l,t}$	[−]	Power Transfer Distribution Factor (PTDF) of line l from node i at time t
$\overline{f}_{l,t}$	[MW]	maximum power flow on line l at time t
$\underline{f}_{l,t}$	[MW]	minimum power flow on line l at time t

Decision Variables

$\gamma_{g,t}$	[0/1]	binary variable (1: production of generator g is interrupted at time t; 0: otherwise)
$\delta_{g,t}$	[0/1]	binary variable (1: scheduled production of generator g is modified, i.e increased or reduced, at time t; 0: otherwise)
$u_{g,t}$	[MW]	increase of scheduled production for generator g at time t
$d_{g,t}$	[MW]	reduction of scheduled production for generator g at time t
$p_{b,t}^{\text{in}}$	[MW]	energy rate from source of storage device b at time t
$p_{b,t}^{\text{out}}$	[MW]	energy rate to load of storage device b at time t
$e_{b,t}$	[MWh]	energy in storage device b at the end of time t
$f_{l,t}$	[MW]	power flow on line l at time t

$$\min \quad dt \sum_{t \in \mathcal{T}} \left[\sum_{g \in \mathcal{G}^D} \left(C_g^{DF} \delta_{g,t} + C_g^{DU} u_{g,t} + C_g^{DD} d_{g,t} \right) + \sum_{g \in \mathcal{G}^{DS}} C_g^S \left(1 - \gamma_{g,t}\right) + \sum_{b \in \mathcal{B}} \left(C_b^{\text{in}} p_{b,t}^{\text{in}} + C_b^{\text{out}} p_{b,t}^{\text{out}} \right) \right] \tag{1}$$

subject to

$$0 \leq u_{g,t} \leq \left(P_g^{\max} - \hat{p}_{g,t}\right) \delta_{g,t} \qquad g \in \mathscr{G}^D, t \in \mathscr{T} \qquad (2)$$

$$0 \leq d_{g,t} \leq \left(\hat{p}_{g,t} - P_g^{\min}\right) \delta_{g,t} \qquad g \in \mathscr{G}^D, t \in \mathscr{T} \qquad (3)$$

$$0 \leq u_{g,t} \leq \left(P_g^{\max} - \hat{p}_{g,t}\right) \delta_{g,t} \qquad g \in \mathscr{G}^{DS}, t \in \mathscr{T} \qquad (4)$$

$$u_{g,t} \leq \left(P_g^{\max} - \hat{p}_{g,t}\right) \gamma_{g,t} \qquad g \in \mathscr{G}^{DS}, t \in \mathscr{T} \qquad (5)$$

$$0 \leq d_{g,t} \leq \hat{p}_{g,t} \delta_{g,t} \qquad g \in \mathscr{G}^{DS}, t \in \mathscr{T} \qquad (6)$$

$$\hat{p}_{g,t} - \hat{p}_{g,t}\gamma_{g,t} \leq d_{g,t} \leq \hat{p}_{g,t} - P_g^{\min}\gamma_{g,t} \qquad g \in \mathscr{G}^{DS}, t \in \mathscr{T} \qquad (7)$$

$$e_{b,t} = \left(\eta_b^h e_{b,t-1} + \eta_b^{\text{in}} p_{b,t}^{\text{in}} - \eta_b^{\text{out}} p_{b,t}^{\text{out}}\right) dt \qquad b \in \mathscr{B}, t \in \mathscr{T} \qquad (8)$$

$$0 \leq e_{b,t} \leq e_b^{\max} \qquad b \in \mathscr{B}, t \in \mathscr{T} \qquad (9)$$

$$e_b^{\text{fin}} \leq e_{b,|\mathscr{T}|} \qquad b \in \mathscr{B}, t \in \mathscr{T} \qquad (10)$$

$$0 \leq p_{b,t}^{\text{in}} \leq p_b^{\max} \qquad b \in \mathscr{B}, t \in \mathscr{T} \qquad (11)$$

$$0 \leq p_{b,t}^{\text{out}} \leq p_b^{\max} \qquad b \in \mathscr{B}, t \in \mathscr{T} \qquad (12)$$

$$\sum_{i \in \mathscr{N}} \left(1 + l_{i,t}^F\right) L_{i,t}^F = \sum_{g \in \mathscr{G}^{ND}} p_{g,t}^F + \sum_{g \in \mathscr{G}^D} \left(\hat{p}_{g,t} + u_{g,t} - d_{g,t}\right) + \sum_{b \in \mathscr{B}} \left(p_{b,t}^{\text{out}} - p_{b,t}^{\text{in}}\right) \qquad t \in \mathscr{T} \qquad (13)$$

$$f_{l,t} = \sum_{i \in \mathscr{N}} \sigma_{i,l,t} \left[\sum_{g \in \mathscr{G}^{ND}} p_{g,t}^F + \sum_{g \in \mathscr{G}^D} \left(\hat{p}_{g,t} + u_{g,t} - d_{g,t}\right) \right. \qquad (14)$$

$$\left. + \sum_{b \in \mathscr{B}} \left(p_{b,t}^{\text{out}} - p_{b,t}^{\text{in}}\right) - \sum_{i \in \mathscr{N}} \left(1 + l_{i,t}^F\right) L_{i,t}^F \right] \qquad l \in \mathscr{L}, t \in \mathscr{T}$$

$$\underline{f}_{l,t} \leq f_{l,t} \leq \overline{f}_{l,t} \qquad l \in \mathscr{L}, t \in \mathscr{T} \qquad (15)$$

The objective function (1) represents the cost, to be minimized, of the DSO control action. For dispatchable generators with non-interruptible production, constraints (2) and (3) guarantee that power output, after variation, is between its minimum and maximum values. For dispatchable generators with interruptible production, constraints (4) to (7) state that power output, after variation, is either between its minimum and maximum values, if $\gamma_{g,t} = 1$ (i.e. if production is not interrupted), or zero, if $\gamma_{g,t} = 0$. Constraints (8) are the intertemporal energy balance constraints of storage device b, in which losses are taken into account. Constraints (9) impose lower and upper bounds to the energy stored in storage device b at the end of time t. Constraints (10) guarantee the required minimum energy in storage device b at the end of the period under consideration. Constraints (11) impose lower and upper bounds to the energy rate from source, and constraints (12) impose lower and upper bounds to the energy rate to load. Constraints (13) are the power balance equations that must hold at every time t: the power output of both dispatchable and non-dispatchable generators plus the net power output of storage devices must equal the sum of loads over all nodes, plus a term that represents the losses in lines, which are taken into

account by means of the loss coefficients $l^F_{i,t}$ associated with nodes. Constraints (14) define the power flows $f_{l,t}$ on line l in period t, which are guaranteed by constraints (15) to be between their lower and upper bounds.

The solution of the MILP problem provides the list of the most efficient devices to be used in the power redispatch. This set of devices is used as data in Step 2 of the procedure, where the AC power redispatch has to be determined, taking into account security constraints (current in lines) and delivered power quality (voltages) for every unit of the discretized time horizon. In order to state the nonlinear model to be used in Step 2, the following sets, parameters and decision variables are defined.

Sets

$\mathscr{N}$	set of nodes, indexed by i
$\mathscr{G}$	set of selected power generators, indexed by g
$\mathscr{B}$	set of selected storage devices, indexed by b
$\mathscr{T}$	set of time units, indexed by t, in which the time horizon is divided
$\mathscr{L}$	set of ordered pairs of nodes (i, j)

The subset $\mathscr{G}_i \in \mathscr{G}$ contains all generators located at node i. An element $(i, j) \in \mathscr{L}$ may represent

- a transformer with On-Load Tap Changer either on the primary winding ($\mathscr{L}_1^{TC} \subset \mathscr{L}$) or on the secondary winding ($\mathscr{L}_2^{TC}$)
- network lines or fixed ratio transformers $\mathscr{L} \setminus (\mathscr{L}_1^{TC} \cup \mathscr{L}_2^{TC})$

For every storage device b a line $(i(b), j(b))$ is defined, where $j(b)$ is the network node where the storage device is located and $i(b)$ is a virtual node defined so as to take into account losses related to charge and discharge operations.

Parameters

$P^0_{g,t}$	[MW]	scheduled active power output for generator g at time t
$Q^0_{g,t}$	[Mvar]	scheduled reactive power output for generator g at time t
$c^{\Delta P+}_{g,t}$	[€/MW]	cost of active power increase of generator g at time t
$c^{\Delta P-}_{g,t}$	[€/MW]	cost of active power decrease of generator g at time t
$c^{\Delta Q+}_{g,t}$	[€/Mvar]	cost of reactive power increase of generator g at time t
$c^{\Delta Q-}_{g,t}$	[€/Mvar]	cost of reactive power decrease of generator g at time t
$c^{\text{out},P}_{b,t}$	[€/MW]	cost of active power discharge of storage device b at time t
$c^{\text{in},P}_{b,t}$	[€/MW]	cost of active power charge of storage device b at time t
$c^{\text{out},Q}_{b,t}$	[€/MW]	cost of reactive power discharge of storage device b at time t
$c^{\text{in},Q}_{b,t}$	[€/MW]	cost of reactive power charge of storage device b at time t
$\overline{P}_g$	[MW]	maximum active power output of generator g
$\underline{P}_g$	[MW]	minimum active power output of generator g
$\overline{Q}_g$	[Mvar]	maximum reactive power output of generator g
$\underline{Q}_g$	[Mvar]	minimum reactive power output of generator g
E^0_b	[MWh]	stored electricity in storage device b
$\overline{E}_b$	[MWh]	maximum electricity that can be stored in storage device b

$\underline{E}_b$	[MWh]	minimum electricity that can be stored in storage device b
η_b^{out}	[−]	discharge loss coefficient of storage device b ($0 \leq \eta_b^{\text{out}} \leq 1$)
η_b^{in}	[−]	charge loss coefficient of storage device b ($\eta_b^{\text{in}} \geq 1$)
$\overline{\theta}_i$	[−]	maximum phase angle of node i
$\underline{\theta}_i$	[−]	minimum phase angle of node i
$\overline{V}_i$	[kV]	maximum voltage magnitude of node i
$\underline{V}_i$	[kV]	minimum voltage magnitude of node i
$C_{i,t}$	[MW]	active load of node i at time t
$D_{i,t}$	[Mvar]	reactive load of node i at time t
G_i	[S]	shunt conductance of node i
B_i	[S]	shunt susceptance of node i
$\overline{TA}_{i,j}$	[MW]	maximum active power on line (i, j)
$\underline{TA}_{i,j}$	[MW]	minimum active power on line (i, j)
$\overline{TR}_{i,j}$	[Mvar]	maximum reactive power on line (i, j)
$\underline{TR}_{i,j}$	[Mvar]	minimum reactive power on line (i, j)
$\overline{TI}_{i,j}$	[kA]	maximum current on line (i, j)
$\underline{TI}_{i,j}$	[kA]	minimum current on line (i, j)
$\delta_{i,j}$	[−]	loss angle of series impedance on line (i, j)
$X_{i,j}$	[S]	transversal conductance of line (i, j)
$Y_{i,j}$	[S]	transversal susceptance of line (i, j)
$Z_{i,j}$	[Ω]	series impedance of line (i, j)
$Vn_{i,j}$	[kV]	rated voltage of transformer primary winding i
$Vn_{j,i}$	[kV]	rated voltage of transformer secondary winding j
$u_{i,j}$	[−]	maximum increase relative to rated voltage for transformers with tap changer installed on the primary winding i
$d_{i,j}$	[−]	minimum increase relative to rated voltage for transformers with tap changer installed on the primary winding i
$u_{j,i}$	[−]	maximum increase relative to rated voltage for transformers with tap changer installed on the secondary winding j
$d_{j,i}$	[−]	minimum increase relative to rated voltage for transformers with tap changer installed on the secondary winding j

Decision Variables

$\Delta P_{g,t}^{+}$	[MW]	increase of active power output for generator g at time t
$\Delta P_{g,t}^{-}$	[MW]	decrease of active power output for generator g at time t
$\Delta Q_{g,t}^{+}$	[Mvar]	increase of reactive power output for generator g at time t
$\Delta Q_{g,t}^{-}$	[Mvar]	decrease of reactive power output for generator g at time t
$P_{b,t}^{\text{out}}$	[MW]	active power discharge of storage device b at time t
$P_{b,t}^{\text{in}}$	[MW]	active power charge of storage device b at time t
$Q_{b,t}^{\text{out}}$	[Mvar]	reactive power injection of storage device b at time t
$Q_{b,t}^{\text{in}}$	[Mvar]	reactive power absorption of storage device b at time t
$P_{g,t}$	[MW]	active power output of generator g at time t
$Q_{g,t}$	[Mvar]	reactive power output of generator g at time t

$P_{b,t}$	[MW]	active power exchange between network and storage device b at time t
$Q_{b,t}$	[Mvar]	reactive power exchange between network and storage device b at time t
$V_{i,t}$	[kV]	voltage of node i at time t
$\theta_{i,t}$	[−]	phase angle of node i at time t
$TI_{i,j,t}$	[kA]	current transit on line (i, j) at time t
$TA_{i,j,t}$	[MW]	active power flow on line (i, j) at time t
$TR_{i,j,t}$	[Mvar]	reactive power flow on line (i, j) at time t
$TC_{i,j,t}$	[kV]	voltage at time t of the primary winding for transformer with OLTC (i, j) installed in the primary winding
$TC_{j,i,t}$	[kV]	voltage at time t of the secondary winding for transformer with OLTC (i, j) installed in the secondary winding

The optimal values of the decision variables are determined by solving the following NLP problem:

$$\min \sum_{t\in\mathcal{T}}\left[\sum_{g\in\mathcal{G}}\left(c_{g,t}^{\Delta P+}\Delta P_{g,t}^{+}+c_{g,t}^{\Delta P-}\Delta P_{g,t}^{-}+c_{g,t}^{\Delta Q+}\Delta Q_{g,t}^{+}+c_{g,t}^{\Delta Q-}\Delta Q_{g,t}^{-}\right)\right.$$
$$\left.+\sum_{b\in\mathcal{B}}\left(c_{b,t}^{\text{out},P}P_{b,t}^{\text{out}}+c_{b,t}^{\text{in},P}P_{b,t}^{\text{in}}+c_{b,t}^{\text{out},Q}Q_{b,t}^{\text{out}}+c_{b,t}^{\text{in},Q}Q_{b,t}^{\text{in}}\right)\right] \tag{16}$$

subject to

$$P_{g,t}=P_{g,t}^{0}+\Delta P_{g,t}^{+}-\Delta P_{g,t}^{-} \qquad g\in\mathcal{G},t\in\mathcal{T} \tag{17}$$
$$Q_{g,t}=Q_{g,t}^{0}+\Delta Q_{g,t}^{+}-\Delta Q_{g,t}^{-} \qquad g\in\mathcal{G},t\in\mathcal{T} \tag{18}$$
$$\sum_{g\in\mathcal{G}_i}P_{g,t}=C_{i,t}+V_{i,t}^{2}G_i-\sum_{(i,j)\in\mathcal{L}}TA_{i,j,t} \qquad i\in\mathcal{N},t\in\mathcal{T} \tag{19}$$
$$\sum_{g\in\mathcal{G}_i}Q_{g,t}=D_{i,t}+V_{i,t}^{2}B_i-\sum_{(i,j)\in\mathcal{L}}TR_{i,j,t} \qquad i\in\mathcal{N},t\in\mathcal{T} \tag{20}$$
$$\eta_b^{\text{out}}P_{b,t}^{\text{out}}-\eta_b^{\text{in}}P_{b,t}^{\text{in}}=TA_{i(b),j(b),t} \qquad b\in\mathcal{B},t\in\mathcal{T} \tag{21}$$
$$Q_{b,t}=TA_{i(b),j(b),t} \qquad b\in\mathcal{B},t\in\mathcal{T} \tag{22}$$
$$P_{b,t}=P_{b,t}^{\text{out}}-P_{b,t}^{\text{in}} \qquad b\in\mathcal{B},t\in\mathcal{T} \tag{23}$$
$$\underline{E}_b\le E_b^{0}-\sum_{\tau\le t}P_{b,\tau}\le\overline{E}_b \qquad b\in\mathcal{B},t\in\mathcal{T} \tag{24}$$

$$TA_{i,j,t} = \left(\frac{\sin\delta_{i,j}}{Z_{i,j}} + \frac{X_{i,j}}{2}\right) V_{i,t}^2 + \frac{\sin\left(\theta_{i,t} - \theta_{j,t} - \delta_{i,j}\right)}{Z_{i,j}} V_{i,t} V_{j,t}$$
$$(i,j) \in \mathscr{L} \setminus (\mathscr{L}_1^{TC} \cup \mathscr{L}_2^{TC}), t \in \mathscr{T} \tag{25}$$

$$TR_{i,j,t} = \left(\frac{\cos\delta_{i,j}}{Z_{i,j}} - \frac{Y_{i,j}}{2}\right) V_{i,t}^2 + \frac{\cos\left(\theta_{i,t} - \theta_{j,t} - \delta_{i,j}\right)}{Z_{i,j}} V_{i,t} V_{j,t}$$
$$(i,j) \in \mathscr{L} \setminus (\mathscr{L}_1^{TC} \cup \mathscr{L}_2^{TC}), t \in \mathscr{T} \tag{26}$$

$$TA_{i,j,t} = \frac{Vn_{i,j}^2}{TC_{i,j,t}^2}\left(\frac{\sin\delta_{i,j}}{Z_{i,j}} + \frac{X_{i,j}}{2}\right) V_{i,t}^2 + \frac{Vn_{i,j}}{TC_{i,j,t}} \frac{\sin\left(\theta_{i,t} - \theta_{j,t} - \delta_{i,j}\right)}{Z_{i,j}} V_{i,t} V_{j,t}$$
$$(i,j) \in \mathscr{L}_1^{TC}, t \in \mathscr{T} \tag{27}$$

$$TR_{i,j,t} = \frac{Vn_{i,j}^2}{TC_{i,j,t}^2}\left(\frac{\cos\delta_{i,j}}{Z_{i,j}} - \frac{Y_{i,j}}{2}\right) V_{i,t}^2 + \frac{Vn_{i,j}}{TC_{i,j,t}} \frac{\cos\left(\theta_{i,t} - \theta_{j,t} - \delta_{i,j}\right)}{Z_{i,j}} V_{i,t} V_{j,t}$$
$$(i,j) \in \mathscr{L}_1^{TC}, t \in \mathscr{T} \tag{28}$$

$$TA_{i,j,t} = \left(\frac{\sin\delta_{i,j}}{Z_{i,j}} + \frac{X_{i,j}}{2}\right) V_{i,t}^2 + \frac{Vn_{j,i}}{TC_{j,i,t}} \frac{\sin\left(\theta_{i,t} - \theta_{j,t} - \delta_{i,j}\right)}{Z_{i,j}} V_{i,t} V_{j,t}$$
$$(i,j) \in \mathscr{L}_2^{TC}, t \in \mathscr{T} \tag{29}$$

$$TR_{i,j,t} = \left(\frac{\cos\delta_{i,j}}{Z_{i,j}} - \frac{Y_{i,j}}{2}\right) V_{i,t}^2 + \frac{Vn_{j,i}}{TC_{j,i,t}} \frac{\cos\left(\theta_{i,t} - \theta_{j,t} - \delta_{i,j}\right)}{Z_{i,j}} V_{i,t} V_{j,t}$$
$$(i,j) \in \mathscr{L}_2^{TC}, t \in \mathscr{T} \tag{30}$$

$$TI_{i,j,t} = \frac{1}{\sqrt{3}V_{i,t}} \sqrt{TA_{i,j,t}^2 + TR_{i,j,t}^2} \qquad (i,j) \in \mathscr{L}, t \in \mathscr{T} \tag{31}$$

$$\underline{\theta}_i \le \theta_{i,t} \le \overline{\theta}_i \qquad i \in \mathscr{N}, t \in \mathscr{T} \tag{32}$$

$$\underline{V}_i \le V_{i,t} \le \overline{V}_i \qquad i \in \mathscr{N}, t \in \mathscr{T} \tag{33}$$

$$\underline{P}_g \le P_{g,t} \le \overline{P}_g \qquad g \in \mathscr{G}, t \in \mathscr{T} \tag{34}$$

$$\underline{Q}_g \le Q_{g,t} \le \overline{Q}_g \qquad g \in \mathscr{G}, t \in \mathscr{T} \tag{35}$$

$$(1 - d_{i,j}) Vn_{i,j} \le TC_{i,j,t} \le (1 + u_{i,j}) Vn_{i,j} \qquad (i,j) \in \mathscr{L}_1^{TC}, t \in \mathscr{T} \tag{36}$$

$$(1 - d_{j,i}) Vn_{j,i} \le TC_{j,i,t} \le (1 + u_{j,i}) Vn_{j,i} \qquad (i,j) \in \mathscr{L}_2^{TC}, t \in \mathscr{T} \tag{37}$$

$$\underline{TA}_{i,j} \le TA_{i,j,t} \le \overline{TA}_{i,j} \qquad (i,j) \in \mathscr{L}, t \in \mathscr{T} \tag{38}$$

$$\underline{TR}_{i,j} \le TR_{i,j,t} \le \overline{TR}_{i,j} \qquad (i,j) \in \mathscr{L}, t \in \mathscr{T} \tag{39}$$

$$\underline{TI}_{i,j} \le TI_{i,j,t} \le \overline{TI}_{i,j} \qquad (i,j) \in \mathscr{L}, t \in \mathscr{T} \tag{40}$$

$$\Delta P_{g,t}^+, \Delta P_{g,t}^-, \Delta Q_{g,t}^+, \Delta Q_{g,t}^+, P_{b,t}^{\text{out}}, P_{b,t}^{\text{in}} \ge 0 \qquad g \in \mathscr{G}, b \in \mathscr{B}, t \in \mathscr{T} \tag{41}$$

The optimization model used in Step 2 is based on a classical Optimal Power Flow as in [8, 9], suitably modified to account for power redispatch problems. Given the resources selected by the MILP model in Step 1, the model in Step 2 determines:

- active and reactive power productions of controllable generators,
- modules and phases of voltages of all nodes,
- current flows in lines,
- rated voltage of tap-changer transformers,

so as to minimize redispatch costs, while satisfying the technical constraints

- load-flow equations, i.e. balance of active and reactive power at every node,
- equations of transit of active and reactive power in all lines,
- equations of current transits, for ensuring security,
- generators capability curves that define the feasible values of active and reactive power production for each generator,
- bound constraints on power productions, current transits, voltage modules and phases.

Equations (17) and (18) establish a relation between power output and power variations. Equations (19) and (20) represent the nodal balance constraints. Equations (21) define the active power exchanges between the storage devices and the network, where charge and discharge losses are taken into account, while Eq. (22) are related to reactive power exchanges between storage devices and the network. Equation (23) define the net power discharge of storage devices, while constraints (24) define the bounds on the energy level at time t. Equations (25)–(30) define the power flow through lines and transformers. Equation (31) defines the current transit in lines and transformers. Box constraints (32)–(40) are the upper and lower bounds on phase angles and nodal voltages of nodes, on active and reactive power outputs of generators, on voltages of tap changers, on active and reactive power transits and current transits on lines and transformers.

An application of the introduced framework may be found in [4]. The proposed procedure can be used as a simulation tool for the DSO to optimize the configuration of MV network, e.g. determine effective positions of storage units. It can also be used as a simulation tool for regulators to analyse the impact of costs associated with the usage of controllable resources.

4 Fast Estimation of Equivalent Capability for Active Distribution Networks

Another relevant application of OPF tools in the operation of medium-voltage grids is related to possible future market interactions between distribution and transmission grids [22]. Indeed, the contribution of both distributed renewable energy sources

and flexible loads (demand side management) on the share of electricity production is going to be more relevant [7] in a near future and the stochastic and non-programmable nature of many renewable resources, along with the decrease of the conventional generation, will impact on the stability of the power system. In order to maintain an adequate balancing reserve, distributed resources are likely to be allowed to participate in the ancillary service market (e.g. voltage management of the transmission network). However, the contribution of the distribution network needs to be considered together with the related network constraints. Different coordination schemes, with different potentialities depending on the market evolution, have been recently proposed to foster the participation of distribution resources to the balancing market [10]: in these schemes, the interface between transmission and distribution network, the so-called *Point of Common Coupling* (PCC), plays a fundamental role since the operation of the HV network requires detailed information on the actual flexibility of each transmission node. Fast and efficient methods need to be developed for computing the equivalent P-Q capability as seen from the HV node, since the actual power provision from resources on the MV grid is affected by the constraints on the MV grid operation; moreover, the active power can change in real time depending on the availability of the primary source. In [22] different methods for estimating the active distribution networks, capabilities have been investigated. The proposed approach for computing the equivalent capability of the distribution network is based on the OPF tool used in Step 2 of the procedure presented in the previous section, with the additional consideration of circular and triangular capabilities of generators, represented by the following constraints:

$$\underline{\varphi}_g P_{g,t} \leq Q_{g,t} \leq \overline{\varphi}_g P_{g,t} \qquad g \in \mathscr{G}^T, t \in \mathscr{T} \tag{42}$$

$$\sqrt{P_{g,t}^2 + Q_{g,t}^2} \leq \left|\overline{S}_g\right| \qquad g \in \mathscr{G}^C, t \in \mathscr{T} \tag{43}$$

where $\mathscr{G}^T$ is the set of generators subject to triangular capability, $\mathscr{G}^C$ is the set of generators subject to circular capability, $\underline{\varphi}_g$ and $\overline{\varphi}_g$ are the minimum value and the maximum value, respectively, of the ratio of reactive power output to active power output for generator g, and $\overline{S}_g$ is the maximum apparent power of generator g. The resulting model allows taking into account resources capabilities, local controllers and intertemporal constraints [19].

When network constraints are not taken into account, the aggregated flexibility can be easily calculated by algebraically summing all the capabilities of the available resources: in the test case considered in [22] and reported in Fig. 2, this aggregated flexibility is similar to a trapezium (green curve in Fig. 3) with rounded edges, due to the circular capability of the storage device. However, this theoretical capability cannot be entirely ensured because of the network operational limits (i.e. voltage and current constraints). These constraints bound the capability of the network when high share of both active power and reactive power has to be exchanged (red curve in Fig. 3).

resource (g)	capability shape	$\overline{\lvert S \rvert}_g$	Q_g^0	$\underline{Q}_g$	$\overline{Q}_g$	P_g^0	$\underline{P}_g$	$\overline{P}_g$	$c_g^{\Delta P^+}$	$c_g^{\Delta P^-}$
		[MVA]	[Mvar]	[Mvar]	[Mvar]	[MW]	[MW]	[MW]	[€/MW]	[€/MW]
Storage	circular	1.1	0	–1	1	0	–1	1	50	0
DG 4	circular	2	0	–1	1	1	1	2	100	80
DG 1	triangular ($pf = 0.9$)	2	0	–0.5	0.5	1	0	1	120	100
DG 2	triangular ($pf = 0.9$)	1	0	–0.5	0.5	0.5	0	0.5	120	100
DG 3	triangular ($pf = 0.9$)	2	0	–0.5	0.5	0	0	1	120	100
Load	constant ($pf = 0.9$)	2	–0.4	–0.4	0	–0.8	–0.8	0	220	—

Fig. 2 Example of distributed resources of a medium-voltage network

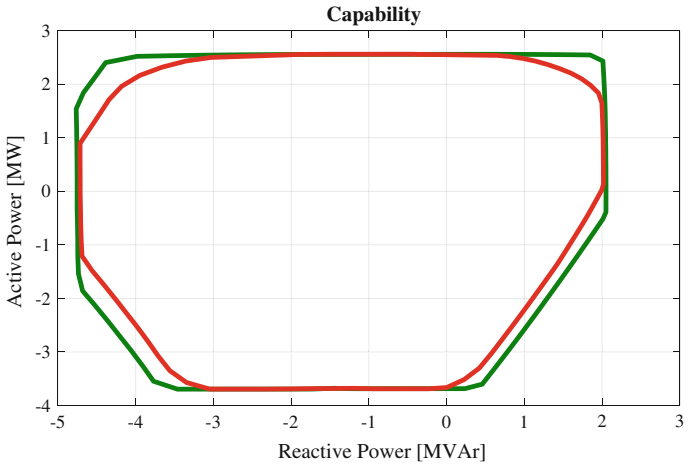


Fig. 3 Unconstrained capability (green) versus network constrained capability (red)

The inner frontier, in red colour, determines the area corresponding to the set of activations and/or modulations of the connected flexible resources that do not determine network congestions. The constrained capability could be potentially computed by defining a grid of (P-Q) values and, for each of the defined couples, solving an OPF problem to check if the active and reactive power can be exchanged with the HV grid. In case the OPF returns an infeasibility, then the point lies outside of the capability area. Albeit defining a quite accurate method of inspection, it is time-consuming to run this methodology with an acceptable resolution. This aspect is extremely important, especially for real-time markets (such as the balancing one) in which the prompt

estimation of the ancillary services provision is fundamental. Moreover, since the working point of operations of distribution grids is subject to continuous variations, the reconstruction of the aggregated capability has to be frequently reprocessed. A faster procedure can be adopted using a conventional OPF tool. The basic principle of calculation can be summarized with the following steps:

1. a dummy unit is added to the network model in correspondence of the PCC;
2. a positive cost is assigned to the dummy unit when its power exchange is nonzero;
3. zero cost is assigned to distribution flexible resources;
4. a power exchange (P_{PCC}, Q_{PCC}) at the PCC is imposed, which has to be outside the capability area determined without considering the network constraints ;
5. a starting operational point (P_0, Q_0) is defined for the network;
6. the OPF is performed.

The general principle behind this method is that the dummy unit will exchange the minimum amount of active and reactive power (P_{FU} and Q_{FU}, respectively), fulfilling the constraints. Therefore, the OPF returns a situation in which the distribution resources will try to exchange the maximum power to minimize the dummy unit contribution (which has a cost) and to maintain the network in its safe operation area. The dummy unit is modelled in order to exchange power with a fixed power factor ($\tan(\phi_{FU})$) and with a cost proportional to $\mid P_{FU} \mid$. Thanks to this, the OPF solution lies on an arbitrarily selected straight line, for which the slope and the position on the (P-Q) plane depend on the imposed power exchange (P_{PCC},Q_{PCC}) and on $\tan(\phi_{FU})$. The capability curve is scanned by polling a series of OPFs for which different PCC power exchanges and FU power factors have been imposed as shown in Fig. 4. According to the working principle describe above, each OPF

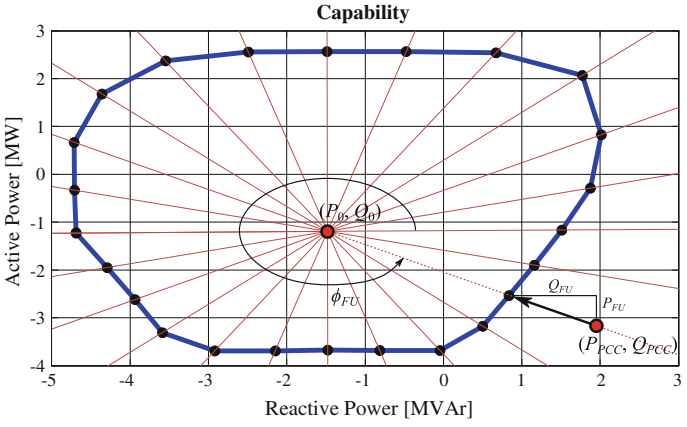


Fig. 4 Radial reconstruction of equivalent capability (fixed power factor method)

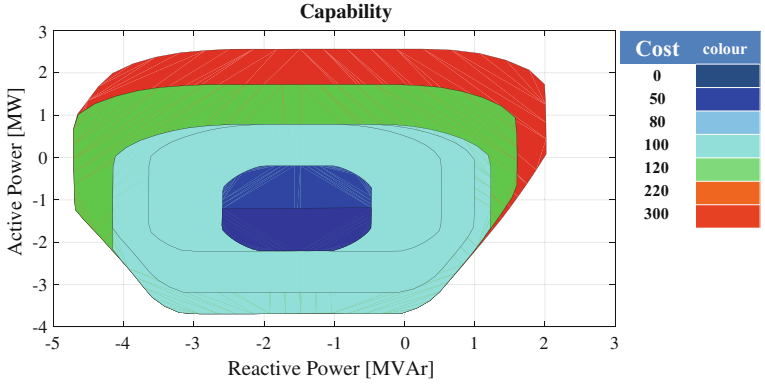


Fig. 5 Maximum capability for increasing number of resources according to the marginal cost, computed with the proportional cost method

converges on a point that corresponds to the intersection between the distribution network capability curve and the selected straight line.

Finally, the proposed methods can be adapted in order to determine an approximation of the costs associated with the (P-Q) points belonging to the capability area. This can be obtained by reprocessing the estimation methods and activating a limited set of resources which is then gradually increased (the first process is executed for the cheapest flexible device, the next one includes also the second cheapest units, etc.). With this procedure, a series of concentric capability areas can be extracted (with the desired cost resolution). Figure 5 reports the obtained results.

The frequent recalibration of the capability area, due to changes in the network conditions, calls for the usage of an efficient solution method for the OPF tool. Primal Dual Interior Point methods, with the Mehrotra Predictor-Corrector modification, provide a suitable approach for the solution of this kind of problem [13, 16–18].

5 Conclusions

The shift towards a decentralized, RES-based power system requires advanced techniques for integrating distributed resources in the electricity grid. Two Optimal Power Flow-based tools have been presented which are expected to be useful for planning and operation of medium-voltage networks in the near future. The first application allows for an efficient use of the network resources and provides assistance in interfacing the medium-voltage grid with the high-voltage grid. The non-convexity of the AC OPF models and the mixed integer nature of selecting the adequate resources require defining special procedures to compute a solution for grid operation. The case studies suggest that the usage of these tools for distribution networks is beneficial for voltage control and for relieving line congestions, thus providing power quality and network security as well as ensuring a continuous provision to the customers. We have also shown that OPF tools can be efficiently used for providing a market interface with the high-voltage grid and allow the distributed resources to participate

in the provision of ancillary services to the transmission network. Namely, the OPF tool can be used to build the equivalent capability region for the provision of active and reactive power from the distribution network to the transmission network and can be used, to some extent, to define bids to be supplied to the transmission network. Nevertheless, these operations require to be frequently reprocessed as the network conditions shift over time. This requires the Distribution System Operator to use efficient algorithms and software for the management of the distribution network.

References

1. Ackermann, T., Andersson, G., Söder, L.: Distributed generation: a definition. Electr. Power Syst. Res. **57**(3), 195–204 (2001)
2. Akorede, M.F., Hizam, H., Pouresmaeil, E.: Distributed energy resources and benefits to the environment. Renew. Sustain. Energy Rev. **14**(2), 724–734 (2010)
3. Barker, P.P., De Mello, R.W.: Determining the impact of distributed generation on power systems. I. Radial distribution systems. In: Power Engineering Society Summer Meeting, Seattle, WA, vol. 3, pp. 1645–1656 (2000)
4. Bosisio, A., Moneta, D., Vespucci, M.T., Zigrino, S.: A procedure for the optimal management of medium-voltage AC networks with distributed generation and storage devices. Procedia—Social Behavioral Sc. **108**, 164–186 (2013)
5. Buijs, P., et al.: Transmission investment problems in europe: going beyond standard solutions. Energy Policy **39**(3), 1794–1801 (2011)
6. Brown, R.E.: Impact of smart grid on distribution system design. Power and Energy Society General Meeting—IEEE Conversion and Delivery of Electrical Energy in the 21st Century. Pittsburg, IEEE, pp. 1–4 (2008)
7. European Commission, 2030 Energy Strategy, available on https://ec.europa.eu/energy/en/topics/energy-strategy/2030-energy-strategy
8. Garzillo, A., Innorta, M., Ricci, M.: The problem of the active and reactive optimum power dispatching solved by utilizing a primal-dual interior point method. Int. J. Electr. Power Energy Syst. **20**(6), 427–434 (1998)
9. Garzillo, A., Innorta, M., Ricci, M.: The flexibility of interior point based optimal power flow algorithms facing critical network situations. Int. J. Electr. Power Energy Syst. **21**(8), 579–584 (1999)
10. H2020: SmartNet EU project Deliverable 1.3, Basic Models for TSO-DSO coordination, available on http://smartnet-project.eu/wp-content/uploads/2016/12/D1.3_20161202_V1.0.pdf (2016)
11. Herzog A., Lipman T., Edwards J.: Renewable energy: a viable choice. Environment (December), 1–34 (2001)
12. Hinrichs, D., Conbere, S., Lobash, M.: Taking control of power supplies. In: Building Operating Management (July), http://www.findarticles.com/p/articles/mi_qa3922/is_200207/ai_n9110155 (2002)
13. Huneault, M., Galiana, F.: A survey of the optimal power flow literature. IEEE Trans. Power Syst. **6**(2), 762–770 (1991)
14. International Energy Agency: Distributed generation in liberalized electricity markets. International Energy Agency, Paris (2002)
15. International Energy Agency: World Energy Outlook 2008. International Energy Agency, Paris (2008)
16. Mehrotra S.: On the implementation of a (primal-dual) interior point method, Tech. Report 90-03, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL (1990)

17. Momoh, J.A., El-Hawary, M., Adapa, R.: A review of selected optimal power flow literature to 1993. Part I: Nonlinear and quadratic programming approaches. IEEE Trans. Power Syst. **14**(1), 96–104 (1999)
18. Momoh, J.A., El-Hawary, M., Adapa, R.: A review of selected optimal power flow literature to 1993 Part II: Newton, linear programming and interior point methods. IEEE Trans. Power Syst. **14**(1), 105–111 (1999)
19. Moneta, D., Gelmini, A., Carlini, C., Belotti, M.: Storage units: possible improvements for voltage control of MV distribution networks. In: Proceedings 17th Power Systems Computation Conference (PSCC) (2011)
20. Lin, J.: Power outage hits industrial park hard. Taipei Times (April 11), p. 10 (2004)
21. Owens, B.: The rise of distributed power. http://www.eenews.net/assets/2014/02/25/document_gw_02.pdf (2014)
22. Rossi, M., Moneta, D., Viganò, G., Vespucci, M.T., Pisciella, P.: Fast estimation of equivalent capability for active distribution networks. In: 24th Conference on Electricity Networks, Glasgow, 12–15, June 2017
23. Silberglitt, R., Ettedgui, E., Hove, A.: Strengthening the Grid: Effect of High-Temperature Superconducting Power Technologies on Reliability, Power Transfer Capacity, and Energy Use. http://www.rand.org/publications/MR/MR1531/ (2002)
24. Viral, R., Khatod, D.: Optimal planning of distributed generation systems in distribution system: a review. Renew. Sustain. Energy Rev. **16**, 5146–5165 (2012)
25. Vita, V., Alimardan, T., Ekonomou, L.: The impact of distributed generation in the distribution networks' voltage profile and energy losses. In: IEEE European Modelling Symposium (EMS), Madrid, pp. 260–265 (2015)

Equilibria and Weighted Complementarity Problems



Florian A. Potra

Abstract We review some results about market equilibria and weighted complementarity problems (wCP). The latter problem consists in finding a pair of vectors (x, s) belonging to the intersection of a manifold with a cone, such that their product in a certain algebra, $x \circ s$, equals a given weight vector w. When w is the zero vector, then wCP reduces to a Complementarity Problem (CP). The motivation for introducing the more general notion of a wCP lies in the fact that several equilibrium problems in economics can be formulated in a natural way as wCP. Moreover, those formulations lend themselves to the development of highly efficient algorithms for solving the corresponding equilibrium problems. For example, Fisher's competitive market equilibrium model can be formulated as a wCP that can be efficiently solved by interior-point methods. The Quadratic Programming and Weighted Centering problem, which generalizes the notion of a Linear Programming and Weighted Centering problem proposed by Anstreicher, can also be formulated as a special linear monotone wCP. The paper summarizes some previous results about the convergence of two interior-point methods for general monotone linear wCP and a corrector-predictor method for sufficient linear wCP. Some recent smoothing Newton methods for monotone linear wCP are also presented.

Keywords Weighted complementarity · Interior-point · Path-following Fisher equilibrium

1 Introduction

The aim of this paper is to review the theoretical properties of the weighted complementarity problem (wCP), a notion introduced by the author in [50], to present some numerical methods for its numerical solution, and to describe several applications to significant problems in economics, science, and engineering.

F. A. Potra (✉)
Department of Mathematics and Statistics, University of Maryland Baltimore County, 1000 Hilltop Circle, Baltimore, MD 22150, USA
e-mail: potra@umbc.edu

© Springer International Publishing AG, part of Springer Nature 2018

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*,
Springer Proceedings in Mathematics & Statistics 235,
https://doi.org/10.1007/978-3-319-90026-1_12

In what follows, we first review some basic facts about the complementarity problem (CP), and then we present the notion of wCP. We try to make the case that not only is wCP a nontrivial generalization of CP, but that formulating an equilibrium problem as a wCP leads to more efficient algorithms for its numerical solution than with known formulations. We also show that wCP is a notion worthy of study in itself. Since it substantially extends CP, one has to determine which properties of CP can easily be generalized to wCP, and which require completely new approaches.

Interior-point methods can be easily extended for solving wCP. By using appropriate extensions of complementarity functions, wCP can be reformulated as a semismooth system of nonlinear equations. This opens the possibility of using semismooth Newton techniques for solving wCP.

The motivation for introducing the more general notion of a wCP lies in the fact that several equilibrium problems in economics can be formulated in a natural way as wCP. We give some details about the Fisher and the Arrow–Debreu competitive market equilibrium problems. We have chosen these problems for illustration because they have been known for a long time (the former was formulated by the American economist Irving Fisher in 1891, while the latter goes back to the work of the French economist Léon Walras from 1874), and because of new exciting results about the computability of their equilibria. The Fisher market equilibrium problem can be modeled in a very natural way as a wCP, and the Arrow–Debreu competitive market equilibrium problem (due to Nobel prize laureates Kenneth Joseph Arrow and Gerard Debreu) can be formulated as a self-dual wCP. In 2009, Chen et al. [12] proved that the 2-player Nash equilibrium problem is PPAD-hard to compute (see also [16]). From this result, it follows that Arrow-Debreu market equilibria are also PPAD-hard to compute [16]. Vazirani and Yannakakis [59] proved that under mild sufficient conditions, Fisher and Arrow-Debreu markets under additively separable, piecewise-linear, concave utility functions (PLC) are in PPAD and that finding an equilibrium for Fisher markets in this setting is PPAD-hard. This effectively rules out the existence of efficient algorithms for such markets.

The challenge is to come up with models that capture some essential elements of real markets, admit equilibria, and for which efficient algorithms can be found. It is known that the Fisher and Arrow–Debreu competitive market equilibrium problems with linear utilities can be solved in weakly polynomial time [18, 63]. If the Fisher problem has a special structure, like the one arising in rate control for communication networks, then it is possible to solve it by a strongly polynomial time exact combinatorial algorithm [28]. Goel and Vazirani showed in 2011 that introducing perfect price discrimination into the Fisher model with PLC utilities renders its equilibrium polynomial time computable. By considering a class of piecewise-linear utility functions that includes Leontief's utility functions, but is different from the class considered in [59], Yinyu Ye developed a weakly polynomial time algorithm for solving the corresponding Fisher equilibrium problem [62]. Ye also showed that solving the Arrow-Debreu problem with the same class of utility functions is equivalent to solving a linear complementarity problem whose algorithmic complexity is finite but not polynomially bounded (see also [13]). In a remarkable paper [65], Zhu, Dang, and Ye present a fully polynomial time approximation scheme for approxi-

mating the solution of a symmetric matrix linear complementarity problem arising from the Nash and Arrow–Debreu competitive economy equilibria. Applications to other problems with different utility functions, including global trading and dynamic spectrum management problems, are also given.

Very good computational complexity results for the Fisher and Arrow-Debreu problems with linear utilities were obtained by Ye [63] in 2008. While Ye's algorithm for solving the Fisher problem is a clearly defined modified path-following algorithm, he only showed that an interior-point method, based on the theory of self-concordant barriers from [47], could be constructed for solving the Arrow–Debreu competitive market equilibrium problem, without giving any specific details on that algorithm.

The computational complexity results for the Fisher problem from [63] have been improved by Anstreicher [5]. He proposes a generalization of the Eisenberg–Gale formulation of the Fisher problem [20], called Linear Programming and Weighted Centering (LPWC) problem, and shows that it possesses a natural dual problem DPWC. He obtains a lower iteration complexity for DPWC by utilizing a combination of the volumetric [4, 56] and logarithmic [47] barriers. It turns out that LPWC generalizes both linear programming (LP) and the problem of finding the weighted analytic center of a polytope [7, 27]. Subsequently, the present author [50] proposed a generalization of LPWC, called Quadratic Programming and Weighted Centering (QPWC), and showed that this problem and its dual lead to a monotone linear wCP.

A large variety of equilibrium problems can be formulated as wCP. As mentioned earlier, this has already been done for the Fisher and Arrow-Debreu models with linear utility functions. We believe that the Arrow-Debreu Nash bargaining game from [58], the perfect price discrimination market model [29], the Fisher model with Leontief utility functions [62], and the Eisenberg–Gale markets [38] can also be phrased as wCPs. The scope of applications of wCP extends beyond market equilibrium problems. For example, they can be applied to equilibrium problems arising in atmospheric chemistry [2, 11, 42] and multibody dynamics [26, 48, 55].

Conventions. We denote by $I\!N$ the set of all nonnegative integers. $I\!R$, $I\!R_+$, and $I\!R_{++}$ denote the set of real, nonnegative real, and positive real numbers, respectively. The symbol e represents the vector of all ones, with dimension given by the context. We denote by $\log t$ the natural logarithm of t.

We denote componentwise operations on vectors by the usual notations for real numbers. Thus, given two vectors u, v of the same dimension we denote by uv the vector with components $u_i v_i$. This notation is consistent as long as componentwise operations always have precedence in relation to matrix operations. Note that $Auv = A(uv) \neq (Au)v$. We denote by u/v the vector whose ith component is equal to u_i/v_i if $u_i v_i \neq 0$, and to 0 if $u_i v_i = 0$. Also, if f is a scalar function and v is a vector, then $f(v)$ denotes the vector with components $f(v_i)$. For example, if $v \in I\!R^n_+$, then $\sqrt{v}$ denotes the vector with components $\sqrt{v_i}$. For a vector $v \in I\!R^n$ we denote $\max v = \max\{v_i \ : \ i = 1, \ldots, n\}$ and $\min v = \min\{v_i \ : \ i = 1, \ldots, n\}$. If $\|\,.\,\|$ is a vector norm on $I\!R^n$ and A is a matrix, then the operator norm induced by $\|\,.\,\|$ is defined by $\| A \| = \max\{\| Ax \| \ ; \ \| x \| = 1\}$. As a particular case we note that if U is the diagonal matrix defined by the vector u, then $\| U \|_2 = \| u \|_\infty$.

For any real number r we denote $r^+ = \max\{0, r\}$ and $r^- = \max\{0, -r\}$, so that $r = r^+ - r^-$. The same notation extends, componentwise, to real vectors.

Throughout this paper, we use the following MATLAB-like notation. If A, B, C are matrices that have the same number of rows, then $[A\ B\ C]$ or $[A,\ B,\ C]$ denotes the matrix obtained by concatenating their rows. If they have the same number of columns, then $[\ A\ ;\ B\ ;\ C\ ]$ denotes the matrix obtained by their column concatenation. In particular if u, v, w are (column) vectors, then $[\ u\ ;\ v\ ]$ and $[\ u\ ;\ v\ ;\ w\ ]$ to denote the column vectors $[u^T v^T]^T$ and $[u^T v^T w^T]^T$, respectively. Given a matrix P, we denote by Ran P its range (or column space) and by Ker P its kernel (or null space).

2 Background and Motivation

Since its introduction more than four decades ago, the notion of complementarity problem (CP) has proved to be an invaluable tool in many areas of research. Applications of complementarity problems are especially prevalent in economics and engineering. Research on complementarity has provided new methodologies, enhanced our understanding of algorithms, and permitted novel applications. A major source of complementarity problems arises from the optimality conditions of general constrained optimization problems. In particular, linear programming and convex quadratic programming can be written as linear complementarity problems. The linear complementarity problem (LCP) has played an important unifying role in operations research. The major developments prior to 1992 in the theory and applications of LCPs are described in the fundamental monograph [14]. The most significant applications analyzed there arise in quadratic programming, bimatrix games, matrix equilibrium, optimal invariant capital stock, optimal stopping, convex hulls, etc. Newer applications of both linear and nonlinear complementarity problems, together with a comprehensive analysis of their theoretical properties and solution algorithms, are described in the two-volume treatise [23]. Other applications of nonlinear complementarity problems are given in [24]. The applications in engineering include contact mechanics, structural mechanics, rigid-body dynamics, structural design, elasto-hydrodynamic lubrication, network design, and traffic equilibrium. Among the applications in economics, we mention multicommodity problems, optimal control, production and consumption problems, taxation and subsidies problems, invariant capital stock problems, game-theoretic models, and market equilibria. Another area of applications of complementarity problems is in financial mathematics. For example, the linear complementarity problem has been successfully applied to option pricing [1, 17, 36, 41, 44, 64]. Esteban-Bravo [21, 22] used interior-point methods for linear and nonlinear complementarity problems in order to compute solutions of various market equilibrium problems, but no computational complexity results are known for those algorithms. Interior-point methods and penalty methods for equilibrium problems are also analyzed in [8].

The notion of wCP significantly extends the notion of CP. Generally speaking, wCP consists in finding a pair of vectors (x, s) belonging to the intersection of a

manifold with a cone, such that their product in a certain algebra, $x \circ s$, equals a given weight vector w. When w is the zero vector, then wCP reduces to CP. With nonzero weight vectors, the theory of wCP becomes more complicated than the theory of CP. However, as shown later in the paper, many of the essential properties of CP extend to wCP. Also, many interior-point methods for CP can be extended to efficient algorithms for solving wCP. Several complementarity functions, such as the Fischer–Burmeister function [25], can be modified to write wCP as a system of nonlinear equations, which can then be solved by semismooth Newton methods.

We have been motivated to introduce the notion of wCP by the fact that wCP can be used for modeling a larger class of problems from science and engineering. Even when a problem can also be modeled by CP, the wCP model leads to a more efficient numerical solution method. For example, we show that the Fisher market equilibrium problem, which can be modeled as a *nonlinear* CP, can also be modeled as a *linear* wCP. The latter can be solved more efficiently than the former.

In the remainder of this section, we introduce CP and wCP in a very simple setting. Then we describe the Fisher market equilibrium problem, and model it both as a nonlinear CP and as a linear wCP. Finally, we comment on the corresponding solution algorithms.

2.1 Complementarity Problems Over $\mathbb{R}^n_+$

A general linear CP over the nonnegative orthant $\mathbb{R}^n_+$ of the n-dimensional Euclidean space $\mathbb{R}^n$ consists in finding vectors $x, s \in \mathbb{R}^n$, and $y \in \mathbb{R}^m$ such that

$$\begin{aligned} xs &= 0 \\ Px + Qs + Ry &= a \\ x, s &\geq 0. \end{aligned} \tag{1}$$

Here $P \in \mathbb{R}^{(n+m)\times n}$, $Q \in \mathbb{R}^{(n+m)\times n}$, $R \in \mathbb{R}^{(n+m)\times m}$ are given matrices, and $a \in \mathbb{R}^{n+m}$ is a given vector (the data of the problem). In the first equation above, xs is the componentwise product of the vectors x and s. The notation $x \geq 0$ means that all components of the vector x are nonnegative, i.e., $x \in \mathbb{R}^n_+$. We note that the relations $xs = 0$ and $x^T s = 0$ are equivalent for any $x, s \in \mathbb{R}^n_+$. Similarly, $x > 0$ means that all components of the vector x are positive, i.e., $x \in \mathbb{R}^n_{++}$.

The linear complementarity problem (1) is called monotone if

$$P\Delta x + Q\Delta s + R\Delta y = 0 \text{ implies } \Delta x^T \Delta s \geq 0\,. \tag{2}$$

The notion of monotony can be extended to nonlinear complementarity problems of the form

$$\begin{aligned} xs &= 0 \\ F(x, s, y) &= 0 \\ x, s &\geq 0, \end{aligned} \tag{3}$$

where $F : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^{n+m}$ is a given mapping. The nonlinear complementarity problem (3) is called monotone if

$$F(x + \Delta x, s + \Delta s, y + \Delta y) = F(x, s, y) \text{ implies } \Delta x^T \Delta s \geq 0\,. \tag{4}$$

We note that monotone linear complementarity problems can be solved very efficiently by a variety of interior-point methods (see, e.g., [49] and the literature cited therein). While, under a scaled Lipschitz assumption, it is possible to construct interior-point methods for monotone nonlinear CP [52], the efficiency of available solvers for monotone nonlinear CP lags much behind the efficiency of available solvers in the linear case.

2.2 Weighted Complementarity Problems over $\mathbb{R}^n_+$

If $w \in \mathbb{R}^n_+$ is a given nonnegative weight vector, then the corresponding linear wCP is defined as

$$\begin{aligned} xs &= w \\ Px + Qs + Ry &= a \\ x, s &\geq 0. \end{aligned} \tag{5}$$

Similarly, the corresponding nonlinear wCP is given by

$$\begin{aligned} xs &= w \\ F(x, s, y) &= 0 \\ x, s &\geq 0. \end{aligned} \tag{6}$$

The linear wCP (5) is called monotone if (2) is satisfied, while the nonlinear wCP (6) is called monotone if (4) holds.

2.3 The Fisher Equilibrium Problem

This problem considers a market composed of n_c consumers and n_p producers. Consumer i has a budget w_i to spend on buying goods from the producers in such a way that an individual utility function is maximized. The price equilibrium is an assignment of prices to goods, so that when every consumer buys a maximal bundle of goods then the market clears, meaning that all the money is spent and all the goods are sold. Without loss of generality, it is assumed that producer j has one unit of some good to sell. Let the individual utility function of consumer i be of the form

$$u_i = \sum_{j=1}^{n_p} u_{ij} x_{ij}, \tag{7}$$

where u_{ij} is the utility coefficient of consumer i for the good produced by producer j, and x_{ij} represents the amount of good bought by consumer i from producer j. It is assumed that the following inequalities are satisfied for all i and j:

$$w_i > 0, \quad u_{ij} \geq 0, \quad \sum_{k=1}^{n_c} u_{kj} > 0, \quad \sum_{k=1}^{n_p} u_{ik} > 0\,. \tag{8}$$

Under these assumptions, Eisenberg and Gale [20] proved that the market clearing prices are given by the optimal Lagrange multipliers for the first n_p equality constraints of the following convex optimization problem:

$$\begin{array}{ll} \underset{u_i,\, x_{ij}}{\text{maximize}} & \sum_{i=1}^{n_c} w_i \log u_i \\ \text{subject to} & \sum_{i=1}^{n_c} x_{ij} = 1, \qquad j = 1, \ldots, n_p \\ & u_i - \sum_{j=1}^{n_p} u_{ij} x_{ij} = 0\,, \; i = 1, \ldots, n_c \\ & u_i \geq 0,\; x_{ij} \geq 0\,, \qquad i = 1, \ldots, n_c\,,\;\; j = 1, \ldots, n_p. \end{array} \tag{9}$$

This optimization problem can be written under the form

$$\begin{array}{ll} \underset{x}{\text{maximize}} & \sum_{i=1}^{n} w_i \log x_i \\ \text{subject to} & Ax = b \\ & x \geq 0\,, \end{array} \tag{10}$$

where x is a n-dimensional vector, with $n = n_c(n_p + 1)$, having its first n_c coordinates formed by $u_1, \ldots, u_{n_c}$, and the remaining $n_c n_p$ coordinates consisting of the variables x_{ij}, i.e.,

$$x = [u_1, \ldots, u_{n_c}, x_{11}, \ldots, x_{1n_p}, x_{21}, \ldots, x_{2n_p}, \ldots, x_{n_c 1}, \ldots, x_{n_c n_p}]^T.$$

The n-dimensional weight vector w has its first n_c coordinates equal to $w_1, \ldots, w_{n_c}$ and the remaining coordinates equal to zero, and b is an m-dimensional vector having its first n_p coordinates equal to one and the remaining n_c coordinates equal to zero, i.e., $b = [e; 0]$, $e \in I\!R^{n_p}$. Here A is the $m \times n$-matrix, with $m = n_c + n_p$ given by

$$A = \begin{pmatrix} 1 & 0 & \cdots & 0 & a^{1T} & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & 0 & a^{2T} & \cdots & 0 \\ \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdots & \cdot \\ 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & a^{n_c T} \\ 0 & 0 & \cdots & 0 & e_1^T & e_1^T & \cdots & e_1^T \\ 0 & 0 & \cdots & 0 & e_2^T & e_2^T & \cdots & e_2^T \\ \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdots & \cdot \\ 0 & 0 & \cdots & 0 & e_{n_p}^T & e_{n_p}^T & \cdots & e_{n_p}^T \end{pmatrix},$$

where

$$a^1 = -[u_{1\,1}, \ldots, u_{1\,n_p}]^T,\ a^2 = -[u_{2\,1}, \ldots, u_{2\,n_p}]^T,\ \ldots, a^{n_c} = -[u_{n_c\,1}, \ldots, u_{n_c\,n_p}]^T.$$

Let us now consider a general optimization problem of the form (10) with arbitrary $w \in \mathbb{R}^n_+$, $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$. We note that if x is the solution of (10), then we must have $x_i > 0$ whenever $w_i > 0$, so that in this case the product $w_i x_i^{-1}$ is well defined. If $w_i = 0$, then we take by definition $w_i x_i^{-1} = 0$ for any value of x_i. With this convention, the KKT conditions for (10) can be written as

$$\begin{aligned} xv &= 0 \\ wx^{-1} + v - A^T y &= 0 \\ Ax &= b \\ x, v &\geq 0. \end{aligned}$$

This is a nonlinear CP of the form (3) with

$$F(x, v, y) = \begin{pmatrix} wx^{-1} + v - A^T y \\ Ax - b \end{pmatrix}.$$

We note that $F(x, v, y)$ is defined for any $v \in \mathbb{R}^n$, $y \in \mathbb{R}^m$, and any $x \in \mathbb{R}^n$ such that $x_i \neq 0$ whenever $w_i > 0$. It is easily seen that F is monotone on its domain of definition.

By denoting $s = A^T y$, we have $s = wx^{-1} + v \geq 0$. Multiplying this equation by x, we obtain the following linear wCP:

$$\begin{aligned} xs &= w \\ s - A^T y &= 0 \\ Ax &= b \\ x, s &\geq 0. \end{aligned} \tag{11}$$

This a particular case of wCP (1) with

$$P = \begin{pmatrix} A \\ 0 \end{pmatrix},\ Q = \begin{pmatrix} 0 \\ I \end{pmatrix},\ R = \begin{pmatrix} 0 \\ -A^T \end{pmatrix},\ a = \begin{pmatrix} b \\ 0 \end{pmatrix},$$

which is obviously skew-symmetric in the sense that

$$P\Delta x + Q\Delta s + R\Delta y = 0 \text{ implies } \Delta x^T \Delta s = 0\,.$$

Since the optimization problem (10) is convex, one can develop interior-point methods for its numerical solution [47, 61]. Primal–dual interior-point methods for this problem are in fact damped modified Newton methods for the nonlinear CP (48). In the present paper, we present several interior point methods for the linear wCP (1) of Sect. 4.1, and we argue that they are more efficient.

If wCP (11) arises from the Fisher problem then, following Ye [63], we can use a starting point $z^0 = [x^0; s^0; y^0]$ in which the first n_c coordinates of x^0 are given by

$$u_i^0 = \frac{1}{n_p} \sum_{k=1}^{n_p} u_{ik}, \quad i = 1, \ldots, n_c \tag{12}$$

and the remaining $n_c n_p$ coordinates are equal to $1/n_p$,

$$x_{ij}^0 = \frac{1}{n_p}, \quad i = 1, \ldots, n_c\,, \;\; j = 1, \ldots, n_p. \tag{13}$$

We clearly have $Ax^0 = b$ and $x^0 > 0$. The dual vector $y^0 = [q^0; p^0]$ has components

$$q_i^0 = \frac{\widehat{\beta}}{u_i^0}\,, \; i = 1, \ldots, n_c\,; \quad p_j^0 = 2n_p\widehat{\beta}, \; j = 1, \ldots, n_p; \quad \widehat{\beta} = \frac{n_p + 1}{2n_p}\,\|w\|_\infty\,. \tag{14}$$

Finally, the components of the slack vector $s^0 = Ay^0$,

$$s^0 = [v_1^0, ..., v_{n_c}^0, s_{11}^0, \ldots, s_{1n_p}^0, s_{21}^0, \ldots, s_{2n_p}^0, \ldots, s_{n_c 1}^0, \ldots, s_{n_c n_p}^0]^T,$$

are given by

$$v_i^0 = q_i^0\,, \; i = 1, \ldots, n_c\,; \quad s_{ij}^0 = p_j^0 - q_i^0 u_{ij}\,, \; i = 1, \ldots, n_c\,, \; j = 1, \ldots, n_p\,. \tag{15}$$

2.4 The Quadratic Programming and Weighted Centering Problem

In this section, we introduce a more general convex optimization problem that leads to a monotone linear wCP. Given an $n \times n$ symmetric positive semidefinite matrix M, a full-rank $m \times n$ matrix A with $m < n$, and vectors $f \in \mathbb{R}^n$, $w \in \mathbb{R}^n_+$, $b \in \mathbb{R}^m$, we consider the following optimization problems:

$$\begin{array}{ll} \underset{x}{\text{minimize}} & \varphi(x) := \frac{1}{2}x^T Mx + f^T x - \sum_{i=1}^n w_i \log x_i \\ \text{subject to} & Ax = b \\ & x \geq 0\,, \end{array} \tag{16}$$

and

$$\begin{array}{ll} \underset{u,s,y}{\text{maximize}} & \varphi(u, s, y) := -\frac{1}{2}u^T Mu + b^T y + \sum_{i=1}^n w_i \log s_i + \sum_{i=1}^n w_i(1 - \log w_i) \\ \text{subject to} & s = Mu - A^T y + f \\ & s \geq 0. \end{array} \tag{17}$$

In (16) and (17), we have tacitly made the convention that if $w_i = 0$ then the corresponding terms $w_i \log x_i$, $w_i \log s_i$, $w_i(1 - \log w_i)$ are set to zero. By denoting

$$\mathscr{I} = \{i \in \{1, \ldots, n\} \ : \ w_i > 0\} \ , \tag{18}$$

we have

$$\begin{aligned} \sum_{i=1}^{n} w_i \log x_i &= \sum_{i \in \mathscr{I}} w_i \log x_i \,, \ \sum_{i=1}^{n} w_i \log s_i \\ &= \sum_{i \in \mathscr{I}} w_i \log s_i \,, \ \sum_{i=1}^{n} w_i \log w_i = \sum_{i \in \mathscr{I}} w_i \log w_i \,. \end{aligned}$$

We call (16) a Quadratic Programming and Weighted Centering (QPWC) problem. For $M = 0$ it reduces to the Linear Programming and Weighted Centering (LPWC) problem of [5]. We say that x is strictly feasible (or an interior point) for (16) if $Ax = b$ and $x > 0$. Similarly (u, s, y) is called strictly feasible (or an interior point) for (17) if $s = Mu - A^T y + f > 0$. We note that in the special case $M = 0$, Anstreicher [5] considers a more general notion of strict feasibility requiring only that $x_i > 0$, $s_i > 0$ for all $i \in \mathscr{I}$. Since in this paper we are concerned with interior-point methods, we only consider the more restrictive notion of strict feasibility defined above. The following theorem is proved in [50].

Theorem 1

1. *(weak duality) If x is feasible for (16) and (u, s, y) is feasible for (17), then $\varphi(x) \geq \varphi(u, s, y)$;*

2. *(optimality conditions) x is an optimal solution for (16) and (u, s, y) is an optimal solution for (17) if and only if*

$$xs = w, \quad Mx = Mu. \tag{19}$$

   In this case (x, s, y) is also an optimal solution for (17), and we have $\varphi(x) = \varphi(u, s, y) = \varphi(x, s, y)$;

3. *(strong duality) If (16) and (17) are strictly feasible, then they have optimal solutions x^*, (x^*, s^*, y^*) with $\varphi(x^*) = \varphi(x^*, s^*, y^*)$.*

The optimality conditions from the above theorem give rise to a linear wCP of the form (5) with

$$P = \begin{pmatrix} A \\ M \end{pmatrix}, \ Q = \begin{pmatrix} 0 \\ -I \end{pmatrix}, \ R = \begin{pmatrix} 0 \\ -A^T \end{pmatrix}, \ a = \begin{pmatrix} b \\ -f \end{pmatrix}.$$

It can be shown that this wCP is monotone. In the particular case when $M = 0$, the wCP is skew-symmetric.

2.5 The Arrow–Debreu Competitive Market Equilibrium and Self-dual wCPs

The Arrow–Debreu competitive market equilibrium model is based on Léon Walras' model from 1874 [60]. The main difference between the Arrow-Debreu and Fisher models is that in the former, each player is both producer and consumer, and more importantly, that the initial endowments of players are not given in advance, but are to be determined by the equilibrium conditions so that they coincide with the clearing market prices. If we take $n_c = n_p = l$ in (9), then the Fisher equilibrium can be computed by solving a wCP of the form (11), where A is a full-rank $m \times n$ matrix, with $m = 2l$, $w \in \mathbb{R}^l_{++}$, and $b \in \mathbb{R}^m$ has its first l components equal to 1 and the remaining l components equal to zero, i.e., $b = [e; 0]$. The first l components of the dual variable y represent the market clearing prices for Fisher's equilibrium model. When (11) comes from the Fisher model, then the following conditions are satisfied:

C1. The primal feasible set $\{x \in \mathbb{R}^n : Ax = b, \quad x \geq 0\}$ is bounded and has a nonempty interior.

C2. $A^T y \geq 0$ implies $y_1 \geq 0, \ldots, y_l \geq 0$.

Using the Kakutani fixed point theorem [32], Ye [63] proved that for any full-rank matrix $A \in \mathbb{R}^{m \times n}$, with $m = 2l < n$, that satisfies **C1** and **C2**, where $b = [e; 0]$, there is a vector $w \in \mathbb{R}^l_+$ such the solution of wCP (11) satisfies $y_1 = w_1, \ldots, y_l = w_l$. In other words, there is a vector of initial endowments w that coincide with the clearing prices obtained from the corresponding Fisher equilibrium. But this is exactly the Arrow–Debreu competitive market equilibrium problem. Therefore, the latter can be solved by finding a solution $z = [x; s; y] \in \mathbb{R}^{2n+m}$ of the following self-dual wCP:

$$\begin{aligned} x_i s_i &= y_i, \ i = 1, \ldots, l \\ x_i s_i &= 0, \ i = l+1, \ldots, n \\ s - A^T y &= 0 \\ Ax &= b \\ x, s &\geq 0. \end{aligned} \tag{20}$$

Since it is based on the Kakutani fixed point theorem, the proof of existence of a solution of the Arrow–Debreu competitive market equilibrium problem given by Ye [63] is nonconstructive, as is the original proof of Arrow and Debreu [6]. However, by using a self-concordant barrier function, and the theory from [47], Ye was able to prove that the computational complexity of the Arrow–Debreu competitive market equilibrium problem is weakly polynomial. More precisely, he proved that there is an interior-point algorithm for generating an ε-approximate solution to the Arrow-Debreu problem in $O(n \log(1/\varepsilon))$ iterations, where each iteration uses $O(n^3)$ arithmetic operations. This worst-case complexity bound is lower than the bounds obtained by Jain [37], who used the ellipsoid method. In the above cited paper, Ye defined a central path that converges to the solution of the Arrow-Debreu problem,

and suggested that primal–dual path-following Newton-based methods could eventually solve the Arrow-Debreu problem. We are not aware of any concrete results having been obtained so far along that line of research. Obviously, (20) can be naturally extended to a more general self-dual wCP of the form

$$\begin{aligned} x_i s_i &= y_i, \ \ i = 1, \ldots, l \\ x_i s_i &= 0, \ \ i = l+1, \ldots, n \\ Px + Qs + Ry &= a \\ x, s &\geq 0, \end{aligned} \tag{21}$$

where the matrices $P \in I\!R^{(n+m)\times n}$, $Q \in I\!R^{(n+m)\times n}$, $R \in I\!R^{(n+m)\times m}$ and the vector $a \in I\!R^{n+m}$ are given.

3 Two Interior-Point Methods for Solving Monotone Linear wCPs

In this section, we present a long-step path-following method and a predictor–corrector method for solving a general linear wCP of the form (5) that is monotone in the sense of (2). The long-step method may be interpreted as a generalization of McShane's largest-step algorithm [45]. The name "largest step algorithm" was given by Gonzaga [30] in the case of monotone complementarity problems (see also [9, 10]). The predictor–corrector method can be considered as a generalization of the Mizuno–Todd–Ye method [46], which was the first algorithm for solving linear programming problems having both polynomial complexity and superlinear convergence. The main difference between the two algorithms to be introduced in this section is that the long-step path-following algorithm uses only one matrix factorization per iteration while the predictor–corrector method uses two factorizations.

3.1 A Long-Step Path-Following Method

Let us denote the feasible set of wCP (5) by

$$\mathscr{F} = \{z = [x; s; y] \in I\!R^{2n+m} \ : \ Px + Qs + Ry = a, \ x \geq 0, \ s \geq 0\}. \tag{22}$$

Its relative interior

$$\mathscr{F}^0 = \{z = [x; s; y] \in I\!R^{2n+m} \ : \ Px + Qs + Ry = a, \ x > 0, \ s > 0\} \tag{23}$$

is called the set of strictly feasible (or interior) points. The solution set of the wCP is denoted by

$$\mathscr{S} = \{z = [x; s; y] \in \mathscr{F} \ : \ \|xs - w\|_2 \leq \varepsilon\}\,, \tag{24}$$

and the set of ε-approximate solutions by

$$\mathscr{S}_\varepsilon = \{z = [x; s; y] \in \mathscr{F} \ : \ \|xs - w\|_2 \leq \varepsilon\}\,. \tag{25}$$

Given a strictly feasible starting point $z^0 = [x^0, s^0; y^0] \in \mathscr{F}^0$, we denote

$$t_0 = \mu(z^0),\ c = x^0 s^0,\ \gamma = \frac{\min\ c}{t_0},\ \ w(t) = (1 - t/t_0)w + (t/t_0)c,\ t \in (0, t_0]. \tag{26}$$

We define the central path of wCP (5) emanating from z^0 as the set of all points $[t; z] = [t; x; s; y]$, with $t \in (0, t_0]$, satisfying

$$\begin{array}{r} xs = w(t) \\ Px + Qs + Ry = a \\ x > 0,\ s > 0\,. \end{array} \tag{27}$$

By construction, $[t^0; z^0]$ belongs to this path. We note that for the Fisher problem with starting point defined by (12)–(15), we have $t_0 = \| w \|_\infty$ and $\gamma > 1/2$.

Given a parameter α such that

$$0 \leq \frac{\gamma}{3} \leq \alpha \leq \frac{2\gamma}{3}\,, \tag{28}$$

we define the following neighborhood of the above central path:

$$\mathscr{N}_2(w, c, \alpha) = \{[t; z] = [t; x; s; y] \in (0, t_0] \times \mathscr{F}^0 \ : \ \|xs - w(t)\| \leq \alpha t\}. \tag{29}$$

For our starting point we have $x^0 s^0 = c = w(t_0)$, so that $[t_0, z^0] \in \mathscr{N}_2(w, c, \alpha)$.

At a typical iteration of our algorithm, we have a point $[t; z] \in \mathscr{N}_2(w, c, \alpha)$, for some $t \leq t_0$. Since $c \geq \gamma t_0\, e$, it follows that

$$xs \geq w(t) - \alpha t e \geq (t/t_0)c - \alpha t e \geq (\gamma - \alpha) t e = \beta t e, \quad \beta = \gamma - \alpha \geq \frac{\gamma}{3} \geq \frac{\alpha}{2}\,. \tag{30}$$

Let us denote

$$t(\theta) = (1 - \theta)t,\ z(\theta) = [x(\theta); s(\theta; y(\theta))] = [x + u(\theta); s + v(\theta); y + d(\theta)]\,, \tag{31}$$

where $u(\theta), v(\theta), d(\theta)$ are the solutions of the linear system

$$\left\{ \begin{array}{rcl} su(\theta) + xv(\theta) & = & w(t(\theta)) - xs \\ Pu(\theta) + Qv(\theta) + Rd(\theta) & = & 0 \end{array} \right.\,. \tag{32}$$

Using the stepsize

$$\theta^+ = \max\{\widehat{\theta} \in [0,1] \ : \ [t(\theta); z(\theta)] \in \mathcal{N}_2(w,a,\alpha), \ \forall \theta \in [0,\widehat{\theta}\,]\}, \tag{33}$$

we obtain the new point

$$[t_+; z^+] := [t(\theta^+); z(\theta^+)] \in \mathcal{N}_2(w,c,\alpha)\,, \tag{34}$$

and we can begin a new iteration.

In order to compute the stepsize (33), we first solve the two linear systems

$$\left\{\begin{array}{rcl} s\check{u} + x\check{v} & = & w(t) - xs \\ P\check{u} + Q\check{v} + R\check{d} & = & 0 \end{array}\right., \quad \left\{\begin{array}{rcl} s\hat{u} + x\hat{v} & = & w - xs \\ P\hat{u} + Q\hat{v} + R\hat{d} & = & 0 \end{array}\right. . \tag{35}$$

The solution of (32) can be written in the form

$$u(\theta) = (1-\theta)\check{u} + \theta\hat{u}, \ v(\theta) = (1-\theta)\check{v} + \theta\hat{v}, \ d(\theta) = (1-\theta)\check{d} + \theta\hat{d}. \tag{36}$$

From (31) and (32), we have

$$x(\theta)s(\theta) = w(t(\theta)) + u(\theta)s(\theta) = w(t) + \theta(t/t_0)(w-c) + u(\theta)s(\theta)\,. \tag{37}$$

By considering the quartic

$$\varphi(\theta) = \|x(\theta)s(\theta) - w(t(\theta))e\|_2^2 - \alpha^2 t(\theta)^2,$$

we deduce that

$$[t(\theta); z(\theta)] \in \mathcal{N}_2(w,c,\alpha) \quad \text{if and only if} \quad \varphi(\theta) \le 0.$$

It can be shown that $\varphi(0) = \left\| \check{u}\check{v} \right\|_2^2 - t^2\alpha^2 < 0$ and $\varphi(1) = \left\| \hat{u}\hat{v} \right\|_2^2 \ge 0$. Therefore, the quartic equation $\varphi(\theta) = 0$ has at least one root in the interval $(0,1]$ and our steplength can be computed as

$$\theta^+ = \text{ the smallest root of the quation } \varphi(\theta) = 0 \text{ from the interval } (0,1]. \tag{38}$$

Therefore, our algorithm can be formally defined as follows.

Algorithm 1 (*Largest Step*)
Given a starting point $z^0 = [x^0; s^0; y^0] \in \mathcal{F}^0$:
Consider the notation from (26);
Choose a parameter α satisfying (28);
Set $k \leftarrow 0$;
repeat
Set $z = [x; s; y] \leftarrow z^k$, $t \leftarrow t_k$;

Solve the linear systems (35);
Compute steplength θ^+ from (38);
Compute t_+, z^+ from (31), (34), and (36);
Set $\theta_k \leftarrow \theta^+$, $z^{k+1} \leftarrow z^+$, $t_{k+1} \leftarrow t_+$;
Set $k \leftarrow k+1$.
continue

We note that the linear systems in (35) have the same matrix, so that the algorithm requires only one matrix factorization per iteration. Let us denote

$$\rho = 1 + \frac{\|c - w\|_2}{t_0} \cdot \tag{39}$$

The following results are proved in [50].

Theorem 2 *If wCP (5) is monotone, then Algorithm 1 is well defined and generates an iteration sequence satisfying the following properties:*

$$[t_k; z^k] \in \mathcal{N}_2(w, c, \alpha);$$
$$t_{k+1} = (1 - \theta_k) t_k;$$
$$\theta_k \geq \frac{\alpha}{9\rho} \cdot$$

Corollary 1 *If wCP (5) is monotone, then Algorithm 1 finds an ε-approximate solution for this problem (i.e., a point $z \in \mathcal{S}_\varepsilon$, where $\mathcal{S}_\varepsilon$ is defined in (25)) in at most*

$$O\left(\frac{x^{0T}s^0/n + \left\| x^0 s^0 - w \right\|_2}{\min x^0 s^0} \log \frac{x^{0T}s^0/n + \left\| x^0 s^0 - w \right\|_2}{\varepsilon}\right)$$

iterations.

Corollary 2 *When applied to the wCP generated by the Fisher problem with starting point z^0 given by (12)–(15), Algorithm 1 finds an ε-approximate solution for this problem in at most*

$$O\left(\sqrt{n_c n_p} \log \frac{(n_c + n_p)\|w\|_\infty}{\varepsilon}\right)$$

iterations.

3.2 A Predictor–Corrector Method

As mentioned in the previous subsection, Algorithm 1 requires only one matrix factorization per iteration. At a cost of two matrix factorizations per iteration, we can generalize the Mizuno–Todd–Ye predictor–corrector algorithm to our setting. The

purpose of the predictor is to improve as much as possible the optimality measure t while not departing too much from the central path (27). The algorithm depends on two parameters α and $\bar{\alpha}$, such that

$$\frac{\gamma}{3} \leq \alpha < \bar{\alpha} \leq \frac{2\gamma}{3}, \quad \frac{4\alpha}{3} \leq \bar{\alpha} \leq \sqrt{2}\alpha. \tag{40}$$

The above relations are satisfied, for example, by $\alpha = \sqrt{2}\gamma/3$ and $\bar{\alpha} = 2\gamma/3$.

3.2.1 The Predictor

At the beginning of the predictor step, we are given a point $[t; z] \in \mathcal{N}(w, c, \alpha)$ and we compute the predictor direction $[u; v; d]$ as the solution of the linear system

$$\begin{cases} su + xv = w - xs \\ Pu + Qv + Rd = 0 \end{cases}. \tag{41}$$

We define

$$\begin{gathered} x(\theta) = x + \theta u, \ s(\theta) = s + \theta v, \ y(\theta) = y + \theta d, \ t(\theta) = (1-\theta)t, \\ z(\theta) = [\, x(\theta)\,;\, s(\theta)\,;\, y(\theta)\,]. \end{gathered} \tag{42}$$

The stepsize along this direction is taken as

$$\bar{\theta} = \max\{\widehat{\theta} \in [0, 1] \ : \ [\, t(\theta)\,;\, z(\theta)\,] \in \mathcal{N}_2(w, c, \bar{\alpha}), \ \forall \theta \in [0, \widehat{\theta}\,]\}, \tag{43}$$

As shown in [50], $\bar{\theta}$ can be computed by solving two quadratic equations.

3.2.2 The Corrector

A corrector step usually follows a predictor step. It starts with a point $[\, t\,;\, z\,] \in \mathcal{N}(w, c, \bar{\alpha})$ and produces a point $\left[\, t\,;\, z^+\,\right] \in \mathcal{N}(w, c, \alpha)$. Note that the measure of optimality t remains unchanged, but the measure of proximity to the path (27) is improved, whence the name corrector. The direction of the corrector is computed as the solution of the linear system

$$\begin{cases} su + xv = w(t) - xs \\ Pu + Qv + Rd = 0 \end{cases}. \tag{44}$$

By taking a unit step along this direction, we obtain the points

$$t^+ = t, \quad x^+ = x + u, \quad s^+ = s + v, \quad y^+ = y + d, \quad z^+ = [\, x^+\,;\, s^+\,;\, y^+\,]. \tag{45}$$

It can be shown that $\left[\, t^+\,;\, z^+\,\right] \in \mathcal{N}(w, c, \alpha)$.

Algorithm 2 (*Predictor-Corrector*)
Given a starting point $z^0 = [x^0; s^0; y^0] \in \mathscr{F}^0$:
Consider the notation from (26);
Choose parameters α and $\bar{\alpha}$ satisfying (40);
Set $k \leftarrow 0$;
repeat
Predictor
Set $z = [x; s; y] \leftarrow z^k$, $t \leftarrow t_k$;
Solve the linear system (41);
Compute steplength $\bar{\theta}$ from (43);
Compute $\bar{t}, \bar{z}$ from (42) with $\theta = \bar{\theta}$;
Set $\bar{\theta}_k \leftarrow \bar{\theta}$, $\bar{z}^k \leftarrow \bar{z}$, $t_{k+1} \leftarrow \bar{t}$;
Corrector
Set $z = [x; s; y] \leftarrow \bar{z}^k$, $t \leftarrow t_{k+1}$;
Solve the linear system (44);
Compute z^+ from (45);
Set $z^{k+1} \leftarrow z^+$;
Set $k \leftarrow k + 1$.
continue

The following results are similar to those obtained for Algorithm 1.

Theorem 3 *If wCP* (11) *is monotone, then Algorithm 2 is well defined and generates an iteration sequence satisfying the following properties:*

$$[t_k; z^k] \in \mathscr{N}_2(w, c, \alpha) \quad , [t_{k+1}; \bar{z}^k] \in \mathscr{N}_2(w, c, \bar{\alpha}) ;$$
$$t_{k+1} = (1 - \bar{\theta}_k) t_k ;$$
$$\bar{\theta}_k \geq \frac{2\alpha}{3\rho} .$$

Corollary 3 *If wCP* (5) *is monotone, then Algorithm 2 finds an ε-approximate solution for this problem (i.e., a point $z \in \mathscr{S}_\varepsilon$, where $\mathscr{S}_\varepsilon$ is defined in* (25)*) in at most*

$$O\left(\frac{x^{0T} s^0 / n + \left\| x^0 s^0 - w \right\|_2}{\min x^0 s^0} \log \frac{x^{0T} s^0 / n + \left\| x^0 s^0 - w \right\|_2}{\varepsilon} \right)$$

iterations.

Corollary 4 *When applied to the wCP generated by the Fisher problem with starting point z^0 given by* (12)–(15)*, Algorithm 2 finds an ε-approximate solution for this problem in at most*

$$O\left(\sqrt{n_c n_p} \log \frac{(n_c + n_p) \|w\|_\infty}{\varepsilon} \right)$$

iterations.

4 Sufficient Weighted Complementarity Problems

The notion of a sufficient linear complementary problem was introduced in 1989 by Cottle et al. [15]. Two years later, Kojima et al. [40] introduced the notion of a P_* linear complementarity problem and showed that a P_*-matrix is column sufficient. Subsequently, Guu and Cottle [35] proved that a P_*-matrix is also row sufficient and therefore the class P_* is included in the class of sufficient matrices. Soon after that Väliaho [57] proved the reverse inclusion. Therefore P_* coincides with the class of sufficient matrices. Since $P_* = \cup_{\kappa \geq 0} P_*(\kappa)$, it follows that a matrix is sufficient if and only if it is a $P_*(\kappa)$-matrix for some $\kappa \geq 0$. The smallest κ with this property is called the handicap of the matrix. Kojima et al. introduced the class P_* because many interior-point methods, originally developed for linear programming, can be extended in a natural way to P_* linear complementarity problems. For example, they proved that the primal–dual potential reduction method can solve a $P_*(\kappa)$ linear complementarity problem in at most $O((1+\kappa)\sqrt{n}L)$ iterations, where n is the dimension of the problem and $L = \log(\varepsilon_0/\varepsilon)$, with ε the required precision (duality gap) and ε_0 the duality gap at the starting point. This is still the best complexity result for solving $P_*(\kappa)$ linear complementarity problems. However, like the vast majority of interior-point methods for solving sufficient linear complementarity problems that appeared subsequently in the literature, the above primal–dual potential reduction method makes explicit use of the parameter κ, so that in order to use it for solving a given sufficient linear complementarity problem, one has first to find an upper bound of the handicap of that problem. It is well known that such an upper bound is very difficult to estimate. Therefore, it is very important to develop interior-point methods for solving sufficient linear complementarity problems that do not depend on the handicap of the problem (see [43, 51]).

In a recent paper [53], we introduced the notion of a sufficient linear wCP and proposed an interior-point method for its numerical solution. This notion extends both the notion of a monotone linear wCP and that of a sufficient linear complementarity problem. We associated with each sufficient linear wCP an appropriate optimization problem, and we showed that a linear wCP is row sufficient if and only if every KKT point of that optimization problem is a solution of the wCP. We proved that every column sufficient linear wCP has a convex (perhaps empty) solution set. If the weight vector is zero, then the reverse implication is also true. This generalizes the well-known characterization of sufficient linear complementarity problems (see [14], [54]). We also showed that if a sufficient linear wCP is strictly feasible, then it is solvable (i.e., its solution set is not empty). We then used the notion of the central path of the wCP introduced in [50] to develop a path-following algorithm. Since neither of the two interior-point methods from [50] can be extended to sufficient wCP without explicitly using the parameter κ, we employed a corrector–predictor approach. This approach was first used in [49] for devising efficient interior-point methods for solving monotone linear complementarity problems in large neighborhoods of the central path, and then in [34, 43] for constructing interior-point methods for sufficient linear complementarity problems that do not depend on the handicap

of the problem. We showed that the proposed corrector–predictor method is well defined and we found upper bounds on its computational complexity in terms of the handicap of the problem and a measure of the quality of the starting point.

4.1 Associated Optimization Problems

The mixed weighted linear complementarity problem (5) is closely related to the following optimization problem:

$$\begin{array}{ll} \underset{x,s,y}{\text{minimize}} & x^T s - \sum_{i=1}^{n} w_i \log x_i s_i \\ \text{subject to} & Px + Qs + Ry = a \\ & x \geq 0, \quad s \geq 0\,. \end{array} \tag{46}$$

Let us consider the index sets

$$\overline{\mathscr{W}} = \{i \in \{1, \ldots, n\} \ : \ w_i > 0\}\,, \quad \widehat{\mathscr{W}} = \{i \in \{1, \ldots, n\} \ : \ w_i = 0\}\,. \tag{47}$$

For any vector $u \in I\!R^n$ we denote by $u_{\overline{\mathscr{W}}}$ the vector formed by components of u that correspond to indices in $\overline{\mathscr{W}}$, and by $u_{\widehat{\mathscr{W}}}$ the vector formed by components of u that correspond to indices in $\widehat{\mathscr{W}}$. We note that if $[\,x\,;\,s\,]$ is in the domain of definition of the objective function in (46) then $x_{\bar{\mathscr{W}}}\, s_{\overline{\mathscr{W}}} > 0$. From the inequality constraints $x \geq 0\,,\ s \geq 0$ it follows that $x_{\bar{\mathscr{W}}} > 0\,,\ s_{\overline{\mathscr{W}}} > 0$. Using the notation u/v described at the end of the Introduction, the KKT conditions for (46) can be written as

$$\begin{array}{l} Px + Qs + Ry = a\,,\ x_{\overline{\mathscr{W}}} > 0\,,\ s_{\overline{\mathscr{W}}} > 0\,, \\ [\,s - w/x - p\,;\ x - w/s - q\,;\ 0\,] \in \mathrm{Ran}[P^T; Q^T; R^T]\,, \\ x \geq 0\,,\ s \geq 0\,,\ p \geq 0\,,\ q \geq 0\,,\ p^T x = 0\,,\ q^T s = 0\,. \end{array} \tag{48}$$

4.2 Sufficiency

The triple (P, Q, R) is called column sufficient if

$$[\,x\,;\ s\,;\ y\,] \in \ \mathrm{Ker}\,[P, Q, R] \text{ and } xs \leq 0 \quad \text{imply } xs = 0, \tag{49}$$

and it is called row sufficient if

$$[\,x\,;\ s\,;\ 0\,] \in \ \mathrm{Ran}\,[\,P^T\,;\ Q^T\,;\ R^T\,] \text{ and } xs \geq 0 \quad \text{imply } xs = 0\,. \tag{50}$$

The triple (P, Q, R) is called sufficient if it is both column sufficient and row sufficient.

Let $\kappa \geq 0$ be a given constant. We say that the triplet (P, Q, R) has the $P_*(\kappa)$ property if

$$(1+4\kappa)\sum_{i\in\mathscr{I}^+} u_i v_i + \sum_{i\in\mathscr{I}^-} u_i v_i \geq 0 \;\; \forall [u; v; y] \in \mathrm{Ker}[P, Q, R]\,, \tag{51}$$

where

$$\mathscr{I}^+ = \{i : u_i v_i > 0\} \text{ and } \mathscr{I}^- = \{i : u_i v_i < 0\}.$$

If $\kappa = 0$, then the triplet (P, Q, R) is monotone (see (2)). We say that the triplet (P, Q, R) has the P_* property if it has the $P_*(\kappa)$ property for some $\kappa \geq 0$. Using Väliaho's result [57] and the results about the equivalence between different formulations of the complementarity problem from [3, 31, 33], it follows that the triplet (P, Q, R) has the P_* property if and only if it is sufficient. If the triplet (P, Q, R) is sufficient, then the smallest κ for which (P, Q, R) has the $P_*(\kappa)$ property is called the handicap of the problem.

We say that the mixed weighted linear complementarity problem (5) is column sufficient, row sufficient, sufficient, or $P_*(\kappa)$, if the triplet (P, Q, R) has the respective property.

4.3 Properties of the Sufficient Weighted Linear Complementarity Problem

The following three theorems are proved in [53]:

Theorem 4 *The following statements are equivalent:*

(a) The triplet (P, Q, R) is row sufficient.
(b) There is a weight vector $w \in \mathbb{R}^n_+$ such that for all vectors $d \in \mathbb{R}^n$ any KKT point of (46) is a solution of (5).
(c) For all weight vectors $w \in \mathbb{R}^n_+$ and all vectors $d \in \mathbb{R}^n$, any KKT point of (46) is a solution of (5).

Theorem 5 *If the triplet (P, Q, R) is column sufficient, then for all weight vectors $w \in \mathbb{R}^n_+$ and all vectors $d \in \mathbb{R}^n$, the mixed linear weighted complementarity problem (5) has a convex (perhaps empty) solution set $\mathscr{F}^*$. Moreover, $x_{\overline{w}}$ and $s_{\overline{w}}$ are uniquely defined for any $[x\,;\, s\,;\, y] \in \mathscr{F}^*$.*

In [53] we gave a simple example showing that the existence of a weight vector $w \in \mathbb{R}^n_+ - \{0\}$ and a vector $a \in \mathbb{R}^m$ such that the solution set of (5) is convex does not imply that the triple (P, Q, R) is column sufficient. We end this subsection by giving a simple sufficient condition for the solvability of (5).

Theorem 6 *If the weighted complementarity problem* (5) *is sufficient and strictly feasible, then it is solvable.*

4.4 A Corrector–Predictor Algorithm for Solving Sufficient wCP's

The interior-point method presented in this section is a path-following method. The path is the one the one considered in (27) for monotone mixed linear wCP. The present interior-point method differs from the two path-following methods from [50] because the natural extensions of those methods to the sufficient case would need explicit use of the handicap of the problem. Given a strictly feasible starting point $z^0 = [\, x^0 \,;\, s^0 \,;\, y^0 \,] \in \mathscr{F}^0$, we consider the notation from (22) to (28). By construction, $[t_0; z^0]$ belongs to the path (27). The proximity of a point $[\, t \,;\, z \,] = [t; x; s; y]$ to this central path can be measured by the function

$$\delta(t, z) = t^{-1} \parallel xs - w(t) \parallel_2 . \tag{52}$$

Obviously, a point $[\, t \,;\, z \,] \in (0, t_0] \times \mathscr{F}^0$ belongs to the central path if and only if $\delta(t, z) = 0$. The neighborhood (29) of the central path (27) can then be written as

$$\mathscr{N}_2(w, c, \alpha) = \{[t; z] = [t; x; s; y] \in (0, t_0] \times \mathscr{F}^0 \ : \ \delta(t, z) \leq \alpha\}. \tag{53}$$

At a typical iteration, we have a point $[\, t \,;\, z \,] \in \mathscr{N}_2(w, c, \alpha)$, for some $t \leq t_0$.
The corrector. The role of the corrector step is to produce a point $[\, t \,;\, \bar{z} \,] = [t; \bar{x} \,;\, \bar{s} \,;\, \bar{y} \,]$ in $(0, t_0] \times \mathscr{F}^0$ with a smaller proximity measure, i.e., $\delta(t, \bar{z}) < \delta(t, z)$. Since the starting point is perfectly centered, i.e., $\delta(t_0, z^0) = 0$, no corrector is needed at the first iteration. Formally, at the first iteration we can take $\bar{z} = z^0$. For all subsequent iterations, the corrector direction $[u; v; h]$ is obtained as the solution of the following linear system

$$\begin{cases} \quad su + xv \quad\quad = w(t) - xs \\ Pu + Qv + Rh = \quad 0 \end{cases} . \tag{54}$$

Let us denote

$$z(\theta) = [\, x(\theta) \,;\, s(\theta) \,;\, y(\theta) \,] = z + \theta[u; v; h] = [\, x + \theta u \,;\, s + \theta v \,;\, y + \theta h \,] \tag{55}$$

and define the steplength $\bar{\theta}$ of the corrector as the minimizer of the proximity measure $\delta(t, z(\theta))$. It turns out that minimizing $\delta(t, z(\theta))$ is equivalent to minimizing a quartic, which can be done very efficiently. With

$$\bar{\theta} = \underset{\theta \geq 0}{\operatorname{argmin}} (1-\theta)^2 \| xs - w(t) \|_2^2 + 2\theta^2 (1-\theta)(uv)^T (xs - w(t)) + \theta^4 \| uv \|_2^2 , \tag{56}$$

the output of the corrector step is the point

$$\bar{z} = [\, \bar{x} \,;\, \bar{s} \,;\, \bar{y} \,] = z(\bar{\theta}) . \tag{57}$$

It can be proved that

$$[\, t \,;\, \bar{z} \,] \in \mathcal{N}_2(w, c, \bar{\alpha}) , \quad \bar{\alpha} := \delta(t, \bar{z}) < \left(1 - \frac{1}{4(1+2\kappa)}\right) \alpha . \tag{58}$$

The predictor.The predictor step follows the corrector step, so that the point $[\, t \,;\, \bar{z} \,] \in \mathcal{N}_2(w, c, \bar{\alpha})$ is available. The purpose of the predictor step is to produce a new point $\left[\, t^+ \,;\, z^+ \,\right] \in \mathcal{N}_2(w, c, \alpha)$ with $t^+ < t$. Given the point (57), we compute the predictor direction $[\, \bar{u} \,;\, \bar{v} \,;\, \bar{h} \,]$ by solving the linear system

$$\begin{cases} \bar{s}\,\bar{u} + \bar{x}\,\bar{v} = w - \bar{x}\,\bar{s} \\ P\bar{u} + Q\bar{v} + R\bar{h} = 0 \end{cases} . \tag{59}$$

We define

$$\begin{gathered} \bar{x}(\theta) = \bar{x} + \theta\bar{u} , \ \bar{s}(\theta) = \bar{s} + \theta\bar{v} , \ \bar{y}(\theta) = \bar{y} + \theta\bar{h} , \ t(\theta) = (1-\theta)t, \\ \bar{z}(\theta) = [\, \bar{x}(\theta) \,;\, \bar{s}(\theta) \,;\, \bar{y}(\theta) \,] . \end{gathered} \tag{60}$$

The stepsize θ^+ along this direction is defined as the largest θ for which $[\, t(\theta) \,;\, \bar{z}(\theta) \,]$ belongs to the neighborhood $\mathcal{N}_2(w, c, \alpha)$, i.e.,

$$\theta^+ = \max\{\hat{\theta} \ : \ [\, t(\theta) \,;\, \bar{z}(\theta) \,] \in \mathcal{N}_2(w, c, \alpha), \ \forall\, 0 < \theta \leq \hat{\theta} \,\} . \tag{61}$$

It turns out that this stepsize is given explicitly using the solution of two quadratic Equations [53]. Having computed this steplength, we obtain the predictor point

$$\left[\, t^+ \,;\, z^+ \,\right] = [\, (1-\theta^+)t \,;\, \bar{x}(\theta^+) \,;\, \bar{s}(\theta^+) \,;\, y(\theta^+) \,] \in \mathcal{N}_2(w, c, \alpha). \tag{62}$$

Hence, we can start a new corrector step from this point followed by the corresponding predictor step. We obtain the following iterative scheme.

Algorithm 3 *(Corrector–Predictor)*
Given a starting point $z^0 = [\, x^0 \,;\, s^0 \,;\, y^0 \,] \in \mathcal{F}^0$:
Consider the notation from (26);
Choose a parameter α satisfying (28);
Set $k \leftarrow 0$;
repeat
Set $z = [\, x \,;\, s \,;\, y \,] \leftarrow z^k$, $t \leftarrow t_k$;

Corrector
If $k = 0$ set $\bar{z} = z^0$, $\bar{\theta} = 0$ and go to *Predictor*;
Solve the linear system (54);
Compute steplength $\bar{\theta}$ from (56) ;
Compute $\bar{z}$ from (55) and (57);

Predictor
Solve the linear system (59);
Compute steplength θ^+ from (61);
Compute t^+, z^+ from (60) and (62);

Set $\theta_k \leftarrow \theta^+, t_{k+1} \leftarrow t^+, \bar{z}^k \leftarrow \bar{z}, z^{k+1} \leftarrow z^+$;
Set $k \leftarrow k + 1$.
continue

4.4.1 The Computational Complexity of the Algorithm

The following results are proved in [53].

Theorem 7 *If wCP (5) is sufficient, then the Corrector–Predictor Algorithm is well defined and generates two iteration sequences satisfying the following properties:*

$$[t_k; z^k] \in \mathcal{N}_2(w, c, \alpha)\,, \ [t_k; \bar{z}^k] \in \mathcal{N}_2(w, c, \bar{\alpha})\,, \ \bar{\alpha} < \alpha\,,$$
$$t_{k+1} = (1 - \theta_k)t_k\,, \ \theta_k \geq \frac{\alpha}{2\rho(1 + 2\kappa)}\,, \ \rho = 1 + \frac{\|c - w\|_2}{t_0}\,,$$

for $k = 0, 1, \ldots$.

Corollary 5 *If wCP (5) is sufficient, then the Corrector–Predictor Algorithm finds an ε-approximate solution for this problem, i.e., a point $z = [x\,;\ s\,;\ y\,] \in \mathcal{F}^0$ such that $\| xs - w \|_2 \leq \varepsilon$, in at most*

$$O\left(\frac{x^{0T}s^0/n + \left\| x^0s^0 - w \right\|_2}{\min x^0s^0}\,(1 + \kappa)\log\frac{x^{0T}s^0/n + \left\| x^0s^0 - w \right\|_2}{\varepsilon}\right)$$

iterations.

5 A Smoothing Algorithm for wCP

By using a smoothing function of the form

$$\phi_\vartheta(t, a, b, c) = \sqrt{a^2 + b^2 - 2\vartheta ab + 2(1 + \vartheta)c + 2t} - a - b, \qquad (63)$$

where $a, b \in \mathbb{R}$, $t, c \in \mathbb{R}_+$, and $\vartheta \in (-1, 1]$, Jian [39] showed that wCP (5) can be written as a system of nonlinear equations. To see this, we first observe that $\phi_\vartheta(0, a, b, c) = 0$ if and only if $a \geq 0,\ b \geq 0$ and $ab = c$. For any $x, s \in \mathbb{R}^n$, $w \in \mathbb{R}^n_+$ and $t \geq 0$ we denote

$$\Phi_\vartheta(t, x, s, w) = [\phi_\vartheta(t, x_1, s_1, w_1); \phi_\vartheta(t, x_2, s_2, w_2); \ldots; \phi_\vartheta(t, x_n, s_n, w_n)]. \quad (64)$$

Using the above observation, we deduce that $\Phi_\vartheta(t, x, s, w) = 0$ if and only if $x \geq 0,\ s \geq 0$ and $xs = w$. Finally, let us consider the nonlinear system $H(z) = 0$, where $z = [t; x; s; y]$ and

$$H(z) = [e^t - 1\ ;\ \Phi_\vartheta(t, x, s, w)\ ;\ Px + Qs + Ry - a]. \quad (65)$$

Since the first entry of $H(z)$ vanishes if and only if $t = 0$ it follows that the solution sets of (5) and $H(z) = 0$ coincide.

Algorithm 4 (*Smoothing Newton*)
Given a starting point $z^0 = [t_0, x^0; s^0; y^0] \in \mathbb{R}_{++} \times \mathbb{R}_n \times \mathbb{R}^n \times \mathbb{R}^m$:
Consider the notation from (65);
Choose parameters $\sigma \in (0, 1/2)$, $\lambda,\ \delta \in (0, 1)$ and $\vartheta \in (-1, 1]$.
Set $k \leftarrow 0$;
repeat
Set $z = [t, x; s; y] \leftarrow z^k$;
Compute Δz by solving the linear system
$H(z) + H'(z)\Delta z = [t \min\{\sigma,\ e^t - 1\}\ ;\ 0]$;
Let p be the smallest nonnegative integer such that
$\|H(z + \delta^p \Delta z)\|^2 \leq \left(1 - \lambda\left(1 - 2\min\{\sigma,\ e^t - 1\}\right)\delta^p\right)\|H(z)\|^2$;
Compute t_+, z^+ from (31), (34), and (36);
Set $z^{k+1} \leftarrow z + \delta^p \Delta z$;
Set $k \leftarrow k + 1$.
continue

In [39], the author proves that if wCP (5) is monotone then Algorithm 4 is well defined and it generates a sequence $\{z^k = [t_k\ x^k; s^k; y^k]\}$ with $0 < t_{k+1} < t_k < \ldots \leq t_0$. Moreover, it is shown that if $\{z^k\}$ is bounded then $\{t_k\}$ converges to zero and that any accumulation point $z^* = [0; x^*; s^*; y^*]$ of the sequence generated by Algorithm 4 provides a solution $x^*; s^*; y^*]$ of wCP (5).

As is usually the case with smoothing methods, no polynomial complexity results are given. However, the numerical results presented in the paper show that the algorithm is efficient in practice.

Similar theoretical results are obtained in a very recent paper [19], where a slightly different smoothing function is used, the linear systems do not have to be solved exactly, and the stepsize is obtained via a nonmonotone line search. The numerical results show that this algorithm performs better than the corresponding algorithm that uses direct methods for the solution of the linear systems and employs a monotone line search.

Acknowledgements This material is based upon work supported by the National Science Foundation under Grant No. DMS-1311923.

References

1. Achdou, Y., Pironneau, O.: *Computational Methods for Option Pricing*, vol. 30 of *Frontiers in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2005
2. Amundson, N.R., Caboussat, A., He, J.W., Seinfeld, J.H.: Primal-dual interior-point method for an optimization problem related to the modeling of atmospheric organic aerosols. J. Optim. Theory Appl. **130**(3), 375–407 (2006)
3. Anitescu, M., Lesaja, G., Potra, F.A.: Equivalence between different formulations of the linear complementarity problem. Optim. Methods Softw. **7**(3), 265–290 (1997)
4. Anstreicher, K.M.: Volumetric path following algorithms for linear programming. Math. Program. **76**(1, Ser. B):245–263 (1997)
5. Anstreicher, K.M.: Interior-point algorithms for a generalization of linear programming and weighted centring. Optim. Methods Softw. **27**(4–5), 605–612 (2012)
6. Arrow, K.J., Debreu, G.: Existence of an equilibrium for a competitive economy. Econometrica **22**, 265–290 (1954)
7. Atkinson, D.S., Vaidya, P.M.: A scaling technique for finding the weighted analytic center of a polytope. Math. Program. **57**(2, Ser. B), 163–192 (1992)
8. Benson, H.Y., Sen, A., Shanno, D.F., Vanderbei, R.J.: Interior-point algorithms, penalty methods and equilibrium problems. Comput. Optim. Appl. **34**(2), 155–182 (2006)
9. Bonnans, J.F., Gonzaga, C.C.: Convergence of interior point algorithms for the monotone linear complementarity problem. Mathe. Oper. Res. **21**, 1–25 (1996)
10. Bonnans, J.F., Potra, F.A.: On the convergence of the iteration sequence of infeasible path following algorithms for linear complementarity problems. Mathe. Oper. Res. **22**(2), 378–407 (1997)
11. Caboussat, A., Leonard, A.: Numerical method for a dynamic optimization problem arising in the modeling of a population of aerosol particles. C. R. Math. Acad. Sci. Paris **346**(11–12), 677–680 (2008)
12. Chen, X., Deng, X., Teng, S.-H.: Settling the complexity of computing two-player Nash equilibria. J. ACM **56**(3):Art. 14, 57 (2009)
13. Chen, X., Huang, L.-S., Teng, S.-H.: Market equilibria with hybrid linear-Leontief utilities. Theoret. Comput. Sci. **410**(17), 1573–1580 (2009)
14. Cottle, R.W., Pang, J.-S., Stone, R.E.: The Linear Complementarity Problem. Academic Press, Boston, MA (1992)
15. Cottle, R.W., Pang, J.-S., Venkateswaran, V.: Sufficient matrices and the linear complementarity problem. Linear Algebra Appl. **114**(115), 231–249 (1989)
16. Daskalakis, C., Goldberg, P.W., Papadimitriou, C.H.: The complexity of computing a Nash equilibrium. SIAM J. Comput. **39**(1), 195–259 (2009)
17. Denault, M., Pigeon, B.: Résolution numérique de problèmes de complémentarité linéaire et évaluation d'options américaines. INFOR Inf. Syst. Oper. Res. **42**(1), 1–21 (2004)
18. Devanur, N.R., Papadimitriou, C.H., Saberi, A., Vazirani, V.V.: Market equilibrium via a primal-dual algorithm for a convex program. J. ACM 55(5):Art. 22, 18 (2008)
19. Dong, L., Tang, J., Song, X.: A non-monotone inexact non-interior continuation method based on a parametric smoothing function for LWCP. Int. J. Comput. Math. 1–13 (2017)
20. Eisenberg, E., Gale, D.: Consensus of subjective probabilities: the pari-mutuel method. Ann. Math. Statist. **30**, 165–168 (1959)
21. Esteban-Bravo, M.: Computing equilibria in general equilibrium models via interior-point methods. Comput. Econom. **23**, 147–171 (2004)

22. Esteban-Bravo, M.: An interior-point algorithm for computing equilibria in economies with incomplete asset markets. J. Econom. Dynam. Control **32**(3), 677–694 (2008)
23. Facchinei, F., Pang, J.S.: Finite-dimensional Variational Inequalities and Complementarity Problems. Springer Series in Operations Research. Springer, New York (2003)
24. Ferris, M.C., Pang, J.S.: Engineering and economic applications of complementarity problems. SIAM Rev. **39**(4), 669–713 (1997)
25. Fischer, A.: A special Newton-type optimization method. Optimization **24**(3–4), 269–284 (1992)
26. Flores, P., Leine, R., Glocker, C.: Modeling and analysis of planar rigid multibody systems with translational clearance joints based on the non-smooth dynamics approach. Multibody Syst. Dyn. **23**(2), 165–190 (2010)
27. Freund, R.M.: Projective transformations for interior-point algorithms, and a superlinearly convergent algorithm for the w-center problem. Math. Program. **58**(3, Ser. A), 385–414 (1993)
28. Garg, D., Jain, K., Talwar, K., Vazirani, V.V.: A primal-dual algorithm for computing Fisher equilibrium in the absence of gross substitutability property. Theoret. Comput. Sci. **378**(2), 143–152 (2007)
29. Goel, G., Vazirani, V.V.: A perfect price discrimination market model with production, and a rational convex program for it. Math. Oper. Res. **36**(4), 762–782 (2011)
30. Gonzaga, C.C.: The largest step path following algorithm for monotone linear complementarity problems. Math. Program. **76**(2, Ser. A), 309–332 (1997)
31. Gowda, M.S.: Reducing a monotone horizontal LCP to an LCP. Appl. Math. Lett. **8**(1), 97–100 (1995)
32. Granas, A., Dugundji, J.: Fixed Point Theory. Springer Monographs in Mathematics. Springer, New York (2003)
33. Güler, Osman: Generalized linear complementarity problems. Math. Oper. Res. **20**(2), 441–448 (1995)
34. Gurtuna, F., Petra, C., Potra, F.A., Shevchenko, O., Vancea, A.: Corrector-predictor methods for sufficient linear complementarity problems. Comput. Optim. Appl. **48**(3), 453–485 (2011)
35. Guu, S.-M., Cottle, R.W.: On a subclass of $\mathbf{P}_0$. Linear. Algebra. Appl. 223/224, 325–335 (1995). Special issue honoring Miroslav Fiedler and Vlastimil Pták
36. Ikonen, S., Toivanen, J.: Operator splitting methods for American option pricing. Appl. Math. Lett. **17**(7), 809–814 (2004)
37. Jain, K.: A polynomial time algorithm for computing an Arrow-Debreu market equilibrium for linear utilities. SIAM J. Comput. 37(1), 303–318 (2007) (electronic)
38. Jain, K., Vazirani, V.V.: Eisenberg-Gale markets: algorithms and game-theoretic properties. Games Econom. Behav. **70**(1), 84–106 (2010)
39. Jian, Z.: A smoothing Newton algorithm for weighted linear complementarity problem. Optim. Lett. **10**(3), 499–509 (2016)
40. Kojima, M., Megiddo, N., Noma, T., Yoshise, A.: A Unified Approach to Interior Point Algorithms for Linear Complementarity Problems. Lecture Notes in Comput, vol. 538. Sci. Springer, New York (1991)
41. Koulisianis, M.D., Papatheodorou, T.S.: Pricing of American options using linear complementarity formulation: methods and their evaluation. Neural Parallel Sci. Comput. **11**(4), 423–443 (2003)
42. Landry, C., Caboussat, A., Hairer, E.: Solving optimization-constrained differential equations with discontinuity points, with application to atmospheric chemistry. SIAM J. Sci. Comput. **31**(5), 3806–3826 (2009)
43. Liu, X., Potra, F.A.: Corrector-predictor methods for sufficient linear complementarity problems in a wide neighborhood of the central path. SIAM J. Optim. **17**(3), 871–890 (2006)
44. McCartin, B.J., Labadie, S.M.: Accurate and efficient pricing of vanilla stock options via the Crandall-Douglas scheme. Appl. Math. Comput. **143**(1), 39–60 (2003)
45. McShane, K.A.: Superlinearly convergent $O(\sqrt{n}L)$-iteration interior-point algorithms for linear programming and the monotone linear complementarity problem. SIAM J. Optim. **4**(2), 247–261 (1994)

46. Mizuno, S., Todd, M.J., Ye, Y.: On adaptive-step primal-dual interior-point algorithms for linear programming. Mathe. Oper. Res. **18**(4), 964–981 (1993)
47. Nesterov, Y., Nemirovsky, A.: Interior Point Polynomial Methods in Convex Programming. SIAM Publications. SIAM, Philadelphia (1994)
48. Pfeiffer, F., Foerg, M., Ulbrich, H.: Numerical aspects of non-smooth multibody dynamics. Comput. Methods Appl. Mech. Engrg. **195**(50–51), 6891–6908 (2006)
49. Potra, F.A.: Corrector-predictor methods for monontone linear complementarity problems in a wide neighborhood of the central path. Math. Program. **111**(1-2, Ser. B), 243–272 (2008)
50. Potra, F.A.: Weighted complementarity problems-a new paradigm for computing equilibria. SIAM J. Optim. **22**(4), 1634–1654 (2012)
51. Potra, F.A., Liu, X.: Predictor-corrector methods for sufficient linear complementarity problems in a wide neighborhood of the central path. Optim. Methods Softw. **20**(1), 145–168 (2005)
52. Potra, F.A., Ye, Y.: Interior-point methods for nonlinear complementarity problems. JOTA **88**(3), 617–647 (1996)
53. Potra, F.A.: Sufficient weighted complementarity problems. Comput. Optim. Appl. **64**(2), 467–488 (2016)
54. Stoer, J.: High order long-step methods for solving linear complementarity problems. Ann. Oper. Res. 103, 149–159 (2001). Optimization and numerical algebra (Nanjing, 1999)
55. Tasora, A., Anitescu, M.: A fast NCP solver for large rigid-body problems with contacts, friction, and joints. In: Multibody Dynamics, vol. 12 of Computer Methods Applied Science, pp. 45–55. Springer, Berlin (2009)
56. Vaidya, P.M.: A new algorithm for minimizing convex functions over convex sets. Math. Program **73**(3, Ser. A), 291–341 (1996)
57. Väliaho, H.: P_*-matrices are just sufficient. Linear Algebra its Appl. **239**, 103–108 (1996)
58. Vazirani, V.V.: The notion of a rational convex program, and an algorithm for the Arrow-Debreu Nash bargaining game. J. ACM **59**(2):Art. 7, 36 (2012)
59. Vazirani, V.V., Yannakakis, M.: Market equilibrium under separable, piecewise-linear, concave utilities. J. ACM **58**(3):Art. 10, 25 (2011)
60. Walras, L.: Éléments d'économie politique pure; ou. Théorie de la richesse socialle. L. Corbaz, Paris (1874)
61. Ye, Y.: Interior Point Algorithms: Theory and Analysis. Wiley-Interscience Series in Discrete Mathematics and Optimization, Wiley, Hoboken (1997)
62. Ye, Y.: Exchange market equilibria with Leontief's utility: freedom of pricing leads to rationality. Theoret. Comput. Sci. **378**(2), 134–142 (2007)
63. Ye, Y.: A path to the Arrow-Debreu competitive market equilibrium. Math. Program. **111**(1-2, Ser. B), 315–348 (2008)
64. Zhu, Y., Wu, X., Chern, I.: Derivative Securities and Difference Methods. Springer Finance. Springer, New York (2004)
65. Zhu, Z. Dang, C., Ye, Y.: A FPTAS for computing a symmetric Leontief competitive economy equilibrium. *Math. Program.*, **131**(1-2, Ser. A), 113–129, 2012

A Mathematical Approach to Living on Sinking Ground



Kees Roos

Abstract Many regions in the world are protected against flooding by a dike, which may be either natural or artificial. We deal with a model for finding the optimal height of such a dike in the future. The model minimizes the sum of investment costs and expected damage costs. It is highly nonlinear, nonconvex, and infinite-dimensional. Despite this, it is shown that the model can be solved analytically. Our model has been used in the Netherlands to define legal safety standards for the coming decades.

Keywords Flood prevention · Cost-benefit analysis · Global optimality Nonconvex optimization

1 Introduction

Dike height optimization is of major importance to the Netherlands because a large part of the country lies below sea level, and high water levels in rivers can also cause floods. Recently, improvements have been made on a cost-benefit model that was introduced by Van Dantzig after a devastating flood in the Netherlands in 1953 [4]. In some recent papers, we dealt with an extension of this model that may also be applicable to other deltas in the world where the ground is sinking and/or the water level is rising [1–3].

The submersible part of the Netherlands is divided in more than 50 so-called dike rings. A dike ring is an area that is surrounded by a dike. Most dike rings consist of several segments that are characterized by different properties. We focus on the case of a one-segment (or homogeneous) dike and show that our model, which is a highly nonlinear, nonconvex, and infinite-dimensional optimization problem, can be solved analytically.

The paper is self-contained and a streamlined version of the (electronic) appendix of [2]. Some proofs are simplified or otherwise modified. To ease the reading, we only

K. Roos (✉)
Delft University of Technology, P.O. Box 5031, 2600, GA Delft, The Netherlands
e-mail: C.Roos@TUDelft.nl

© Springer International Publishing AG, part of Springer Nature 2018

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*,
Springer Proceedings in Mathematics & Statistics 235,
https://doi.org/10.1007/978-3-319-90026-1_13

summarized the contents of some elementary lemmas, thereby referring to related results in [2].

Let us mention that for nonhomogeneous dikes an analytic solution is out of reach. In that case, dynamic programming can be used to solve a discretized version of the model if the number of segments is not too large. However, if the number of segments is larger than about five, this approach explodes. For that case, we developed a mixed integer nonlinear optimization model in [1, 3].

2 Mathematical Model for a Homogeneous Dike

Our model uses the following functions and parameters.

H_t : dike height at time t,
$P_t = P_0 e^{\alpha\eta t} e^{-\alpha(H_t - H_0)}$: flood probability at time t,
α : parameter in exponential distribution for extreme water levels (1/cm),
η : ground level decrease plus water level increase (cm/year),
$V_t = V_0 e^{\gamma t} e^{\zeta(H_t - H_0)}$: loss by flooding at time t (million euros),
γ : rate of growth of wealth in area protected by the dike (per year),
ζ : increase of loss per cm dike heightening (1/cm),
δ : discount rate (1/year),

The expected loss at time t is then

$$S_t = P_t V_t = P_0 e^{\alpha\eta t} e^{-\alpha(H_t - H_0)} \cdot V_0 e^{\gamma t} e^{\zeta(H_t - H_0)} = S_0 e^{\beta_1 t} e^{-\theta h_t},$$

where

$$S_0 = P_0 V_0, \quad \beta_1 = \alpha\eta + \gamma, \quad \theta = \alpha - \zeta, \quad h_t = H_t - H_0.$$

We assume that upgrades of the dike take place at moments t_k, $k = 1, 2 \ldots$. The value of the upgrade at moment t_k is denoted as u_k. We require that

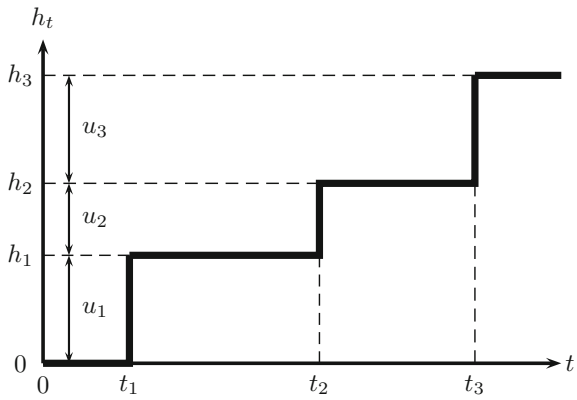
$$u_k > 0, \quad t_{k+1} > t_k \geq t_0 = 0, \qquad k \geq 1. \tag{1}$$

The height (relative to the heigth at $t = 0$) at moment t_k includes the upgrade u_k and is denoted as h_k (see Fig. 1). So we have

$$h_0 = 0, \quad h_k = h_{k-1} + u_k, \qquad k \geq 1. \tag{2}$$

We define infinite sequences u, h, and τ as follows:

Fig. 1 Graph of h_t



$$u := (u_1; u_2; u_3; \ldots),$$
$$h := (h_1; h_2; h_2; \ldots),$$
$$\tau := (t_1; t_2; t_3; \ldots).$$

It is clear that if we know u, then h follows from (2) and vice versa. So we may consider the total cost as a function $f(u, \tau)$ of u and τ alone. One has

$$f(u, \tau) = I(u, \tau) + A(u, \tau),$$

where $I(u, \tau)$ represents the total *investment cost*, and $A(u, \tau)$ the total *expected damage*. The cost for an upgrade of the dike with u_k at t_k is given by

$$D(u_k)e^{\lambda h_k}e^{-\delta t_k}, \quad D(u_k) = c + bu_k,$$

where $\lambda \geq 0$ and the factor $e^{-\delta t_k}$ takes care of discounting the future costs. We may now write

$$I(u, \tau) = \sum_{k=1}^{\infty} D(u_k)e^{\lambda h_k - \delta t_k},$$
$$A(u, \tau) = \sum_{k=1}^{\infty} \int_{t_{k-1}}^{t_k} S_t e^{-\delta t}\, dt = \sum_{k=1}^{\infty} \int_{t_{k-1}}^{t_k} S_0 e^{\beta_1 t} e^{-\theta(H_t - H_0)} e^{-\delta t}\, dt$$
$$= \sum_{k=1}^{\infty} \frac{S_0}{\beta} \left[e^{\beta t_k} - e^{\beta t_{k-1}}\right] e^{-\theta h_{k-1}},$$

where $\beta = \beta_1 - \delta$ and where we used that the height of the dike in the interval $[t_{k-1}, t_k)$ equals h_{k-1}. The parameters in this model are assumed to satisfy

$$S_0 > 0, \quad b \geq 0, \quad c > 0, \quad \lambda \geq 0, \quad \delta > 0, \quad \theta > 0, \quad \beta \neq 0.$$

We call the pair (u, τ) admissible if (1) and (2) are satisfied. Our aim is to find an admissible pair (u, τ) that minimizes $f(u, \tau)$. Thereby, we also use that

$$\beta + \delta > 0, \qquad \theta\delta - \lambda\beta > 0, \qquad \lambda + b > 0, \tag{3}$$

because these inequalities are satisfied for all dikerings in The Netherlands. To simplify the presentation, we assume $\beta \neq 0$, though the arguments used below can easily be adapted to the case where $\beta = 0$.

3 First-Order Optimality Conditions With Respect To t_k

By computing the partial derivative of $f(u, \tau)$ with respect to t_k, we obtain

$$\frac{\partial f(u,\tau)}{\partial t_k} = -\delta D(u_k) e^{\lambda h_k - \delta t_k} + S_0 e^{\beta t_k - \theta h_{k-1}} - S_0 e^{\beta t_k - \theta h_k}.$$

Hence, the pair (u, τ) is stationary with respect to t_k if and only if

$$\delta D(u_k) e^{\lambda h_k - \delta t_k} = S_0 e^{\beta t_k - \theta h_{k-1}} - S_0 e^{\beta t_k - \theta h_k}, \quad k \geq 1. \tag{4}$$

Because of (2), this condition can be written as

$$\delta D(u_k) e^{(\lambda+\theta) h_k - (\beta+\delta) t_k} = S_0 \left(e^{\theta u_k} - 1\right). \tag{5}$$

For $x > 0$, we define the function $\varrho(x)$ as follows:

$$\varrho(x) := \ln \frac{\delta D(x)}{S_0 \left(e^{\theta x} - 1\right)}, \quad x > 0. \tag{6}$$

By taking logarithms at both sides in (5), the condition for stationarity with respect to t_k becomes

$$\varrho(u_k) = (\beta + \delta) t_k - (\lambda + \theta) h_k, \quad k \geq 1. \tag{7}$$

As already established, if u is known, then h can be computed. Then, the above relation reveals that the sequence τ can also be computed, because using $\beta + \delta > 0$ we may write

$$t_k = \frac{\varrho(u_k) + (\lambda + \theta) h_k}{\beta + \delta}. \tag{8}$$

Hence, a stationary pair (u, τ) is completely determined by its sequence u. Therefore, we call the sequence u a *stationary sequence* if the corresponding pair (u, τ) is

stationary. Similarly, u is said to be admissible (optimal) if (u, τ) is admissible (optimal).

In the sequel, we also use the function

$$\kappa(x) := \frac{D(x)}{e^{\theta x} - 1}, \quad x > 0, \tag{9}$$

which is related to the function $\varrho(x)$ via

$$S_0 e^{\varrho(x)} = \frac{\delta D(x)}{e^{\theta x} - 1} = \delta \kappa(x). \tag{10}$$

This yields a second way of characterizing stationarity with respect to t_k:

$$S_0 e^{\beta t_k - \theta h_k} = \delta \kappa(u_k) e^{\lambda h_k - \delta t_k}. \tag{11}$$

4 First-Order Optimality Conditions With Respect To u_k

Since $h_\ell = \sum_{k=1}^{\ell} u_k$, we have

$$\frac{\partial h_\ell}{\partial u_k} = \begin{cases} 1, & \text{if } \ell \geq k, \\ 0, & \text{if } \ell < k. \end{cases}$$

Using this we get, for $k \geq 1$,

$$\frac{\partial f(u,\tau)}{\partial u_k} = b e^{\lambda h_k - \delta t_k} + \sum_{\ell \geq k} \lambda D(u_\ell) e^{\lambda h_\ell - \delta t_\ell} - \sum_{\ell - 1 \geq k} \frac{\theta S_0}{\beta} \left(e^{\beta t_\ell} - e^{\beta t_{\ell-1}} \right) e^{-\theta h_{\ell-1}}.$$

Due to (11), it follows that stationarity with respect to u_k implies

$$\begin{aligned} b e^{\lambda h_k - \delta t_k} + \sum_{\ell=k}^{\infty} \lambda D(u_\ell) e^{\lambda h_\ell - \delta t_\ell} &= \sum_{\ell=k+1}^{\infty} \frac{\theta S_0}{\beta} \left(e^{\beta t_\ell} - e^{\beta t_{\ell-1}} \right) e^{-\theta h_{\ell-1}} \\ &= \sum_{\ell=k+1}^{\infty} \frac{\theta S_0}{\beta} \left(e^{\beta t_\ell - \theta h_\ell + \theta u_\ell} - e^{\beta t_{\ell-1} - \theta h_{\ell-1}} \right) \\ &= \sum_{\ell=k+1}^{\infty} \frac{\theta \delta}{\beta} \left(\kappa(u_\ell) e^{\lambda h_\ell - \delta t_\ell} e^{\theta u_\ell} - \kappa(u_{\ell-1}) e^{\lambda h_{\ell-1} - \delta t_{\ell-1}} \right), \end{aligned} \tag{12}$$

where the last equality uses the condition (11) for stationarity with respect to t_k, for each $k \geq 1$. Defining

$$\Delta(u_k) = b + \lambda D(u_k), \quad k \geq 1, \tag{13}$$

this can be written as

$$\Delta(u_k)e^{\lambda h_k-\delta t_k}+\sum_{\ell=k+1}^{\infty}\lambda D(u_\ell)e^{\lambda h_\ell-\delta t_\ell}$$
$$=\sum_{\ell=k+1}^{\infty}\frac{\theta\delta}{\beta}\left(\kappa(u_\ell)e^{\lambda h_\ell-\delta t_\ell}e^{\theta u_\ell}-\kappa(u_{\ell-1})e^{\lambda h_{\ell-1}-\delta t_{\ell-1}}\right),\quad k\geq 1,$$

which is equivalent to

$$\Delta(u_{k+1})e^{\lambda h_{k+1}-\delta t_{k+1}}+\sum_{\ell=k+2}^{\infty}\lambda D(u_\ell)e^{\lambda h_\ell-\delta t_\ell}$$
$$=\sum_{\ell=k+2}^{\infty}\frac{\theta\delta}{\beta}\left(\kappa(u_\ell)e^{\lambda h_\ell-\delta t_\ell}e^{\theta u_\ell}-\kappa(u_{\ell-1})e^{\lambda h_{\ell-1}-\delta t_{\ell-1}}\right),\quad k\geq 0.$$

Subtracting the second equation from the first gives

$$\Delta(u_k)e^{\lambda h_k-\delta t_k}-\left(\Delta(u_{k+1})-\lambda D(u_{k+1})\right)e^{\lambda h_{k+1}-\delta t_{k+1}}$$
$$=\frac{\theta\delta}{\beta}\left(\kappa(u_{k+1})e^{\lambda h_{k+1}-\delta t_{k+1}}e^{\theta u_{k+1}}-\kappa(u_k)e^{\lambda h_k-\delta t_k}\right),\quad k\geq 1.$$

Using $\Delta(u_{k+1})=b+\lambda D(u_{k+1})$, we get

$$\Delta(u_k)e^{\lambda h_k-\delta t_k}-be^{\lambda h_{k+1}-\delta t_{k+1}}=\frac{\theta\delta}{\beta}\left(\kappa(u_{k+1})e^{\lambda h_{k+1}-\delta t_{k+1}}e^{\theta u_{k+1}}-\kappa(u_k)e^{\lambda h_k-\delta t_k}\right),$$

which, after multiplying both sides with $e^{-\lambda h_k}$ and rearranging terms, reduces to

$$\left(\Delta(u_k)+\frac{\theta\delta}{\beta}\kappa(u_k)\right)e^{-\delta t_k}=\left(b+\frac{\theta\delta}{\beta}\kappa(u_{k+1})e^{\theta u_{k+1}}\right)e^{\lambda u_{k+1}-\delta t_{k+1}},\quad k\geq 1. \tag{14}$$

In order to eliminate t_k (and t_{k+1}), we introduce parameters $\bar{\delta}$ and q as follows:

$$\bar{\delta}:=\frac{\delta}{\beta+\delta},\quad q:=\bar{\delta}(\lambda+\theta)-\lambda. \tag{15}$$

Then, by using (11) once more, we may write, for $k\geq 1$,

$$e^{-\delta t_k}=\left(e^{(\beta+\delta)t_k}\right)^{-\bar{\delta}}=\left(\frac{\delta\kappa(u_k)}{S_0}e^{(\theta+\lambda)h_k}\right)^{-\bar{\delta}}=\left(\frac{S_0}{\delta}\right)^{\bar{\delta}}\kappa(u_k)^{-\bar{\delta}}e^{-(q+\lambda)h_k}. \tag{16}$$

Hence, we obtain

$$\frac{e^{-\delta t_{k+1}}}{e^{-\delta t_k}} = \frac{\left(\frac{S_0}{\delta}\right)^{\bar{\delta}} \kappa(u_{k+1})^{-\bar{\delta}} e^{-(q+\lambda)h_{k+1}}}{\left(\frac{S_0}{\delta}\right)^{\bar{\delta}} \kappa(u_k)^{-\bar{\delta}} e^{-(q+\lambda)h_k}} = \frac{\kappa(u_{k+1})^{-\bar{\delta}} e^{-(q+\lambda)u_{k+1}}}{\kappa(u_k)^{-\bar{\delta}}}.$$

Substitution into (14) yields, for $k \geq 1$,

$$\kappa(u_k)^{-\bar{\delta}} \left[\Delta(u_k) + \frac{\theta\delta}{\beta} \kappa(u_k) \right] = \kappa(u_{k+1})^{-\bar{\delta}} e^{-qu_{k+1}} \left[b + \frac{\theta\delta}{\beta} \kappa(u_{k+1}) e^{\theta u_{k+1}} \right]. \quad (17)$$

To simplify the notation, we introduce

$$L(x) := \Delta(x) + \frac{\theta\delta}{\beta} \kappa(x), \quad (18)$$

$$R(x) := e^{-qx} \left(b + \frac{\theta\delta}{\beta} \kappa(x) e^{\theta x} \right), \quad (19)$$

and define $\mathscr{L}(x)$ and $\mathscr{R}(x)$ as follows:

$$\mathscr{L}(x) := \kappa(x)^{-\bar{\delta}} L(x), \qquad \mathscr{R}(x) := \kappa(x)^{-\bar{\delta}} R(x). \quad (20)$$

Then, we may rewrite the condition for stationarity with respect to u_k in the following compact form:

$$\mathscr{L}(u_k) = \mathscr{R}(u_{k+1}), \quad k \geq 1. \quad (21)$$

In the next section, we show the remarkable fact that the objective value $f(u, \tau)$ for any stationary sequence u is completely determined by the value of the function $\mathscr{R}$ in u_1.

5 Objective Value at a Stationary Point

Taking the sum of all stationarity conditions (4) for t_k ($k \geq 1$) yields

$$\sum_{k=1}^{\infty} \delta D(u_k) e^{\lambda h_k - \delta t_k} = \sum_{k=1}^{\infty} \left(S_0 e^{\beta t_k - \theta h_{k-1}} - S_0 e^{\beta t_k - \theta h_k} \right).$$

This yields a linear relation between $I(u, \tau)$ and $A(u, \tau)$ at a stationary point as follows:

$$\begin{aligned} \delta I(u, \tau) &= \sum_{k=1}^{\infty} S_0 e^{\beta t_k - \theta h_{k-1}} - \sum_{k=2}^{\infty} S_0 e^{\beta t_{k-1} - \theta h_{k-1}} \\ &= S_0 e^{\beta t_0 - \theta h_0} + \sum_{k=1}^{\infty} S_0 \left[e^{\beta t_k} - e^{\beta t_{k-1}} \right] e^{-\theta h_{k-1}} \\ &= S_0 + \beta A(u, \tau). \end{aligned} \quad (22)$$

On the other hand, the stationarity condition for u_k in (12) yields, for $k = 1$,

$$\begin{aligned} be^{\lambda h_1 - \delta t_1} + \sum_{\ell=1}^{\infty} \lambda D(u_\ell) e^{\lambda h_\ell - \delta t_\ell} &= \sum_{\ell=2}^{\infty} \frac{\theta S_0}{\beta} \left(e^{\beta t_\ell} - e^{\beta t_{\ell-1}}\right) e^{-\theta h_{\ell-1}} \\ &= \theta \left(A(u,\tau) - \frac{S_0}{\beta} e^{\beta t_1 - \theta h_0} + \frac{S_0}{\beta} e^{\beta t_0 - \theta h_0} \right) \\ &= \theta A(u,\tau) - \frac{\theta\delta}{\beta} \kappa(u_1) e^{\lambda h_1 - \delta t_1} e^{\theta u_1} + \frac{\theta S_0}{\beta}, \end{aligned}$$

where we used $h_0 = t_0 = 0$ and $h_1 = u_1$. Thus, we obtain, after rearranging terms, a second linear relation between $I(u,\tau)$ and $A(u,\tau)$:

$$\begin{aligned} -\lambda I(u,\tau) + \theta A(u,\tau) &= be^{\lambda h_1 - \delta t_1} + \frac{\theta\delta}{\beta} \kappa(u_1) e^{\lambda h_1 - \delta t_1} e^{\theta u_1} - \frac{\theta S_0}{\beta} \\ &= \left(b + \frac{\theta\delta}{\beta} \kappa(u_1) e^{\theta u_1} \right) e^{\lambda h_1 - \delta t_1} - \frac{\theta S_0}{\beta} \\ &= e^{q u_1} R(u_1) e^{\lambda h_1 - \delta t_1} - \frac{\theta S_0}{\beta}. \end{aligned}$$

Using (16), with $k = 1$, and also $h_1 = u_1$, we obtain

$$e^{\lambda h_1 - \delta t_1} = e^{\lambda h_1} \left(\frac{S_0}{\delta} \right)^{\bar{\delta}} \kappa(u_1)^{-\bar{\delta}} e^{-(q+\lambda)u_1} = \left(\frac{S_0}{\delta} \right)^{\bar{\delta}} \kappa(u_1)^{-\bar{\delta}} e^{-q u_1}.$$

Substituting this, we find

$$-\lambda I(u,\tau) + \theta A(u,\tau) = \left(\frac{S_0}{\delta} \right)^{\bar{\delta}} \mathscr{R}(u_1) - \frac{\theta S_0}{\beta} \tag{23}$$

As a consequence, we now have two linear relations between $I(u,\tau)$ and $A(u,\tau)$ given as (22) and (23). The determinant of the coefficient matrix equals $\delta\theta - \lambda\beta$, which by (3) is positive. Hence, the system has a unique solution, namely

$$\begin{aligned} (\theta\delta - \beta\lambda)\, A(u,\tau) &= \lambda S_0 + \delta \left(\frac{S_0}{\delta} \right)^{\bar{\delta}} \mathscr{R}(u_1) - \frac{\delta\theta S_0}{\beta} \\ (\theta\delta - \beta\lambda)\, I(u,\tau) &= \theta S_0 + \beta \left(\frac{S_0}{\delta} \right)^{\bar{\delta}} \mathscr{R}(u_1) - \frac{\beta\theta S_0}{\beta}. \end{aligned}$$

By taking the sum at both sides, we obtain

$$(\theta\delta - \beta\lambda)\, f(u,\tau) = (\theta + \lambda) S_0 + (\beta + \delta) \left(\frac{S_0}{\delta} \right)^{\bar{\delta}} \mathscr{R}(u_1) - (\beta + \delta) \frac{\theta S_0}{\beta}.$$

The definition (15) of q gives

$$q = \bar{\delta}(\lambda+\theta) - \lambda = \delta\frac{\lambda+\theta}{\beta+\delta} - \lambda = \frac{\delta(\lambda+\theta) - \lambda(\beta+\delta)}{\beta+\delta} = \frac{\theta\delta - \lambda\beta}{\beta+\delta}.$$

Furthermore, the coefficient of S_0 in the above expression for $(\theta\delta - \beta\lambda)\, f(u,\tau)$ equals

$$\theta + \lambda - (\beta+\delta)\frac{\theta}{\beta} = \lambda - \delta\frac{\theta}{\beta} = \frac{\beta\lambda - \theta\delta}{\beta}.$$

Hence, it follows that at a stationary point the objective value is given by

$$f(u,\tau) = \frac{1}{q}\left(\frac{S_0}{\delta}\right)^{\bar{\delta}} \mathscr{R}(u_1) - \frac{S_0}{\beta}. \tag{24}$$

It comes as a surprise that the objective value at a stationary point is completely determined by u_1 and is a positive constant times $\mathscr{R}(u_1)$ minus a constant. Hence, it remains to find a stationary point for which $\mathscr{R}(u_1)$ is minimal. As we show in the next section, this is possible due to some very nice properties of the functions $\mathscr{L}(x)$ and $\mathscr{R}(x)$.

Before proceeding to the next section, an important observation is that (24) indicates that β and $\mathscr{R}(u_1)$ must have the same sign, since otherwise one would get a negative value for the total cost $f(u,\tau)$, which would be absurd. Indeed, more generally one has

$$\beta\mathscr{R}(x) > 0, \quad x > 0. \tag{25}$$

For a formal proof of this inequality, see the appendix in [2, p. 34].

6 Properties of $\mathscr{L}(x)$ and $\mathscr{R}(x)$

By way of example, the graphs of $\mathscr{L}(x)$ and $\mathscr{R}(x)$ are shown in Fig. 2, for an arbitrary homogeneous dike. One can show that these functions always share the following properties (for the proofs of these properties, see the appendix in [2, Sect. A.4]):

Prop. 1: The graphs of $\mathscr{L}(x)$ and $\mathscr{R}(x)$ intersect at exactly one point, where $x = \nu$.
Prop. 2: $\mathscr{L}(x)$ and $\mathscr{R}(x)$ are bounded from below and minimal for the same value $\bar{\nu} < \nu$.
Prop. 3: $\mathscr{L}(x)$ and $\mathscr{R}(x)$ are monotonically decreasing for $x < \bar{\nu}$ and increasing for $x > \bar{\nu}$.
Prop. 4: One has $\mathscr{L}(x) < \mathscr{R}(x)$ if $x < \nu$, and $\mathscr{L}(x) > \mathscr{R}(x)$ if $x > \nu$.

The value of ν is determined by the equation $\mathscr{L}(\nu) = \mathscr{R}(\nu)$, which is equivalent to $L(\nu) = R(\nu)$, i.e.,

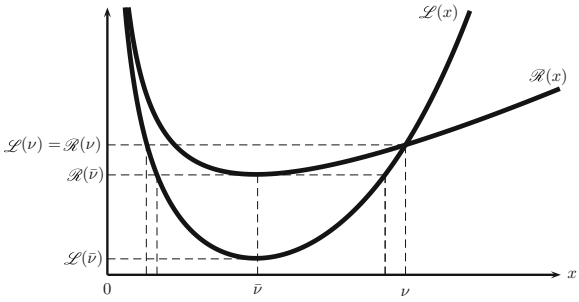


Fig. 2 Typical form of the graphs of $\mathscr{L}(x)$ and $\mathscr{R}(x)$

$$\Delta(\nu) + \frac{\theta\delta}{\beta}\kappa(\nu) = e^{-q\nu}\left(b + \frac{\theta\delta}{\beta}\kappa(\nu)e^{\theta\nu}\right), \quad \kappa(\nu) = \frac{D(\nu)}{e^{\theta\nu} - 1}.$$

As Prop. 1 states, this nonlinear equation determines ν uniquely.

Recall that the sequence $u = (u_1; u_2; u_3; \ldots)$ is stationary if and only if $\mathscr{L}(u_k) = \mathscr{R}(u_{k+1})$ for each $k \geq 1$. It may be understood from Fig. 2 that there exist infinitely many stationary sequences. As Fig. 2 also shows, the sequence $u = (\nu; \nu; \nu; \ldots)$ is stationary. We call this the *periodic sequence*. In Sect. 5, it became clear that a stationary sequence u is optimal if $\mathscr{R}(u_1)$ is minimal. Using this, we need a few lemmas to establish that the periodic sequence is optimal (provided that it is admissible).

Lemma 1 *If u_k occurs in a stationary sequence, then $\mathscr{L}(u_k) \geq \mathscr{L}(\nu)$.*

Proof Suppose that $\mathscr{L}(u_k) < \mathscr{L}(\nu)$. Since $\mathscr{L}(x)$ is monotonically increasing for $x > \bar{\nu}$, by Prop. 3, and $\nu > \bar{\nu}$, by Prop. 2, we necessarily have $u_k < \nu$. If u_k occurs in a stationary sequence, its successor u_{k+1} satisfies $\mathscr{R}(u_{k+1}) = \mathscr{L}(u_k) < \mathscr{L}(\nu) = \mathscr{R}(\nu)$, where we used Prop. 1. So we have $\mathscr{R}(u_{k+1}) < \mathscr{R}(\nu)$. For the same reasons as for $\mathscr{L}$, this implies $u_{k+1} < \nu$. Since $\mathscr{L}(x) < \mathscr{R}(x)$ for all $x < \nu$, by Prop. 4, it follows that $\mathscr{L}(u_{k+1}) < \mathscr{R}(u_{k+1}) = \mathscr{L}(u_k)$. Thus, we have shown that $\mathscr{L}(u_k) < \mathscr{L}(\nu)$ implies $\mathscr{L}(u_{k+1}) < \mathscr{L}(u_k)$ and as a consequence also $\mathscr{L}(u_{k+1}) < \mathscr{L}(\nu)$. This implies that the $\mathscr{L}$-values of the successors of u_k in the sequence u form a strictly decreasing sequence. Since this sequence is bounded below by $\mathscr{L}(\bar{\nu})$, the sequence must converge. A similar argument applies to the sequence of the values $\mathscr{R}(u_k)$. Due to $\mathscr{R}(u_{k+1}) = \mathscr{L}(u_k)$, for each k, the limits of the two sequences must be equal. But this is impossible, since the limits occur in the interval where $\mathscr{L}(x) < \mathscr{R}(x)$. This contradiction proves the lemma. □

Lemma 2 *If u_k occurs in a stationary sequence, then $\mathscr{R}(u_k) \geq \mathscr{R}(\nu)$.*

Proof If $u_k \geq \nu$, then we have $\mathscr{R}(u_k) \geq \mathscr{R}(\nu)$, because $\mathscr{R}(x)$ is increasing for $x \geq \nu$. On the other hand, if $u_k \leq \nu$, then $\mathscr{R}(u_k) \geq \mathscr{L}(u_k)$, by Prop. 3. By Lemma 1, we have $\mathscr{L}(u_k) \geq \mathscr{L}(\nu)$. Hence, $\mathscr{R}(u_k) \geq \mathscr{L}(\nu)$. Since $\mathscr{L}(\nu) = \mathscr{R}(\nu)$, we get $\mathscr{R}(u_k) \geq \mathscr{R}(\nu)$, as desired. □

Theorem 1 *If* $\varrho(\nu) + (\lambda + \theta)\,\nu \geq 0$, *then the periodic sequence is optimal.*

Proof Lemma 2 implies that $\mathscr{R}(u_1) \geq \mathscr{R}(\nu)$ for every stationary sequence u. According to (24), this establishes the theorem, provided that $t_k \geq 0$, for $k \geq 1$. Since

$$t_k = \frac{\varrho(u_1) + (\lambda + \theta)h_k}{\beta + \delta} = \frac{\varrho(\nu)}{\beta + \delta} + k\,\frac{\lambda + \theta}{\beta + \delta}\nu, \quad k \geq 1,$$

for the periodic sequence, the condition holds only if $\varrho(\nu) + (\lambda + \theta)\,\nu \geq 0$, which completes the proof. □

Theorem 1 only holds if the value of t_1 in the periodic solution is nonnegative. One may easily deduce from the definitions (6) of $\varrho(x)$ and (9) of $\kappa(x)$ that this holds if and only if

$$S_0 \leq \delta\kappa e^{(\lambda+\theta)\nu}, \tag{26}$$

i.e., if and only if the expected damage costs at $t = 0$ do not exceed the threshold value $\delta\kappa e^{(\lambda+\theta)\nu}$. So, if the value of t_1 is negative, there is a backlog in the maintenance of the dike. If this happens, we say that the dike is *unhealthy*, otherwise *healthy*. It remains to deal with the unhealthy case. In that case, an immediate heightening of the dike is desirable. It seems natural to expect that to eliminate the backlog in maintenance, a heightening of at least ν will be necessary. So, we will then have $t_1 = 0$. In the next section, we deal with the question of how to find the value of u_1 that minimizes $f(u, \tau)$. As we will see, it seems impossible to find an analytic solution of this problem. But we show that the optimal value of u_1 in the unhealthy case can be found by a standard routine to minimize a convex function on a finite interval.

7 The Unhealthy Case

In the unhealthy case, the constraint $t_1 \geq 0$ in (1) is active. This means that $f(u, \tau)$ is not stationary with respect to t_1. As a consequence, at optimality, we have stationarity with respect to t_k if and only if $k \geq 2$ and stationarity with respect to u_k if and only if $k \geq 1$. Using this, and by using the same approach as in Sect. 5 for the healthy case, one can show that the objective value at a stationary point is now given by

$$f(u, \tau) = D(u_1)e^{\lambda u_1} + \frac{e^{-qu_1}}{q}\left(\frac{S_0}{\delta}\right)^{\bar{\delta}}\mathscr{R}(u_2) - \frac{S_0}{\beta}e^{-\theta u_1}.$$

With u_1 fixed, by using the same arguments as in the healthy case, it follows that $u_2 = \nu$ yields the smallest value. Due to (20) and (10), we write

$$\left(\frac{S_0}{\delta}\right)^{\bar{\delta}}\mathscr{R}(\nu) = \left(\frac{S_0}{\delta}\right)^{\bar{\delta}}\kappa(\nu)^{-\bar{\delta}}R(\nu) = \left(\frac{\delta}{S_0}\kappa(\nu)\right)^{-\bar{\delta}}R(\nu) = R(\nu)e^{-\bar{\delta}\varrho(\nu)}. \tag{27}$$

Hence, denoting the resulting function of u_1 as $F(u_1)$, we obtain

$$F(u_1) = D(u_1)e^{\lambda u_1} + \frac{1}{q}R(\nu)e^{-\bar{\delta}\varrho(\nu)-qu_1} - \frac{S_0}{\beta}e^{-\theta u_1}.$$

It remains to find the value of u_1 that minimizes $F(u_1)$. In considering this minimization problem, we should respect the conditions (1) for admissibility of $u = (u_1; \nu; \nu; \ldots)$. Since the solution is periodic from $t = t_2$ on, (1) boils down to the simple condition $t_2 > 0$. By (8), and since $u_2 = \nu$ and $h_2 = u_1 + \nu$, this holds if and only if

$$\varrho(\nu) + (\lambda + \theta)(u_1 + \nu) > 0. \tag{28}$$

On the other hand, the dike is unhealthy if and only if the periodic solution gives $t_1 < 0$, which means that $\varrho(\nu) + (\lambda + \theta)\nu < 0$. So, we have

$$\varrho(\nu) + (\lambda + \theta)\nu < 0 < \varrho(\nu) + (\lambda + \theta)(u_1 + \nu). \tag{29}$$

It will be convenient to introduce the number u_0 as

$$u_0 = -\frac{\varrho(\nu)}{\lambda + \theta} - \nu.$$

This value is chosen in a such a way that

$$\varrho(\nu) + (\lambda + \theta)(u_0 + \nu) = 0. \tag{30}$$

Combining this relation with (29) gives $(\lambda + \theta)\nu < (\lambda + \theta)(u_0 + \nu) < (\lambda + \theta)(u_1 + \nu)$, which implies

$$0 < u_0 < u_1.$$

We derive from (10) and (30) that u_0 is also uniquely defined by

$$S_0 e^{-\theta(u_0+\nu)} = \delta\kappa(\nu)e^{\lambda(u_0+\nu)}. \tag{31}$$

We are now ready to analyze the behavior of $F(u_1)$ and to find its global minimizer, which is denoted as u_1^*. As just established, we have $u_1^* \in (u_0, \infty)$. We start with computing the first two derivatives of $F(u_1)$ with respect to u_1. One has

$$F'(u_1) = \Delta(u_1)e^{\lambda u_1} - R(\nu)e^{-\bar{\delta}\varrho(\nu)-qu_1} + \theta\frac{S_0}{\beta}e^{-\theta u_1}, \tag{32}$$

$$F''(u_1) = \lambda\left[b + \Delta(u_1)\right]e^{\lambda u_1} + qR(\nu)e^{-\bar{\delta}\varrho(\nu)-qu_1} - \theta^2\frac{S_0}{\beta}e^{-\theta u_1}. \tag{33}$$

Lemma 3 *$F(u_1)$ is convex for large enough values of u_1.*

Proof Note that the last two terms in (33) converge to zero if u_1 grows to infinity. If $\lambda > 0$, then the first term exponentially grows to infinity. So, if $\lambda > 0$, then $F(u_1)$ is convex for large values of u_1. This also holds if $\lambda = 0$, as we show next. We then have

$$F''(u_1) = \left(qR(\nu)e^{-\bar{\delta}\varrho(\nu)} - \theta^2 \frac{S_0}{\beta} e^{-ru_1} \right) e^{-qu_1},$$

where, due to (15),

$$r = \theta - q = \theta - \bar{\delta}(\lambda + \theta) + \lambda = (1 - \bar{\delta})(\lambda + \theta) = \beta \frac{\lambda + \theta}{\beta + \delta}.$$

Due to (3), it follows that r and β have the same sign. Due to (25), also $\mathscr{R}(\nu)$, and hence $R(\nu)$, has this sign. This implies that $F''(u_1)$ is positive for large values of u_1, because it converges to $qR(\nu)e^{-\bar{\delta}\varrho(\nu)}$ if $\beta > 0$, and to ∞ if $\beta < 0$. This completes the proof. □

Next, we prove that if $F(u_1)$ is not convex for all $u_1 > u_0$, then there exists a value $\tilde{u} > u_0$ such that $F(u_1)$ is concave for $u_1 < \tilde{u}$ and convex for $u_1 > \tilde{u}$. This is a consequence of the next lemma.

Lemma 4 *$F(u_1)$ has at most one inflection point on the interval (u_0, ∞).*

Proof From (33), we derive

$$F''(u_1) = e^{\lambda u_1} \left[\lambda \left[b + \Delta(u_1) \right] + qR(\nu)e^{-\bar{\delta}\varrho(\nu) - (\lambda+q)u_1} - \theta^2 \frac{S_0}{\beta} e^{-(\lambda+\theta)u_1} \right].$$

We prove the lemma by showing that the expression between square brackets vanishes for at most one value of u_1. Recall that $R(\nu)$, β, and r are nonzero and have the same sign. Therefore, this expression has the form of the function $f(x)$ written as

$$f(x) = \lambda(A + Bx) + Ce^{-(\lambda+q)x} - De^{-(\lambda+\theta))x},$$

where $x = u_1$ and the parameters A, B, C, and D are so that

$$A \geq 0, \quad B \geq 0, \quad CD > 0, \quad rD > 0.$$

Using $\theta = q + r$, it follows that

$$\begin{aligned} f'(x) &= \lambda B - (\lambda + q)Ce^{-(\lambda+q)x} + (\lambda + \theta)De^{-(\lambda+\theta)x} \\ &= \lambda B - (\lambda + q)Ce^{-(\lambda+q)x} + (\lambda + q + r)De^{-(\lambda+\theta)x} \\ &= \lambda B + rDe^{-(\lambda+\theta)x} - (\lambda + q)\left(f(x) - \lambda(A + Bx) \right). \end{aligned}$$

Assuming $f(x) = 0$, we get

$$f'(x) = \lambda B + rDe^{-(\lambda+\theta)x} + (\lambda + q)\lambda(A + Bx) \geq rDe^{-(\lambda+\theta)x} > 0.$$

This proves that $f'(x) > 0$ whenever $f(x) = 0$, which implies that $f(x)$ vanishes for at most one value of x. Hence, the lemma follows. □

Let $\tilde{u} > u_0$ denote a point of inflection, if it exists. If there is no such point, then $F(u_1)$ is convex for all $u_1 > u_0$, and hence, the minimizer u_1^* of $F(u_1)$ can be found by standard numerical techniques for minimizing a strictly convex function.

Hence, from now on, we assume that there is an inflection point $\tilde{u}$, and $\tilde{u} > u_0$. Then, $F(u_1)$ is concave for $u_0 < u_1 < \tilde{u}$, and convex for $u_1 > \tilde{u}$. We proceed by showing that $F(u_1)$ is strictly convex at $u_1 = u_0 + \nu$. This will imply $\tilde{u} \in (u_0, u_0 + \nu)$.

Lemma 5 *$F(u_1)$ is strictly convex at $u_1 = u_0 + \nu$.*

Proof Substitution of $u_1 = u_0 + \nu$ into (33) yields the following expression for $F''(u_0 + \nu)$:

$$\lambda(b + \Delta(u_0 + \nu))e^{\lambda(u_0+\nu)} + qR(\nu)e^{-\bar{\delta}\varrho(\nu)-q(u_0+\nu)} - \frac{\theta^2 S_0}{\beta}e^{-\theta(u_0+\nu)}.$$

By using (15) and (30) successively, we obtain

$$\begin{aligned} -\bar{\delta}\varrho(\nu)-q(u_0+\nu) &= -\bar{\delta}\varrho(\nu) - \left[\bar{\delta}(\lambda+\theta) - \lambda\right](u_0+\nu) \\ &= -\bar{\delta}\left[\varrho(\nu) + (\lambda+\theta)(u_0+\nu)\right] + \lambda(u_0+\nu) \\ &= \lambda(u_0+\nu). \end{aligned} \tag{34}$$

Using this, $R(\nu) = L(\nu)$ and (31), we get

$$\begin{aligned} F''(u_0+\nu) &= \lambda(b + \Delta(u_0+\nu))e^{\lambda(u_0+\nu)} + qR(\nu)e^{\lambda(u_0+\nu)} - \theta\frac{\theta\delta}{\beta}\kappa(\nu)e^{\lambda(u_0+\nu)} \\ &= e^{\lambda(u_0+\nu)}\left[\lambda\left[b + \Delta(u_0+\nu)\right] + qL(\nu) - \theta\frac{\theta\delta}{\beta}\kappa(\nu)\right]. \end{aligned}$$

Hence, it remains to show that the bracketed expression is positive. We have $\Delta(u_0 + \nu) = \lambda b u_0 + \Delta(\nu)$ and $\lambda^2 b u_0 \geq 0$. Furthermore, since $\theta = q + r$, and

$$\frac{\delta r}{\beta} = \frac{\delta(\theta - q)}{\beta} = \frac{\delta\theta - \beta\lambda + \beta\lambda - \delta q}{\beta} = \frac{(\beta+\delta)q + \beta\lambda - \delta q}{\beta} = \lambda + q,$$

we obtain

$$qL(\nu) - \theta\frac{\theta\delta}{\beta}\kappa(\nu) = q\left(L(\nu) - \frac{\theta\delta}{\beta}\kappa(\nu)\right) - r\frac{\theta\delta}{\beta}\kappa(\nu) = q\Delta(\nu) - \theta(\lambda+q)\kappa(\nu).$$

We therefore have

$$\lambda\left[b+\Delta(u_0+\nu)\right]+qL(\nu)-\theta\frac{\theta\delta}{\beta}\kappa(\nu)\geq\lambda b+(\lambda+q)\Delta(\nu)-\theta(\lambda+q)\kappa(\nu).$$

So, it suffices for the proof to show that $g(x)>0$ at $x=\nu$, where

$$g(x):=\lambda b+(\lambda+q)\left(\Delta(x)-\theta\kappa(x)\right).$$

By elementary means, one can show that $\kappa'(x)<0$, for all $x>0$ (cf. [2, Lemma 1 in Appendix A]). Hence, $\kappa(x)$ is monotonically decreasing. Since $\Delta(x)$ is nondecreasing, it follows that $g(x)$ is monotonically increasing. Therefore, since $\bar{\nu}<\nu$, it suffices for the proof of the lemma to show that $g(\bar{\nu})\geq 0$. Due to the definition of $\bar{\nu}$, we have $\mathscr{L}'(\bar{\nu})=0$ and

$$\begin{aligned}\mathscr{L}'(x)&=\kappa(x)^{-\bar{\delta}-1}\left[-\bar{\delta}\kappa'(x)L(x)+\kappa(x)L'(x)\right]\\&=\kappa(x)^{-\bar{\delta}-1}\left[-\bar{\delta}\kappa'(x)\left(\Delta(x)+\frac{\theta\delta}{\beta}\kappa(x)\right)+\kappa(x)\left(\lambda b+\frac{\theta\delta}{\beta}\kappa'(x)\right)\right].\end{aligned}$$

The coefficient of $\kappa(x)\kappa'(x)$ in the bracketed expression equals

$$-\bar{\delta}\frac{\theta\delta}{\beta}+\frac{\theta\delta}{\beta}=(1-\bar{\delta})\frac{\theta\delta}{\beta}=\left(1-\frac{\delta}{\beta+\delta}\right)\frac{\theta\delta}{\beta}=\frac{\beta}{\beta+\delta}\frac{\theta\delta}{\beta}=\theta\bar{\delta}.$$

Thus, it follows that

$$\mathscr{L}'(x)=\kappa(x)^{-\bar{\delta}-1}\left[\lambda b\kappa(x)+\bar{\delta}\kappa'(x)\left[\theta\kappa(x)-\Delta(x)\right]\right].$$

Hence, $\bar{\nu}$ satisfies

$$\lambda b\kappa(x)+\bar{\delta}\kappa'(x)\left[\theta\kappa(x)-\Delta(x)\right]=0$$

This implies

$$\Delta(\bar{\nu})-\theta\kappa(\bar{\nu})=\frac{\lambda b\kappa(\bar{\nu})}{\bar{\delta}\kappa'(\bar{\nu})},\tag{35}$$

which is less than or equal to zero, since $\kappa'(x)<0$, for all $x>0$. Hence, we get, by also using $\lambda+q=\bar{\delta}(\lambda+\theta)$ once more,

$$g(\bar{\nu})=\lambda b+\bar{\delta}(\lambda+\theta)\frac{\lambda b\kappa(\bar{\nu})}{\bar{\delta}\kappa'(\bar{\nu})}=\frac{\lambda b}{\kappa'(\bar{\nu})}\left[\kappa'(\bar{\nu})+(\lambda+\theta)\kappa(\bar{\nu})\right].$$

Straightforward computations yield that for any $x>0$,

$$\kappa'(x)+(\lambda+\theta)\kappa(x)=\frac{\Delta(x)-\theta\kappa(x)}{e^{\theta x}-1}.$$

Thus, we obtain

$$g(\bar{\nu}) = \frac{\lambda b}{\kappa'(\bar{\nu})} \frac{\Delta(\bar{\nu}) - \theta\kappa(\bar{\nu})}{e^{\theta\bar{\nu}} - 1} = \frac{1}{e^{\theta\bar{\nu}} - 1} \frac{\lambda b}{\kappa'(\bar{\nu})} \frac{\lambda b\kappa(\bar{\nu})}{\bar{\delta}\kappa'(\bar{\nu})} = \frac{1}{e^{\theta\bar{\nu}} - 1} \frac{\lambda^2 b^2 \kappa(\bar{\nu})}{\bar{\delta}\kappa'(\bar{\nu})^2} \geq 0.$$

This completes the proof. □

The above lemma implies that $\tilde{u} \in (u_0, u_0 + \nu)$. We proceed by showing that $u_1^* \in (\tilde{u}, u_0 + \nu]$. For this, we need two more lemmas.

Lemma 6 *If $\lambda b > 0$, then $F'(u_0 + \nu) > 0$. Otherwise, i.e., if $\lambda b = 0$, then $F'(u_0 + \nu) = 0$.*

Proof One has

$$F'(u_0 + \nu) = \Delta(u_0 + \nu)e^{\lambda(u_0+\nu)} - R(\nu)e^{-\bar{\delta}\varrho(\nu)}e^{-q(u_0+\nu)} + \theta\frac{S_0}{\beta}e^{-\theta(u_0+\nu)}.$$

Using successively (34), (31), and (18), we get

$$\begin{aligned} F'(u_0 + \nu) &= \left[\Delta(u_0 + \nu) - L(\nu) + \frac{\theta\delta}{\beta}\kappa(\nu)\right] e^{\lambda(u_0+\nu)} \\ &= \left[\Delta(u_0 + \nu) - \Delta(\nu)\right] e^{\lambda(u_0+\nu)} \\ &= \lambda b u_0 e^{\lambda(u_0+\nu)}. \end{aligned}$$

Since $\lambda \geq 0$, $b \geq 0$, and $u_0 > 0$, the proof is complete. □

Since $F(u_1)$ is strictly convex for $u_1 > \tilde{u}$ and since $u_0 + \nu > \tilde{u}$, we may conclude from Lemma 6 that $u_1^* \leq u_0 + \nu$. In the special case where $\lambda b = 0$, the lemma makes clear that $u_1^* = u_0 + \nu$. Otherwise, the minimizer of $F(u_1)$ on the interval $(\tilde{u}, u_0 + \nu)$ can be found by standard numerical methods for minimizing $F(u_1)$ on the interval. We claim that under some mild condition, this procedure yields the global minimizer of $F(u_1)$. The condition is

$$c > bu_0\left(e^{\lambda\nu} - 1\right). \tag{36}$$

Let us emphasize that (36) is satisfied by all dikes for which the data are known. Moreover, if (36) does not hold, then there is something wrong with the data, as we explain later.

One should have noted that there is another candidate for being a global minimizer of $F(u_1)$, namely u_0. To illustrate this, we refer to Fig. 3. In this figure, we see that u_0 is a local minimizer of $F(u_0)$, but not a global minimizer. The next lemma shows that this always holds if condition (36) holds. Before dealing with this lemma, it is worth to discuss the physical meaning of (36).

Roughly speaking, it requires that the fixed cost term in the investment cost function not be too small. Since $c > 0$, the condition is certainly satisfied if $\lambda b = 0$. In

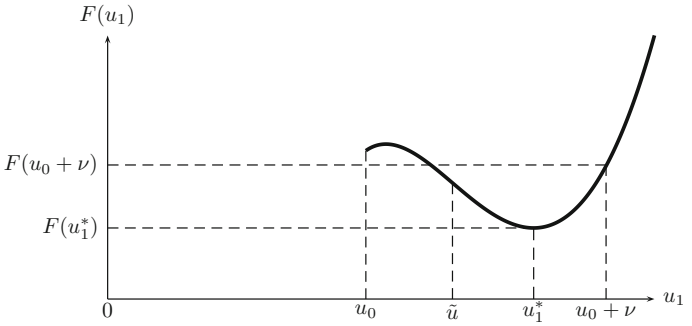


Fig. 3 Typical graph of $F(u_1)$, $u_1 > u_0$

order to understand the physical meaning of this inequality better, one may easily verify that it can be rewritten as follows:

$$D(u_0)e^{\lambda u_0} - bu_0e^{\lambda(u_0+\nu)} > 0,$$

which in turn is equivalent to the inequality

$$D(u_0)e^{\lambda u_0} + D(\nu)e^{\lambda(u_0+\nu)} > D(u_0+\nu)e^{\lambda(u_0+\nu)}. \tag{37}$$

The left-hand side expression in (37) represents the cost for a heightening with u_0 at $t = 0$, 'followed' by a heightening with ν, also at $t = 0$. On the other hand, the right-hand side expression equals the cost for a heightening at $t = 0$ with $u_0 + \nu$, in one step. In the second case, the fixed cost for heightening with ν is avoided. One should therefore expect that a combined heightening with $u_0 + \nu$ is more profitable than its splitting into two subsequent heightenings with u_0 and ν, respectively. But, if (36) is not satisfied, there may be no advantage in combining two heightenings at the same time into one heightening. From a practical point of view, this would indicate that there is something wrong with at least one of the parameters c, b, and λ and maybe also with some of the other parameters involved with the definitions of ν and u_0. Below, we therefore assume that (36) and hence also (37) hold. The next lemma then guarantees that u_0 is not a global minimizer of $F(u_0)$.

Lemma 7 $F(u_0) = \left[D(u_0)e^{\lambda u_0} - bu_0e^{\lambda(u_0+\nu)}\right] + F(u_0+\nu)$.

Proof One has

$$F(u_0) = D(u_0)e^{\lambda u_0} + \frac{1}{q}R(\nu)e^{-\bar{\delta}\varrho(\nu)-qu_0} - \frac{S_0}{\beta}e^{-\theta u_0},$$
$$F(u_0+\nu) = D(u_0+\nu)e^{\lambda(u_0+\nu)} + \frac{1}{q}R(\nu)e^{-\bar{\delta}\varrho(\nu)-q(u_0+\nu)} - \frac{S_0}{\beta}e^{-\theta(u_0+\nu)}.$$

By using (34) and (31), we may therefore write

$$F(u_0) - D(u_0)e^{\lambda(u_0)} = \frac{1}{q}R(\nu)e^{\lambda(u_0+\nu)+q\nu} - \frac{\delta}{\beta}\kappa(\nu)e^{(\lambda+\theta)\nu+\lambda u_0},$$

and also using $D(u_0 + \nu) = bu_0 + D(\nu)$,

$$F(u_0 + \nu) - bu_0e^{\lambda(u_0+\nu)} = D(\nu)e^{\lambda(u_0+\nu)} + \frac{1}{q}R(\nu)e^{\lambda(u_0+\nu)} - \frac{\delta}{\beta}\kappa(\nu)e^{\lambda(\nu+u_0)}.$$

The proof follows if the right-hand side members in the above equations are equal. After dividing by the common factor $e^{\lambda(u_0+\nu)}$, we get

$$\frac{1}{q}e^{q\nu}R(\nu) - \frac{\delta}{\beta}\kappa(\nu)e^{\theta\nu} = D(\nu) + \frac{1}{q}R(\nu) - \frac{\delta}{\beta}\kappa(\nu).$$

Since $\kappa(\nu)e^{\theta\nu} - \kappa(\nu) = D(\nu)$, this further reduces to

$$\frac{1}{q}e^{q\nu}R(\nu) - \frac{\delta}{\beta}D(\nu) = D(\nu) + \frac{1}{q}R(\nu),$$

which is equivalent to

$$e^{q\nu}R(\nu) - q\frac{\beta+\delta}{\beta}D(\nu) = R(\nu).$$

Using the definitions of q and $R(x)$, the left-hand side expression reduces to $L(\nu)$ as follows:

$$\begin{aligned} e^{q\nu}R(\nu) - q\frac{\beta+\delta}{\beta}D(\nu) &= b + \frac{\theta\delta}{\beta}\kappa(\nu)e^{\theta\nu} - \frac{\theta\delta-\lambda\beta}{\beta}D(\nu) \\ &= b + \frac{\theta\delta}{\beta}(\kappa(\nu) + D(\nu)) - \frac{\theta\delta-\lambda\beta}{\beta}D(\nu) \\ &= b + \lambda D(\nu) + \frac{\theta\delta}{\beta}\kappa(\nu) = L(\nu). \end{aligned}$$

Since $L(\nu) = R(\nu)$, by the definition of ν, the proof is complete. □

As a result of Lemma 7 and (37), we have $F(u_0) > F(u_0 + \nu)$, establishing that u_0 is not a global minimizer.

We have shown that in the unhealthy case, the solution is periodic from $t = t_2$ on. Except for t_2, this part of the solution is independent of S_0. All subsequent upgrades have the same value ν and occur periodically; the interval between two subsequent upgrades is independent of S_0. The value of S_0 influences only the value of u_1.

If S_0 is equal to the threshold value in (26), then we will have $u_1 = \nu$, as in the periodic solution, because then the periodic solution has $t_1 = 0$. If S_0 increases, it means that the backlog in maintenance increases, and to compensate for this, a larger

value of u_1 will be required. This physical argument explains that we will always have $u_1 > \nu$ in the unhealthy case. Unfortunately, we were not able to prove this inequality mathematically. We leave it as a challenge for the reader.

References

1. Brekelmans, R.C.M., den Hertog, D., Roos, C., Eijgenraam, C.J.J.: Safe dike heights at minimal costs: the nonhomogeneous case. Oper. Res. **60**(6), 1342–1355 (2012)
2. Eijgenraam, C.J.J., Brekelmans, R.C.M., den Hertog, D., Roos, C.: Optimal strategies for flood prevention. Manage. Sci. pp. 1 – 15 (2016). https://doi.org/10.1287/mnsc.2015.2395
3. Eijgenraam, C.J.J., Kind, J., Bak, C., Brekelmans, R.C.M., den Hertog, D., Duits, M., Roos, C., Vermeer, P., Kuijken, W.: Economically efficient standards to protect the Netherlands against flooding. Interfaces **44**(1), 7–21 (2014)
4. van Dantzig, D.: Economic decision problems for flood prevention. Econometrica **24**(3), 276–287 (1956)

NAOIV-2017 Conference Participants

Invited Speakers

Adil M. Bagirov, Federation University, Australia
Oleg Burdakov, Linkoping University, Sweden
Andrew R. Conn, IBM Research Center, USA
Yu-Hong Dai, Chinese Academy of Science, China
Iain Duff, Rutherford Appleton Laboratory, UK
Giovanni Fasano, University of Venice Ca Foscari, Italy
David M. Gay, AMPL Optimization Inc, USA
Michael Hintermüller, Humboldt-Universitaet zu Berlin, Germany
Nezam Mahdavi-Amiri, Sharif University of Technology, Iran
Hans Mittelmann, Arizona State University, USA
Dominique Orban, Ecole Polytechnique de Montreal, Canada
Amiya Kumar Pani, India Institute of Technology Bombay, India
Florian A. Potra, University of Maryland, USA
Cornelis Roos, Delft University of Technology, The Netherlands
Ekkehard W. Sachs, University of Trier, Germany
Michael Saunders, Stanford University, USA
Philippe L. Toint, University of Namur, Belgium
Maria Teresa Vespucci, Bergamo University, Italy

Organizing Committee

Mehiddin Al-Baali (Chair), Sultan Qaboos University, Oman
Magda Al-Hinai, Sultan Qaboos University, Oman
Fatma Al-Kharousi, Sultan Qaboos University, Oman
Nasser Al-Salti, Sultan Qaboos University, Oman
Amina Al Sawaii, Sultan Qaboos University, Oman
Easwaran Balakrishnan, Sultan Qaboos University, Oman

© Springer International Publishing AG, part of Springer Nature 2018

M. Al-Baali et al. (eds.), *Numerical Analysis and Optimization*,
Springer Proceedings in Mathematics & Statistics 235,
https://doi.org/10.1007/978-3-319-90026-1

Lucio Grandinetti, Calabria University, Italy
Bernhard Heim, German University of Technology, Oman
Sebti Kerbal, Sultan Qaboos University, Oman
Issam Moghrabi, Gulf University for Science and Technology, Kuwait
Amar Oukil, Sultan Qaboos University, Oman
Anton Purnama, (Co-Chair), Sultan Qaboos University, Oman
Florian Rupp, German University of Technology, Oman
Muhammad Syam, United Arab Emirates University, U.A.E.
Chefi Triki, Sultan Qaboos University, Oman
Wasila Al-Busaidi (Webmaster), Sultan Qaboos University, Oman

Liaison Committee of SQU, Oman

Salma Al-Kindy (Chair), Dean of College of Science
Khidir Abdelbasit, Head of Department of Mathematics and Statistics
Mehiddin Al-Baali, Chair of the Organizing Committee
Anton Purnama, Co-Chair of the Organizing Committee
Khamis Al Hadhrami, Representative of the SQU Public Relations Office
Saeed A. Al-Muharami, Administrative Director of College of Science

International Programme Committee

Mehiddin Al-Baali, Sultan Qaboos University, Oman
Iain Duff, Rutherford Appleton Laboratory, UK
Masao Fukushima, Kyoto University, Japan
Lucio Grandinetti, Calabria University, Italy
Luigi Grippo, Rome University, Italy
Jorge Mòre, Argonne National Laboratory, USA
Dominique Orban, Ecole Polytechnique de Montreal, Canada
Martin Reed, University of Bath, UK
Michael Saunders, Stanford University, USA
Robert C. Sharpley, University of South Carolina, USA
Brian Straughan, University of Durham, UK
Philippe L. Toint, University of Namur, Belgium

Student Committee

Muna S. M. Al-Ghabshi, Sultan Qaboos University, Oman
Fatma A. M. Al-Kharusi, Sultan Qaboos University, Oman
Amal Y. A. Al-Kharusi, Sultan Qaboos University, Oman

Fatma S. K. Al-Musalhi, Sultan Qaboos University, Oman
Amani Al-Rubkhi, Sultan Qaboos University, Oman
Tahani Al-Sariri, Sultan Qaboos University, Oman

Participants

Aziza Al-Farsi, Sultan Qaboos University, Oman
Faisal Ababneh, Sultan Qaboos University, Oman
Khidir Abdelbasit, Sultan Qaboos University, Oman
Reda Abu-Elwan, Sultan Qaboos University, Oman
Afaq Ahmad, Sultan Qaboos University, Oman
Muhammad Idrees Ahmad, Sultan Qaboos University, Oman
Safa Al-Abdlie, Sultan Qaboos University, Oman
Fahad Al-Abri, Sultan Qaboos University, Oman
Masood Alam, Sultan Qaboos University, Oman
Moosa Al-Amri, Sultan Qaboos University, Oman
Mehiddin Al-Baali, Sultan Qaboos University, Oman
Salih Al-Battashi, Sultan Qaboos University, Oman
Sayyid Samir Al-Busaidi, Sultan Qaboos University, Oman
Muna S. M. Al-Ghabshi, Sultan Qaboos University, Oman
Mohammed Al-Ghafri, Sultan Qaboos University, Oman
Asma Al-Ghassani, Sultan Qaboos University, Oman
Omar Al-Hammadi, Sultan Qaboos University, Oman
Asma Khalid Al-Harthi, Sultan Qaboos University, Oman
Mahmoud Al-Hashami, Sultan Qaboos University, Oman
Amal Al-Hashimy, Sultan Qaboos University, Oman
Khadjah Al-Hassani, Sultan Qaboos University, Oman
Mohammed M. Al-Hatmi, Sultan Qaboos University, Oman
Magda Al-Hinai, Sultan Qaboos University, Oman
Ishtiaq Ali, COMSATS Institute of Information Technology, Pakistan
Montaz Ali, University of the Witwatersrand, South Africa
Bushra Al-Kalbani, Sultan Qaboos University, Oman
Khamis Saif Al-Kalbani, Sultan Qaboos University, Oman
Fatma Al-Kharousi, Sultan Qaboos University, Oman
Amal Y. A. Al-Kharusi, Sultan Qaboos University, Oman
Fatma A. M. Al-Kharusi, Sultan Qaboos University, Oman
Iman Alkindi, Higher College of Technology, Oman
Ibtisam Al-Maani, German Univeristy of Technology, Oman
Ahmed Al-Maashari, Sultan Qaboos University, Oman
Khaled S. Al-Mashrafi, General Directorate of Education in Sharqiyah, Oman
Qasem M. Al-Mdallal, UAE University, United Arab Emirates
Huda Al-Memari, Sultan Qaboos University, Oman
Asim Al-Mumari, Sultan Qaboos University, Oman

Fatma S. K. Al-Musalhi, Sultan Qaboos University, Oman
Mohammed Al-Refai, UAE University, United Arab Emirates
Amani Al-Rubkhi, Sultan Qaboos University, Oman
Amal Al-Saidi, Sultan Qaboos University, Oman
Nasser Al-Salti, Sultan Qaboos University, Oman
Tahani Al-Sariri, Sultan Qaboos University, Oman
Amina Al-Sawaii, Sultan Qaboos University, Oman
Hamed Al-Senidi, Sultan Qaboos University, Oman
Ahmed Al-Siyabi, Sultan Qaboos University, Oman
Fayeza Al-Sulti, Sur College of Applied Sciences, Oman
Hamood Al-Wardi, Rustaq College of Applied Sciences, Oman
Talal Shaban Amer, Sultan Qaboos University, Oman
Medhat Awadalla, Sultan Qaboos University, Oman
Mojgan Momenie Azandariani, K. N. Toosi University of Technology, Iran
Isa Abdullahi Baba, Near East University, Turkish Republic of North Cyprus
Adil M. Bagirov, Federation University, Australia
Issam Bait Bahadur, Sultan Qaboos University, Oman
Easwaran Balakrishnan, Sultan Qaboos University, Oman
Noureddine Benrabia, Badji Moktar University, EPST, Algeria
Fatima Boudaoud, LAMAP, Oran1 University Ahmed Benbella, Algeria
Oleg Burdakov, Linkoping University, Sweden
Pallath Chandran, Sultan Qaboos University, Oman
Nabil Channouf, Sultan Qaboos University, Oman
Andrew R. Conn, IBM Research Center, USA
Yu-Hong Dai, Chinese Academy of Science, China
Iain Duff, Rutherford Appleton Laboratory, UK
Tayfour El-Bashir, Sultan Qaboos University, Oman
Ibrahim Elmojtaba, Sultan Qaboos University, Oman
Ibrahim Eltayeb, Sultan Qaboos University, Oman
Godfrey Engwau, Sultan Qaboos University, Oman
Mohammad Fares, Rustaq College of Applied Sciences, Rustaq, Oman
Giovanni Fasano, University of Venice Ca Foscari, Italy
David M. Gay, AMPL Optimization Inc, USA
Ahmed F. Ghaleb, Cairo University, Egypt
Ashish Madhukar Gujrathi, Sultan Qaboos University, Oman
Elzain Gumma, International University of Africa, Sudan
Paresh Halder, Sultan Qaboos University, Oman
Bernhard Heim, German Univeristy of Technology, Oman
Michael Hintermuller, Humboldt-Universitaet zu Berlin, Germany
Afzal Husain, Sultan Qaboos University, Oman
Tariq Jamil, Sultan Qaboos University, Oman
Aref Kamal, Sultan Qaboos University, Oman
Samir Karaa, Sultan Qaboos University, Oman
Taliya Kashoub, Sultan Qaboos University, Oman
Sebti Kerbal, Sultan Qaboos University, Oman

Mohammad S. Khan, Sultan Qaboos University, Oman
Qamar J. Khan, Sultan Qaboos University, Oman
Sudeep Kundu, India Institute of Technology Bombay, India
Kenneth K. Kwikiriza, Sultan Qaboos University, Oman
Ding Ma, Stanford University, USA
Nezam Mahdavi-Amiri, Sharif University of Technology, Iran
Jasbir S. Manhas, Sultan Qaboos University, Oman
Hans Mittelmann, Arizona State University, USA
Hassan Mohammad, Sultan Qaboos University, Oman
Ahmed M. Mohammed, Sultan Qaboos University, Oman
Haniffa M. Nasir, Sultan Qaboos University, Oman
Mohamed Yasin Noor, Center for Preparatory Studies, SQU
Dominique Orban, Ecole Polytechnique de Montreal, Canada
Amar Oukil, Sultan Qaboos University, Oman
Amiya Kumar Pani, India Institute of Technology Bombay, India
Florian A. Potra, University of Maryland, USA
Anton Purnama, Sultan Qaboos University, Oman
Mohammad Mansur Rahman, Sultan Qaboos University, Oman
Cornelis Roos, Delft University of Technology, The Netherlands
Florian Rupp, German Univeristy of Technology, Oman
Nirmal Sacheti, Sultan Qaboos University, Oman
Ekkehard W. Sachs, University of Trier, Germany
Amadou Sarr, Sultan Qaboos University, Oman
Michael Saunders, Stanford University, USA
Mohamed Lamine Sayari, Sultan Qaboos University, Oman
Irina Skhomenko, Sultan Qaboos University, Oman
Nasser H. Sweilam, Cairo University, Egypt
Muhammed I. Syam, UAE University, United Arab Emirates
Debasish Tidakar, Sultan Qaboos University, Oman
Philippe L. Toint, University of Namur, Belgium
Chefi Triki, Sultan Qaboos University, Oman
Md Jashim Uddin, Sultan Qaboos University, Oman
Maria Teresa Vespucci, Bergamo University, Italy
Gerald Wanjala, Sultan Qaboos University, Oman
Basuki Widodo, Institute of Technology Sepuluh November, Indonesia
Fengmin Xu, Xian Jiaotong University, China
Riadh Zaier, Sultan Qaboos University, Oman
Muhammad Ziad, Sultan Qaboos University, Oman

المؤتمر الدولي الرابع حول
التحليل العددي و الحلول المثلى

برنامج حفل الافتتاح
الإثنين
2 كانون الثاني/ يناير 2017
قاعة المؤتمرات
جامعة السلطان قابوس
مسقط، سلطنة عمان

The 4th International Conference on
Numerical Analysis and Optimization

Opening Ceremony Programme
Monday
2 January 2017
Conference Hall
Sultan Qaboos University
Muscat, Sultanate of Oman



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 IV 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 Bagirov
University of Ballarat, Australia
Oleg Burdakov
Linkoping University, Sweden
Andrew R. Conn
IBM Research Center, USA
Yu-Hong Dai
Chinese Academy of Sciences, China
Iain Duff
Rutherford Appleton Laboratory, UK
Giovanni Fasano
University of Venice Ca' Foscari, Italy
David M. Gay
AMPL Optimization Inc., USA
Francesca Guerriero
Calabria University, Italy
Michael Hintermuller
Humboldt-Universität zu Berlin, Germany
Nezam Mahdavi-Amiri
Sharif University of Technology, Iran
Hans Mittelmann
Arizona State University, USA
Dominique Orban
Ecole Polytechnique de Montreal, Canada
Amiya Kumar Pani
IIT, Bombay, India
Florian A. Potra
University of Maryland, USA
Cornelis Roos
Delft University of Technology, Netherlands
Ekkehard W. Sachs
University of Trier, Germany
Michael Saunders
Stanford University, USA
Abul Hasan Siddiqi
Sharda University, India
Philippe L. Toint
University of Namur, Belgium
Maria Teresa Vespucci
Bergamo University, Italy

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
Sebti Kerbal, SQU, Oman
Amar Oukil, SQU, Oman
Anton Purnama (Co-Chair), SQU, Oman
Chefi Triki, SQU, Oman
Bernhard Heim, GUtech, Oman
Florian Rupp, GUtech, Oman
Issam Moghrabi, GUST, Kuwait
Muhammad Syam, UAE Univ., UAE
Lucio Grandinetti, Calabria Univ., Italy

Venue:
Conference Hall, SQU, Oman
Email:
nao2017@squ.edu.om
naoiv2017@gmail.com
Website:
https://conference.squ.edu.om/Default.aspx?tabid=2157

Sponsors:


NAOIV
Hasan 1911