# The Homogeneous Interior-Point Algorithm: Nonsymmetric Cones, Warmstarting, and Applications

**Skajaa, Anders; Hansen, Per Christian; Jørgensen, John Bagterp**

*Publication date:*
2013

*Document Version*
Publisher's PDF, also known as Version of record

# The Homogeneous Interior-Point Algorithm: Nonsymmetric Cones, Warmstarting, and Applications

Anders Skajaa

Kongens Lyngby 2013
IMM-PhD-2013-311

Technical University of Denmark
DTU Compute
Department of Applied Mathematics and Computer Science
Matematiktorvet, building 303B
DK-2800 Kongens Lyngby, Denmark.
Phone: +4545253031. Email: compute@compute.dtu.dk
Web: www.compute.dtu.dk IMM-PhD-2013-311

# Summary (English)

The overall topic of this thesis is *convex conic optimization*, a sub-field of mathematical optimization that attacks optimization problem with a certain geometric structure. These problems allow for modelling of an extremely wide range of real-world problems, but the availability of solution algorithms for these problems is still limited.

The goal of this thesis is to investigate and shed light on two computational aspects of homogeneous interior-point algorithms for convex conic optimization: The first part studies the possibility of devising a homogeneous interior-point method aimed at solving problems involving constraints that require nonsymmetric cones in their formulation. The second part studies the possibility of warmstarting the homogeneous interior-point algorithm for conic problems.

The main outcome of the first part is the introduction of a completely new homogeneous interior-point algorithm designed to solve nonsymmetric convex conic optimization problems. The algorithm is presented in detail and then analyzed. We prove its convergence and complexity. From a theoretical viewpoint, it is fully competitive with other algorithms and from a practical viewpoint, we show that it holds lots of potential, in several cases being superior to other solution methods.

The main outcome of the second part of the thesis is two new warmstarting schemes for the homogeneous interior-point algorithm for conic problems. Again, we first motivate and present the schemes and then analyze them. It is proved that they, under certain circumstances, result in an improved worst-case complexity as compared to a normal coldstart. We then move on to present an

extensive series of computational results substantiating the practical usefulness of these warmstarting schemes. These experiments include standard benchmarking problem test sets as well as an application from smart energy systems.

# Summary (Danish)

Det overordnede emne for denne afhandling er *konveks konisk optimering*, et underområde af matematisk optimering som angriber problemer med en bestemt geometrisk struktur. Disse problemer muliggør modellering af en ekstremt bred vifte af virkelige problemer, men tilgængeligheden af løsningsalgoritmer til disse problemer er stadig meget begrænset.

Målet for denne afhandling er at undersøge og kaste lys over to beregningsmæssige aspekter af den homogene indre-punkts algoritme til løsning af konvekse koniske optimeringsproblemer: Den første del studerer muligheden for at udvikle en homogen indre-punkts metode målrettet løsning af problemer, der indeholder begrænsninger, som kræver ikke-symmetriske kegler i deres beskrivelse. Den anden del studerer muligheden for at varmstarte den homogene indre-punkts algorithme til koniske problemer.

Hovedresultat af den første del er introduktionen af en helt ny homogen indre-punkts algoritme designet til at løse ikke-symmetriske konvekse koniske optimeringsproblemer. Denne algoritme præsenteres i detaljer og derefter analyseret. Vi beviser dens konvergens og kompleksitet. Fra et teoretisk synspunkt er den fuldt kompetitiv med mere generelle metoder og fra et praktisk synspunkt viser vi, at den indeholder stort potentiale; mange gange er den at foretrække frem for andre løsningsmetoder.

Hovedresultatet af den anden del af afhandlingen er to nye varmstart metoder til den homogene indre-punkts algorithm til koniske problemer. Som før motiverer og præsenterer vi først metoderne og derefter analyseres de. Det bevises, at de, under visse omstændigheder, resulterer i en forbedret værste-falds kompleksitet

når man sammenligner med sædvanlig koldstart. We fortsætter derefter med præsentationen af en omfattende serie af beregningsresultater der understøtter den praktiske anvendelighed af disse varmstart metoder. Eksperimenterne inkluderer standard benchmark problemsamlinger såvel som en anvendelse, der stammer fra smarte energisystemer.

# Preface

This thesis was prepared at the Department of Applied Mathematics and Computer Science (DTU Compute) at the Technical University of Denmark in fulfilment of the requirements for acquiring an PhD degree in informatics.

The thesis deals with various aspects of interior-point algorithms for convex conic optimization problems. The main contributions fall in the area of interior-point methods for nonsymmetric conic optimization and warmstarting interior-point methods. The thesis consists of four main chapters: Firstly, an introductory chapter outlining the background material necessary. Secondly, three chapters all concerned with interior-point methods: first for linear programming, then for convex conic programming and finally warmstarting of interior-point methods.

Included as part of the thesis are the two journal papers which both can be found in the appendix.

1. **Anders Skajaa** and Yinyu Ye: *A Homogeneous Interior-Point Algorithm for Nonsymmetric Convex Conic Optimization.* Submitted to Mathematical Programming. Accepted for publication on March 26th, 2014.
2. **Anders Skajaa**, Erling D. Andersen and Yinyu Ye: *Warmstarting the Homogeneous and Self-Dual Interior-Point Method for Linear and Conic Quadratic Problems.* Published in Mathematical Programming Computation. Volume 5, Issue 1, Pages 1–25, 2013.

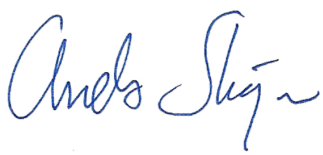
As part of the PhD programme, the original research contained in this thesis was presented at the following conferences and seminars:

1. SIAM Conference on Optimization (SIAM OP11). May 16th, 2011, Darmstadt, Germany.
2. Seminar on Optimization, May 24th, 2011. Center for Operations Research and Econometrics (CORE), Uni. Catholique de Louvain, Belgium.
3. Seminar on Linear Algebra and Optimization. January 19th, 2012, Institute for Computational and Mathematical Engineering (ICME), Stanford University, USA.
4. International Sympositum on Mathematical Programming (ISMP12). August 24th, 2012, Berlin, Germany.

Three external research stays were completed during the course of the PhD:

1. Center for Operations Research and Econometrics (CORE), Uni. Catholique de Louvain, Belgium, May 23rd – 25th, 2011. Host: Prof. Francois Glineur.
2. Institute for Computational and Mathematical Engineering (ICME), Stanford University, USA, January 16th – 25th, 2012. Host: Prof. Yinyu Ye.
3. Institute for Computational and Mathematical Engineering (ICME), Stanford University, USA, July 12th – August 15th, 2012. Host: Prof. Yinyu Ye.

Lyngby, 30-June-2013



Anders Skajaa

# Acknowledgements

During the course of my PhD programme, I have drawn support from many people to whom I am very thankful. Obvious to anyone who has completed a PhD study, this process can not be realized without a great deal of academic and personal support.

First and foremost, I thank my advisors Professor Per Christian Hansen and Associate Professor John Bagterp Jørgensen. John has providing me with inspiration from a range of interesting and relevant applications of mathematical optimization and initiated and facilitated much valuable collaboration between myself and co-workers ultimately leading to stimulating research. Per Christian has offered a constant availability to discuss anything ranging from trivial to highly intricate technical issues, from office politics to good academic writing.

This project has been highly influenced by several external collaborators whose input and guidance I have greatly appreciated. Particularly I want to thank experts in the field of optimization Erling D. Andersen and Joachim Dahl of Mosek ApS for offering many hours of their time and numerous of their ideas to me. This collaboration in turn led to a fruitful collaboration with Professor Yinyu Ye, whom, aside from unbelievable intuition and insight into mathematical optimization, I thank for great hospitality during my two research stays at ICME at Stanford University. Similarly, I am grateful to Prof. Francois Glineur for a lot of early inspiration when he welcomed me at CORE at UC Louvain.

Finally, I am happy to have spent three years in great company at IMM at the Technical University of Denmark. Many inspiring hours have been spent talking to co-workers about everything and nothing: Leo Emil Sokoler, Rasmus Halvgaard, Carsten Völcker, Tobias Hovgaard, Dimitri, Andrea, Fabrice and particularly my office-mate and friend Jakob H. Jørgensen.

Above all, I am grateful to my family for enduring the ups and downs that the PhD process causes: Thank you, Johan and Gitte.

# Notation

The table below shows most of the notation used repeatedly in this thesis. The notation is aimed at being as consistent with standard interior-point litterature as possible without being confusing or ambiguous.

| **Symbol** | **Meaning** |
|---|---|
| $\mathbb{R}$ | The real numbers |
| $\emptyset$ | The empty set |
| $\mathcal{C}$ | A set (usually a convex set) |
| $\mathcal{K}$ | A cone (usually the primal cone in a conic program) |
| $\mathcal{K}^*$ | The dual cone of $\mathcal{K}$ |
| $\text{int}(\mathcal{C})$ | The interior of a set $\mathcal{C}$ |
| $\partial\mathcal{C}$ | The boundary of a set $\mathcal{C}$ |
| $F$ | Primal barrier function |
| $F^*$ | The conjugate of $F$ |
| $\|\cdot\|$ | A norm |
| $\mu(x,s)$ | Complementarity gap defined as $x^T s/n$ |
| $\sigma$ | Centering parameter |
| $\tau$ | Parameter indexing the central path points |
| $\nu$ | Barrier parameter |
| $\circ$ | Jordan-product |
| $X \succeq 0$ | Denotes that $X$ is a positive semi-definite matrix |
| $\text{diag}(x)$ | The diagonal matrix with $x$ in the diagonal |
| $L_2$ | The Lorentz cone (2-norm cone) |
| $S_+^n$ | The cone of symmetric, positive semi-definite $n \times n$-matrices |
| $\mathcal{K}_{\text{exp}}$ | The exponential cone |
| $\mathcal{K}_\alpha$ | The power cone with parameter $\alpha$ |

Some acronyms are used throughout the thesis for simplicity and clarity. They are listed in the following table.

| **Acronym** | **Meaning** |
|---|---|
| IPM | Interior-Point Method |
| PDIPM | Primal-Dual Interior-Point Method |
| LHSCB | Logarithmically Homogeneous Self-Concordant Barrier |
| HSD | Homogeneous and Self-Dual |
| LP | Linear Program |
| QP | Quadratic Program |
| SDP | Semidefinite Program |
| MPC | Model Predictive Control |

# Contents

Chapter 1

# Introduction

It is not hard to argue why the field of mathematical optimization is useful and important — both in science, engineering and society in general: it is concerned with determining the *best* possible decision amongst a large number of possible decisions. Of course, it is not successful in answering every such question but nevertheless successful enough that it has found applications in a vast number of industries. Examples include portfolio management, automatic control, artificial intelligence, web commerce, logistics, production scheduling, engineering design, automated trading and much more.

A number of challenges lie between the wish to find this optimal decision and the possibility of using mathematical optimization to do so. Firstly, one must *formulate* his problem using only a strict and standardized mathematical language. This can often prove difficult because it requires the exact identification a several quantities: (1) the *decision variables*, which represent the choices that we can actually control, (2) the *objective function*, which is the object we want to minimize (or maximize) — for example a cost (or profit) and (3) the *constraints*, which precisely state which combinations of variables are not allowed, for example because of physical limitations.

Secondly, some way to *solve* the problem must be devised. For all but the very simplest and uninteresting optimization problems, it is not possible to derive a finite formula describing the solution. Instead, mathematicians construct

*algorithms* that, step by step, build a solution to the problem by following a predefined set of rules which ultimately lead to a solution of the problem. An algorithm is designed to be effective for a certain subset of problems. In other words, the rules defining the algorithm are specifically chosen to work particularly well for that specific subset of problems. If the subset is very small, very targeted rules can be chosen and we therefore expect a very effective (i.e. fast and robust) algorithm. On the other hand, if the subset is very large, the rules must be more general, and thus the algorithm is expected to work less efficiently.

The sub-field of this thesis is *convex optimization*, which describes a class of problems with a useful geometric property: convexity. This sub-field pursues the strategy of "smaller subsets of problems" and "more targeted algorithm rules" in the sense outlined above. Indeed, the mathematical language that must be used in formulating the optimization problem is stricter. The reward is more efficient algorithms.

In this thesis, we take the slightly more specific viewpoint of *conic optimization*. Theoretically, it is no less general than convex optimization, but the mathematical formulation required of the problem is in a certain sense even stricter. The problems that are allowed must be described as compositions of convex cones. The advantage lies again in the ability to make even more efficient algorithms, which are designed to take advantage of the specific geometric properties of each type of cone allowed. The idea is then to accumulate a repertoire of cones that can be combined in every imaginable way. In this sense, the cones play the role of *atoms* and problems the roles of *molecules*. The analysis of algorithms then becomes very elegant in the sense that the atoms can be treated separately even if the molecules are very complex.

One could fear that this demanded conic structure would severely limit the amount of problems that are practical to formulate. But this turns out not to be the case. As well shall see, the conic structure is very versatile and allows for modelling of an extremely wide range of problems.

**Thesis overview.** We begin the thesis by providing the background material necessary to presenting the main contributions which appear later in the thesis. This background material includes such core concepts as convexity, cones, Newton's method for self-concordant functions, barriers and conjugate barriers. We then present general convex optimization problems, their dual problems and optimality conditions. We then move on to conic problems and introduce for this class the homogeneous and self-dual model. Readers familiar with all of the above material can safely skip the entire Chapter 2.

The remaining part of the thesis is concerned only with interior-point methods and applications thereof. As the primordial example of a convex conic optimization problem, we first review in Chapter 3 interior-point algorithms for linear programming. This problem class is attractive to study because it is sufficiently complex to capture the core ideas of interior-point methods yet sufficiently simple to not require too much tedious notation and technicalities. We introduce the crucial concept of the central path and consider four of the historically most prominent interior-point algorithms. Thus, this chapter also contains no original contributions but instead the concepts discussed will serve as reference points for the contributions that follow in the two following chapters.

In Chapter 4, we finally reach the subject at the core of this thesis: interior-point methods for convex conic programming. The chapter mainly discusses one of the core difficulties with this problem class: the symmetric linearization of the complementarity conditions. We describe the algorithmic consequences of the convex cone begin nonsymmetric and then move on to present the main contribution of the thesis: the introduction of an efficient interior-point algorithm to solve nonsymmetric convex conic optimization problems. The core results are outlined in the main text, but the details and proofs are diverted to the appendix where a research paper is included in full length.

Finally, Chapter 5 deals with the issue of initializing an interior-point algorithm: Which point should the algorithm use as its starting point? This is still a relatively undecided problem, albeit it is not uncommon that the iteration count for an interior-point algorithm starting in one particular point may be several times that of a good point for the same problem. For this reason alone, this issue deserves attention. We first outline the common heuristics that are somewhat agreed upon constitute good pointers as to what a good starting point is. We then move on to the second main original contribution of this thesis: *warmstarting* an interior-point algorithm. How does one utilize information from one optimization problem when solving a similar, but different problem? Again, we outline the results in the main text but refer to the full-length research paper which is included in the appendix. Following the presentation of our paper, we demonstrate in a case study about smart energy systems and automatic control, how our warmstarting scheme may be used to speed-up the internal computations that take place when a process is being governed by an automatic control regime. This case study thus serves as a demonstration of the practical applicability of warmstarting of interior-point methods.

Chapter 2

# Optimization Background

The purpose of this chapter is to introduce the fundamental concepts essential to presenting this thesis' main contributions which appear in later chapters. We aim at providing a clear and simplistic presentation with focus on light notation and include only those proofs that highlight important properties while remaining of predominantly non-technical nature.

We first define and give examples of convex sets, functions, cones and dual cones, all of which are absolutely core material in the theory of interior point methods. We then move on to present the classical Newton's method for unconstrained optimization starting from self-concordant functions, which allow for a particularly nice and simple analysis. This naturally continues into logarithmically homogeneous and self-concordant barrier functions and their conjugates which are absolutely crucial in the analysis of interior-point methods. We then review general convex constrained optimization problems, their duals and optimality conditions and consider a few examples of such problems. This extends into convex conic optimization problem, the problem class at the core of this thesis. We highlight the remarkable dual symmetry present with this family of problems and give a series of examples of problems, all of which will be treated again later in the thesis. Finally, we present the so-called homogeneous and self-dual model which offers itself as the solution to two practical issues when solving convex conic optimization problems: the identification of infeasible problems and determining a suitable starting point for the algorithm.

The content of this chapter is overall well-known material. Readers familiar with the above terms can therefore safely skip this entire chapter.

For a more comprehensive treatment of the contents of this chapter, the reader is directed to the following references: [7, 12, 21, 40, 45, 49].

## 2.1 Convexity

### 2.1.1 Convex sets

A set $\mathcal{C} \subseteq \mathbb{R}^n$ is convex if

$$\forall x, y \in \mathcal{C} \text{ and } \lambda \in [0,1]: \quad \lambda x + (1-\lambda) y \in \mathcal{C}. \tag{2.1.1}$$

That is: the convex combination of any two points in $\mathcal{C}$ is also contained in $\mathcal{C}$. Geometrically, one can think of convexity in the following way: Take any two points in $\mathcal{C}$ and draw the line segment connecting them. The line segment must be contained in $\mathcal{C}$. If this holds for any two points in $\mathcal{C}$, then $\mathcal{C}$ is convex.

A few examples of convex sets:

1. The empty set $\emptyset$ and $\mathbb{R}^n$.
2. An orthant, for example: $\{x \in \mathbb{R}^n : x_i \geq 0\}$.
3. A norm-ball: $\{x \in \mathbb{R}^n : \|x\| \leq r\}$ for some norm $\|\cdot\|$.
4. An affine subspace: $\{x \in \mathbb{R}^n : Ax = b\}$.
5. The intersection of two convex sets.

### 2.1.2 Convex functions

A function $F : \mathcal{C} \mapsto \mathbb{R}$ is convex if $\mathcal{C}$ is convex and

$$\forall x, y \in \mathcal{C}, \lambda \in [0,1]: \quad F(\lambda x + (1-\lambda) y) \leq \lambda F(x) + (1-\lambda) F(y) \tag{2.1.2}$$

Geometrically, this means that the line segment connecting the two points $(x, F(x))$ and $(y, F(y))$ is an upper bound of $F$ on the interval $[x, y]$.

If $F$ is convex and also differentiable, then another equivalent definition of convexity is

$$\forall x, y \in \mathcal{C}: \quad F(y) \geq F(x) + \nabla F(x)^T (y - x) \tag{2.1.3}$$

which means that the hyperplane tangent to a point on the graph of $F$ is a lower bound on $F$.

Finally, if $F$ is twice differentiable, then convexity is equivalent to

$$\forall x \in \mathcal{C}: \quad \nabla^2 F(x) \succeq 0 \tag{2.1.4}$$

i.e. the Hessian of $F$ must be positive semidefinite on $\mathcal{C}$. See for example [12] for examples and many other properties of convex functions.

## 2.2 Convex cones

A convex cone $\mathcal{K}$ is a set that satisfies

$$\forall x, y \in \mathcal{K},\ \alpha, \beta > 0: \quad \alpha x + \beta y \in \mathcal{K}. \tag{2.2.1}$$

This means that a convex cone is a convex set that further satisfies that if a point is in the cone, then the ray starting in the origin and passing through the point is also contained in the cone.

A few examples of convex cones:

1. The set $\mathbb{R}^n_+$ is clearly a convex cone for if $x$ and $y$ are vectors with non-negative elements then also $\alpha x + \beta y$ for $\alpha, \beta > 0$ will have non-negative elements.

2. For a parameter $\alpha$, the set

$$\mathcal{K}_\alpha = \left\{(x_1, x_2, x_3) \in \mathbb{R} \times \mathbb{R}^2_+ : |x_1| \leq x_2^\alpha x_3^{1-\alpha}\right\} \tag{2.2.2}$$

   is a convex cone. $\mathcal{K}_\alpha$ is called the three-dimensional *power cone* and can used to model constraints involving power functions.

   As an example, consider the problem of finding the point in a given affine space with the property that its $p$-norm is least among all points in the space:

$$\begin{array}{ll} \min_x & \|x\|_p \\ \text{s.t.} & Ax = b \end{array}$$

   where $p \geq 1$ and $A \in \mathbb{R}^{m \times n}$. It is not hard to see [21] that this problem is equivalent to

$$\begin{array}{ll} \min_{x,y,t} & t \\ \text{s.t.} & Ax = b,\ e^T y = t \\ & (x_j, y_j, t) \in \mathcal{K}_{1/p}, \quad j = 1, \ldots, n. \end{array}$$

3. The set

$$S_+^n = \left\{X \in \mathbb{R}^{n\times n} : \ X \text{ symmetric and } X \succeq 0\right\} \tag{2.2.3}$$

is a convex cone. Here, $X \succeq 0$ means that $X$ is a positive semi-definite matrix. Notice that because the members of $S_+^n$ must be symmetric matrices, this set can be identified with a smaller space with only $n(n+1)/2$ degrees of freedom.

A quite non-restrictive assumption often required by cones in theoretical texts is that it be *proper*. This means that the cone is convex, closed, pointed and solid. The first, we have already defined. A closed set is one that contains its own boundary, i.e. its complement is open. A pointed cone is one that contains no straight line of infinite length. A cone is solid if its interior is not empty. Therefore, it is meaningful to think of a proper cone as a *non-degenerate* cone.

### 2.2.1 Dual cones

Given a convex cone $\mathcal{K}$, let us define the set

$$\mathcal{K}^* = \{s \in \mathbb{R}^n : s^T x \geq 0, \ \forall x \in \mathcal{K}\}.$$

This set is called the *dual cone* of $\mathcal{K}$ and it is easy to see that indeed it is a cone. It further holds that when $\mathcal{K}$ is proper, $\mathcal{K}^*$ is also proper.

As an example, consider the cone $\mathbb{R}_+^n$. Since it clearly holds that

$$\forall x \in \mathbb{R}_+^n : x^T y \geq 0 \quad \Leftrightarrow \quad y \geq 0$$

we have $(\mathbb{R}_+^n)^* = \mathbb{R}_+^n$. It is therefore said that $\mathbb{R}_+^n$ is *self-dual*. Similarly, it can be shown that the set $S_+^n$ defined above is self-dual.

### 2.2.2 Homogeneity and symmetry

If the set (group) of all linear maps $\mathcal{L}$ such that $\mathcal{L}\mathcal{K} = \mathcal{K}$ acts *transitively* on $\mathcal{K}$, then the cone $\mathcal{K}$ is called *homogeneous*. This means that for a homogeneous cone $\mathcal{K}$, we have

$$\forall x, y \in \mathcal{K} : \quad \exists \text{ linear operator } \mathcal{L} \text{ such that } \mathcal{L}\mathcal{K} = \mathcal{K} \text{ for which } \mathcal{L}x = y.$$

A convex cone that is both homogeneous and self-dual is called *symmetric*.

In [25], it was shown that any symmetric cone consists of a (unique) Cartesian product of *irreducible* symmetric cones, of which only five exist. These irreducible cones can therefore be thought of as a basis of all symmetric cones. For optimization, currently only *two* of these cones are of practical interest. They are:

1. The Lorentz cone, defined by

$$L_2^n = \left\{(x,t) \in \mathbb{R}^{n+1} : \quad \|x\|_2 \leq t\right\} \tag{2.2.4}$$

   is symmetric (see e.g. [12]). It is known under different names, e.g. the *ice-cream cone*, the *second order cone* or the *2-norm cone*.

   This cone is used in modelling constraints involving quadratic terms. It allows, for example, formulation of quadratically constrained quadratic programs, second order cone programs and constraints involving rational powers of the variables (see [40]).

2. The set of all positive semidefinite and real symmetric matrices, $S_+^n$, is also a symmetric cone. See [12] for a proof of this fact. This cone is used in formulating so called semidefinite programs. These programs are very versatile and can be used in a variety of optimization models.

Notice that the positive orthant $\mathbb{R}_+^n$ is a direct product of $n$ copies of the latter cone. This is clear because $S_+^n$ for the case $n = 1$ reduces to containing non-negative scalars.

## 2.3 Self-concordant functions

Let $F$ be a convex function defined on the set $\mathcal{D}$. It is very convenient to define the following *local* Hessian-norm for a point $y$:

$$\|y\|_x = \sqrt{y^T H(x) y} \tag{2.3.1}$$

where $H(x) = \nabla^2 F(x)$, the Hessian of $F$ at $x$.

Let us also define the open ball centered at $y$ of radius $r$ measured in the local Hessian-norm:

$$B_x(y,r) = \{v : \ \|v - y\|_x < r\}. \tag{2.3.2}$$

The concept of *self-concordant* functions plays an important role in analyzing the efficiency of the classical Newton's method. A self-concordant function $F$ is one that satisfies the two conditions

$$x \in \mathcal{D} \quad \Rightarrow \quad B_x(x,1) \subseteq \mathcal{D} \tag{2.3.3}$$

$$y \in B_x(x,1) \quad \Rightarrow \quad 1 - \|y-x\|_x \leq \frac{\|v\|_y}{\|v\|_x} \leq \frac{1}{1-\|y-x\|_x} \tag{2.3.4}$$

for any $v \neq 0$.

Notice that this definition, which is used in [49], is somewhat different from the original definition of self-concordant function given in [40]. Although there are slight technical differences, the above conditions define the same family of functions but where "degenerate" functions are eliminated from the familiy. See [49, §2.5] for a precise statement about the equivalence of the two definitions. We use the definition (2.3.3)-(2.3.4) because it is better suited for the later analysis.

## 2.4 Unconstrained optimization and Newton's method

Given a convex function $F$ on a convex domain $\mathcal{D}$, consider the unconstrained optimization problem

$$\min_{x \in \mathcal{D}} F(x). \tag{2.4.1}$$

Let us define the *Newton step* for $F$ at $x$ by

$$N_F(x) = -\nabla^2 F(x)^{-1} \nabla F(x). \tag{2.4.2}$$

If the current iterate is $x$, Newton's method finds the next iterate $x^+$ from

$$x^+ = x + N_F(x). \tag{2.4.3}$$

One way to motivate this step is to consider the quadratic model of $F$ around $x$ (the second order Taylor expansion):

$$Q_x(y) = F(x) + (y-x)^T g(x) + \tfrac{1}{2}(y-x)^T H(x)(y-x)$$

where again $H(x) = \nabla^2 F(x)$ and $g(x) = \nabla F(x)$. In order to minimize $Q_x(y)$ w.r.t. $y$, let us find the point where $\nabla_y Q_x(y) = 0$:

$$\begin{aligned} \nabla_y Q_x(y) = g(x) + H(x)(y-x) &\stackrel{!}{=} 0 \quad \Rightarrow \\ y &= x - H(x)^{-1} g(x) \\ &= x + N_F(x). \end{aligned}$$

So the next iteration in Newton's method is indeed the minimizer of the quadratic model around $x$. The function $F$ being convex, the intuitive rationale is that when we approach a minimizer of $F$, the quadratic approximation $Q$ will be an increasingly better model of $F$ and hence (2.4.3) will bring a large improvement.

When $F$ is assumed to be self-concordant, the convergence analysis of Newton's method is particularly elegant and simple. It is expressed very clearly in terms of the local Hessian-norm defined in Section 2.3. Because Newton's method is at the core of the analysis of interior point methods, we include below a few results explaining the convergence behavior of Newton's method under these assumptions. The presentation here is based on that found in [49]. We skip the proofs (which are never more than 10 lines long) and instead comment on the meaning and role of each of the theorems.

The first theorem shows how the local norm of the Newton step decreases.

**Theorem 2.2.4 of [49].** *If $F$ is self-concordant and $\|N_F(x)\|_x < 1$, then*

$$\|N_F(x^+)\|_{x^+} \leq \left(\frac{\|N_F(x)\|_x}{1 - \|N_F(x)\|_x}\right)^2. \tag{2.4.4}$$

This theorem states that when $\|N_F(x)\|_x$ is sufficiently small, the Newton step converges quadratically to zero when measured in the local Hessian norm. If, for example, $\|N_F(x)\|_x = 1/k$, the theorem implies that $\|N_F(x^+)\|_{x^+} \leq 1/(k-1)^2$.

The theorem only states, however, that once the Newton step is small enough, it decreases quickly to zero. Does this also mean that we are close to a solution of (2.4.1)? The following theorem answers this question positively:

**Theorem 2.2.5 of [49].** *If $F$ is self-concordant and $\|N_F(x)\|_x < 1/4$, then $F$ has a minimizer $x^\star$ and*

$$\|x^\star - x\|_x \leq \|N_F(x)\|_x + \mathcal{O}(\|N_F(x)\|_x^2). \tag{2.4.5}$$

Because of (2.4.4), we can apply this theorem to the next iterate $x^+$. When $\|N_F(x)\|_x \leq 1/4$, we have $\|N_F(x^+)\|_{x^+} \leq 1/9 < 1/4$ and therefore (dropping the higher order term):

$$\|x^\star - x^+\|_{x^+} \overset{(2.4.5)}{\leq} \mathcal{O}\big(\|N_F(x^+)\|_{x^+}\big) \overset{(2.4.4)}{\leq} \mathcal{O}\big(\|N_F(x)\|_x^2\big)$$

This shows that $\|x^\star - x\|_x$, i.e. the distance from the iterates to the minimizer, decreases quadratically to zero.

The quantity $\|N_F(x)\|_x$ also provides a suitable stopping criterion for Newton's method. It holds that (see [12, p. 502])

$$F(x) - F(x^\star) \leq \|N_F(x)\|_x^2$$

when $\|N_F(x)\|_x < 0.68$.

The previous results explain the behavior of Newton's method once a sufficiently small neighborhood of the optimal point $x^\star$ has been reached. This neighborhood can be characterized by $\|N_F(x)\|_x \leq 1/4$. Once this is the observed, the algorithm is said to be in the *quadratically convergent phase*, since it converges quadratically to the solution. In practice, this is extremely fast as it means that a tolerance on the order of machine precision can be reached within 6–8 iterations.

In order to make sure that the quadratically convergent phase is eventually reached, Newton's method must be slightly modified:

$$x^+ = x + tN_F(x). \tag{2.4.6}$$

where $t \in (0, 1]$ is a step size chosen in each iteration to satisfy certain criteria ensuring a sufficient decrease in $F$. Under mild conditions, such a $t$ always exists and it then holds (see [12, p. 504]) that there is a constant $\gamma > 0$ dependent only on these criteria so that

$$F(x^+) - F(x) \leq -\gamma. \tag{2.4.7}$$

That is, the function $F$ is reduced at least by a fixed amount in each iteration. This phase is called the *Damped Newton phase*. Because of (2.4.7) the quadratically convergent phase will eventually be reached. The full behavior for the method (2.4.6) can thus be summarized:

There is a positive constant $\eta \leq 1/4$ so that

> Case 1: $\|N_F(x)\|_x > \eta$ (Damped Newton phase). The step size $t$ can be chosen so that
>
> $$F(x^+) - F(x) \leq -\gamma.$$
>
> That is, a constant reduction in $F$ can be achieved.
>
> Case 2: $\|N_F(x)\|_x \leq \eta$ (quadratically convergent phase). With the step size $t = 1$, we have
>
> $$\|x^\star - x^+\|_{x^+} \leq \|N_F(x^+)\|_{x^+} \leq \mathcal{O}\big(\|N_F(x)\|_x^2\big).$$
>
> That is, the iterates approach the optimal point with quadratic convergence speed. Further, once this phase is entered, the method never leaves this phase again.

The latter statement follows from (2.4.4): If $\eta < 1/4$, then $(\eta/(1-\eta))^2 \leq 1/9 < 1/4$ so the next step also satisfies the condition of Case 2.

## 2.5 Barrier functions

The class of functions most important for interior-point methods is the *barrier functions*. A function $F$ is said to be a barrier function for the convex set $\mathcal{C}$ if

$$F(x) \to \infty \quad \text{for} \quad x \to \partial\mathcal{C}. \tag{2.5.1}$$

This should be understood as: take any sequence of point $x_i$ that approaches a point on $\partial\mathcal{C}$, the boundary of $\mathcal{C}$. Then if $F(x_i)$ approaches $\infty$, $F$ is called a barrier function of $\mathcal{C}$.

As an example, let $\mathcal{C} = \mathbb{R}_{++}$, the positive real numbers. Then

$$F(x) = -\log(x)$$

is a barrier function for $\mathcal{C}$.

### 2.5.1 Logarithmic homogeneity and self-concordance

If a barrier function $F$ satisfies for any $x$ in its domain that

$$F(\tau x) = F(x) - \nu \log \tau \tag{2.5.2}$$

for a fixed constant $\nu$ characteristic of the function, then $F$ is said to be logarithmically homogeneous with barrier parameter $\nu$. The property (2.5.2) implies the following properties (see [42]):

$$\nabla^2 F(x)x = -\nabla F(x) \tag{2.5.3}$$

$$\nabla F(\tau x) = \tau^{-1}\nabla F(x) \tag{2.5.4}$$

$$\nabla^2 F(\tau x) = \tau^{-2}\nabla^2 F(x) \tag{2.5.5}$$

$$x^T \nabla F(x) = -\nu \tag{2.5.6}$$

$$\|x\|_x^2 = \nu \tag{2.5.7}$$

If $F$ further is self-concordant, we call it a LHSCB-function, which is an acronym for Logarithmically Homogeneous Self-Concordant Barrier function.

As an example generalizing the example from before, consider now $\mathcal{C} = \mathbb{R}^n_+$, the non-negative orthant. The following function is certainly a barrier for $\mathcal{C}$:

$$F_+(x) = -\sum_{j=1}^{n} \log(x_j).$$

This is a self-concordant function (see [49, p. 24]). Let us see that it is also logarithmically homogeneous:

$$\begin{aligned} F_+(\tau x) &= -\sum_{j=1}^{n} \log(\tau x_j) \\ &= -\sum_{j=1}^{n} \log(x_j) - \sum_{j=1}^{n} \log(\tau) \\ &= F_+(x) - n\log(\tau), \end{aligned}$$

which shows that $F_+$ is logarithmically homogeneous with parameter $\nu = n$.

As a second example, let $\mathcal{C} = L_2^n$, the Lorentz cone of dimension $n+1$. See (2.2.4) for a definition of this convex cone. The function

$$F_2(x,t) = -\log(t^2 - \|x\|_2^2)$$

is self-concordant. Further, it is a logarithmically homogeneous barrier. Let us compute its barrier parameter:

$$\begin{aligned} F_2(\tau x, \tau t) &= -\log\left((\tau t)^2 - \|\tau x\|_2^2\right) \\ &= -\log\left(\tau^2\left(t^2 - \|x\|_2^2\right)\right) \\ &= -\log\left(t^2 - \|x\|_2^2\right) - \log\tau^2 \\ &= F_2(x,t) - 2\log\tau \end{aligned}$$

which shows that the barrier parameter of $F_2$ is $\nu = 2$.

### 2.5.2 The conjugate barrier

Given a LHSCB-function $F$ with domain $\mathcal{C}$, let us define

$$F^*(s) = \sup_{x\in\mathcal{C}} \left\{-s^T x - F(x)\right\} \tag{2.5.8}$$

which is called the *conjugate function* of $F$. This function is particularly interesting when $\mathcal{C}$ is a proper cone $\mathcal{K}$ (see Section 2.2) because then $F^*$ has domain $\mathcal{K}^*$ and is a LHSCB-function for this cone .

The connection between $F$ and $F^*$ is very important for IPM theory and, as we shall later see, helps to explain the relationship between a convex conic optimization problem and its dual.

The following relations connect the two functions (see [42]):

$$-\nabla F(x) \in \operatorname{int}(\mathcal{K}^*) = \operatorname{domain}(F^*) \tag{2.5.9}$$

$$-\nabla F^*(s) \in \operatorname{int}(\mathcal{K}) = \operatorname{domain}(F) \tag{2.5.10}$$

$$\nabla F\left(-\nabla F^*(s)\right) = -s \tag{2.5.11}$$

$$\nabla F^*\left(-\nabla F(x)\right) = -x \tag{2.5.12}$$

$$\nabla^2 F(-\nabla F^*(s)) = \nabla^2 F^*(s)^{-1} \tag{2.5.13}$$

$$\nabla^2 F^*(-\nabla F(x)) = \nabla^2 F(x)^{-1}. \tag{2.5.14}$$

The above relations show that there is a certain symmetry between the "primal" side where the domain is $\mathcal{K}$ on which $F$ is a LHSCB-function and the dual side where the domain is $\mathcal{K}^*$ on which $F^*$ is a LHSCB-function. As we will see, this symmetry can be extensively exploited when devising interior-point algorithms for optimization problems involving conic constraints.

A major question remains: Can $F^*$ be easily computed when $F$ is given? Unfortunately, the answer in the general case is no.

Let us consider an example where the answer is yes: Let again $\mathcal{C} = \mathcal{K} = \mathbb{R}^n_+$, the non-negative orthant. The LHSCB-function is again

$$F(x) = -\sum_{j=1}^{n} \log\left(x_j\right).$$

Let $1/v$ denote the elementwise reciprocal of the vector $v$, i.e. $1/v = (1/v_1, \ldots, 1/v_n)$. To determine when the supremum in (2.5.8) is attained, we compute

$$\begin{aligned} \nabla_x(-s^T x - F(x)) = -s + 1/x \overset{!}{=} 0 \quad &\Rightarrow \\ x = 1/s. \end{aligned}$$

Therefore,

$$\begin{aligned} F^*(s) &= -s^T(1/s) + \sum_{j=1}^{n} \log\left(1/s_j\right) \\ &= -\sum_{j=1}^{n} \log\left(s_j\right) - n. \end{aligned}$$

So we see that $F$ and $F^*$ in this case are equal up to a constant difference.

## 2.6 Convex constraints

A convex optimization problem is one with a convex objective function and convex constraints:

$$\begin{array}{rl} \min_x & f(x) \\ \text{s.t.} & x \in \mathcal{C} \end{array} \tag{2.6.1}$$

where $f$ is a convex function on the convex set $\mathcal{C}$. The *appearance* of problem (2.6.1) can be simplified slightly by defining

$$\bar{\mathcal{C}} = \{(x,t) \in \mathcal{C} \times \mathbb{R}_+ : f(x) \leq t\}$$

This set is clearly convex since it is the intersection of $\mathcal{C} \times \mathbb{R}_+$ with $\{(x,t) : f(x) \leq t\}$ which is convex since $f$ is convex. Then problem (2.6.1) can be equivalently written as

$$\begin{array}{rl} \min_{(x,t)} & t \\ \text{s.t.} & (x,t) \in \bar{\mathcal{C}}. \end{array} \tag{2.6.2}$$

Notice that (2.6.1) and (2.6.2) have the same form except that (2.6.2) is simpler: its objective function is linear.

Reformulations like this serve no real purpose other than simple convenience, however. For an optimization problem to be accessible by an algorithm executed on a computer, the crucial point is that it must be possible to communicate the constraints, in this case $\mathcal{C}$, to the computer. As the reformulation from (2.6.1) into (2.6.2) shows, the objective function can be subsumed into the constraints.

In this section, we describe first the general standard explicit representation of convex optimization problems. We then reach the optimization problem class at the core of this thesis: *convex conic programs*. As is the case in many other fields, systematic structure in the problem can usually be exploited to yield more efficient algorithms. As we shall see, the class of problems that can be cast as conic programs appears to precisely encompass the problems that can be solved by highly practically efficient primal-dual interior-point algorithms.

### 2.6.1 General case

One way to explicitly describe the feasible set $\mathcal{C}$ is by a linear matrix equality and an elementwise vector inequality:

$$\begin{array}{rl} \min_x & f(x) \\ \text{s.t.} & Ax = b \\ & h(x) \leq 0 \end{array} \tag{2.6.3}$$

where $f$ is still a convex function, $h$ is a vector valued function where each component is a convex function, $A$ is a matrix of suitable size and $b$ is a vector. The inequality $h(x) \leq 0$ should be understood in the elementwise sense. Then we have

$$\mathcal{C} = \{x : h(x) \leq 0,\ Ax = b\}. \tag{2.6.4}$$

Notice that it again is possible to subsume the general objective function $f$ into the constraints and replace it by a linear function. Similarly, the equation $Ax = b$ could be included in $h(x) \leq 0$. It is, however, preferable to maintain the form (2.6.3) as it reveals certain exploitable structure in the problem.

### 2.6.2 The dual problem

The optimization problem known as the *dual problem* of (2.6.3) is

$$\begin{array}{rl} \max_{(y,s)} & g(y,s) \\ \text{s.t.} & s \geq 0 \end{array} \tag{2.6.5}$$

where

$$g(y,s) = \inf_x L(x,y,s) \tag{2.6.6}$$

and

$$L(x,y,s) = f(x) + s^T h(x) + y^T(b - Ax), \tag{2.6.7}$$

the Lagrangian. The infimum in (2.6.6) must be taken over the intersection of the domains of $f$ and $h$. Let us mention a few important properties about the primal and dual problem pair. Assume $x^\star$ is an optimal solution to (2.6.3) and $(y^\star, s^\star)$ are optimal solutions to (2.6.5).

1. Duality gap: The difference

$$f(x^\star) - g(y^\star, s^\star) \tag{2.6.8}$$

   is known as the *duality gap*.

2. Weak duality: It always holds that

$$g(y^\star, s^\star) \leq f(x^\star). \tag{2.6.9}$$

   That is: the optimal value of the dual problem (2.6.5) is a lower bound on the optimal value of the primal problem (2.6.3). This means that the duality gap is always non-negative.

3. Strong duality: If

$$g(y^\star, s^\star) = f(x^\star) \tag{2.6.10}$$

then we say that *strong duality* holds. Not all convex optimization problem pairs satisfy strong duality (see [12] for examples). Different sufficient conditions for strong duality have been etablished. One such condition is *Slater's condition*: If there exists $x$ such that $Ax = b$ and $h(x) < 0$, then strong duality holds. That is, the problem is *strictly feasible*, i.e. there is a point $x$ that satisfies the convex inequalities of (2.6.3) *strictly*. This corresponds to saying that the relative interior of the set $\mathcal{C}$ defined in (2.6.4) is non-empty.

### 2.6.3 Karush-Kuhn-Tucker conditions

The Karush-Kuhn-Tucker conditions (or *KKT conditions*) for problem (2.6.3) are (see [12]): Any points $(x^\star, y^\star, s^\star)$ that satisfy

$$Ax^\star = b \tag{2.6.11}$$

$$h(x^\star) \leq 0 \tag{2.6.12}$$

$$s^\star \geq 0 \tag{2.6.13}$$

$$s^\star \circ h(x^\star) = 0 \tag{2.6.14}$$

$$\nabla f(x^\star) + [Dh(x^\star)]\, s^\star - A^T y^\star = 0. \tag{2.6.15}$$

are primal and dual optimal for (2.6.3) and (2.6.5) and have zero duality gap. Specifically, $x^\star$ is optimal for the primal problem (2.6.3) and $(y^\star, s^\star)$ are optimal for the dual problem (2.6.5) and $g(y^\star, s^\star) = f(x^\star)$. Here, $\circ$ denotes elementwise product[1] of vectors and $Dh$ is the Jacobian matrix of the function $h$.

The KKT conditions are particularly important in convex optimization because they provide necessary and sufficient conditions for optimality. Therefore, many optimization algorithms, including interior-point methods, are constructed with the overall goal of solving the KKT equations (2.6.11)–(2.6.15)

### 2.6.4 Examples

Some particularly important convex optimization problems deserve individual mention:

[1] Some texts use $\odot$ to denote the elementwise product. To emphasize the connection to Jordan algebra, we maintain the notation $\circ$. See e.g. [52] for further details.

1. Linear programming (LP): An LP in standard form is the following problem:

$$\begin{aligned} \min_x \quad & c^T x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0 \end{aligned} \tag{2.6.16}$$

where $x \in \mathbb{R}^n$ and $A, b$ and $c$ are a matrix and two vectors of suitable dimensions. Using the terminology of the previous sections, we have $f(x) = c^T x$ and $h(x) = -x$. By inserting these functions in (2.6.5), we see that the dual problem of (2.6.16) is

$$\begin{aligned} \max_x \quad & b^T y \\ \text{s.t.} \quad & A^T y + s = c \\ & s \geq 0. \end{aligned} \tag{2.6.17}$$

Similarly, we can find the KKT conditions from (2.6.11)–(2.6.15): Optimal primal and dual solutions $(x, y, s)$ to (2.6.16) and (2.6.17) satisfy

$$\left.\begin{aligned} Ax &= b \\ x &\geq 0 \end{aligned}\right\} \text{ primal feasibility} \tag{2.6.18}$$

$$\left.\begin{aligned} A^T y + s &= c \\ s &\geq 0 \end{aligned}\right\} \text{ dual feasibility} \tag{2.6.19}$$

$$\left.\begin{aligned} x \circ s = 0 \end{aligned}\right\} \text{ complementary slackness.} \tag{2.6.20}$$

The requirement (2.6.18) simply states that the primal solution must be feasible in the primal problem (2.6.16). Similarly, the requirement (2.6.19) states that the dual solution must be feasible in the dual problem (2.6.17). The last condition (2.6.20) is called *complementary slackness* and is, for a primal and dual feasible pair $x$ and $s$, sufficient for optimality of the linear program. As we shall see, one of the absolute most important questions to answer when designing algorithms to solve convex conic programs (of which LP is an example), is how to linearize the complementary slackness conditions (2.6.20).

2. Quadratic program (QP): A convex quadratic program is the following problem

$$\begin{aligned} \min_x \quad & \tfrac{1}{2} x^T Q x + c^T x \\ \text{s.t.} \quad & Ax = b \\ & Ex \leq d, \end{aligned} \tag{2.6.21}$$

where $Q \in \mathbb{R}^{n \times n}$ is a positive semi-definite and symmetric matrix, $A$ and $E$ are matrices of suitable dimensions and $c, b$ and $d$ are vectors of suitable length. Identifying each component of the problem with the corresponding elements of (2.6.3), we find $f(x) = \frac{1}{2} x^T Q x + c^T x$ and $h(x) = Ex - d$.

## 2.7 Conic constraints

A particularly important kind of convex optimization problem (2.6.1) is one where $\mathcal{C}$ is the intersection of an affine subspace and a proper cone (see Section 2.2):

$$\begin{array}{rl} \min_x & c^T x \\ \text{s.t.} & Ax = b \\ & x \in \mathcal{K} \end{array} \tag{2.7.1}$$

where $\mathcal{K}$ is a proper cone. A problem of this type is called a *convex conic program*. In all that follows, we make the non-restrictive assumption that $A$ has full row rank.

Let us first remark that, in principle, the constraint $x \in \mathcal{K}$ is no less general than the constraint $x \in \mathcal{C}$, where $\mathcal{C}$ is any convex set. This inclusion can be obtained by choosing $\mathcal{K}$ as the intersection of the conical hull of $\mathcal{C}$ and an affine subspace (see [40]). This reformulation has mostly theoretical interest, however. The main motivation for studying the convex conic program (2.7.1) is that particularly efficient primal-dual interior-point algorithms can be developed for certain convex cones and direct products of these cones. The fundamental reason is that by specifying convex optimization problems in this way, certain structure is revealed that can be exploited by the algorithm. In practice, the cone $\mathcal{K}$ is specified as

$$\mathcal{K} = \mathcal{K}_1 \times \mathcal{K}_2 \times \cdots \times \mathcal{K}_k \tag{2.7.2}$$

where each $\mathcal{K}_i$ is one of a number of *basic cones*. These basic cones have been very well studied and a lot is known as to how their structure should be exploited in the algorithm. In Section 2.7.3, a number of these basic cones are presented. Also examples of optimization problems that can be modelled in this way are presented.

### 2.7.1 Conic duality

A particularly pleasing feature about the convex conic program (2.7.1) is the symmetry that exists between the primal problem (2.7.1) and its Lagrangian dual problem. We associate with the conic constraint $x \in \mathcal{K}$ a Lagrange multiplier $s$ such that $s^T x \geq 0$. Notice that this is the same as $s \in \mathcal{K}^*$ (see Section 2.2.1). We then get the Lagragian

$$L(x, y, s) = c^T x - s^T x + y^T (b - Ax). \tag{2.7.3}$$

To find where the infimum of (2.7.3) is attained, we determine where its gradient vanishes:

$$\nabla_x L(x,y,s) = c - s - A^T y \overset{!}{=} 0 \quad \Rightarrow \tag{2.7.4}$$

$$c = A^T y + s. \tag{2.7.5}$$

When (2.7.5) is satisfied, we therefore find $L(x,y,s) = b^T y$, which is the objective function of the dual problem of (2.7.1). Therefore we arrive at the following primal and dual pair of convex conic optimization problems:

$$\text{Primal} \begin{cases} \min_x & c^T x \\ \text{s.t.} & Ax = b \\ & x \in \mathcal{K} \end{cases} \qquad \text{Dual} \begin{cases} \max_{y,s} & b^T y \\ \text{s.t.} & A^T y + s = c \\ & s \in \mathcal{K}^*,\ y \in \mathbb{R}^m. \end{cases} \tag{2.7.6}$$

It is readily seen that the pair (2.7.6) satisfies weak duality:

$$c^T x - b^T y = y^T A x + s^T x - b^T y = y^T b + s^T x - b^T y = s^T x \geq 0. \tag{2.7.7}$$

Following [49], let us make apparent the complete symmetry between the two problems. Assume that the two problems are feasible and fix an arbitrary primal point $\hat{x}$ satisfying $A\hat{x} = b$ and arbitrary dual points $(\hat{y}, \hat{s})$ satisfying $A^T\hat{y} + \hat{s} = c$. Then, similarly to (2.7.7) we have

$$x^T \hat{s} = \quad c^T x - b^T \hat{y} \tag{2.7.8}$$

$$\hat{x}^T s = -b^T y + c^T \hat{x}. \tag{2.7.9}$$

Notice that the second term in both (2.7.8) and (2.7.9) is constant while the first term is the primal respectively negative dual objective function. Now let $\mathcal{L}$ denote the nullspace of $A$:

$$\mathcal{L} = \{w : \ Aw = 0\}.$$

Then, we can write $Ax = b$ as $x \in \mathcal{L} + \hat{x}$. Similarly, we can write $A^T y + s = c$ as $s \in \mathcal{L}^\perp + \hat{s}$. Using (2.7.8) and (2.7.9) and that an additive constant in the objective function does not change the solution of an optimization problem, we can write the primal-dual pair (2.7.6) as

$$\text{Primal} \begin{cases} \min_x & x^T \hat{s} \\ \text{s.t.} & x \in \mathcal{L} + \hat{x} \\ & x \in \mathcal{K} \end{cases} \qquad \text{Dual} \begin{cases} \min_s & s^T \hat{x} \\ \text{s.t.} & s \in \mathcal{L}^\perp + \hat{s} \\ & s \in \mathcal{K}^* \end{cases} \tag{2.7.10}$$

which should make the symmetry clear. Because $(\mathcal{K}^*)^* = \mathcal{K}$, it is also clear that the dual of the dual problem in (2.7.10) is again the primal. Notice that $y$ was eliminated from the dual problem. Given $s$ and using the assumption that $A$ has full row rank, $y$ is fixed from the equation $A^T y + s = c$.

### 2.7.2 Karush-Kuhn-Tucker conditions

We saw in (2.7.7) that the conic problem pair satisfied weak duality. Similarly to the general convex problem (2.6.3), strong duality is satisfied under Slater's condition: If there exist $x \in \operatorname{int}(\mathcal{K})$ and $(y, s)$ such that $s \in \operatorname{int}(\mathcal{K}^*)$, $Ax = b$ and $A^T y + s = c$, then strong duality holds: There exist feasible optimal points $x^\star$ and $y^\star$ so that $b^T y^\star = c^T x^\star$.

Therefore, if Slater's condition holds, we need only search for points that satisfy the KKT conditions in order to find solutions for (2.7.6):

$$\begin{aligned} Ax - b &= 0 \\ -A^T y - s + c &= 0 \\ x^T s &= 0 \\ x \in \mathcal{K},\ s \in \mathcal{K}^*,\ y &\in \mathbb{R}^m \end{aligned} \tag{2.7.11}$$

It should be mentioned, that because of the extra geometric structure that the cone $\mathcal{K}$ and its dual cone $\mathcal{K}^*$ introduce in the convex conic problem pair, more can be said about solvability, boundedness and feasibility without assuming Slater's condition. For example, if it is only assumed that there are dual interior points $(y, s)$ so that $A^T y + s = c$ and $s \in \operatorname{int}(\mathcal{K}^*)$, then (a) if the dual problem is unbounded, the primal problem is infeasible (b) otherwise, the primal is feasible and strong duality holds. For many more details and results in this direction, see [52] and [7].

### 2.7.3 Examples

**1**. $\mathcal{K} = \mathbb{R}^n_+$. The primordial example of convex conic programming is linear programming LP:

$$\begin{aligned} \min_x \quad & c^T x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0. \end{aligned} \tag{2.7.12}$$

Notice that this problem is the result of simply replacing $\mathcal{K}$ in (2.7.1) by the cone $\mathbb{R}^n_+$. Some texts (e.g. [12]) emphasize the connection between linear programming and convex conic programming by defining a generalized inequality operator w.r.t. a cone: $x \succeq_{\mathcal{K}} 0 :\Leftrightarrow x \in \mathcal{K}$. Then the convex conic program (2.7.1) can be written $\min_x \{c^T x : \ Ax = b,\ x \succeq_{\mathcal{K}} 0\}$.

As we already saw in Section 2.2.2, the non-negative orthant $\mathbb{R}^n_+$ is a symmetric cone (i.e. it is homogeneous and is its own dual). Therefore, the conic constraint

in the dual problem in (2.7.6) is simply $s \geq 0$.

**2**. $\mathcal{K} = L_2^n$. When replacing $\mathcal{K}$ in (2.7.1) by the Lorentz cone

$$L_2^n = \left\{(x,t) \in \mathbb{R}^{n+1} : \quad \|x\|_2 \leq t\right\}, \tag{2.7.13}$$

we obtain a *quadratic cone program*. This cone allows for modelling of a great variety of optimization problems involving quadratic constraints.

As an example, let us consider a quadratic program:

$$\begin{array}{rl} \min_x & \frac{1}{2}x^T Q x \\ \text{s.t.} & Ax = b \\ & x \geq 0. \end{array}$$

where $Q \in \mathbb{R}^{n\times n}$ is a positive definite and symmetric matrix, $A$ is a matrix of suitable dimension and $b$ a vector of suitable length. This means that $Q$ can be Cholesky factorized. Let $L$ be the Cholesky factor of $Q$ so that $L^T L = Q$. Through the linear change of variables $w = Lx$, the quadratic program above becomes

$$\begin{array}{rl} \min_w & \frac{1}{2}\|w\|_2^2 \\ \text{s.t.} & AL^{-1}w = b \\ & L^{-1}w \geq 0 \end{array}$$

which we may write as (see [2])

$$\begin{array}{rll} \min & t & \\ \text{s.t.} & AL^{-1}w = b, & L^{-1}w - z = 0 \\ & u + v - \sqrt{2}t = 0, & u - v = \sqrt{2} \\ & z \geq 0,\ t \geq 0 & \\ & (w,v,u) \in L_2^{n+1}. & \end{array} \tag{2.7.14}$$

The problem (2.7.14) is now in the primal conic form (2.7.1). The decision variable is $(z,t,w,v,u)$ and the cone is $\mathcal{K} = \mathbb{R}_+^{n+1} \times L_2^{n+1}$.

Thus we can formulate quadratic programs as convex conic optimization problems by use of the two specific cones $\mathbb{R}_+$ and $L_2^n$. In fact, the quadratic cone is more general. For example, it also allows modelling of qudratically constrained quadratic programs and constraints involving rational powers (see [7]).

**3**. $\mathcal{K} = S_+^n$. As was mentioned in Section 2.2.2, the set of all positive semidefinite real symmetric matrices, $S_+^n$, is also a cone. When we replace $\mathcal{K}$ in (2.7.1) by this cone, we obtain a *semidefinite program*. The members of this cone are matrices rather than vectors of a real vector space. However, the identification between the two are readily made: The space $S_+^n$ is isomorphous

to the space $\mathbb{R}_+^{n(n+1)/2}$. The inner product $c^T x$ in $\mathbb{R}_+^{n(n+1)/2}$ should therefore be understood as the inner product trace$(CX)$ in $S_+^n$ where $C$ and $X$ are the matrices in $S_+^n$ corresponding to $c$ respectively $x$ in $\mathbb{R}_+^{n(n+1)/2}$ according to the isomorphism between the two spaces.

The semidefinite cone is quite general and very versatile. For example, it is more general than the two cones previously mentioned in this section. That is, any constraint $x \in \mathbb{R}_+^n$ or $x \in L_2^n$ can in principle be written as a semidefinite constraint. From an algorithmic viewpoint, this is usually not fruitful as the explicit structure of the two less general cones reveals structure in the problem that can be exploited.

**4.** $\mathcal{K} = \mathcal{K}_\alpha$. The three-dimensional *power cone* is defined by

$$\mathcal{K}_\alpha = \left\{(x_1, x_2, x_3) \in \mathbb{R} \times \mathbb{R}_+^2 : |x_1| \leq x_2^\alpha x_3^{1-\alpha}\right\} \tag{2.7.15}$$

where $\alpha \in [0, 1]$ is a parameter. The cone $\mathcal{K}_{1/2}$ is a rotated Lorentz cone. The power cone is useful in modelling constraints involving powers of variables with exponents no smaller than 1.

As an example, let us consider the optimization problem of finding the point in an affine space with the smallest $p$-norm:

$$\begin{array}{rl} \min_x & \|x\|_p \\ \text{s.t.} & Ax = b. \end{array} \tag{2.7.16}$$

Using the three-dimesional power cone, this problem can be modelled as

$$\begin{array}{rll} \min_{x,y,t} & t & \\ \text{s.t.} & Ax = b,\ \sum_{j=1}^n y_j = t & \\ & (x_j, y_j, t) \in \mathcal{K}_{(1/p)}, & j = 1, \ldots, n \end{array} \tag{2.7.17}$$

To see this, notice that the conic constraints are equivalent to

$$|x_j| \leq y_j t^{p-1}$$

and therefore

$$\sum_{j=1}^n |x_j|^p \leq t^{p-1} \sum_{j=1}^n y_j = t^p \quad \Rightarrow \quad \left(\sum_{j=1}^n |x_j|^p\right)^{1/p} \leq t$$

or $\|x\|_p \leq t$. Of course, $t$ can not be required to lie in multiple cones, so to conform to the format (2.7.1), it is necessary rewrite (2.7.17) into

$$\begin{array}{rll} \min_{x,y,t} & t_1 & \\ \text{s.t.} & Ax = b,\ \sum_{j=1}^n y_j = t_1 & \\ & t_1 = t_2 = \cdots = t_n & \\ & (x_j, y_j, t_j) \in \mathcal{K}_{(1/p)} & j = 1, \ldots, n. \end{array}$$

It should be mentioned, however, that an algorithm internally does not need to store $n$ copies of $t$.

**5.** $\mathcal{K} = \mathcal{K}_{\exp}$. The three-dimensional *exponential cone* is defined by

$$\begin{aligned} \mathcal{K}_{\exp} = \operatorname{closure}\{(x_1, x_2, x_3) \in \mathbb{R} \times \mathbb{R}_+ \times \mathbb{R}_{++} : \\ \exp(x_1/x_3) \leq x_2/x_3\}. \end{aligned} \tag{2.7.18}$$

This cone can be used for modelling constraints involving exponentials and logarithms.

As an example, consider the entropy maximization problem:

$$\begin{array}{ll} \min_x & \sum_{j=1}^n d_j x_j \log x_j \\ \text{s.t.} & Ax = b \\ & x \geq 0 \end{array} \tag{2.7.19}$$

which is easily seen to be equivalent to

$$\begin{array}{lll} \min_{x,u} & -d^T u & \\ \text{s.t.} & Ax = b & \\ & v_j = 1 & j = 1, \ldots, n \\ & (u_j; v_j; x_j) \in \mathcal{K}_{\exp} & j = 1, \ldots, n. \end{array} \tag{2.7.20}$$

The class of problems known as *geometric programs* can be formulated using the exponential cone. The derivation is slightly longer, but is quite straight forward and underlines the usefulness of the exponential cone.

### 2.7.4 Homogeneous and self-dual model

When presented with a convex conic optimization problem of the form (2.7.1), we know that if Slater's condition holds, that is, if there are interior primal and dual feasible points, then we can find a solution by searching for points that satisfy the KKT conditions:

$$\begin{aligned} Ax - b &= 0 \\ -A^T y - s + c &= 0 \\ x^T s &= 0 \\ x \in \mathcal{K},\ s \in \mathcal{K}^*,\ y \in \mathbb{R}^m. \end{aligned} \tag{2.7.21}$$

It is, however, not practical to be forced to verify that such interior feasible points indeed exist. At the same time, it is extremely important in virtually all real-world applications of optimization to be able to identify when a problem is infeasible — i.e. has no solution. Therefore, a good algorithm for solving optimization problems should for any problem at least (a) provide an approximate

(to some specified tolerance) optimal solution when it exists or (b) detect if the primal or dual problem is infeasible or unbounded and provide a certificate thereof.

A particularly elegant way to achieve these properties in an algorithm is via the so-called simplified *homogeneous self-dual embedding* [60]. The idea is to solve a slightly larger optimization problem which is known to have an optimal solution and which has the following properties: From the solution of this new problem, either an optimal solution to the original problem can be easily computed or a certificate of infeasibility can be easily computed — and the two can be distinguished.

In this section, this simplified homogeneous self-dual (HSD) model for convex conic optimization problems will be presented and its properties will be discussed.

The KKT conditions of the conic problem pair (2.7.6) are shown in (2.7.21). Instead of looking for points that satisfy these conditions, we introduce two extra non-negative scalar variables $\tau$ and $\kappa$ and seek to find $x, \tau, y, s, \kappa$ such that

$$\left.\begin{array}{lrrrrcl} \min & 0 & & & & & \\ \text{s.t.} & & Ax & -b\tau & & = & 0 \\ & -A^T y & & +c\tau & -s & = & 0 \\ & b^T y & -c^T x & & -\kappa & = & 0 \\ & \multicolumn{6}{l}{(x,\tau) \in \mathcal{K} \times \mathbb{R}_+, \quad (s,\kappa) \in \mathcal{K}^* \times \mathbb{R}_+, \quad y \in \mathbb{R}^m.} \end{array}\right\} \qquad \text{(HSD)}$$

Notice that the two first equations of (HSD) correspond to primal and dual feasibilty, cf (2.7.21), when normalizing the equations by $\tau$. In case a pair of primal and dual feasible points are found, we know that $c^T x - b^T y = x^T s$. Therefore the third equation of (HSD) measures how close the points are to satisfying complementarity slackness. When all three equations are satisfied along with the conic constraints, the problem (HSD) is solved, since the objective function is zero. Therefore, (HSD) is in fact a feasibility problem.

The main motivation for solving this slightly enlarged problem can be summarized in the following points:

Assume $(x, \tau, y, s, \kappa)$ solves HSD. Then

1. $(x, \tau, y, s, \kappa)$ is *complementary*. That is: $x^T s + \tau\kappa = 0$.
2. If $\tau > 0$ then $(x, y, s)/\tau$ is optimal for (2.7.6).
3. If $\kappa > 0$ then one or both of $b^T y > 0$ and $c^T x < 0$ hold. If the first holds, then (2.7.6) is primal-infeasible. If the second holds, then (2.7.6) is

dual-infeasible.

In order to justify these statements, let us first define

$$G := \begin{pmatrix} 0 & A & -b \\ -A^T & 0 & c \\ b^T & -c^T & 0 \end{pmatrix}$$

and notice that $G$ is skew-symmetric: $G = -G^T$. Now observe that we can write the equality constraints of (HSD) as

$$G(y, x, \tau)^T - (0, s, \kappa)^T = 0. \tag{2.7.22}$$

The three statements above are now readily proven by:

1. Pre-multiplying (2.7.22) by $(y, x, \tau)^T$ gives $x^T s + \tau\kappa = 0$.
2. $\tau > 0$ implies $\kappa = 0$ and hence $b^T(y/\tau) - c^T(x/\tau) = 0$ and therefore $x^T s = 0$. Dividing the two first linear equality constraints of (HSD) by $\tau$, we obtain the equality constraints of (2.7.21). Thus $(x, y, s)/\tau$ is optimal for (2.7.6).
3. If $\kappa > 0$ then $\tau = 0$ so $Ax = 0$ and $A^T y + s = 0$. Further $c^T x - b^T y = -\kappa < 0$ so not both $c^T x$ and $-b^T y$ can be non-negative. Assume $-b^T x < 0$. If (2.7.6) is primal-feasible then there exists $\hat{x} \in \mathcal{K}$ such that $A\hat{x} = b$. But then $0 > -b^T y = -\hat{x}^T A^T y = \hat{x}^T s \geq 0$, a contradiction. We can argue similarly if $c^T x < 0$.

These points mean that any solution to (HSD) with $\tau + \kappa > 0$ provides either an optimal solution to our original problems (2.7.6) or a certificate of infeasibility of (one of) the original problems (see [30] for further details).

The advantages of solving the homogeneous and self-dual model therefore include

- It solves the original primal-dual problem pair (2.7.6) without assuming anything concerning the existence of optimal or feasible solutions.
- The dimension of the problem is not essentially larger than that of the original primal-dual pair (2.7.6).
- If the original primal-dual pair (2.7.6) is infeasible, a certificate of this infeasibility is produced.

A further very useful feature of (HSD) is that it is a self-dual optimization problem, i.e. it is its own dual. This implies that we can apply a primal-dual interior-point algorithm to the problem (HSD) *without* doubling the dimension of the problem — i.e. there is no need to handle and store variables from the dual of (HSD) since they are identical to those of the primal.

Chapter 3

# Linear Programming

Linear programming has, perhaps because of the speed and robustness of solution methods, grown into what may now be called a *technology*. Applications of linear programming range from classical production scheduling and logistics to the more modern network flow problems, financial optimization problems and determining the maximal utilization of assets in a smart electricity grid.

The previous sections introduced the concepts needed to understand certain theoretical aspects of interior-point methods, which, in different forms, will be the sole focus of the remaining part of this thesis. In this chapter, interior-point methods for linear programming are explained and reviewed. From a theoretical viewpoint, this problem class is attractive to study because it is sufficiently complex to capture the core ideas of interior-point methods yet sufficiently simple to not require too much tedious notation and technicalities.

With a starting point in the Karush-Kuhn-Tucker condition for linear programs, we begin our presentation by defining the central path and its most common neighborhoods. We then present the framework for interior-point methods for linear programs as a Newton-type method that tracks the central path. Next, we review some historically relevant interior-point methods: the feasible short-step algorithm, the Mizuno-Todd-Ye predictor-corrector algorithm, the feasible long-step and the infeasible long-step algorithm. Finally, we briefly address a computationally important aspect: how to efficiently solve the linear system of

equations arising in each iteration of an interior-point algorithm.

For a historical review of linear programming and further details about the theoretical aspects, the reader is referred to [21, 29, 58, 62]. Many interesting examples of linear programs can be found in [9, 12].

## 3.1 The central path

The aim is to solve the standard form linear programming problem

$$\begin{aligned} \min_x \quad & c^T x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0 \end{aligned} \tag{3.1.1}$$

where $x, c \in \mathbb{R}^n, A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$ and we assume that $m \leq n$ and that $\text{rank}(A) = m$. We have already seen that the KKT conditions for this problem are

$$\begin{aligned} Ax - b &= 0 \\ -A^T y - s + c &= 0 \\ x \circ s &= 0 \\ x \geq 0,\ s \geq 0,\ y &\in \mathbb{R}^m. \end{aligned} \tag{3.1.2}$$

where again $\circ$ denotes elementwise product of vectors. For now, we assume that the problem has solutions and that strong duality holds (see Section 2.6.2).

A core concept in interior-point methods is that of *the central path*. It is defined as the set of stricly feasible points that satisfy

$$\begin{aligned} Ax - b &= 0 \\ -A^T y - s + c &= 0 \\ x \circ s &= \tau e \\ x \geq 0,\ s &\geq 0. \end{aligned} \tag{3.1.3}$$

for $\tau > 0$ where $e$ denotes the vector of all ones. The KKT conditions require $x_i s_i = 0$ for every $i$ while a point on the central path satisfies $x_i s_i = \tau$ for some $\tau > 0$. Let $(x_\tau, y_\tau, s_\tau)$ denote the central path point that satisfies $x \circ s = \tau$. The entire central path if thus the set

$$\mathcal{CP} = \{(x_\tau, y_\tau, s_\tau) : \ \tau > 0\}.$$

The central path is useful because it guides us towards a solution of (3.1.1): As $\tau \to 0$, the points on the central path approach a point that satisfies the KKT conditions (3.1.2). Therefore, primal-dual interior-point methods take steps that loosely track the central path towards the solution. As we shall see, this tracking is realized by taking Newton steps.

### 3.1.1 Central path neighborhoods

First, let us define what we mean by "loosely tracking": It is too restrictive to require that the iterates generated by an algorithm be exactly on the central path. Instead, we require that they stay inside some neighborhood of the central path. We define the *function*

$$\mu(x, s) = x^T s/n \tag{3.1.4}$$

i.e. the average value of the pairwise products of elements $x$ and $s$. This quantity is called the *complementarity gap*. Let us also define the interior *feasible region* as the set

$$\mathcal{F} = \left\{(x, y, s) : \; Ax = b, \;\; -A^T y - s = c, \quad x, s > 0\right\}. \tag{3.1.5}$$

Let us finally define a *centrality function* whose magnitude (norm) is a measure of the proximity of a point to the central path:

$$\psi(x, s) = x \circ s - \mu(x, s)e \tag{3.1.6}$$

When $\psi = 0$, the iterate is *on* the central path, cf (3.1.3). We can then define the following neighborhoods of the central path:

$$\mathcal{N}_2(\eta) = \{(x, y, s) \in \mathcal{F} : \; \|\psi(x, s)\|_2 \quad \leq \eta\mu(x, s)\} \tag{3.1.7}$$

$$\mathcal{N}_{-\infty}(\eta) = \{(x, y, s) \in \mathcal{F} : \; \|\psi(x, s)\|_{-\infty} \leq \eta\mu(x, s)\} \tag{3.1.8}$$

where $\eta$ is a parameter in $(0, 1)$. Here, $\|z\|_{-\infty} = \|z^-\|_\infty$ with $z_i^- = \min\{0, z_i\}$.

It clearly holds that

$$\mathcal{CP} = \mathcal{N}_2(0) = \mathcal{N}_{-\infty}(0) \subset \mathcal{N}_2(\eta) \subset \mathcal{N}_{-\infty}(\eta) \subset \mathcal{F}$$

for any $\eta \in (0, 1)$. These neighborhoods are going to play a crucial role in defining and analyzing interior-point methods. They will define the limits of how far away from the central path we will allow iterates to go.

## 3.2 Path-following algorithms

Let us first make the assumption that we know a point in the feasible set $\mathcal{F}$ from which we can initialize the algorithm. That is, we have available $(x, y, s) \in \mathcal{F}$.

Most of the algorithms discussed in this section use as search direction the solution $(d_x, d_y, d_s)$ to the following system of linear equations:

$$\begin{aligned} Ad_x &= 0 \\ -A^T d_y - d_s &= 0 \\ s \circ d_x + x \circ d_s &= -x \circ s + \sigma\mu e. \end{aligned} \tag{3.2.1}$$

The solution to this system is the Newton direction for the equations (3.1.3) (see Section 2.4). Unlike using Newton's method for an *unconstrained* optimization problem, there is no guarantee that the full Newton step will stay inside the feasible region. The solution $d = (d_x, d_y, d_s)$ is dependent only on $x$, $s$ and $\sigma$, so we may write $d = d(x, s, \sigma)$.

Let us briefly study the search direction $d$. For simplicity, we will write $\mu = \mu(x, s)$ and define for an $\alpha \in [0, 1]$:

$$\begin{aligned} x^+ &= x + \alpha d_x \\ s^+ &= y + \alpha d_y \\ y^+ &= s + \alpha d_s \\ \mu^+ &= \mu(x^+, s^+) \\ \psi^+ &= \psi(x^+, s^+). \end{aligned}$$

We have $(x, y, s) \in \mathcal{F}$ by assumption, so it follows from the two first equations of (3.2.1) that also $(x^+, y^+, s^+) \in \mathcal{F}$. Further, using elementary linear algebra, we readily see that

$$\mu^+ = (1 - \alpha)\mu + \alpha\sigma\mu \tag{3.2.2}$$

$$\psi^+ = (1 - \alpha)\psi(x, s) + \alpha^2(d_x \circ d_s). \tag{3.2.3}$$

From (3.2.2), it follows that if $\alpha$ were fixed, the greatest reduction in the function $\mu(\cdot)$ is obtained for small $\sigma$ (idealy $\sigma = 0$). Since it is our overall goal to find $(x^\star, y^\star, s^\star) \in \mathcal{F}$ with $\mu(x^\star, s^\star) = 0$ (cf. (3.1.2)), this makes a small $\sigma$ attractive. From (3.2.3), we see how the centrality measure changes along the direction $d$. The next centrality measure is reduced by a factor $(1 - \alpha)$ relative to the previous centrality measure plus the term $\alpha^2(d_x \circ d_s)$.

The direction obtained by setting $\sigma = 0$, i.e. the direction $d(x, y, 0)$ is known as the *affine scaling* direction. It is the pure Newton step for the KKT conditions (3.1.2) and seen in that light, it is reasonable to consider it the direction that reduces $\mu(\cdot)$ the most (we say it improves *optimality* the fastest). On the other hand, the Newton step for the central path equations (3.1.3) would be $d(x, y, 1)$. Since this direction aims at a central path point, we expect it reduces $\|\psi\|$ the fastest, i.e. it improves *centrality* the most. Therefore, it is known as the *centering* direction.

**Algorithm 1** Short-step PDIPM for LP

**Input:** Data $A$, $b$ and $c$, Parameters $\eta$ and $\sigma$, initial point $(x^0, y^0, s^0) \in \mathcal{N}_2(\eta)$.
**For** $k = 0, 1, 2, \ldots$
$(x^{k+1}, y^{k+1}, s^{k+1}) = (x^k, y^k, s^k) + d(x^k, s^k, \sigma)$
**End**

### 3.2.1 Feasible short-step algorithm

Perhaps the simplest algorithm within this framework is the so-called *short-step* primal-dual interior-point algorithm (PDIPM). The algorithm takes only relatively short steps in the search direction defined in (3.2.1). It is shown in Algorithm 1.

Notice that in each iteration, the *full* Newton step is taken. By a sufficiently conservative choice of the parameters $\eta$ and $\sigma$, it is possible to ensure that the next iterate is strictly feasible and still contained in $\mathcal{N}_2(\eta)$. For example, the choices $\eta = 0.4$ and $\sigma = 1 - 0.4/\sqrt{n}$ are sufficient [58]. This would ensure that $d_x$ and $d_s$ stay small enough that the iterates never leave the neighborhood $\mathcal{N}_2(\eta)$, cf. (3.2.3).

The short step PDIPM has the following theoretical properties (see [58] for proofs):

1. All iterates are in $\mathcal{F}$.
2. All iterates stay within the narrow neighborhood $\mathcal{N}_2(\eta)$.
3. The complementarity gap is reduced according to

$$\mu(x^{k+1}, s^{k+1}) \leq \sigma \mu(x^k, s^k). \tag{3.2.4}$$

The latter property (3.2.4) means that the complementarity gap $\mu(\cdot)$ is reduced geometrically with a rate $(1 - \Omega(1/\sqrt{n}))$ which implies that the algorithm reaches an iterate $(x^\star, y^\star, s^\star) \in \mathcal{F}$ with $\mu(x^\star, s^\star) \leq \epsilon \mu(x^0, s^0)$ in

$$\mathcal{O}\big(\sqrt{n} \log(1/\epsilon)\big) \tag{3.2.5}$$

iterations. Here, we used the "big-O"-style notation meaning

$$w = \mathcal{O}(f(n)) \quad :\Leftrightarrow \quad w \leq C_1 f(n), \quad \text{for } n \to \infty \tag{3.2.6}$$

$$w = \Omega(f(n)) \quad :\Leftrightarrow \quad w \geq C_2 f(n), \quad \text{for } n \to \infty \tag{3.2.7}$$

for some (possibly large) positive constant $C_1$ and some (possibly small) positive constant $C_2$, both independent of $n$.

**Algorithm 2** MTY predictor-corrector PDIPM for LP

**Input:** Data $A$, $b$ and $c$, initial point $(x^0, y^0, s^0) \in \mathcal{N}_2(1/4)$.
**For** $k = 0, 1, 2, \ldots$
Compute maximal $\alpha_k$ so that
$(x', y', s') := (x^k, y^k, s^k) + \alpha_k d(x^k, s^k, 0) \in \mathcal{N}_2(1/2)$
$(x^{k+1}, y^{k+1}, s^{k+1}) = (x', y', s') + d(x', s', 1)$
**End**

### 3.2.2 Feasible Mizuno-Todd-Ye predictor-corrector algorithm

In the short-step PDIPM (Algorithm 1), the search direction is $d(x, y, \sigma)$ with $\sigma \in (0, 1)$. Since $\sigma$ is neither 0 nor 1, we can consider the search direction a combination of the affine scaling direction $d(x, s, 0)$ and the centering direction $d(x, s, 1)$. In order for the theoretical results outlined in the previous section to hold, we must, however, choose $\sigma$ quite close to 1, which in practice makes progress towards the optimal solution quite slow.

One remedy to this situation was suggested by Mizuno, Todd and Ye in [35]. Instead, the algorithm alternates between 1. taking the maximal step possible in the affine scaling direction $d(x, s, 0)$ while remaining in a pre-defined central path-neighborhood and 2. taking a full step in the centering direction $d(x, s, 1)$. The details are shown in Algorithm 2.

The Mizuno-Todd-Ye (MTY) predictor-corrector algorithm has the following theoretical properties (see [62] for proofs):

1. The step lengths satisfy $\alpha_k = \Omega(1 - 1/\sqrt{n})$ uniformly for all $k$.
2. $(x^k, y^k, s^k) \in \mathcal{N}_2(1/4)$ for all $k$.
3. The complementarity gap is reduced according to

$$\mu(x^{k+1}, s^{k+1}) \leq (1 - \alpha_k)\mu(x^k, s^k). \tag{3.2.8}$$

Because of the first and third property, we immediately see

$$\mu(x^{k+1}, s^{k+1}) = (1 - \Omega(1/\sqrt{n}))\mu(x^k, s^k).$$

and therefore that the algorithm reaches an iterate $(x^\star, y^\star, s^\star) \in \mathcal{F}$ with $\mu(x^\star, s^\star) \leq \epsilon\mu(x^0, s^0)$ in

$$\mathcal{O}\big(\sqrt{n}\log(1/\epsilon)\big) \tag{3.2.9}$$

**Algorithm 3** Long-step PDIPM for LP

**Input:** Data $A$, $b$ and $c$, Parameters $\eta \in (0,1)$ and $\sigma_1$, $\sigma_2$ so that $0 < \sigma_1 < \sigma_2 < 1$ and initial point $(x^0, y^0, s^0) \in \mathcal{N}_{-\infty}(\eta)$.
**For** $k = 0, 1, 2, \ldots$
Choose $\sigma_k \in [\sigma_1, \sigma_2]$
Compute maximal $\alpha_k$ so that
$(x^{k+1}, y^{k+1}, s^{k+1}) := (x^k, y^k, s^k) + \alpha_k d(x^k, s^k, \sigma_k) \quad \in \mathcal{N}_{-\infty}(\eta)$
**End**

iterations. This complexity result matches the one found for the short-step PDIPM in the previous section, compare (3.2.9) and (3.2.5). As is often the case for numerical algorithms, the above estimates are *worst-case* analysis and it is therefore equally important to investigate the actual performance of the algorithms when applied to real problems.

From a practical viewpoint, the MTY predictor-corrector may look less efficient than the short-step method: The former requires *two* solutions of the system (3.2.1) in each iteration while the latter requires only one. Therefore each iteration is twice as computationally expensive. However, practical evidence shows that indeed the MTY predictor-corrector method is overall faster. In the average case, the step in the affine scaling direction brings so much progress towards optimality (reduction in $\mu(\cdot)$) that it offsets the extra costs required to bring the iterate back close to the central path, specifically inside the narrow neighborhood $\mathcal{N}_2(1/4)$. So, although one iteration is twice as expensive, the number of iterations is usually less than half that of the short-step PDIPM making the MTY predictor-corrector overall superior in the average case.

### 3.2.3 Feasible long-step algorithm

Let us finally discuss the *long-step* PDIPM. This method is sometimes also called the *wide-neighborhood* PDIPM [62]. In each iteration, the longest step possible in the direction $d(x, s, \sigma)$ is taken, subject to the next iterate staying inside $\mathcal{N}_{-\infty}(\eta)$. The details are shown in Algorithm 3.

The long-step PDIPM has the following theoretical properties (see [62] for proofs):

1. The complementarity gap is reduced according to
$$\mu(x^{k+1}, s^{k+1}) \leq (1 - \alpha_k(1 - \sigma_k))\mu(x^k, s^k). \tag{3.2.10}$$
2. It holds that $(1 - \alpha_k(1 - \sigma_k))$ is asymptotically $\Omega(1 - 1/n)$ where the constant depends on $\sigma_1$, $\sigma_2$ and $\eta$ but not $n$.

These two points allow the conclusion that the algorithm reaches an iterate $(x^\star, y^\star, s^\star) \in \mathcal{F}$ with $\mu(x^\star, s^\star) \leq \epsilon\mu(x^0, s^0)$ in

$$\mathcal{O}(n \log(1/\epsilon)) \tag{3.2.11}$$

iterations. Notice that this complexity result is theoretically worse than the complexity estimates for the short-step and for the MTY predictor-corrector algorithm, cf (3.2.5) and (3.2.9). The estimate is worse by a factor $\sqrt{n}$.

In practice, however, the long-step method almost always outperforms both of the two methods presented in the previous sections. Similar to the short-step PDIPM, the long-step method requires only one solution of (3.2.1) per iteration. At the same time, it allows for the longest possible step subject to staying in $\mathcal{N}_{-\infty}(\eta)$, which is similar to the first step of the MTY predictor-corrector PDIPM (except that the latter uses a more narrow neighborhood).

The long-step algorithm allows a certain degree of adaptivity: The choice of $\sigma_k$ in each iteration can be done using various strategies and different strategies lead to different variants of the algorithm. For example, $\sigma_k$ can be chosen close to $\sigma_2$ if the current iterate, according to some measure, is deemed not sufficiently close to the central path. Otherwise, $\sigma_k$ is chosen close to $\sigma_1$ thus allowing a faster improvement in terms of optimality since $\sigma_1$ is closer to 0.

### 3.2.4 Infeasible long-step algorithm

In Sections 3.2.1 through 3.2.3 we have assumed that an initial point

$$(x^0, y^0, s^0) \in \mathcal{F}$$

was available to us. If such a point is not known, one can be found using a so-called phase I algorithm. This algorithm works by applying a PDIPM to a modified problem for which an initial feasible point is trivially identified. Therefore, it is equally expensive to find a feasible starting point as solving the original instance, starting from this point. Essentially, this means that unless a point in $\mathcal{F}$ is known beforehand, the PDIPM must be run twice in order to solve a problem from scratch. This is clearly not very satisfactory.

This issue is addressed via the so-called *infeasible-start* PDIPMs. These algorithms allow starting from a point not necessarily feasible with respect to the linear equality constraints of (3.1.2). Where the algorithms of the previous sections always maintain a feasible point and attempt to reduce $\mu(\cdot)$ in order to progress towards optimality, the infeasible PDIPM simultaneously improve feasibility and optimality, as we now outline.

**Algorithm 4** Infeasible long-step PDIPM for LP

**Input:** Data $A$, $b$ and $c$, Parameters $\eta \in (0,1)$ and $\sigma_1$, $\sigma_2$ so that $0 < \sigma_1 < \sigma_2 < 1$ and initial point $(x^0, y^0, s^0) \in \mathcal{N}(\eta)$.
**For** $k = 0, 1, 2, \ldots$
Solve (3.2.12) for search direction $(d_x, d_y, d_s)$ with $\sigma = \sigma_k \in [\sigma_1, \sigma_2]$.
Compute maximal $\alpha_k$ so that
$(x^{k+1}, y^{k+1}, s^{k+1}) := (x^k, y^k, s^k) + \alpha_k(d_x, d_y, d_s) \quad \in \mathcal{N}(\eta)$
**End**

If the current iterate is $(x, y, s)$, the search direction in an infeasible PDIPM is the solution $(d_x, d_y, d_s)$ to the system

$$\begin{aligned} Ad_x &= -(Ax - b) \\ -A^T d_y - d_s &= -(-A^T y - s + c) \\ s \circ d_x + x \circ d_s &= -x \circ s + \sigma\mu e. \end{aligned} \tag{3.2.12}$$

Consider the first equation $Ad_x = -(Ax - b)$. Its solution $d_x$ is the Newton step towards solving the equation $Ax = b$. Notice that if the iterate is feasible, the equation becomes $Ad_x = 0$, i.e. it reduces to the corresponding equation used in the feasible PDIPM, cf. (3.2.1). The same observation applies to the second equation. The third equation is unchanged as compared to (3.2.1).

The analysis of infeasible PDIPM is somewhat more complicated than that of a feasible PDIPM. This is primarily because we no longer have $d_x^T d_s = 0$ which simplifies many derivations in the proofs.

An outline of the infeasible PDIPM is seen in Algorithm 4. The neighborhood $\mathcal{N}(\eta)$ used in Algorithm 4 is a slight generalization of $\mathcal{N}_{-\infty}(\eta)$. To avoid unnecessary complication, we can, for now, think of $\mathcal{N}(\eta)$ as equivalent to $\mathcal{N}_{-\infty}(\eta)$ but allowing also points that are not feasible w.r.t. to the linear equality constraints of (3.1.2).

The algorithm has the following theoretical properties (see [58] for proofs).

1. The linear residuals are reduced according to

$$\begin{aligned} Ax^{k+1} - b &= (1 - \alpha_k)(Ax^k - b) \\ -A^T y^{k+1} - s^{k+1} + c &= (1 - \alpha_k)(-A^T y^k - s^k + c) \end{aligned} \tag{3.2.13}$$

2. The step lengths $\alpha_k$ can be chosen so that

$$\mu(x^{k+1}, s^{k+1}) \leq (1 - \alpha_k(1 - \sigma_k))\mu(x^k, s^k)$$

3. There exists a uniform lower bound on $\alpha_k$ so that
$$\alpha_k = \Omega(1 - 1/n^2).$$

These points imply that the algorithm reaches an iterate $(x^\star, y^\star, s^\star)$ with
$$\begin{aligned} Ax^\star - b &\leq \epsilon(Ax^0 - b) \\ -A^T y^\star - s^\star + c &\leq \epsilon(-A^T y^0 - s^0 + c) \\ \mu(x^\star, s^\star) &\leq \epsilon\mu(x^0, s^0) \end{aligned}$$
in no more than
$$\mathcal{O}\big(n^2 \log(1/\epsilon)\big) \tag{3.2.14}$$
iterations. This complexity result is even worse than the one obtained for the feasible long-step algorithm, compare (3.2.14) and (3.2.11). However, in practice, we usually observe that the infeasible long-step PDIPM outperforms all previously discussed PDIPMs. In terms of number of iterations, it is often comparable to the feasible long-step PDIPM, but the latter requires a feasible initial point. Therefore, the infeasible PDIPM is usually preferable.

The discrepancy often observed between the theoretical worst-case behavior and the practically observed behavior for PDIPMs in part motivates one of the main contributions of this thesis. As will be described in much more detail in Section 4.5, our implementation of an interior-point algorithm for nonsymmetric conic optimization problem is aimed at obtaining the best possible *practical* performance. To obtain this, we take inspiration from the techniques that seem to improve performance for the infeasible long-step PDIPM just described, as compared to the feasible long-step PDIPM.

## 3.3 Solving for the search direction

The main computational task in PDIPMs is to solve the system defining the search direction. These systems have the general form (cf. (3.2.1) and (3.2.12))
$$\begin{pmatrix} A & 0 & 0 \\ 0 & -A^T & -I \\ S & 0 & X \end{pmatrix} \begin{pmatrix} d_x \\ d_y \\ d_s \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix} \tag{3.3.1}$$
where $X = \operatorname{diag}(x_1, x_2, \ldots, x_n)$, $S = \operatorname{diag}(s_1, s_2, \ldots, s_n)$ and $r_1, r_2, r_3$ are arbitrary right-hand sides. Eliminating $d_s$ from this system of equations, we get
$$\begin{pmatrix} H & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} d_x \\ v \end{pmatrix} = \begin{pmatrix} r_4 \\ r_1 \end{pmatrix} \tag{3.3.2}$$

where we defined $H = S^{-1}X$, $v = -d_y$ and $r_4 = r_2 + X^{-1}r_3$. Notice that any multiplication with either $X$, $S$ or their inverses is computationally cheap since they are diagonal matrices. The existence of the matrices $X^{-1}$ and $S^{-1}$ is ensured by $x$ and $s$ being *strictly* feasible points, i.e. $x > 0$ and $s > 0$.

There are several ways to solve the KKT system (3.3.2). Because all elements of the diagonal matrix $H$ are strictly positive and since we are assuming that $A$ has full row rank, the system matrix in (3.3.2) is nonsingular. Therefore, the system has a unique solution for any right-hand side. Probably the most often used general method is to reduce the system to the so-called *normal equation* system: Eleminate $d_x$ to obtain

$$(AH^{-1}A^T)v = r \tag{3.3.3}$$

where $r = r_1 - AH^{-1}r_4$. The matrix $AH^{-1}A^T$ is symmetric and positive definite, so it has a Cholesky decomposition: $AH^{-1}A^T = R^TR$ where $R$ is an upper triangular matrix.

Since the sparsity pattern of $AH^{-1}A^T$ is constant regardless of the stage of the algorithm, a pivoting order for the Cholesky factorization aimed at minimizing the number of non-zeros in $R$ can be chosen at initialization time of the PDIPM. It is in general a difficult problem to determine the optimal pivoting order, but several efficient heuristics exist [20].

Once the Cholesky factor has been computed, which requires $\mathcal{O}((1/3)n^3)$ flops for a dense matrix, recovering $v$ from (3.3.3) requires two back-substitutions:

$$v = R^{-1}R^{-T}r.$$

Each of these operations requires $\mathcal{O}(n^2)$ flops and thus are negligeble compared to the factorization. Once $v$ is known, $d_y$, $d_x$ and $d_s$ are readily recovered by simply substituting back into (3.3.2) and (3.3.1).

There exist a number of other ways to speed up the solution, particularly when a certain exploitable structure in $A$ is present. A particularly important example is when $A$ has a single or a few dense columns, but is otherwise sparse. In this case, these dense columns cause the entire matrix $AH^{-1}A^T$ to be dense, since it essentially consists of sums of multiples of the outer products of the columns of $A$. This, in turn, causes the Cholesky factor $R$ to be completely dense, significantly slowing down the solution process. Fortunately, dense columns can be handled separately through the so-called modified Schur complement method. This corresponds to viewing $AH^{-1}A^T$ as a sum of a dense low-rank matrix (the outer product of the dense columns) and the remaining full-rank but sparse matrix. Using this method, the sparsity in $AH^{-1}A^T$ is not completely destroyed even if $A$ contains several dense columns. See [4] and [54] for further details.

Chapter 4

# Convex Conic Programming

Interior-point methods were first developed for linear programming problems. In their early existence, interior-point methods for linear programming certainly offered a theoretically attractive alternative to the already well-established simplex method: the theoretical complexity of interior-point methods is superior to that of the simplex algorithm, which is exponential in worst-case running time. However, from a practical viewpoint it was not, and still is not, established which algorithm will perform better on a given problem — each of the two types of algorithm offers advantages and disadvantages.

A very compelling argument in favor of interior-point algorithms came with the generalization to convex conic programming — which is not directly possible for the simplex algorithm. Albeit important differences exist, it turns out that most of the core concepts in interior-point algorithms for linear programming are very elegantly generalizable to the much broader class of problems known as convex conic programs. This family of problems encompasses so many different-natured optimization problems that likely only a small fraction of them are in application today. A much deeper penetration into industry is expected in the future.

In this chapter, we first develop the framework for interior-point methods for convex conic programs by first describing the central path of a linear program as a family of barrier problems. This is the natural starting point for the gen-

eralization to convex conic programs, which we turn to afterwards. Next, we treat the somewhat intricate issue of symmetrically linearizing the complementarity conditions when the cone in question is symmetric. We then describe the difficulties that arise when the cone is no longer symmetric. This finally brings us to the main contribution of this thesis. After outlining previous work in the area of nonsymmetric conic optimization, we briefly present the contents of a scientific paper by Skajaa and Ye [51] submitted[1] to the journal *Mathematical Programming* in 2012 which addresses these problems — the paper may be found in its full length in the appendix.

Currently, examples of applications of convex conic optimization include truss-topology optimization, different eigenvalue optimization problems, second-order cone problems in financial optimization and robust optimization. Also classical quadratic programming can be seen as a convex conic optimization problem. All of these examples involve symmetric cones. A still relatively unexplored area is those problems involving nonsymmetric cones, although some applications are known: entropy optimization problems, geometric programs and applications from financial optimization. Further applications are expected by this author to surface over the coming years as solution methods become even more accurate and robust.

The generalization of interior-point algorithms from linear programming to convex conic optimization problems has had many contributers. The theoretical foundation includes, amongst many others, the following references: [7, 40, 41, 42, 43, 49]. Interesting examples of applications of convex conic programs can be found in [10, 12, 19].

## 4.1 A family of barrier problems

In order to better understand how PDIPMs are generalized to this more general problem class, it is fruitful to first derive the key concept from the point of view of barrier functions (see Section 2.5.1).

We can reduce the standard form LP

$$\begin{array}{rl} \min_x & c^T x \\ \text{s.t.} & Ax = b \\ & x \geq 0 \end{array} \tag{4.1.1}$$

[1] The paper is currently under review.

to a family of unconstrained *barrier problems*

$$\begin{array}{rl} \min_x & c^T x + \tau F(x) \\ \text{s.t.} & Ax = b \end{array} \tag{4.1.2}$$

where

$$F(x) = -\sum_{j=1}^{n} \log(x_j) \tag{4.1.3}$$

and $\tau > 0$ is a parameter indexing the family of problems. When $\tau$ goes to 0, the solution of (4.1.2) goes to a solution of (4.1.1). As we saw in Section 2.5.1, the function in (4.1.3) is a logarithmically homogeneous self-concordant barrier function (a LHSCB-function) for the cone $\mathbb{R}^n_+$. The KKT conditions for (4.1.2) are

$$\begin{aligned} Ax &= b \\ c + \tau \nabla F(x) - A^T y &= 0. \end{aligned} \tag{4.1.4}$$

By introducing the variable $s = -\tau \nabla F(x)$, we can rewrite (4.1.4) into

$$\begin{aligned} Ax - b &= 0 \\ -A^T y - s + c &= 0 \\ s + \tau \nabla F(x) &= 0. \end{aligned} \tag{4.1.5}$$

Notice that $x$ is implicitely assumed to contain only positive elements, as it is in the domain of $F$ (cf. (4.1.3)).

When $F$ is specifically given by (4.1.3), we have $\nabla F(x) = -X^{-1}e$ and $\nabla^2 F(x) = X^{-2}$ where $X$ is the diagonal matrix with $x$ on the diagonal. Since $s = -\tau \nabla F = \tau X^{-1} e$, also $s$ contains only positive elements. So the third equation of (4.1.5) is

$$\begin{aligned} s - \tau X^{-1} e &= 0 \\ x > 0,\ s &> 0. \end{aligned}$$

By multiplying this equation by $\nabla^2 F(x)^{-1/2} = X$, we get $x \circ s = \tau e$, the familiar relaxed complementarity conditions for LP. We have thus arrived at the same equations by which we defined the central path, (3.1.3) (except for last inequalities which in fact can be relaxed to non-strict inequalities as the third equation anyway implies they are strict). Therefore, the primal point $x_\tau$ on the central path is the solution to the barrier problem (4.1.2), and the corresponding dual point is $s_\tau = -\tau \nabla F(x_\tau)$.

The above derivations lead to the generalization to convex conic cones. The cone $\mathbb{R}^n_+$ is replaced by the general cone $\mathcal{K}$ so that the problem under consideration

now is

$$\begin{array}{rl} \min_x & c^T x \\ \text{s.t.} & Ax = b \\ & x \in \mathcal{K}. \end{array} \tag{4.1.6}$$

The family of barrier problems is then still (compare with (4.1.2))

$$\begin{array}{rl} \min_x & c^T x + \tau F(x) \\ \text{s.t.} & Ax = b \end{array} \tag{4.1.7}$$

indexed by $\tau > 0$, where $F$ is now a LHSCB-function for the cone $\mathcal{K}$. Similarly to before, we define the central path for (4.1.6) as the points that satisfy the KKT conditions for (4.1.7) for some $\tau > 0$:

$$\begin{array}{r} Ax - b = 0 \\ -A^T y - s + c = 0 \\ s + \tau \nabla F(x) = 0 \end{array} \tag{4.1.8}$$

where, for the same reason as described for LP, we implicitely assume $x \in \mathcal{K}$ and $s \in \mathcal{K}^*$. The overall idea of a PDIPM to solve (4.1.6) is then again to track the central path (4.1.8) towards a point that is optimal for (4.1.6).

The success of the method depends crucially on at least two issues: 1. The properties (particularly the barrier parameter) of the barrier function $F$ for $\mathcal{K}$ and 2. how the third equation of the KKT conditions (4.1.8) is linearized to define the search direction. This latter issue will be the topic of the following sections.

## 4.2 Linearization of the complementarity conditions

Recall that the conic problem pair is given by:

$$\text{Primal} \left\{ \begin{array}{rl} \min_x & c^T x \\ \text{s.t.} & Ax = b \\ & x \in \mathcal{K} \end{array} \right. \qquad \text{Dual} \left\{ \begin{array}{rl} \max_{y,s} & b^T y \\ \text{s.t.} & A^T y + s = c \\ & s \in \mathcal{K}^*,\ y \in \mathbb{R}^m. \end{array} \right. \tag{4.2.1}$$

The derivation in the previous section led to the KKT conditions in (4.1.8). Had we performed the same derivation starting instead from the dual problem with a LHSCB-function $G$ for $\mathcal{K}^*$, we would arrive at the conditions on the right in

(4.2.2) below. For comparison we also repeat those from the primal:

$$\text{Primal}\begin{cases} Ax - b = 0 \\ -A^T y - s + c = 0 \\ s + \tau\nabla F(x) = 0 \\ x \in \mathcal{K},\ s \in \mathcal{K}^*. \end{cases} \qquad \text{Dual}\begin{cases} Ax - b = 0 \\ -A^T y - s + c = 0 \\ x + \tau\nabla G(s) = 0 \\ x \in \mathcal{K},\ s \in \mathcal{K}^* \end{cases} \tag{4.2.2}$$

Clearly the two systems in (4.2.2) differ only in the last equation. That is

$$\begin{aligned} \text{Primal:} \quad & s + \tau\nabla F(x) = 0 \\ \text{Dual:} \quad & x + \tau\nabla G(s) = 0. \end{aligned} \tag{4.2.3}$$

If we do not assume any relation between $F$ and $G$ the two sets of equations in (4.2.2) do not generally define the same set of points — i.e. the same central path. This property, however, can be achieved if we in place of $G$ use the conjugate barrier $F^*$ which is a barrier for $\mathcal{K}^*$ (see Section 2.5.2). The two equations in (4.2.3) then become

$$\begin{aligned} \text{Primal:} \quad & s + \tau\nabla F(x) \ = 0 \\ \text{Dual:} \quad & x + \tau\nabla F^*(s) = 0. \end{aligned} \tag{4.2.4}$$

These equations now define the same set of points. This is quickly seen using the relations (2.5.9)–(2.5.14): Applying the function $\nabla F^*$ to both sides of $-\nabla F(x) = \tau^{-1}s$, we get

$$\nabla F^*\left(-\nabla F(x)\right) = \nabla F^*\left(\tau^{-1}s\right)$$

which, by (2.5.12) and (2.5.4) is the same as $x+\tau\nabla F^*(s) = 0$. Therefore the two sets of equations indeed define the same points. Notice that requiring $G = F^*$ is a strong requirement since computing $F^*$ and its derivates is not necessarily simple, cf. the definition (2.5.8) and Section 2.5.2.

The primal and dual problems in (4.2.1) have the same general form and are completely symmetric in duality (see Section 2.7.1). Thus there is no reason in particular that one should be called "the primal" and the other "the dual". They are best understood as a pair of problems which are dual to each other, each containing information about the other. For this reason, it is generally undesireable to let an algorithm that solves them both simultaneously, such as a PDIPM, place emphasis on one of the problems in some way. In other words, an algorithm should be consistent w.r.t. duality so that it would not be forced to, for each problem instance, to determine which of the two problems to emphasize. It should therefore produce exactly the same iterates if we interchange the roles of the two problems.

The iterates of a PDIPM depend on the search directions and the step lengths. The search directions arise from the linearization of the KKT conditions. Linearization here refers to computing the Newton step of a set of equations.

Although the two equations (4.2.4) define the same points, this is unfortunately no guarantee that their linearizations result in equations defining the same search direction. This is a consequence of the fact that linearization of a set of equations depends not on the solution set but the particular algebraic representation of the points.

As an illustrative example, consider the problem of finding $x > 0$ such that $f(x) = x^2 - 4 = 0$. Clearly the solution is $x^\star = 2$. The Newton step is determined by $f'(x)d_x = -f(x)$. If the current iterate is $x = 1$, we find $d_x = 3/2$. On the other hand, if the problem is to find $x > 0$ such that $f(x) = x - 2 = 0$, we would in the same way find $d_x = 1$ although the two equations have the same solution set.

Reiterating this point for the equations occuring in interior-point algorithms: The relaxed complementarity conditions for LP are $x \circ s = \tau e$ but the same conditions derived from the barrier formulation are $s - \tau X^{-1} e = 0$. The multiplication of the latter with $X$ to obtain the first changes the linearization. In [15] it was shown that indeed the two search directions obtained from linearizing these two equations are always different in all but degenerate situations.

Therefore, a problem now arises: If the algorithm for the general conic programming should be consistent w.r.t. duality, which of the two sets of equations in (4.2.2) should be linearized? Considering again the case of LP: We see that the operation

$$s - \tau X^{-1} e = 0 \quad \overset{\text{mult. } X}{\Rightarrow} \quad x \circ s = \tau e \tag{4.2.5}$$

*symmetrizes* the complementarity conditions: The equation on the right in (4.2.5) is clearly symmetric in $x$ and $s$. I.e. interchanging the primal and dual problems would lead to $x$ and $s$ in (4.2.5) being interchanged which clearly makes no difference for the equation on the right.

Whether a transformation similar to (4.2.5) exists for the conic problem depends on the fundamental geometric properties of the cone $\mathcal{K}$. It turns out than when the cone is self-dual and homogeneous (see Section 2.2.2), the search direction can be defined so that it is symmetric w.r.t. duality. When the cone lacks these properties, this is not the case. The following sections discuss these issues in more detail.

## 4.3 Symmetric cones

The property needed to ensure that a search direction symmetric w.r.t. duality can be defined is that the cone be *self-scaled*. This property was introduced in [41], developed further in [42] and was originally defined as a property of the barrier function:

**Definition 2.1 of [42].** *Assume $F$ is a* LHSCB*-function for $\mathcal{K}$ with barrier parameter $\nu$. Then $F$ is called self-scaled if the following two properties hold: For any $x, w \in \mathcal{K}$,*

$$\nabla^2 F(w)x \in \operatorname{int}(\mathcal{K}^*)$$
$$F^*(\nabla^2 F(w)x) = F(x) - 2F(w) - \nu$$

It is now common to say that a cone is self-scaled and one can take this as meaning that the cone in question admits a self-scaled barrier — i.e., the geometric structure of the cone is rich enough that a self-scaled barrier can be defined on it.

The defining properties look quite technical at first but one crucial property that they imply is that of the existence of a so-called scaling point for any primal-dual pair:

**Theorem 3.2 of [41].** *For each pair $x \in \mathcal{K}$ and $s \in \mathcal{K}^*$, there exists a unique point $w \in int(\mathcal{K})$ such that*

$$s = \nabla^2 F(w)x. \tag{4.3.1}$$

The point $w$ is sometimes called the *scaling point*. When this concept was first introduced in [41], the authors, Nesterov and Todd, were unaware of a fact proved by Güler around the same time [25]: The self-scaled cones coinside with the cones that are homogeneous and self-dual (see Section 2.2.2 for definitions) — i.e. those that are *symmetric*. As we mentioned in Section 2.2.2, any symmetric cone is a direct (Cartesian) product of the a combination of the five basic symmetric cones:

1. The Lorentz (second-order) cone $L_2^n$ (see definition in (2.2.4)).
2. The cone of positive semi-definite symmetric real matrices $S_+^n$ (see definition in (2.2.3))
3. The cone of positive semi-definite Hermitian complex matrices

4. The cone of positive semi-definite Hermitian quaternion matrices
5. A special 27-dimensional cone

These irreducible cones can therefore be thought of as a basis of all symmetric cones. For optimization, currently only the *two* first of the above cones are of practical interest. Notice that the positive orthant is a special case of the above: $\mathbb{R}^n_+$ is the direct product of $n$ copies of $S^1_+$.

In practice, it is relevant to know explicit expressions for the scaling point (4.3.1) for specific cones. As an example, let $\mathcal{K} = S^n_+$, then one can find

$$w = x^{1/2}\left(x^{1/2}sx^{1/2}\right)^{-1/2}x^{1/2} \tag{4.3.2}$$

where (4.3.1) and (4.3.2) should be understood in the sense of the isomorphism between $S^n_+$ and $\mathbb{R}^{n(n+1)}$ described on page 24. If $n = 1$, we clearly have $w = \sqrt{x/s}$ and therefore for $\mathcal{K} = \mathbb{R}^n_+$, we get $w = (\sqrt{x_1/s_1}, \ldots, \sqrt{x_n/s_n})$.

Let us now see how a search direction in a PDIPM can be defined symmetrically w.r.t. duality by making explicit use of the scaling point. In the general conic case, the only nonlinear equation of the KKT conditions (4.2.4) arising from the primal barrier problem is

$$s + \tau\nabla F(x) = 0. \tag{4.3.3}$$

Its linearization is

$$\tau\nabla^2 F(x)d_x + d_s = -\left(s + \tau\nabla F(x)\right). \tag{4.3.4}$$

Now let $w$ be the scaling point for $x$ and $s$. That is, $s = \nabla^2 F(w)x$. Then a symmetric search direction is obtained by replacing $\tau\nabla^2 F(x)$ in (4.3.4) by $\nabla^2 F(w)$:

$$\nabla^2 F(w)d_x + d_s = -\left(s + \tau\nabla F(x)\right). \tag{4.3.5}$$

To see that the direction (4.3.5) is indeed symmetric, left-multiply the equation by $\nabla^2 F(w)^{-1}$ to obtain

$$d_x + \nabla^2 F(w)^{-1}d_s = -\nabla^2 F(w)^{-1}\left(s + \tau\nabla F(x)\right). \tag{4.3.6}$$

Now notice that by taking the "dual-side scaling point" $v = -\nabla F(w) \in \operatorname{int}(\mathcal{K}^*)$, we have from (2.5.12) and (2.5.13) that

$$\nabla^2 F^*(v)s = x \tag{4.3.7}$$

$$\nabla^2 F^*(v) = \nabla^2 F(w)^{-1}. \tag{4.3.8}$$

We can therefore write (4.3.6) as

$$\nabla^2 F(v) d_s + d_x = -\left(x + \tau \nabla F^*(s)\right). \tag{4.3.9}$$

Comparing (4.3.9) with (4.3.5), it is clear that they are symmetric w.r.t. duality. The above derivation was carried through for the centering direction. If we set $\tau = 0$, the direction is the affine scaling direction, and the symmetry can be made particularly clear. The affine scaling direction is

$$\nabla^2 F(w) d_x + d_s = -s. \tag{4.3.10}$$

Let $B$ denote the unique square root of $\nabla^2 F(w)$ so that $B^2 = \nabla^2 F(w)$. Then (4.3.1) can be written $Bx = B^{-1}s$. Let us define $\lambda := Bx = B^{-1}s$. Then by left-multiplying by $B^{-1}$, we can write the equation (4.3.10) as

$$B d_x + B^{-1} d_s = -\lambda. \tag{4.3.11}$$

If the roles of $s$ and $x$ (and hence $d_x$ and $d_s$) were switched, $B$ would be inverted because of (4.3.8), and thus (4.3.11) would be left invariant. The symmetric picture is thus complete. The view of the primal and dual problems as mirror images in "problem space" is therefore maintained in "algorithm space" when the algorithm uses the general search direction (4.3.5), known as the Nesterov-Todd direction.

It is not simple vanity that makes this property desireable. It has been widely verified empirically that the practially most efficient algorithms within the field of interior-point methods are those that respect the symmetry w.r.t. duality. The Nesterov-Todd direction discussed above has proven very efficient for problems involving the Lorentz cone $L_2^n$, the positive semi-definite cone $S_+^n$ and combinations of them. See for example [2, 52, 54] for results substantiating these claims.

## 4.4 Nonsymmetric cones

One of the main contributions in this thesis is to develop an algorithm that can solve convex conic optimization problems of the form (4.1.6) when the cone $\mathcal{K}$ is *not* symmetric. When the cone no longer admits a self-scaled barrier, the existence of a unique scaling point $w$ for each primal and dual pair of points $x$ and $s$ is no longer guaranteed. This means that we can not use the approach outlined in Section 4.3 to obtain a search direction symmetric w.r.t. duality.

The main contribution in this direction is included as an original research paper in appendix A. In this section, we first describe two approaches previously employed to develop PDIPMs for solving such general nonsymmetric conic programs.

Here, it should be stressed that these methods deal with methods to solve *general* nonsymmetric conic programs. Other more specialized approaches have been studied with some success elsewhere, see e.g. [5, 61] and the introduction in the paper in appendix A.

Following the description of these two previous lines of study, we provide a brief overview of the general results obtained in our paper and otherwise refer the reader to the paper in the appendix.

### 4.4.1 Nesterov-Todd-Ye 1998

The method discussed in this section was published in [43] in 1998. Two main conceptual algorithms are developed in the paper. The second specifically requires that the problem defining cone $\mathcal{K}$ be self-scaled and is therefore of no interest to us.

The first method, however, is aimed at solving general, not necessarily self-scaled convex conic optimization problem. The method has two important characteristics:

1. It solves the self-dual and homogeneous (HSD) model, see Section 2.7.4.
2. It assumes availability of both the primal barrier function and its conjugate barrier (see Section 2.5.2), their gradients and Hessians.

The first of these characteristics implies all the advantageous properties of solving the HSD model. The second allows the construction of a search direction symmetric w.r.t. duality, which, as we described in the previous section, is a highly desirable property. However, this feature also implies at least two crucial disadvantages:

1. In many cases, the conjugate (dual) barrier, its gradient and Hessian are not available in closed form or are not easily computable.
2. The use of Hessians of both the primal and dual barrier in defining the search direction means that the linear systems that must be solved in each iteration double in size compared to the standard PDIPM for self-scaled cones.

The first of these two points means that the methods are restricted to be applicable only to problems involving cones for which the conjugate barrier and

its derivatives are easily computable. This, however, is not the case even for simple examples. As an example, consider the three dimensional power cone (see definition in (2.7.15)). The function

$$F_\alpha(x) = -\log\left(x_2^{2\alpha} x_3^{2(1-\alpha)} - x_1^2\right) - \log x_2 - \log x_3 \tag{4.4.1}$$

was shown by Nesterov in [39] to be a LHSCB-function with parameter 4 for the three-dimensional power cone. However, its conjugate function can not be written in closed form. Although a method was derived in [39] to compute the *value* of the conjugate barrier for this particular cone, still the gradient and Hessian are unavailable. So indeed the assumption of availability of both primal and conjugate barriers and their derivates is quite restrictive.

The second disadvantage mentioned above is more straight-forward: The definition of the search direction involves Hessians of both the primal and conjugate barrier. They appear in the definition in such a way that the system of linear equations needed to be solved is twice the size of that possible in a PDIPM for self-scaled cones. This makes each iteration significantly slower as the solution of the linear system is the dominating computational work in each iteration. So even though the iteration bound obtained in [43] matches the best-known bounds for PDIPMs, namely $\mathcal{O}(\sqrt{n}\log(1/\epsilon))$, each iteration is slower and in this respect, the algorithms therefore compare infavorably.

### 4.4.2 Nesterov 2006

The method discussed in this section was published in [39] in 2006. In this paper, some of the disadvantages of the algorithms mentioned in the previous section are remedied.

First and foremost, the primal barrier, its gradient and Hessian and just the *value* of the conjugate (dual) barrier are required. I.e. the gradient and Hessian of the conjugate barrier are not used. Secondly the search direction is defined so that the linear system that must be solved has the same size as those occuring in a PDIPM for self-scaled problems. This latter property greatly increases the practical performance of the algorithm.

Drawbacks of the algorithm include that it allows only starting from a strictly feasible starting point. Indeed, it proposes a phase-I method (see Section 5.1) to produce such a starting point. As we discussed in Section 5.1, this is not a very desirable property from a practical viewpoint as one must solve *two* equally sized problems in order to solve the original problem. In spite of the inferior worst-case complexity estimate of infeasible-start IPMs (see Section 3.2.4), their

real-life popularity and applications far exceed that of feasible-start algorithms. Since a similar discrepancy between theory and practice might be present for the case of nonsymmetric cones, it may leave room for improvement to solely focus on a feasible-start algorithm. It should also be mentioned that the analysis of the algorithm is somewhat simpler when only feasible starting points are considered. This is a significant advantage from a theoretical viewpoint.

## 4.5 Skajaa-Ye 2012: A Homogeneous Interior-Point Algorithm for Nonsymmetric Convex Conic Optimization

Our contribution to the area of interior-point methods for nonsymmetric convex conic optimization is included as an original research paper in appendix A.

### 4.5.1 Overview

The main objective of the paper is to introduce and analyze a novel interior-point algorithm able to solve convex conic optimization problems involving nonsymmetric cones.

The contribution to the field is three-fold:

1. The new algorithm introduced remedies some of the drawbacks of the algorithms briefly outlined in the two previous sections. Specifically, it uses exclusively the primal barrier function, its gradient and its Hessian. Thus the dual barrier is not needed in any form which is significant practical improvement compared to previous methods. A proof of convergence and worst-case complexity is included (see outline further below).

2. An extensive series of numerical experiments with the algorithm is carried out. Comparisons are made with previous less general algorithms. Thus the paper adds to the extremely limited body of knowledge concerning practical performance of interior-point methods for nonsymmetric cones. The numerical results are overall positive, particularly when considering the very general nature of this algorithm.

3. The algorithm is designed with high practical performance in mind. Firstly, the algorithm solves the homogeneous model (see Section 2.7.4) which im-

plies the ability to start from any infeasible point and allows for infeasibility detection. Secondly, we introduce two heuristic techniques to speed up convergence. These techniques are inspired by similar practice employed in IPMs for symmetric cones.

The paper is organized in two main parts. In the first, theoretical issues are discussed, the algorithm is presented and a proof that the method converges in $\mathcal{O}(\sqrt{\nu}\log(1/\epsilon))$ iterations is carried through via a series of lemmas. Here, $\nu$ denotes the barrier parameter of the barrier for the cone in the conic formulation of the problem (see Section 2.5.1). This complexity estimate thus matches the best-known estimates for IPMs symmetric cones, cf. Section 3.2.1. All theoretical results are included in the main text, emphasizing asymptotic complexity behavior, but the proofs are diverted to the appendix to make the main text clean and maximally readable. In the second part, details related to the actual implementation of the algorithm are presented. We introduce heuristic methods to increase convergence speed and then present an extensive series of computational results substantiating the effectiveness and practical applicability of our algorithm.

### 4.5.2 Theoretical results

The proof of convergence and the worst-case iteration complexity result constitutes a major part of this paper. The algorithm is composed of two main phases: 1. Prediction and 2. Correction.

The prediction phase essentially consists in taking a step in the direction approximately tangent to the central path. This improves optimality, but possibly worsens centrality.

The correction phase consists of repeatedly improving the centrality enough that another prediction step can be taken, thus returning to prediction.

The proof procedes by first analyzing the prediction phase. First, a lower bound on the step length is established ensuring the we do not stray to far from the central path while still maintaining feasibility (Lemmas 4 and 5). Lemma 6 then shows that the correction phase needs no more than two steps in order to sufficiently restore centrality. Finally, Theorem 1 guarantees that the algorithm reaches an $\epsilon$-optimal point in no more than $\mathcal{O}(\sqrt{\nu}\log(1/\epsilon))$ iterations.

### 4.5.3 Computational results

The second part of the paper first introduces two heuristic ways to speed up convergence.

The first is an adaptation of the Mehotra second order correction [34] term, which is known to significantly improve practical performance of IPMs for linear and quadratic conic problems [34, 2, 54]. With the same goal in mind, we suggest a new way to compute a search direction containing second order information for the general (possibly non-self-scaled) conic problem. We show how the final search direction can be viewed as a type of Runge-Kutta method to solve an ordinary differential equation.

The second improvement is an application of quasi-Newton updating of the Hessian of the barrier function to reduce the number of full matrix factorizations needed. It is shown how this can be done in a way retaining the possibility to exploit sparsity in $A$.

Following the description of these techniques, an extensive series of computational experiments are presented. All the problems solved can be modelled using the non-negative cone, the exponential cone and the power cone (see Section 2.7.3). The performance of the algorithm is evaluated on 1. facility location problems, 2. $p$-cone problems, 3. geometric programs and 4. entropy maximization problems. The instances of the two latter are based on "real-life" data, the $p$-cone problems use data from NETLIB [44] and the facility location problems are randomly generated. The algorithm is compared to MOSEK [36] and `CVX/SeDuMi` [24]. The full details about computational performance of the algorithm is included in tables in the paper in appendix A. Please note that because of space constraints, the tables occuring in the main text of the paper have been shortened, while the full-length tables are to be offered as *electronic supplements*. For the sake of completeness of this thesis, the full-length tables have been included in the very end of the paper, i.e. following the references within the paper.

Chapter 5

# Initialization and Warmstarting

In Section 3, we described both feasible start and infeasible start algorithms for linear programming. When discussing the first kind, we tacitly assumed the availability of a feasible starting point. As we saw, infeasible-start algorithms allow the use of a starting point that does not necessarily satisfy the linear equality constraints. Complexity results for the infeasible-start long-step IPM guarantee convergence to the solution set for any starting point in the positive orthant. However, this is a large set and from a theoretical point of view, nothing gives us information about which one to choose.

Nevertheless, it is not uncommon that the iteration count for an interior-point algorithm starting in one particular point may be several times that of a good point for the same problem. For this reason alone, this issue deserves attention. In this chapter, we first outline the common heuristics that are somewhat agreed upon constitute good pointers as to what a good starting point is.

We then move on to the question of *warmstarting* an interior-point algorithm: How does one utilize information from one optimization problem when solving a similar, but different problem? It is well established, that this is a property present with active set methods, and it is probably one of several reasons for some to prefer e.g. the simplex method over an interior-point method. Indeed

it is widely perceived that is difficult to successfully warmstart an interior-point method. After describing the issue in greater detail, we move on to presenting our contribution to this field: a scientific paper by Skajaa, Andersen and Ye published in the journal *Mathematical Programming Computation* in 2013. The full reference is [50]. In it, we develop theory and provide empirical support for two new methods to generate warm starting points for interior-point methods based on the homogeneous and self-dual model. We include only parts of the results in the main text — the paper may be found in its full length in appendix B.

Following the presentation of our paper [50], we demonstrate in an application closer to reality, how our warmstarting scheme may be used to speed-up the internal computations that take place when a process is being controlled under the regime called Model Predictive Control (MPC). In this thorough case study, we work out the details of applying our warmstarting scheme in a smart energy system where flexible consumers (a fleet of electric vehicles) help balance the electricity grid which is supplied partly by classical power plants and partly by stochastic renewable energy sources (wind turbines). To further substantiate the applicability of the warmstarting schemes in model predictive control, we measure their performance against a standard benchmark known as the OPTEC Online QP Benchmark Collection, which is a collection of series of quadratic programs all stemming from real applications in optimal control.

Warmstarting interior-point methods has so far been studied mostly for the case of linear programming, although even for that class of problems, it is still not a very well understood subject. The most recent references in this direction are [8, 14, 16, 22, 23, 27, 31, 48, 63].

## 5.1 Initialization of interior-point methods

The feasible start algorithms described in Section 3 require a starting point $(x^0, y^0, s^0)$ satisfying

$$\begin{aligned} Ax^0 &= b \\ A^T y^0 + s^0 &= c \\ x^0 > 0, s^0 &> 0 \end{aligned} \tag{5.1.1}$$

The problem of determining such a feasible starting is generally of the same complexity as solving the linear program itself with a feasible point available! In practice, this task can be carried out by solving a different but similarly sized LP whose solution is a feasible point for the original LP. This procedure

is commonly called a phase-I method. Thus, we must complete two executions of the solution algorithm to find the final solution. If we use the feasible long-step method (see Section 3.2.3), they have the worst-case iteration complexity $\mathcal{O}(n \log(1/\epsilon))$.

An alternative is offered by the infeasible start long-step interior point algorithm which we described in Section 3.2.4. It allows starting from any point that satisfies $x^0 > 0, s^0 > 0$. That is, the two first equations of (5.1.1) need not be satisfied. This eliminates the need for a phase-I method but it increases the worst-case complexity to $\mathcal{O}\big(n^2 \log(1/\epsilon)\big)$.

One might argue that two executions of an algorithm with iteration complexity $\mathcal{O}(n \log(1/\epsilon))$ is much better that one execution with iteration complexity $\mathcal{O}\big(n^2 \log(1/\epsilon)\big)$, since, in all but trivial cases, we have $n^2 \gg 2n$. However, worst-case complexity results are not predictions of the actual practical behavior of an algorithm. Indeed the infeasible-start long-step IPMs are by far the most popular and applied types of interior-point algorithms today because they perform superiorly in practice.

Theoretical complexity results for the infeasible-start long-step IPM guarantee convergence to the solution set for any starting point satisfying $x^0 > 0$ and $s^0 > 0$. Still, the question remains: which one to choose?

This is largely an unanswered question. Computational experience suggests, however, that the performance of an interior-point algorithm depends crucially on a good starting point. It is not uncommon that the iteration count for one starting point can be several times that of a good one for the same problem.

It is the general perception that a good starting point should satisfy at least two properties: First, it should be well centered. That is, the quantites $x_i^0 s_i^0$, $i = 1, 2, \ldots$ should not be too different. This is measured by use of some centrality measure, e.g. $\|\psi(x^0, s^0)\|$, see Section 3.1.1. Secondly, the magnitude of infeasibility should be roughly the same as that of the initial complementarity gap. That is, the point should not have $\|Ax^0 - b\|$ very large while $\mu(x^0, s^0) = (x^0)^T s^0/n$ is tiny or vice versa. This is due to the fact that the algorithm decreases the linear residuals and the complementarity gap at roughly the same rate. Therefore, if one is much smaller than the other, it will lead to blocking in the step size, preventing fast progress in the larger of the two. This will eventually deteriorate performance.

It is trivial to find a well-centered point (take, for example, $x = e, s = e$ where $e$ is the vector all ones), but not one that is simultaneously not too infeasible. There exist heuristics for choosing a starting point aiming at the above properties, see for example [58].

In Section 2.7.4, we described the HSD model: a linear program (in fact, a feasibility problem) that, when solved, provided either a solution to the original LP or a certificate of infeasibility. The chosen starting point is, *by construction*, feasible for the problem. Similarly to the infeasible-start long-step algorithm, this eliminates the need for a phase-I method. But unlike the latter, the HSD model has the advantage, that the point $x = e, s = e, y = 0, \tau = 1, \kappa = 1$ is feasible *and* is perfectly centered. Therefore, we might expect, although we have no theoretical justification for this expectation, that this trivial starting point is a good one. Indeed, some computational experience shows that solving the HSD model performs well when simply using this trivial starting point[1, 2, 54]. This is a property that we will exploit in our contribution to this field, see Section 5.3.

The discussion above applies to linear programming problem as well as to general conic programs (see Section 4). Here, the initial point must satisfy $x \in \mathcal{K}$ and $s \in \mathcal{K}^*$ and in this area, the existing knowledge about starting points is even more limited that in the case of linear programming.

## 5.2 Warmstarting

Although finding a consistently good starting point for an IPM, dependent on the data $A, b, c$, would be very desirable, it may be a goal set too high. Instead, a situation that has been studied to a certain extent is that of *warmstarting* (sometimes called *hotstarting*) optimization algorithms. Here, one attempts to construct a good starting point for the algorithm using available information about the specific optimization problem at hand. This information may, for example, be in the form of a solution of a different but similar optimization problem of the same type.

Assume that one needs to solve a sequence of different but presumably related optimization problems. Let $x^\star$ denote the solution to an optimization problem $\mathcal{P}$. The aim is then to use the information contained in $x^\star$ to initialize the optimization algorithm at a particularly good (i.e. "warm") point when solving $\hat{\mathcal{P}}$, a related but different problem. Hopefully this will enable us to solve $\hat{\mathcal{P}}$ using less computational effort than had we not known or used $x^\star$.

It is well established computationally that active set methods, such as the simplex method for linear programming, are well able to use the information from $x^*$ when restarted to solve $\hat{\mathcal{P}}$. Since the simplex method works by identifying the optimal basis, the success of the warmstart depends on how many of the active constraints change to inactive and vice versa. Generally, however, the

solution of an optimization problem does *not* depend continuously on the data, i.e. the solution can "jump" even when the problem is perturbed an arbitrarily small amount. So even for the simplex method, there is no theoretical guarantee of a large improvement when warmstarting.

In fact, from a theoretically viewpoint, it seems sensible to be modest in our expectations about the gains from warmstarting in general. Let the linear program $\{\min_x c^T x, \text{ s.t. } Ax = b,\ x \geq 0\}$ be denoted by $\text{LP}(A, b, c)$ and let $x^\star$ be its optimal primal solution. It was shown in [33] that the existence of a strongly polynomial time algorithm[1] for $\{\text{given } x^\star, \text{ solve } \text{LP}(A, b, c)\}$ would imply the existence of a strongly polynomial time algorithm for $\{\text{solve } \text{LP}(A, b, c)\}$. Here "solve" means (a) finding an optimal solution *and* a certificate of optimality or (b) certify that no such solution exists. Thus even *checking* whether a given point is primal optimal (even if the point actually is a primal optimal solution) may potentially be as hard as simply solving the problem from scratch.

Yet computational experience for the simplex method shows that in many cases, warmstarting can indeed bring a large improvement. It is interesting to ask whether warmstarting can also bring practical improvement to interior-point methods in spite of the above theoretical result. Since interior-point methods are applicable to a wider range of problems than active set methods (an example is general convex conic problems), the overall gain may be even larger if succesful warmstarting schemes can be devised for interior-point algorithms.

It is widely perceived that it is hard to warmstart IPMs. The main reason is that if the solution $x^\star$ of $\mathcal{P}$ is on the boundary of the feasible region, then $x^\star$ is likely to also be close to the boundary for $\hat{\mathcal{P}}$ but not well-centered. At an iterate that is close to the boundary but not well-centered, IPMs generally behave badly producing either ill conditioned linear systems or search directions that allow only tiny step sizes. For that reason, progress towards the solution of $\hat{\mathcal{P}}$ is very slow and often it would have been better to simply coldstart the IPM. For the problem classes usually considered $x^\star$ is effectively always on the boundary of $\mathcal{P}$.

Recent work on this topic includes [8, 14, 16, 22, 23, 27, 31, 48, 63], most often for the case of Linear Programming (LP). Common to several of these approaches is the requirement of more information from the solution process of $\mathcal{P}$ than just the final solution $x^\star$. In both [23] and [63], for example, a pool of primal and dual (non-final) iterates from the solution process of $\mathcal{P}$ is required. Other approaches include (a) further perturbing $\hat{\mathcal{P}}$ to move the boundary and in that way avoid tiny stepsizes [31] and (b) allowing decreasing infeasibility of nonnegativity constraints yielding an "exterior point" method, see e.g. [48].

[1] see [62] for a definition

Computational results from several of the above references are generally positive in that they obtain reductions in the number of interior point iterations on the order of about 50% when perturbations are not too large. A problem often incurred, however, is a relatively costly procedure to compute the warm point. This is in particular seen in the comparisons of different warmstarting schemes in [27]. Very recently, a warm-starting method based on a *slack-approach* was introduced in [16]. Extra artificial variables are introduced to avoid any of the two above mentioned drawbacks and the method exibits promising numerical results. For further information about previous work on warmstarting IPMs, see also the thorough overview in [16].

In the following section, we briefly describe our first original contribution to this line of research in the form of the original research paper [50].

## 5.3 Skajaa-Andersen-Ye 2012: Warmstarting the homogeneous and self-dual interior point method for linear and conic quadratic problems

### 5.3.1 Overview

The contents of this and the following sections is an overview of the paper in appendix B.

The contribution of the paper is to introduce two warmstart strategies that use *only the final* optimal iterate of the solution of $\mathcal{P}$ and has low computational complexity. Further they are applicable to general convex conic problems. One of the strategies, $\mathrm{W_P}$, uses only the primal optimal solution $x^\star$ while the other, $\mathrm{W_{PD}}$, uses the primal $x^\star$ and the dual optimal solution $(y^\star, s^\star)$ of $\mathcal{P}$.

There are several reasons motivating these schemes. Firstly, optimization software is often used as black-box subroutines that output only *final* iterates. Hence intermediate non-optimal iterates or internal algorithmic variables may not be available at all. In such a situation, both strategies are useful. Secondly, sometimes just one optimization problem is to be solved, but a user with technical insight into the particular problem may know a good guess for the optimal primal solution. This information should be possible to utilize without requiring a guess for the dual solution as well. In this situation, the strategy $\mathrm{W_P}$ is useful.

In overview, the warmstarting schemes introduced have the following distinctive

features when compared with previously suggested warmstarting strategies:

- They both require virtually zero computational effort to compute
- They both require *only* the final solution from the previous solution process (no pool of previous iterates)
- One requires only the previous primal solution
- The other requires both the previous primal and dual solution
- They are devised specially for the HSD model and thus maintain the advantages introduced by solving this model (see Section 2.7.4).
- They are applicable to general convex conic programming

Assume that $x^\star$ is the primal optimal solution and $(y^\star, s^\star)$ the dual optimal solution of a linear program $\mathcal{P}$. Further let $\lambda \in [0,1)$ and $\mu^0 > 0$ be (user chosen) parameters. The two warm starting points for the initialization of a related but different linear program $\hat{\mathcal{P}}$ are then:

$$(\mathrm{W_P}) \begin{cases} x^0 &= \lambda x^\star + (1-\lambda)e \\ s^0 &= \mu^0 (x^0)^{-1} \\ y^0 &= 0 \\ \tau^0 &= 1 \\ \kappa^0 &= \mu^0 \end{cases} \qquad (\mathrm{W_{PD}}) \begin{cases} x^0 &= \lambda x^\star + (1-\lambda)e \\ s^0 &= \lambda s^\star + (1-\lambda)e \\ y^0 &= \lambda y^\star \\ \tau^0 &= 1 \\ \kappa^0 &= (x^0)^T s^0 / n \end{cases} \tag{5.3.1}$$

Here, $(x^0)^{-1}$ denotes the *elementwise* reciprocal of $x^0$.

As previously discussed, some computational experience[1, 2, 54] indicates that the starting point $\mathrm{C} := (e, 1, 0, e, 1)$ seems to work well for the initialization of an interior point method to solve the HSD-model. We will call this starting point the *cold* point. We can view the starting point $\mathrm{W_{PD}}$ in (5.3.1) as a convex combination of $(x^\star, y^\star, s^\star)$ and the cold starting point C. Thus, hopefully, $\mathrm{W_{PD}}$ is a point closer (in some sense) to the solution of $\hat{\mathcal{P}}$, but incorporation of $(1-\lambda)\mathrm{C}$ introduces enough centrality to avoid tiny step sizes. The point $\mathrm{W_P}$ is identical to $\mathrm{W_{PD}}$ for the primal variable, but, as we restrict ourselves to using *only primal information*, we cannot do the same for the dual variables. Instead we choose $s^0$ so that the point is perfectly centered and has a prescribed duality gap, namely $\mu^0$.

The paper contains two main parts: 1. Some mainly theoretical results proving improved theoretical complexity when warmstarting under certain conditions and 2. A series of numerical experiments substantiating that the warmstarting strategies are useful in practice.

### 5.3.2 Theoretical results

The proof in the first part establishes that under certain conditions, the warmstarting schemes indeed improve worst-case theoretical complexity. This is done by noting that by applying the Mizuno-Todd-Ye feasible predictor-corrector IPM (see Section 3.2.2), the worst-case iteration complexity is $\mathcal{O}\left(\sqrt{n}\log\left(\Psi(z^0)/\epsilon\right)\right)$ when using the initial point $z^0 \in \mathcal{N}_2(\eta)$ with $\eta \in (0,1)$, see Section 3.1.1. Here,

$$\Psi(z) = \max\left\{\mu(z), \|Ax - b\tau\|, \|A^T y + s - c\tau\|\right\}.$$

See [62] for a proof of this complexity result.

To obtain a better worst-case complexity than starting in C, we would need to initialize the algorithm in a point $z^0$ satisfying $\Psi(z^0) < \Psi(\text{C})$, which is certainly satisfied if

$$\mu(z^0) < \mu(\text{C}), \quad \|Ax^0 - b\tau^0\| < \|Ae - b\|, \quad \|A^T y^0 + s^0 - c\tau^0\| < \|e - c\|. \tag{5.3.2}$$

Our paper's proof of improved worst-case complexity when warmstarting then proceeds by showing the conditions exist under which the three inequalities in (5.3.2) and $z^0 \in \mathcal{N}_2(\eta)$ hold. See the paper in appendix B for further details.

### 5.3.3 Computational results

The second part of the paper present an extensive series of computational experiment documenting that the warmstarting schemes indeed are useful in practice. The proposed schemes are tested on the generic test set of LPs from NETLIB [44] as well as real-life problems arising in financial portfolio optimization. The latter problems are mixed linear and quadratic conic problems and the positive results thus support the previous claim that the schemes are applicable to more general convex conic programming.

To quantify the performance improvement from warmstarting, we use the following measures. Let $\mathcal{I}^{\text{cold}}$ and $\mathcal{I}^{\text{warm}}$ denote the number of iterations needed to solve $\hat{\mathcal{P}}$ from a cold- and warmstart respectively. We then define the measure

$$\mathcal{R} = \mathcal{I}^{\text{warm}}/\mathcal{I}^{\text{cold}}$$

to quantify the gain from warmstarting. If $\mathcal{R} < 1$ the warmstarted run was more efficient than the coldstarted and vice versa.

For an entire sequence of problems $\mathcal{P}_1, \ldots, \mathcal{P}_K$ we define

$$\begin{aligned}
\mathcal{G}_{\mathcal{I}}^{\text{cold}} &= \sqrt[K]{\mathcal{I}_1^{\text{cold}} \cdots \mathcal{I}_K^{\text{cold}}} \\
\mathcal{G}_{\mathcal{I}}^{\text{warm}} &= \sqrt[K]{\mathcal{I}_1^{\text{warm}} \cdots \mathcal{I}_K^{\text{warm}}} \\
\mathcal{G}_{\mathcal{R}} &= \sqrt[K]{\mathcal{R}_1 \cdots \mathcal{R}_K} \\
&= \mathcal{G}_{\mathcal{I}}^{\text{warm}} / \mathcal{G}_{\mathcal{I}}^{\text{cold}},
\end{aligned}$$

the geometric means of the quantities $\mathcal{I}_i^{\text{cold}}$, $\mathcal{I}_i^{\text{warm}}$ and $\mathcal{R}_i$. When $\mathcal{G}_{\mathcal{R}} < 1$, the warmstarting strategy was more efficient in solving the sequence of problems alltogether than had we coldstarted the algorithm — and vice versa. We count iterations since the main iterations of an interior point method constitutes the majority of computational work involved. Since the computational effort needed to compute the warm points is negligible, cf (5.3.1), the reduction in CPU-time will be the same as that measured in terms of iterations.

The experiments generally show work reductions when warmstarting compared to coldstarting in the range 30%–75% depending on the problem class and magnitude of the problem perturbation.

A particularly illustrative result is that of solving the NETLIB benchmark test set of LPs with varying problem perturbation. In this experiment, we perform the following sequence for each problem in the NETLIB test set.

1. Solve the problem $\mathcal{P} = \text{LP}(A, b, c)$, and store solution in $x^\star, y^\star, s^\star$.
2. Compute the warm starting point $\text{W}_\text{P}$ and $\text{W}_\text{PD}$ according to (5.3.1).
3. Randomly perturb the problem into a new problem $\hat{\mathcal{P}} = \text{LP}(\hat{A}, \hat{b}, \hat{c})$
4. Solve the new problem $\hat{\mathcal{P}}$ initializing the algorithm from C, $\text{W}_\text{P}$ and $\text{W}_\text{PD}$.
5. Count the number of iterations spent solving the new problem both from the cold point C and the warm points $\text{W}_\text{P}$ and $\text{W}_\text{PD}$.

Let $v$ be a vector we want to perturb randomly (think of either $b, c$ or the vector of nonzeros of $A$). Assume $v$ has $M$ elements. An element in $v$ is changed if a $[0, 1]$-uniform randomly chosen number is less than $\min\{0.1, 20/M\}$. Thus on average, we change 10% but at most 20 elements of $v$. Notice that we this way preserve the sparsity pattern of $v$. An element $v_i$ is changed by setting

$$v_i := \begin{cases} \delta r & \text{if } |v_i| \leq 10^{-6} \\ (1 + \delta r) v_i & \text{otherwise} \end{cases}$$

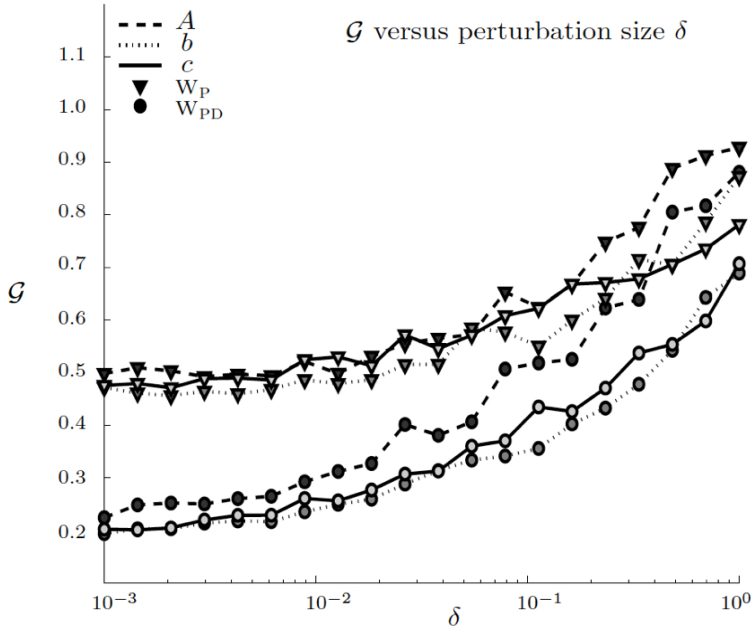



**Figure 5.1:** Results from the NETLIB test set of LPs with $\lambda = 0.99$ and $\mu^0 = 0.01$ and varying $\delta$. Each data point in the figure corresponds to solving all problems in the NETLIB test set with the problem-perturbation specified in the legend for a certain value of $\delta$.

where $r$ is a number chosen randomly from a uniform distribution on $[-1, 1]$. The scalar $\delta$ is a parameter that controls the pertubation magnitude.

Figure 5.1 presents results for the three cases where $v$ is either $b$, $c$ or the nonzeros of $A$. The figure shows the relation between the magnitude of the perturbation $\delta$ and reduction in the geometric mean of number of iterations. As expected, we clearly observe that the reduction depends crucially on $\delta$. The size of the reduction is significant as long as $\delta$ is small enough. It is apparent that $W_{PD}$ is consistently better than $W_P$. This is of course reasonable since $W_{PD}$ uses more information from the solution of $\mathcal{P}$ than $W_P$. Notice, however, that the gap between $W_P$ and $W_{PD}$ narrows as $\delta$ grows. This too is reasonable, because as the problem is perturbed more, the information from the primal or the dual points can no longer be expected to be good. Thus both behave more and more like a coldstart.

## 5.4 Accelerating computations in model predictive control using interior-point warmstarting

As a more application oriented contribution to the area of warmstarting interior-point methods, this section focusses on the optimization problems that arise in optimal control when following the regime of Model Predictive Control (MPC).

### 5.4.1 Overview

Economic Model Predictive Control (MPC) for linear systems is useful for solution of a number of control problems arising in smart energy systems. Economic MPCs can be formulated as sequences of linear programs. In this section we apply the homogeneous and self-dual interior-point algorithm to solve these linear programs. This enables the use of the novel warmstarting strategy for mixed linear and conic quadratic optimization problems presented in Section 5.3. We demonstrate the efficiency of this warmstarting strategy when used for economic MPC as well as MPC resulting in a sequence of quadratic programs. Compared to an implementation not making use of warmstarting, our scheme results in a reduced number of iterations and therefore a more efficient method which is useful in time-critical applications.

The charging of the batteries in electrical vehicles as well as the control of a portfolio of power generators are examples of processes that can be efficiently controlled by economic MPC. The series of LPs that must be solved comprises problems where each instance is closely related to the next. We conduct computational experiments with these problems, showing that in this case computational work load can be reduced by about 25–40% using our warmstarting scheme.

We then further substantiate the usefulness of our warmstarting scheme by conducting experiments for five different test sets of QPs arising from different linear MPC applications. These sets are from the OPTEC Online QP Benchmark Collection [46]. For these problems, the warmstarting scheme exhibits work reductions of 65%—83%.

### 5.4.2 Introduction

Model predictive control requires solution of optimization problems in real-time. Consequently, until recently MPC has been limited to slow systems with ample time for solution of the optimization problem. To expand the types of processes that can be controlled by MPC, intensive research has been conducted to improve the computational efficiency of optimization algorithms. Examples include automatically generated tuned code [32] and intelligent warmstarting of active set methods [18]. In the MPC regime, an optimization problem $\mathcal{P}$ — most often either a linear program (LP) or quadratic program (QP) — must be solved in each sampling instant. Since each problem usually does not differ much from the next, the entire solution process involves solving a sequence of closely related optimization problems. In case online solution of the optimization problem is needed, an obvious idea to reduce computation time is to somehow utilize the information contained in the solution $x^\star$ of the problem $\mathcal{P}$ when solving $\hat{\mathcal{P}}$, the next problem in the sequence. In this section, we demonstrate how to implement this idea by using *warmstarting* of the homogeneous and self-dual interior-point method (IPM) for mixed linear and conic quadratic optimization. Specifically, we show how the warmstarting strategy from [50] (included in appendix B) can be suitably modified to work well for the sequences of problems that arise from economic MPC generating sequences of LPs and for linear MPC generating sequences of QPs.

The following sections are structured as follows: First the general framework for MPC is introduced followed by a presentation of how to generate the warm starting point and a demonstration of how to efficiently use it for the case of MPC. We then turn to a particularly interesting case study, which focusses on smart energy systems and its model predictive control scheme. For this case study, we present simulation results as well as numerical results showing that the warmstarting strategy indeed is efficient in practice for these sequences of LPs. Finally, the efficiency of the warmstarting strategy is further substantiated, where we use it on sequences of QPs from different linear MPC applications.

### 5.4.3 Model predictive control

In this work, we exclusively deal with MPC for linear discrete time state space systems on the form

$$x_{k+1} = Ax_k + Bu_k + Ed_k \tag{5.4.1a}$$

$$y_k = Cx_k + Du_k + Fd_k. \tag{5.4.1b}$$

Here, $x$ is the state vector, $u$ the manipulable variable, $d$ a disturbance and $y$ the output. Subscripts are indices for time-steps.

In traditional tracking control, the objective is to minimize the error between a given reference $r$ and the measured output. When implemented as an MPC, the error is often penalized using the Euclidean norm. This approach leads to the following optimization problem to be solved over the prediction horizon $N$:

$$\min_{\{y,u,x\}} \quad f(y,u,x) \tag{5.4.2a}$$

$$\text{s.t.} \quad x_{k+1} = Ax_k + Bu_k + Ed_k \tag{5.4.2b}$$

$$y_k = Cx_k + Du_k + Fd_k \tag{5.4.2c}$$

$$u_{\min} \leq u_k \leq u_{\max} \tag{5.4.2d}$$

$$y_{\min} \leq y_k \leq y_{\max} \tag{5.4.2e}$$

i.e. minimize some cost function (or penalty function) subject to the variables respecting the discrete time state space system and stay within predefined bounds.

For tracking problems in which the Euclidean norm is used as a penalty for deviation, (5.4.2) becomes a QP with an objective function of the form

$$f(y,u,x) = \frac{1}{2}\sum_{k=0}^{N-1} ||y_k - r_k||_Q^2 + ||u_k - u_{k-1}||_R^2 \tag{5.4.3}$$

The weights $Q$ and $R$ are tunable and a regularization term has been added. At every time step $k$ the goal is to compute $\{u_k\}_{k=0}^{N-1}$ such that the predicted output trajectory $\{y_k\}_{k=0}^{N-1}$ follows the specified output trajectory $\{r_k\}_{k=0}^{N-1}$ as well as possible. $N$ is the prediction horizon, which is normally chosen quite large in order to avoid discrepancies between open loop and closed loop profiles. The first control input $u_0^*$ is then applied to the system. As new information becomes available at the next sampling time, we redo the process of solving the optimization problem using a moving horizon and keep applying the first control input $u_0^*$ of the solution to the system. The input and output constraints are inherently taken into consideration and handled by this optimal controller.

In the case where the objective function is instead the actual cost of operating the system we obtain a so-called economic MPC. When there is only a cost associated with control action $u$ the objective is linear and the problem (5.4.2) reduces to an LP with

$$f(y,u,x) = \sum_{k=0}^{N-1} c_k^T u_k. \tag{5.4.4}$$

The costs enter the optimization problem as the coefficients $c_k$, which must be forecast over the entire prediction horizon, i.e. for $k = 0, \ldots, N-1$. Similarly, we must model and forecast the disturbances $\{d_k\}_{k=0}^{N-1}$. If the forecasts are of high quality and there are no unmodelled disturbances, then the solution does not change very much from one time step to the next. We might expect that the solution for time $k = 0$ shifted one time step is very similar to the solution at $k = 1$. Shifting in time corresponds to removing the first element in $u_0^*$ and adding an end point at $k = N-1$ with a qualified guess of the solution, for example the previous last element of $u^*$. Consequently the MPC-setup is an ideal candidate for warmstarting. Both LPs and QPs as described above will be treated in the following sections.

### 5.4.4 Warmstarting problem in MPC

The MPC problems from Section 5.4.3 have the following generic form after eliminating the states from (5.4.2):

$$\begin{array}{lcccc} \min_z & & & f(z) & \\ \text{s.t.} & \ell & \leq & z & \leq & u \\ & d_\ell & \leq & Kz & \leq & d_u \end{array} \tag{5.4.5}$$

which we will bring into standard form as follows: The simple bounds $\ell \leq z \leq u$ are equivalent to $z_1 + z_2 = u - \ell$ with $z_1, z_2 \geq 0$ and $z_1 = z - \ell$. Defining $z_3 = Kz - d_\ell$, we further get that $d_\ell \leq Kz \leq d_u$ gives $-Kz_1 + z_3 = K\ell - d_\ell$ and $z_1 + z_4 = d_u - d_\ell$ where $z_3, z_4 \geq 0$. Altogether we get the standard form-problem

$$\min_{z_1,z_2,z_3,z_4} f(z_1 + \ell), \quad \text{s.t.} \tag{5.4.6a}$$

$$\begin{pmatrix} I_n & I_n & 0 & 0 \\ -K & 0 & I_{m_K} & 0 \\ 0 & 0 & I_{m_K} & I_{m_K} \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{pmatrix} = \begin{pmatrix} u - \ell \\ K\ell - d_\ell \\ d_u - d_\ell \end{pmatrix} \tag{5.4.6b}$$

$$(z_1, z_2, z_3, z_4) \geq 0 \tag{5.4.6c}$$

where $m_K$ is the number of rows in $K$, $n$ is the number columns in $K$ and $I_m$ denotes the identity matrix of size $m \times m$. Notice that the number of nonzeros in the linear constraint matrix in 5.4.6b is essentially the same as that of $K$. The function $f$ is the cost function which, for the case of MPC problems in this study, is either a linear function making (5.4.6) an LP or a quadratic function making (5.4.6) a QP.

In the case of an LP, the problem is directly applicable to the homogeneous and self-dual interior-point algorithm which was implemented and used extensively

in computation experiments in the paper from Section 5.3. For the full describtion of the algorithm, see Section 4 in the paper in appendix B. In order to apply it to a QP, we need to model the quadratic cost function using the quadratic cone to bring the problem onto the stadard conic form (2.7.1). For this purpose, consider the optimization problem $\min_{z\in\mathcal{A}}\{z^TQz/2\}$, where $\mathcal{A}$ denotes an affine set and $Q$ is symmetric and positive definite with Cholesky factor $L$. We can then model this problem by

$$\min_{p,t,v,w} \quad t \tag{5.4.7a}$$
$$\text{subject to} \quad v + w = \sqrt{2}t \tag{5.4.7b}$$
$$v - w = \sqrt{2} \tag{5.4.7c}$$
$$p \in L\mathcal{A} \tag{5.4.7d}$$
$$\|(y, w)\|_2 \leq v \tag{5.4.7e}$$

which contains a quadratic cone constraint in (5.4.7e) and the problem is indeed on the form (2.7.1).

Let us now describe in detail an effective way to apply the warmstarting strategies of the paper in appendix B to our MPC problems. Since we have available both the primal and the dual solution of the previous problem in the sequence, we will use the strategy denoted $\text{W}_{\text{PD}}$ (see [50]).

Assume $(z_1^\star, z_2^\star, z_3^\star, z_4^\star, m_1^\star, m_2^\star, m_3^\star, s_1^\star, s_2^\star, s_3^\star, s_4^\star)$ is the primal and dual optimal solution of $\mathcal{P}$, an optimization problem of the type (5.4.6) in the sequence of problems that we need to solve. Here, $z_i$ denote the primal variables, $m_i$ the dual multipliers and $s_i$ the dual slacks. The next problem in the sequence is denoted $\hat{\mathcal{P}}$, for which the horizon has moved one time step. It is then reasonable to expect that the point $\hat{z}_i := ([z_i^\star]_{2:n}, [z_i^\star]_n)$, for $i = 1, \ldots, 4$ is close to the primal optimal solution of $\hat{\mathcal{P}}$. Define similarly $\hat{s}_i := ([s_i^\star]_{2:n}, [s_i^\star]_n)$. The dual problem of (5.4.6) is linearly constrained by

$$\begin{pmatrix} I_n & -K^T & 0 \\ I_n & 0 & 0 \\ 0 & I_{m_K} & I_{m_K} \\ 0 & 0 & I_{m_K} \end{pmatrix} \begin{pmatrix} m_1 \\ m_2 \\ m_3 \end{pmatrix} + \begin{pmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \end{pmatrix} = \begin{pmatrix} c \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{5.4.8}$$

from which we can deduce $m_1 = -s_2$, $m_2 = -s_3 - s_4$ and $m_3 = -s_4$. For that reason we define $\hat{m}_1 = -\hat{s}_2$, $\hat{m}_2 = -\hat{s}_3 - \hat{s}_4$ and $\hat{m}_3 = -\hat{s}_4$. We will then use the starting point $\text{W}_{\text{PD}}$ to initialize our homogeneous interior-point algorithm to

solve $\hat{\mathcal{P}}$:

$$\begin{aligned} z^0 &= \lambda\hat{z} + (1-\lambda)e \\ s^0 &= \lambda\hat{s} + (1-\lambda)e \\ m^0 &= \lambda\hat{m} \\ \tau^0 &= 1 \\ \kappa^0 &= (z^0)^T s^0/N \end{aligned} \tag{5.4.9}$$

where $\lambda \in [0,1]$ is a parameter and $e$ is unit vector defined in Section 4.4 in [50] (Appendix B). The starting point is thus a convex combination of the shifted previous solution $(\hat{z}, \hat{m}, \hat{s})$ and the point $(e, 0, e)$, which we will call the *cold starting point*. This cold point is commonly used in IPMs for solving the HSD-model, see e.g. [53]. Choosing $\lambda$ close to 1 implies a certain a risk of starting close to the boundary of the feasible region while $\lambda = 0$ results in the perfectly centered cold point. It is shown in [50] that, under certain conditions, the initialization of an IPM to solve HSD in the above starting point results in a better theoretical worst-case complexity.

### 5.4.5 Case study: Smart energy system

We now study a particular MPC scenario from smart energy systems. This case study results in linear economic MPC, which as usual requires the solution of a sequence of LPs and is therefore well suited for the warmstarting scheme outlined in the previous section. We first present the system under consideration and then show simulations results and numerical results from warmstarting experiments.

A smart energy system essentially consists of a number of flexible consumers along with a stochastically varying power production. Here "flexible" means that the consumers are able to expedite or delay their consumption upon request while staying within certain limits. This situation occurs when part of the power production comes from renewable energy sources that are difficult to accurately predict such as wind or solar. Balancing the power at all time scales therefore requires a high level of coordination between producers and consumers. On the timescale of minutes conventional power plants must control their production in the cheapest possible way while meeting the load demand imposed by consumers.

In this case study we use economic MPC to minimize the power plant production costs. An aggregated fleet of electric vehicles (EVs) is available and can be used as flexible storage to help balance the load but they have limited storage and their predefined charging needs must simultaneously be satisfied. Finally, the model must also take into account the flexible consumers. This setup results in a large-scale optimization problem to be solved in each time instance. Using

warmstarting, we demonstrate that computation time can be reduced which will allow the solution of even larger systems.

**A smart energy system.** The following transfer function models describe the dynamics of the individual components considered in this case study of a smart energy system:
*(a) Power plant:*

$$y_p = \frac{K_p}{(\tau_p s + 1)^3} u_p \tag{5.4.10}$$

where $u_p$ denotes the set point controlling the output power production $y_p$. The parameters $\tau_p$ and $K_p$ represent time constant and gain [26].
*(b) Wind farm:*

$$x_w = \frac{K_w}{\tau_w s + 1} d_w \tag{5.4.11}$$

where $d_w$ is the wind speed input and $x_w$ is the produced power.
*(c) Electric vehicle:*

$$y_v = \frac{1}{s}\left(\eta^+ u^+ - 1/\eta^- u^- - d_v\right). \tag{5.4.12}$$

Here, $y_v$ denotes the state of charge influenced by the charge and discharge signals, $u^+$ respectively $u^-$, with efficiency $\eta$. The charging need of the EVs is denoted $d_v$ and are thus modelled by a disturbance. In order to reduce the problem size, the EVs are modeled as one large aggregated battery storage. The resulting aggregated charge and discharge plan can be submitted to a lower level controller or aggregator that can manage each EV and make sure that the aggregated response is fulfilled. Note that the charging and discharging constraints should be quite conservative since they depend on the availability of the EVs, i.e. it is assumed that an EV is not connected to the grid and able to charge while driving. Based on driving pattern analysis the availability has been reported to be more than 90% assuming that the EV is able to charge whenever it is parked [59].

The EV batteries naturally have a finite battery capacity $Q_c$ that limits the size of the EV storage such that

$$0 \leq y_v \leq Q_c n_{ev} \tag{5.4.13}$$

where $n_{ev}$ is the number of EVs in the fleet. The power balance $y_t \geq 0$ must be nonnegative in order to meet the demand. So we set

$$y_t = y_1 + y_2 + x_w - u_+ + u_- - d_b \geq 0$$

where $d_b$ is some base load from other nonflexible consumers in the energy system.

**State space model.** We consider the case with $n_p = 2$ power plants, $n_w = 1$ wind farm and $n_v = 1$ EV fleet aggregated by $n_{ev} = 10.000$ EVs. Also a base load $n_b = 1$ is set as a disturbance, i.e. the reference load from all other unmodeled power consumers that must also be supplied with energy.

The transfer functions (5.4.10)-(5.4.12) are realized in a discrete time state space model (5.4.1) with sampling period $T_s = 5$s. For $n_p = 2$, $n_w = 1$, $n_v = 1$, $n_b = 1$ we get the states, $x$, the manipulables $u$, disturbances $d$, and the outputs $y$:

$$
\begin{aligned}
x &= \begin{bmatrix} \ddot{x}_1 & \dot{x}_1 & x_1 & \ddot{x}_2 & \dot{x}_2 & x_2 & x_w & x_v \end{bmatrix}^T \\
u &= \begin{bmatrix} u_1 & u_2 & u_+ & u_- \end{bmatrix}^T \\
d &= \begin{bmatrix} d_b & d_w & d_v \end{bmatrix}^T \\
y &= \begin{bmatrix} y_1 & y_2 & y_v & y_t \end{bmatrix}^T
\end{aligned}
$$

The disturbances are modeled here but could be forecasted directly from some other routine. Notice that the charge inputs for the EV are measured and introduces a direct control action in the output through the $D$ matrix, cf. (5.4.2c).

**Economic model predictive control.** Let us now apply economic MPC to control the entire power system planning. Economic MPC for intelligent energy systems has previously been proposed in [26]. The MPC will minimize the electricity costs of operating a number of power plants, fleets of EVs, wind farms and consumers based on predictions of the demand, production and operating costs over the prediction horizon. The economic MPC can in full be formulated as

$$
\begin{aligned}
\min_{u} \quad & \sum_{k=0}^{N-1} p_k^T u_k + \sum_{k=0}^{N} \rho v_k \\
\text{s.t.} \quad & x_{k+1} = Ax_k + Bu_k + Ed_k \\
& y_k = Cx_k + Du_k + Fd_k \\
& u_{\min} \leq u_k \leq u_{\max} \\
& \Delta u_{\min} \leq \Delta u_k \leq \Delta u_{\max} \\
& y_{\min} - v_k \leq y_k \leq y_{\max} + v_k \\
& v_k \geq 0
\end{aligned}
\tag{5.4.14}
$$

where $k \in \{0, 1, \ldots, N\}$ and $N$ is the prediction horizon. Note that soft constraints on the power balance output are used. It is crucial to meet power demands at all times and any imbalances will be economically penalized in a real power market. In case of imbalance, external spinning reserves will be activated to restore the balance. Consequently, in our case the slack variable penalty $\rho$ could also be an actual cost or penalty for not providing enough power.

| | Value | Unit | Description |
|---|---|---|---|
| $n_p$ | 2 | — | Number of power plants |
| $n_w$ | 1 | — | Number of wind farms |
| $n_v$ | 1 | — | Number of EV fleets |
| $n_b$ | 1 | — | Number of base loads |
| $n_{ev}$ | 10.000 | — | Number of EVs |
| $n_x$ | $3n_p + n_w + n_v$ | — | Number of states |
| $n_u$ | $n_p + 2n_v$ | — | Number of inputs |
| $n_y$ | $n_p + n_v + 1$ | — | Number of outputs |
| $n_d$ | $n_w + n_v$ | — | Number of disturbances |
| $K$ | $[1, 1]$ | — | Power plant gain |
| $K_w$ | 1 | MW/(m/s) | Wind farm gain |
| $\tau$ | $[1, 1]$ | s | Power plant time constant |
| $\tau_w$ | 0.7 | s | Wind farm time constant |
| $y_p$ | var. | MW | Power plant output power |
| $x_w$ | var. | MW | Wind farm output power |
| $y_v$ | var. | MWh | EV fleet state of charge (SOC) |
| $u^+$ | var. | MW | EV charge input |
| $u^-$ | var. | MW | EV discharge input |
| $Q_c$ | 24 | kWh | EV battery capacity |
| $\eta^+$ | 0.9 | — | EV charge efficiency |
| $\eta^-$ | 0.9 | — | EV discharge efficiency |
| $d_w$ | var. | m/s | Wind speed at wind farm |
| $d_v$ | var. | MW | EV charge demand |
| $d_b$ | var. | MW | Base load power demand |
| $u_{min}$ | [0,0,0,0] | MW | Minimum control input |
| $u_{max}$ | [5,7,3,3] | MW | Maximum control input |
| $\Delta u_{min}$ | [-2,-0.2,-0.6,-0.6] | MW/s | Minimum control ramp input |
| $\Delta u_{max}$ | [2,0.2,0.6,0.6] | MW/s | Maximum control ramp input |
| $y_{min}$ | [0,0,0,0] | — | Minimum output |
| $y_{max}$ | [5,7,240,$\infty$] | — | Maximum output |
| $p$ | [10,5,0,5] | $\mathrm{MW}^{-1}$ | Production costs |

**Table 5.1:** *Case study parameters*

The optimal power production within the prediction horizon is the solution to (5.4.14) and is denoted $U^* = \{u_k^*\}_{k=0}^{N-1}$. This control action is calculated at every time step $k$ and represents a decision plan, stating when to produce and with how much power. The EV storage charge and discharge is also part of the decision plan. The control action is optimal in terms of economy and is the cheapest based on the predictions and model assumptions available at time $k = 0$. The first decision of the plan, $u_0^*$, is implemented, i.e. a certain amount of power is delivered to the battery at the present time step $k = 0$. This process is repeated at every time step. This is the general principle of model predictive control.

**Simulation.** Figure 5.2 on the next page shows the results of a closed loop simulation of the power system described. We have used the parameters from Table 5.1 on the preceding page. A prediction horizon of 6 min was chosen with a 5s sampling period. The first plot from above shows the power plant production and their set points. The expensive but fast power plant is used to balance the load while the slower but cheaper power plant ramps up production. The second plot shows how the EV fleet is controlled and the resulting charge and discharge power. In the beginning of the simulation the EV fleet discharges to the grid to boost production. Consequently, the EV batteries are gradually depleted and the state of charge decreases. The charge demand from the EVs was modeled as a sinusoid. The third plot shows overall power balance including a base load disturbance and the wind power production. Also the power balance $y_t$ is shown expressing the imbalances from the positive production and negative consumption. Since we have no uncertainty on the load forecasts this imbalance is mostly zero.

Finally, in the bottom plot the number of iterations used by our algorithm when coldstarting (C) and warmstarting (W) is shown. We always use the solution of $\mathcal{P}$ to compute our warm starting point for the initialization of our algorithm to solve $\hat{\mathcal{P}}$. This was done as described in the preceeding pages.

The results are also shown in Table 5.2 on page 76. We notice a relatively large variety in the improvement of using warmstarting over the individual problems. For some problems the iteration count is about 50% of the corresponding cold start while it for others cuts away only about 10%. Overall, the total work reduction is about 25%.

We remark further, that the gain from warmstarting for these problems is quite sensitive to the parameters involved. If, for example, the sampling time is reduced, the neighboring problems are more alike, and warmstarting is even more useful. Another example is the initial state of charge of the EVs which, if

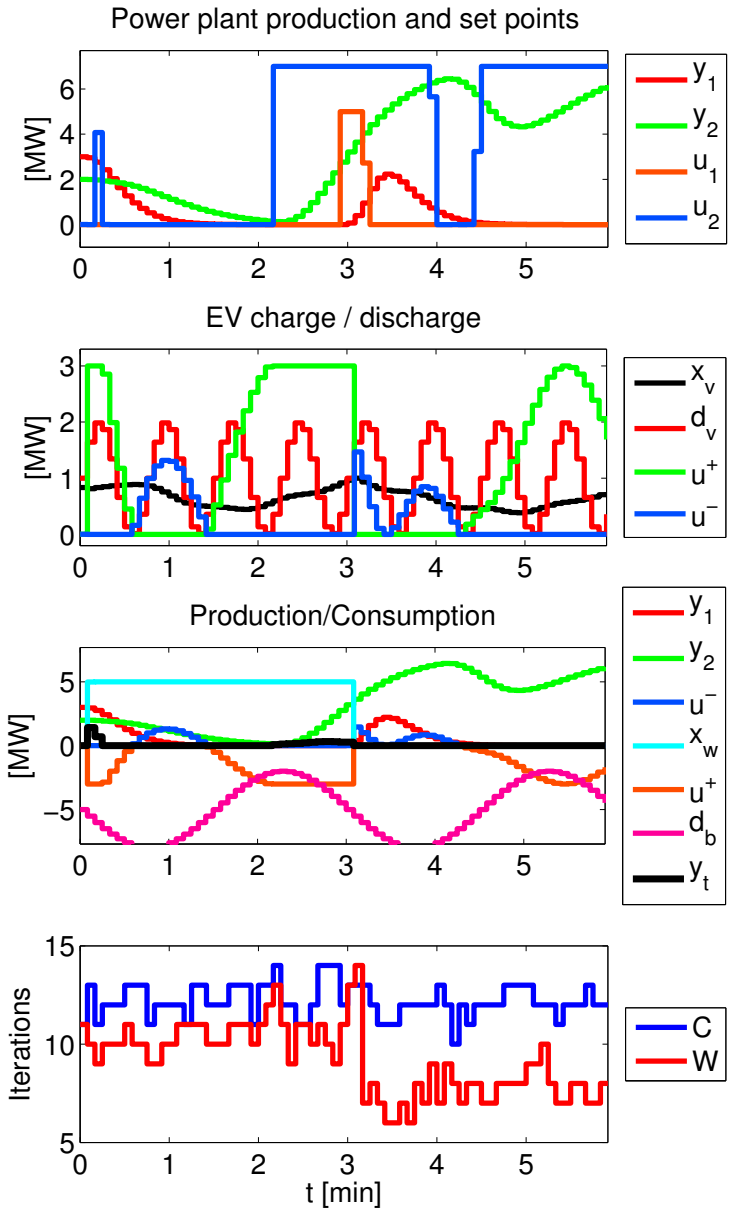



**Figure 5.2:** *Case study simulation with two power plants, a wind farm, a large EV fleet and a base load consumption. Shows the resulting closed loop economic MPC decisions of production and consumption over 6 minutes. Performance when warmstarting our algorithm (W) is compared to standard coldstarting (C) at the bottom (see also Table 5.2).*

| P | $\mathcal{I}^{\text{cold}}$ | $\mathcal{I}^{\text{warm}}$ | $\mathcal{R}$ |
|---|---|---|---|
| 1 | 11 | 11 | 1.00 |
| 2 | 13 | 11 | 0.85 |
| 3 | 11 | 10 | 0.91 |
| . | . | . | . |
| 13 | 12 | 10 | 0.83 |
| 14 | 12 | 11 | 0.92 |
| 15 | 11 | 12 | 1.09 |
| 16 | 13 | 14 | 1.08 |
| 17 | 13 | 12 | 0.92 |
| . | . | . | . |
| 69 | 13 | 7 | 0.54 |
| 70 | 12 | 7 | 0.58 |
| 71 | 12 | 8 | 0.67 |
| 72 | 12 | 8 | 0.67 |
| $\mathcal{G}$ | 12.2 | 9.3 | **0.76** |

**Table 5.2:** *LPs from the Electric Vehicle Control Problem. Columns two and three show iteration counts and the fourth column their ratio. The last row shows geometric means (see Section 5.4.6).*

relatively high, results in few charging periods and thus less varying predictions. This also improves warmstarting performance. Generally, warmstarting is most useful when the simulation is "uneventful" in the sense that few changes occur. These considerations suggest an adaptive strategy: When model predictions are relatively uneventful, use warmstarting. In the opposite case, use coldstart.

### 5.4.6 Further computational results: Quadratic programs

In this section, we conduct a number of computational experiments to further substantiate the usefulness of the warmstarting strategy when used in MPC applications. These experiments also show the versatility of the strategy. Here it is applied to sequences of QPs that stem from different MPC applications.

**Methodology.** Let $\mathcal{I}^{\text{cold}}$ and $\mathcal{I}^{\text{warm}}$ denote the number of iterations needed to solve $\hat{\mathcal{P}}$ from a cold- and warmstart, respectively. We then define the measure

$$\mathcal{R} = \mathcal{I}^{\text{warm}}/\mathcal{I}^{\text{cold}}$$

to quantify the gain from warmstarting. If $\mathcal{R} < 1$ the warmstarted run was more efficient than the coldstarted and vice versa. To compute the starting point (5.4.9), we use for $(x^\star, y^\star, s^\star)$ the solution to $\mathcal{P}$, the previous problem in the sequence.

For an entire sequence of problems $\mathcal{P}_1, \ldots, \mathcal{P}_K$ we define

$$\begin{aligned}
\mathcal{G}_{\mathcal{I}}^{\text{cold}} &= \sqrt[K]{\mathcal{I}_1^{\text{cold}} \cdots \mathcal{I}_K^{\text{cold}}} \\
\mathcal{G}_{\mathcal{I}}^{\text{warm}} &= \sqrt[K]{\mathcal{I}_1^{\text{warm}} \cdots \mathcal{I}_K^{\text{warm}}} \\
\mathcal{G}_{\mathcal{R}} &= \sqrt[K]{\mathcal{R}_1 \cdots \mathcal{R}_K} \\
&= \mathcal{G}_{\mathcal{I}}^{\text{warm}} / \mathcal{G}_{\mathcal{I}}^{\text{cold}},
\end{aligned}$$

the geometric means of the quantities $\mathcal{I}_i^{\text{cold}}$, $\mathcal{I}_i^{\text{warm}}$ and $\mathcal{R}_i$. When $\mathcal{G}_{\mathcal{R}} < 1$, the warmstarting strategy was more efficient in solving the sequence of problems alltogether than had we coldstarted the algorithm — and vice versa. We count just the number of iterations since the main iterations of an interior point method consume the vast majority of computational work involved. Therefore, the reduction in CPU-time will be the same as that measured in terms of iterations.

**QPs from the OPTEC online QP benchmark collection.** We test the strategy on test problems obtained from the OPTEC Online QP Benchmark Collection [46]. Quoting from the source of this test problem collection, it is apparent that our strategy is well suited for these problems:

> "In our opinion, it is not sufficient just to apply fast offline QP solvers to problems arising in MPC applications. Instead, fast online QP solvers must take into account the special structure of the problems. Normally, the problems within the MPC context do not differ much from one QP to the next making it essential to incorporate this knowledge into efficient solvers."

Each problem set from the collection provides a sequence of QPs of the form

$$\min_z \quad \tfrac{1}{2} z^T H z + g^T z \tag{5.4.15a}$$

$$\text{s.t.} \quad \begin{array}{ccccc} \ell & \leq & z & \leq & u \\ d_\ell & \leq & Kz & \leq & d_u \end{array} \tag{5.4.15b}$$

where $H$ and $K \in \mathbb{R}^{m_K \times n}$ are assumed to be constant throughout the sequence of problems. Thus only $g$ in (5.4.15a) and the constraints $\ell$, $u$, $d_\ell$ and $d_u$ in (5.4.15b) change from problem to problem. It is in this sense that the problems in the sequence are close to each other.

Below we will, for each of the problem sets on which we test our warmstarting strategy, explain which MPC-problem gave rise to the sequence of problems in question.

| P | $\mathcal{I}^{\text{cold}}$ | $\mathcal{I}^{\text{warm}}$ | $\mathcal{R}$ |
|---|---|---|---|
| 1 | 10 | 10 | 1.00 |
| 2 | 10 | 4 | 0.40 |
| 3 | 10 | 4 | 0.40 |
| 4 | 10 | 4 | 0.40 |
| . | . | . | . |
| 18 | 13 | 5 | 0.38 |
| 19 | 13 | 5 | 0.38 |
| 20 | 13 | 5 | 0.38 |
| 21 | 13 | 5 | 0.38 |
| 22 | 11 | 5 | 0.45 |
| . | . | . | . |
| 98 | 10 | 2 | 0.20 |
| 99 | 10 | 3 | 0.30 |
| 100 | 10 | 3 | 0.30 |
| 101 | 10 | 2 | 0.20 |
| $\mathcal{G}$ | 10.4 | 3.7 | **0.35** |

**Table 5.3:** *Problem 1: Hanging Chain*

| P | $\mathcal{I}^{\text{cold}}$ | $\mathcal{I}^{\text{warm}}$ | $\mathcal{R}$ |
|---|---|---|---|
| 1 | 11 | 11 | 1.00 |
| 2 | 12 | 5 | 0.42 |
| 3 | 12 | 7 | 0.58 |
| 4 | 12 | 6 | 0.50 |
| . | . | . | . |
| 18 | 11 | 7 | 0.64 |
| 19 | 11 | 7 | 0.64 |
| 20 | 11 | 9 | 0.82 |
| 21 | 11 | 9 | 0.82 |
| 22 | 11 | 7 | 0.64 |
| . | . | . | . |
| 98 | 12 | 2 | 0.17 |
| 99 | 12 | 2 | 0.17 |
| 100 | 11 | 2 | 0.18 |
| 101 | 13 | 2 | 0.15 |
| $\mathcal{G}$ | 11.8 | 3.4 | **0.29** |

**Table 5.4:** *Problem 2: Hanging Chain +State Constraints*

**Problems 1 and 2.** We include here a detailed description of one of the problems sets to give an idea of the type of problems found in [46]. We quote from [46]: "This test problem aims at regulating a chain of nine masses connected by springs into a certain steady-state. One end of the chain is fixed on a wall while the three velocity components of the other end are used as control inputs with fixed lower and upper bounds. The prediction horizon of 16 seconds is divided into 80 control intervals. The model equations are derived from linearisation of the nonlinear ODE model (with 57 states) at the steady-state. Deviation from the steady-state, the velocities of all masses and the control action are penalised via the objective function. In order to obtain the QP series we simulated in a closed-loop manner integrating the nonlinear ODE system to obtain the movements of the chain. Starting at the steady-state, a strong perturbation was exerted to the chain by moving the free end with a given constant velocity for 3 seconds. Then the MPC controller took over and tried to return the chain into its original steady-state."

Problem 2 is the same as Problem 1 with the addition of state constraints that prevent the chain from hitting the vertical wall. The original reference for these problems is [57].

The results of our experiments are seen in tables 5.3 and 5.4. We see from the tables that the work needed to solve the entire series of problems is reduced by about 70%. Hence, problems that usually demand around 10–13 iterations to solve, need just 2–4 iterations when we warmstart. This is clearly a very significant reduction in computational effort.

| P | $\mathcal{I}^{\text{cold}}$ | $\mathcal{I}^{\text{warm}}$ | $\mathcal{R}$ |
|---|---|---|---|
| 1 | 12 | 12 | 1.00 |
| 2 | 11 | 4 | 0.36 |
| 3 | 11 | 5 | 0.45 |
| 4 | 10 | 3 | 0.30 |
| . | . | . | . |
| 101 | 9 | 3 | 0.33 |
| 102 | 10 | 4 | 0.40 |
| 103 | 11 | 4 | 0.36 |
| 104 | 10 | 3 | 0.30 |
| 105 | 11 | 3 | 0.27 |
| . | . | . | . |
| 597 | 13 | 3 | 0.23 |
| 598 | 11 | 3 | 0.27 |
| 599 | 10 | 4 | 0.40 |
| 600 | 9 | 4 | 0.44 |
| $\mathcal{G}$ | 10.6 | 3.7 | **0.34** |

**Table 5.5:** *Problem 3: Diesel Engine*

| P | $\mathcal{I}^{\text{cold}}$ | $\mathcal{I}^{\text{warm}}$ | $\mathcal{R}$ |
|---|---|---|---|
| 1 | 16 | 16 | 1.00 |
| 2 | 16 | 4 | 0.25 |
| 3 | 16 | 4 | 0.25 |
| 4 | 16 | 4 | 0.25 |
| . | . | . | . |
| 155 | 16 | 4 | 0.25 |
| 156 | 17 | 4 | 0.24 |
| 157 | 16 | 6 | 0.38 |
| 158 | 16 | 4 | 0.25 |
| 159 | 16 | 4 | 0.25 |
| . | . | . | . |
| 918 | 16 | 4 | 0.25 |
| 919 | 16 | 4 | 0.25 |
| 920 | 16 | 4 | 0.25 |
| 921 | 16 | 4 | 0.25 |
| $\mathcal{G}$ | 15.8 | 4.4 | **0.28** |

**Table 5.6:** *Problem 4: Crane*

**Problem 3.** This problem set stems from the simulation of a controller governing a turbo charged direct injection diesel engine. This is a time-critical situation so this is an example of a problem where warmstarting might be particularly useful. Faster computation times imply that higher re-optimization frequency can be applied resulting in a more efficient operation of the engine. The original source is [17].

The results are shown in table 5.5. We see again a clearly significant work reduction, this time about 65%.

**Problem 4.** This problem deals with the optimal load movement of a boom crane. The variables are the piecewise constant highest derivative of the load position reference trajectory and slack variables for some of the inequality constraints. The constraints are load position and acceleration bounds and approximate position-dependent load velocity restrictions. The original source is [6].

The results are shown in table 5.6. For this problem we see reductions in work at about 70%.

**Problem 5.** In this problem, a crude distillation unit (CDU) is modelled. We leave out the details of the model, but simply refer the reader to the original reference [47]. This problem as well as the sequence as a whole was by far the

| P | $\mathcal{I}^{\text{cold}}$ | $\mathcal{I}^{\text{warm}}$ | $\mathcal{R}$ |
|---|---|---|---|
| 1 | 12 | 12 | 1.00 |
| 2 | 12 | 2 | 0.17 |
| 3 | 12 | 2 | 0.17 |
| 4 | 12 | 2 | 0.17 |
| . | . | . | . |
| 14 | 12 | 2 | 0.17 |
| 15 | 12 | 2 | 0.17 |
| 16 | 12 | 2 | 0.17 |
| 17 | 12 | 2 | 0.17 |
| 18 | 12 | 2 | 0.17 |
| . | . | . | . |
| 77 | 12 | 2 | 0.17 |
| 78 | 12 | 2 | 0.17 |
| 79 | 12 | 2 | 0.17 |
| 80 | 12 | 2 | 0.17 |
| $\mathcal{G}$ | 12.0 | 2.0 | **0.17** |

**Table 5.7:** *Problem 5: Crude Distillation Unit*

largest of all the five test sets. Each problem had 800 variables, 800 inequality constraints and no simple bounds. There were 7201 problems in the set, but to keep total computation times limited, we solved only the first 80 problems. We did, however, solve other sub-sequences of 80 problems from the series and got roughly the same results. We therefore see no reason that the warmstarting strategy should perform any differently on average had we solved the entire sequence. The results are shown in table 5.7. Again we see significant work reductions, this time on the order of 83%, always reducing work to just 2 interior-point iterations!

**The parameter $\lambda$.** The warm starting point (5.4.9) depends on the parameter $\lambda$ which we are free to choose in the interval $[0, 1]$. A smaller value of $\lambda$ is more conservative since then the warm point is closer to the standard (cold) starting point. Larger values of $\lambda$ are more aggresive, but might also be closer to the boundary of the feasible region of $\hat{\mathcal{P}}$. Hence, one might suspect that larger values of $\lambda$ would lead to more blocking and potentially a slow or even failed run.

For problems 1 and 2 of section 5.4.6, we used $\lambda = 0.99$ and $\lambda = 0.95$ respectively. For problems 3 and 4 of the same section we used $\lambda = 0.99$ and $\lambda = 0.75$. This smaller value of $\lambda$ was used simply as it was more effective.

It is clear that our strategy calls for some adaptive way of choosing $\lambda$. We have experimented with the following simple heuristic to adjust $\lambda$ adaptively during the solution of an entire sequence of problems: We start with $\lambda = 0.99$. If a problem is encountered where the warmstarted run fails, we set $\lambda := \eta\lambda$, where

| Problem Set | $n$ | $m_B$ | $\mathcal{G}_{\mathcal{I}}^{\text{cold}}$ | $\mathcal{G}_{\mathcal{I}}^{\text{warm}}$ | $\mathcal{G}_{\mathcal{R}}$ |
|---|---|---|---|---|---|
| LP-EV | 1160 | 292 | 12.2 | 9.3 | **0.76** |
| QP–Hanging Chain | 240 | 0 | 10.4 | 3.7 | **0.35** |
| QP–Hanging Chain +SC | 240 | 709 | 11.8 | 3.4 | **0.29** |
| QP–Diesel Engine | 20 | 20 | 10.6 | 3.7 | **0.34** |
| QP–Crane | 57 | 160 | 15.8 | 4.4 | **0.28** |
| QP–Crude Distillation Unit | 800 | 800 | 12.0 | 2.0 | **0.17** |

**Table 5.8:** *Overview of results from all test sets. $n$ denotes the number of variables, $m_B$ the number of inequality constraints.*

$\eta < 1$, e.g. $\eta = 0.9$. If, on the other hand, $K$ consecutive problems were solved successfully using warmstarting, we set $\lambda := \min\{0.99, \eta^{-1}\lambda\}$. For example, $K = 10$. This method seems to work quite well in practice for the problems we have tested. Indeed it improves the results for the crane-problem (problem 4 of section 5.4.6) to $\mathcal{G}_{\mathcal{R}} = 0.24$, but — (a) since the above method obviously should be refined and analyzed, (b) to emphasize the importance of $\lambda$ and (c) for simplicity — we used static values of $\lambda$ in our experiments.

### 5.4.7 Conclusion

In the previous sections, we demonstrated how to apply the warmstarting strategy of [50] to sequences of LPs and QPs arising from different MPC applications. Numerous numerical experiments have shown that the strategy indeed is efficient and useful for real problems. See Table 5.8 for an overview of all the results.

Appendix A

# Paper: A Homogeneous Interior-Point Algorithm for Nonsymmetric Convex Conic Optimization

**Mathematical Programming manuscript No.**
(will be inserted by the editor)

# A Homogeneous Interior-Point Algorithm for Nonsymmetric Convex Conic Optimization

**Anders Skajaa · Yinyu Ye**

*Accepted for publication in Mathematical Programming on March 26th, 2014.*

Received: date / Accepted: date

**Abstract** A homogeneous interior-point algorithm for solving nonsymmetric convex conic optimization problems is presented. Starting each iteration from the vicinity of the central path, the method steps in the approximate tangent direction and then applies a correction phase to locate the next well-centered primal-dual point. Features of the algorithm include that it makes use only of the primal barrier function, that it is able to detect infeasibilities in the problem and that no phase-I method is needed. We prove convergence to $\epsilon$-accuracy in $\mathcal{O}(\sqrt{\nu}\log(1/\epsilon))$ iterations. To improve performance, the algorithm employs a new Runge-Kutta type second order search direction suitable for the general nonsymmetric conic problem. Moreover, quasi-Newton updating is used to reduce the number of factorizations needed, implemented so that data sparsity can still be exploited. Extensive and promising computational results are presented for the $p$-cone problem, the facility location problem, entropy maximization problems and geometric programs; all formulated as nonsymmetric convex conic optimization problems.

**Keywords** Convex Optimization · Nonsymmetric Conic Optimization · Homogeneous Self-dual Model · Interior-point Algorithm

Anders Skajaa
Department of Informatics and Mathematical Modelling
Technical University of Denmark, DK-2800, Kgs. Lyngby, Denmark.
E-mail: andsk@dtu.dk

Yinyu Ye
Department of Management Science and Engineering
Stanford University, CA 94305-4121, USA.
E-mail: yinyu-ye@stanford.edu

## 1 Introduction

This paper is concerned with conic optimization problem pairs of the form

$$\text{PRIMAL} \begin{cases} \min_x \ c^T x \\ \text{s.t.} \quad Ax = b \\ \qquad\ \ x \in \mathcal{K} \end{cases} \qquad \text{DUAL} \begin{cases} \max_{y,s} \ b^T y \\ \text{s.t.} \quad A^T y + s = c \\ \qquad\ \ s \in \mathcal{K}^*, \ y \in \mathbb{R}^m \end{cases} \tag{PD}$$

where $x, c \in \mathbb{R}^n, A \in \mathbb{R}^{m\times n}, b \in \mathbb{R}^m$, $\mathcal{K} \subset \mathbb{R}^n$ is a proper cone (i.e. it is convex, pointed, closed and has non-empty interior) and $\mathcal{K}^* = \{s \in \mathbb{R}^n : s^T x \geq 0, \ \forall x \in \mathcal{K}\}$ is its dual cone, which is also proper. We are further assuming that $m \leq n$ and that $\text{rank}(A) = m$.

If $\mathcal{K}$ is the positive orthant $\mathbb{R}^n_+$, then (PD) is a linear programming (LP) problem in standard form and its dual. Solution methods for LP have been studied for long in different settings and until the emergence of interior-point methods (IPMs), the most prominent method was the simplex method, developed by Dantzig in the 1940s. The introduction of IPMs is usually ascribed to Karmarkar [10] in 1984 and since then, research in the area has been extensive.

In [17], it was studied how to extend the ideas of IPMs to the nonlinear case. If $\mathcal{K}$ admits a *self-scaled* barrier function $F$: $\mathcal{K}^\circ \mapsto \mathbb{R}$, problems of the type (PD) are efficiently solvable using long-step symmetric primal-dual IPMs [18,19]. The practical efficiency of these algorithms has been widely verified, see e.g. [1,2,25].

In [9], Güler demonstrated that self-scaled cones are identical to those that are *symmetric*; a class that comprises just five cones of which only two are interesting for optimization. These cones are the Lorentz cone (leading to quadratic cone programming which generalizes quadratic programming and second order cone programming) and the positive semidefinite cone (leading to semidefinite programming). Notice that linear programming is a subset of semidefinite programming.

Although these two self-scaled cones allow for modelling of a great variety of constraints [4], many important types of constraints do not fall in this class. Examples include entropy type constraints: $x \log x \leq t$, $p$-cone constraints: $\|x\|_p \leq t$, and constraints arising in geometric programming [5]. Some of these constraints can be modelled using self-scaled cones, but this usually requires the introduction of many extra variables and constraints [4].

Theoretically, one can solve problems involving *any* convex constraint using a purely primal short-step barrier method and still obtain an algorithm with the best-known worst-case computational complexity. Such an algorithm is known to be practically inefficient compared to a long-step primal-dual IPM. Other approaches are also possible and special algorithms for certain subclasses of problems exist [27,30]. Another approach known to be effective for general convex problems is to solve the monotone complementarity problem, see for example [3].

It may be beneficial to model nonsymmetric constraints more directly using *non*-self-scaled cones (nonsymmetric cones) such as the power cone or the exponential cone. This approach was employed by Nesterov in [16]. He proposed

a method that mimics the ideas of a long-step primal-dual IPM for symmetric cones by splitting each iteration into two phases. First, a pure primal correction phase is used to find a primal central point $x$ and a scaling point $w$. These points are used to compute a feasible dual point $s$ such that an *exact* scaling relation is satisfied: $s = \nabla^2 F(w)x$. Second, a truly symmetric primal-dual step in the approximate tangent direction is taken (a prediction step). This algorithm, however, assumes the existence of a strictly feasible primal-dual point and *requires* a strictly feasible initial primal point to start.

If knowledge of both the primal and the dual barrier function, their gradients and Hessians is assumed, truly primal-dual symmetric search directions can be constructed. This approach was used in [20] to solve a homogeneous model of the general convex conic problem (PD). This leads to a method with some desirable properties but at the same time it has two crucial disadvantages: Firstly, the linear systems that must be solved in each iteration are twice the size compared to algorithms for self-scaled cones therefore increasing total computation time by a factor of $2^3 = 8$ for problems of equal dimension. Secondly, it can be difficult or impossible to find an expression for the dual barrier and its derivatives. The so-called doubly non-negative cone[1] is an example of the latter situation. A simple barrier for the primal cone is known, but no explicit barrier function for the dual cone is known.

Building on the algorithms of [16] and [20], we present in this paper a primal-dual interior-point algorithm for a homogeneous model of (PD). This approach has proven successful for self-scaled cones [29, 2, 24] because it implies several desirable properties, among which are the ability to detect infeasibility in the problem pair and the ease of finding a suitable starting point, eliminating the need for a phase-I method. Unlike the algorithm in [20], our algorithm uses only the primal barrier function and therefore our linear systems are no larger than those appearing in IPMs for self-scaled cones.

In addition to the advantages induced by using a homogeneous model, we suggest the following improvements to reduce computational load. The Mehotra second order correction [12] is known to significantly improve practical performance of IPMs for linear and quadratic conic problems [12, 2, 25]. With the same goal in mind, we suggest a new way to compute a search direction containing second order information for the general (possibly non-self-scaled) conic problem. This search direction is inspired by Runge-Kutta methods for ordinary differential equations. Further, we employ BFGS-updating of the Hessian of the barrier function to reduce the number of full matrix factorizations needed. It is shown how this can be done in a way retaining the possibility to exploit sparsity in $A$.

We will assume that $\mathcal{K}$ is a proper cone and that a logarithmically homogeneous self-concordant barrier function $F$ for $\mathcal{K}$, its gradient $\nabla F$ and its Hessian $\nabla^2 F$ are available and can be efficiently computed for all $x$ in the interior of $\mathcal{K}$. The definition of the barrier parameter $\nu$ of $F$ and many of the useful properties of these functions are listed in appendix A.

[1] A positive semidefinite matrix with all non-negative entries is called *doubly non-negative*.

This paper is organized in two main parts. In the first, which consists of Sections 2 through 4, we discuss theoretical issues, present our algorithm and prove that the method converges in $\mathcal{O}(\sqrt{\nu}\log(1/\epsilon))$ iterations. We state all theoretical results in the main text, emphasizing asymptotic complexity behavior, but divert all proofs to the appendix to keep the main text clean and free of technical details. Sections 5 and 6 make up the second part. Here, we present and discuss details related to the implementation of our algorithm. We introduce heuristic methods to increase convergence speed and then present an extensive series of computational results substantiating the effectiveness and practical applicability of our algorithm. We finally draw conclusions in Section 7.

## 2 Homogeneous and self-dual model

If there exist $x \in \mathcal{K}^\circ$ such that $Ax = b$ and $s \in (\mathcal{K}^*)^\circ, y \in \mathbb{R}^m$ such that $A^T y + s = c$, then strong duality holds for the primal-dual problem pair (PD). In this case, any primal optimal $x$ and dual optimal $(y, s)$ must satisfy

$$\begin{aligned} Ax - b &= 0 \\ -A^T y - s + c &= 0 \\ x^T s &= 0 \\ x \in \mathcal{K},\ s \in \mathcal{K}^*,\ y &\in \mathbb{R}^m. \end{aligned} \tag{1}$$

We propose solving a homogeneous model of problems (PD). We therefore introduce two extra non-negative scalar variables $\tau$ and $\kappa$ and seek to find $x, \tau, y, s, \kappa$ that solve the following problem:

$$\left.\begin{array}{lrrrl} \text{minimize} & 0 & & & \\ \text{subject to} & & Ax & -b\tau & = 0 \\ & -A^T y & & +c\tau\ -s & = 0 \\ & b^T y & -c^T x & & -\kappa = 0 \\ & \multicolumn{4}{r}{(x,\tau) \in \mathcal{K} \times \mathbb{R}_+,\ \ (s,\kappa) \in \mathcal{K}^* \times \mathbb{R}_+,\ \ y \in \mathbb{R}^m.} \end{array}\right\} \tag{HSD}$$

The motivation for doing so is summarized in the following.

**Lemma 1** *Assume $(x, \tau, y, s, \kappa)$ solves* (HSD). *Then*

1. $(x, \tau, y, s, \kappa)$ *is* complementary. *That is:* $x^T s + \tau\kappa = 0$.
2. *If $\tau > 0$ then $(x, y, s)/\tau$ is optimal for* (PD).
3. *If $\kappa > 0$ then one or both of $b^T y > 0$ and $c^T x < 0$ hold. If the first holds, then* (PD) *is primal-infeasible. If the second holds, then* (PD) *is dual-infeasible.*

*Proof* See appendix B.1.

Lemma 1 shows that any solution to (HSD) with $\tau + \kappa > 0$ provides either an optimal solution to our original problems (PD) or a certificate of infeasibility of (one of) the original problems.

Any useful algorithm aimed at solving (HSD) must therefore at least have the following two properties: If (PD) is both primal and dual feasible and has zero duality gap, the algorithm must produce a solution (or approximate solution) to (HSD) with $\tau > 0$. Conversely, if (PD) is either primal or dual infeasible, the algorithm must produce a solution (or approximate solution) to (HSD) with $\kappa > 0$. In [11], it is thoroughly demonstrated which properties are sufficient for an algorithm to meet these goals. The algorithm that we later present in Section 4 indeed has these properties. However, since this paper concentrates on algorithmic aspects we omit the details regarding the various primal and dual infeasibility cases and what they mean for the homogeneous model and algorithm. Instead, we refer the reader to [20] and particularly [11] for a much more detailed discussion in this direction.

There is another desirable feature of the homogeneous model (HSD):

**Lemma 2** *The optimization problem* (HSD) *is self-dual.*

*Proof* See appendix B.2.

Lemma 2 implies that we can apply a primal-dual interior-point algorithm to the problem (HSD) without doubling the dimension of the problem. Specifically, there is no need to handle and store variables from the dual of (HSD) since they are identical to those of the primal.

Given the two lemmas above and the comments following Lemma 1, we can state the following desirable consequences of solving the homogeneous model (HSD) with our algorithm to be presented later:

- If the original problem (PD) is primal and dual feasible and has zero duality gap, an optimal primal-dual solution is found and a certificate of optimality is produced.
- If the original primal-dual pair (PD) is primal or dual infeasible, a certificate of this infeasibility is produced.
- The dimension of the problem is not essentially larger than that of the original primal-dual pair (PD) and does not require more computational effort to solve.
- The algorithm can be initialized in a point not necessarily feasible w.r.t. the linear constraints of (HSD).

When $\mathcal{K}$ is not equal to $\mathbb{R}^+_n$, it is possible that the pair (PD) is primal and dual feasible but has strictly positive duality gap. In this situation, a tiny perturbation to the problem data exists such that the perturbed problem has a solution with $\tau + \kappa > 0$. Thus, the problem is ill-posed. See [4] for further discussion and examples of this exceptional case.

## 3 Nonsymmetric path following

Path following methods are usually motivated by considering a family of barrier problems parametrized by $\mu > 0$:

$$\min_x \; c^T x + \mu F(x), \quad \text{s.t. } Ax = b, \;\; x \in \mathcal{K}^\circ \tag{2}$$

where $F$ again is a logarithmically homogeneous self-concordant barrier function[2] (LHSCB) with barrier parameter $\nu$. The Karush-Kuhn-Tucker (KKT) conditions of problem (2) are: If $x \in \mathcal{K}^\circ$ is optimal for (2), then there exist $s \in (\mathcal{K}^*)^\circ$ and $y \in \mathbb{R}^m$ so that

$$\begin{aligned} Ax - b &= 0 \\ -A^T y - s + c &= 0 \\ s + \mu \nabla F(x) &= 0 \\ x \in \mathcal{K},\ s \in \mathcal{K}^*,\ y &\in \mathbb{R}^m \end{aligned} \tag{3}$$

The points that satisfy (3) are known as the primal-dual *central path*. Let us denote a point in this set by $u(\mu) = (x(\mu), y(\mu), s(\mu))$. Using relation (20) from Appendix A, it is easy to see that central path points satisfy $c^T x(\mu) - b^T y(\mu) = x(\mu)^T s(\mu) = \nu\mu$. The idea of a path-following method is to loosely track $u(\mu)$ towards $u(0)$, thus obtaining a point eventually being approximately optimal for (PD), compare (3) to (1).

Experience shows that it is most efficient to take steps that are combinations of two directions: 1. The direction approximately tangent to the central path (the predictor direction), that is, the direction $u'(\mu)$ and 2. the direction pointing towards the central path as the current iterate may not be exactly on the central path. This correction direction is the Newton step for the equations (3), we will denote it $p(\mu)$.

If the iterate is not exactly on the central path, the search direction $u'(\mu)$ can still be computed so that it is symmetric. Here *symmetric* refers to the search direction (and thus the iterates) being *the same* regardless of whether the roles of the primal and dual problems in (PD) are interchanged [26]. Thus no particular emphasis is put on either the primal or the dual problem, which is a desirable feature of an algorithm. If $\mathcal{K}$ is self-scaled, a symmetric $u'(\mu)$ can be computed using a *scaling point* [18,19]. If the cone is not self-scaled (nonsymmetric), a symmetric $u'(\mu)$ can be computed by using both the Hessian of the primal and the dual barrier. As discussed in the introduction, this, however, leads to an algorithm that must solve linear systems double the size of those occurring in a symmetric IPM. A further disadvantage is that one must be able to compute both the primal and the dual barriers, their gradients and Hessians, which may prove difficult.

Nesterov showed in [16] that a scaling point determined during an iterative centering (correction) procedure can be used to compute a symmetric search direction $u'(\mu)$. Let us briefly describe the concept underlying the algorithm from [16]. The following *proximity measure* is needed:

$$\Psi(x, y, s) = F(x) + F^*(s) + \nu \ln \frac{x^T s}{\nu} + \nu$$

which is $\geq 0$ and $= 0$ only if $(x, y, s)$ is on the central path. Here, $F^*$ denotes the dual barrier of $F$, see appendix A for properties of these two functions.

The general algorithm can then be outlined as below. Assume we start with an initial point $(x, y, s) \in \mathcal{K} \times \mathbb{R}^m \times \mathcal{K}^*$ with $\Psi(x, y, s) < \eta$. Then

[2] See Appendix A for a list of properties of this class of functions.

**Repeat**

1. $(x, y, s) := (x, y, s) + \alpha u'(\mu)$
   $\mu = x^T s/\mu$.
2. **while** $\Psi(x, y, s) > \eta$
   $(x, y, s) := (x, y, s) + \hat{\alpha} p(\mu)$
   **end while**

where $\alpha$ in step 1 is chosen so that $\Psi(x, y, s) < \beta$ after step 1 and $\hat{\alpha}$ is chosen to be $\lambda/(1 + \lambda)$, where $\lambda$ is the Newton decrement. In [16], it is proved that with appropriate choices of $\eta$, $\beta$ and $\alpha$, the above algorithm converges in $\mathcal{O}(\sqrt{\nu} \log(1/\epsilon))$ iterations. This method uses only the Hessian $\nabla^2 F(\cdot)$ of the primal barrier but still the *value* of the dual barrier $F^*(\cdot)$. Two serious practical drawbacks of the method are that it assumes that the original problems are strictly feasible and that it *requires* a strictly feasible initial primal point to start therefore needing a phase-I method.

The approach of [20] is instead to compute a symmetric $u'(\mu)$ by using both the Hessian of the primal and the dual barriers. Again, the iteration complexity result $\mathcal{O}(\sqrt{\nu} \log(1/\epsilon))$ is obtained and the two practical drawbacks of [16] are alleviated by the use of a homogeneous model. Two major disadvantages of the method of [20] are that one must know (or be able to compute) Hessians of both barriers and that the linear systems that must be solved are double in size.

Our goal in this paper is to construct an efficient algorithm utilizing the main ideas of [16] and [20], but adapted to be efficient for the homogeneous model (HSD) *without* using the Hessians of the primal *and* the dual barrier. In fact, our method does not make *any* use of the dual barrier — not even the function value. Unlike [16] and [20], our prediction direction $u'(\mu)$ will *not* be exactly symmetric unless the iterate is exactly on the central path, which is rarely the case. However, it turns out to be sufficient that we ensure that the iterate is "close to" the central path. This will guarantee a high enough quality of the prediction direction. In exchange for dropping the "exact symmetric" tangent direction we obtain a method that does not suffer from any of the above mentioned drawbacks of the methods in either of [16] and [20] while still maintaining the $\mathcal{O}(\sqrt{\nu} \log(1/\epsilon))$ iteration complexity result.

Thus, compared to [20], this work represents the following improvements:

1. We need only to know the primal barrier function, its gradient and Hessian (no need for the dual barrier and its derivatives)
2. The linear systems that need to be solved in each iteration are half the dimension (i.e. a factor 8 faster in terms of computation time).

Likewise, in relation to [16], this work represent the following improvements:

1. We need only to know the primal barrier function, its gradient and Hessian (no need for the dual barrier function value)

2. We do not require a feasible starting point (no phase-I method needed)
3. Our method detects infeasibilities in the problem.

We are also aware of the apparent gap between IPM complexity theory and state-of-the-art implementations, see e.g. the introduction of [16] for a discussion about this issue in the case of convex conic programming. In the realm of interior-point algorithms, it is often the case in practice that methods with inferior complexity estimates convincingly outperform algorithms with best-known complexity estimates. See e.g. [1,25] for implementations of such fast algorithms for the case of self-scaled cones. Furthermore, in industry-standard software, heuristic techniques to speed up convergence rates are often employed, although they invalidate the proofs of convergence in the purely theoretical sense. A standard example of such a practice is PDIPMs for linear programming in which it is common to use different primal and dual step lengths. Since a similar discrepancy between theory and practice might be present for the case of a nonsymmetric cone, we expect to be able to improve the performance of our algorithm by employing techniques similar to those used to accelerate the fastest PDIPMs for self-scaled problems.

## 4 Homogeneous algorithm

### 4.1 Notation

To simplify notation, we will make use of the following notation. For the concatenation of two vectors $v$ and $w$, we will sometimes use the MATLAB-inspired notation $(v; w)$ and otherwise the usual $\begin{pmatrix} v \\ w \end{pmatrix}$. We will further simplify notation by writing

$$\bar{x} = \begin{pmatrix} x \\ \tau \end{pmatrix} = (x; \tau), \qquad \bar{s} = \begin{pmatrix} s \\ \kappa \end{pmatrix} = (s; \kappa)$$
$$\bar{F}(\bar{x}) = F(x) - \log \tau, \qquad \bar{F}^*(\bar{s}) = F^*(s) - \log \kappa$$

and

$$\bar{\mathcal{K}} = \mathcal{K} \times \mathbb{R}_+, \qquad \bar{\mathcal{K}^*} = \mathcal{K}^* \times \mathbb{R}_+, \qquad \bar{\nu} = \nu + 1$$

This notation is consistent with that of [20]. Notice that $\bar{F}$ and $\bar{F}^*$ are logarithmically homogeneous self-concordant barrier functions for the cones $\bar{\mathcal{K}}$ and $\bar{\mathcal{K}^*}$ respectively.

We will also use a higher level of aggregation: $z = (\bar{x}; y; \bar{s}) = (x; \tau; y; s; \kappa) \in \mathcal{F} := \bar{\mathcal{K}} \times \mathbb{R}^m \times \bar{\mathcal{K}^*}$ and define the *complementarity gap* of $z$ by $\mu(z) := (\bar{x}^T \bar{s})/\bar{\nu}$. We will write $g_{\bar{x}} = \nabla \bar{F}(\bar{x})$ and $H_{\bar{x}} = \nabla^2 \bar{F}(\bar{x})$ and make use of the following local norms:

$$\|u\|_{\bar{x}} = \|H_{\bar{x}}^{1/2} u\|, \;\; \|v\|_{\bar{x}}^* = \|H_{\bar{x}}^{-1/2} v\|, \quad \text{for } u \in \bar{\mathcal{K}} \text{ and } v \in \bar{\mathcal{K}^*},$$

where $\|\cdot\|$ denotes the standard Euclidean norm. See also appendix A for more properties of these local norms. In our new notation, we can write the homogeneous model simply as

$$G\begin{pmatrix} y \\ \bar{x} \end{pmatrix} - \begin{pmatrix} 0 \\ \bar{s} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad z = \begin{pmatrix} \bar{x} \\ y \\ \bar{s} \end{pmatrix} \in \mathcal{F} \tag{4}$$

where $G$ is the skew-symmetric matrix

$$G := \begin{pmatrix} 0 & A & -b \\ -A^T & 0 & c \\ b^T & -c^T & 0 \end{pmatrix}. \tag{5}$$

Equations such as (4) will usually be written as $G(y;\bar{x}) - (0;\bar{s}) = (0;0)$ to save vertical space. Notice that the expression $G(y;\bar{x})$ involves a *multiplication* between $G$ and $(y;\bar{x})$ and the parenthesis thus do not denote arguments to a function. This latter situation will be clear from the context.

### 4.2 The central path in the homogeneous model

For $\bar{x} \in \bar{\mathcal{K}}$, $\bar{s} \in \bar{\mathcal{K}}^*$ and a scalar $t$, let us define the function

$$\psi(\bar{x}, \bar{s}, t) := \bar{s} + t g_{\bar{x}}. \tag{6}$$

We initialize our algorithm in $z^0 \in \mathcal{F}$. Denote $\mu^0 = \mu(z^0)$. Parametrized by $\gamma \in [0,1]$, we define the central path of the homogenized problem (4) by the points $z_\gamma$ that satisfy

$$G(y_\gamma;\bar{x}_\gamma) - (0,\bar{s}_\gamma) = \gamma\left(G(y^0;\bar{x}^0) - (0;\bar{s}^0)\right) \tag{7}$$

$$\psi(\bar{x}_\gamma, \bar{s}_\gamma, \gamma\mu^0) = 0 \tag{8}$$

In the homogeneous model, the central path connects the point $z^0$ (at $\gamma = 1$) with a solution of the problem (4) as $\gamma \to 0$. Therefore, the main idea of the algorithm, as in other path-following algorithms, is to approximately track the central path towards a solution.

For a fixed parameter $\eta \in [0,1]$, we define the set

$$\mathcal{N}(\eta) = \{z = (\bar{x}; y; \bar{s}) \in \mathcal{F} : \ \|\psi(\bar{x}, \bar{s}, \mu(z))\|_{\bar{x}}^* \le \eta\mu(z)\} \tag{9}$$

which, in view of (8), can be considered a neighborhood of the feasible central path — that is, the path that would arise from using $z^0$ in (7)–(8) such that $G(y^0;\bar{x}^0) - (0;\bar{s}^0) = 0$.

In the case of LP with the usual barrier $F(x) = -\sum_j \log x_j$, we remark that equation (8) is the same as the familiar $\bar{X}\bar{s} = \gamma\mu^0 e$ where $\bar{X} = \mathrm{diag}(\bar{x})$ and $e = (1,\ldots,1)$. Similarly, the inequality in (9) reduces to $\|\bar{X}\bar{s} - \mu e\| \le \eta\mu(z)$.

### 4.3 Prediction

The direction $d_z$ tangent to the central path (also called the predictor direction) is determined by differentiating (7)–(8) with respect to $\gamma$. For equation (8), this yields

$$d_{\bar{s}_\gamma} = -\mu^0 g_{\bar{x}_\gamma} - \gamma\mu^0 H_{\bar{x}_\gamma} d_{\bar{x}_\gamma}$$

where $d_{\bar{x}_\gamma}$ denotes $\bar{x}_\gamma$ differentiated w.r.t. $\gamma$ and similarly for other variables. By (8), we have $\gamma^{-1}\bar{s}_\gamma = -\mu^0 g_{\bar{x}_\gamma}$, which we insert and get

$$d_{\bar{s}_\gamma} + \gamma\mu^0 H_{\bar{x}_\gamma} d_{\bar{x}_\gamma} = \gamma^{-1}\bar{s}_\gamma.$$

The same operation on (7) gives the equations defining the direction $d_z$:

$$G(d_y; d_{\bar{x}}) - (0; d_{\bar{s}}) = -\left(G(y;\bar{x}) - (0;\bar{s})\right) \tag{10}$$

$$d_{\bar{s}} + \mu(z) H_{\bar{x}} d_{\bar{x}} = -\bar{s} \tag{11}$$

where we have dropped the argument $\gamma$ for readability and put $\mu(z)/\mu^0 = \gamma$. Notice also that we have rescaled the equations by $-\gamma$ to make the notation consistent with the general IPM literature. Determining the direction $d_z$ thus amounts to solving the system of linear equations (10)–(11).

In the rest of this section, we will use the notation

$$\begin{aligned} z^+ &= (\bar{x}^+, y^+, \bar{s}^+) = (\bar{x} + \alpha d_{\bar{x}}, y + \alpha d_y, \bar{s} + \alpha d_{\bar{s}}) = z + \alpha d_z \\ \psi &= \psi(\bar{x}, \bar{s}, \mu(z)) \\ \psi^+ &= \psi(\bar{x}^+, \bar{s}^+, \mu(z^+)) \\ d_z &= \text{solution of (10)–(11).} \end{aligned}$$

The next lemma explains how the linear residuals and the complementarity gap are reduced along the predictor direction.

**Lemma 3** *The direction $d_z$ satisfies*

$$\begin{aligned} G(y^+; \bar{x}^+) - (0; \bar{s}^+) &= (1-\alpha)\left(G(y;\bar{x}) - (0;\bar{s})\right) \\ \mu(z^+) &= (1-\alpha)\mu(z) + (1-\alpha)\alpha\nu^{-1}\psi^T d_{\bar{x}}. \end{aligned}$$

*Proof* See appendix C.1.

The first relation shows that the linear residuals are reduced by the factor $1-\alpha$ along the direction $d_z$. The complementarity gap $\mu$ is reduced in a slightly more complicated way depending on the vector $\psi$. If $z$ is precisely on the central path, $\psi = 0$, so $\mu(z^+) = (1-\alpha)\mu(z)$ and also the complementarity gap is reduced by the factor $1-\alpha$. As we shall see, we can, similarly to other interior-point algorithms, choose $\alpha = \Omega(1/\sqrt{\nu})$ so that $\mu(z^+) \leq (1 - \Omega(1/\sqrt{\nu}))\mu(z)$. Here, we use the "big-$\Omega$"-notation meaning that $\alpha$ is asymptotically bounded below by $1/\sqrt{\nu}$ times a positive (possibly small) constant as $\nu \to \infty$.

**Lemma 4** *Assume $z \in \mathcal{N}(\eta)$. Then we can choose $\alpha = \Omega(1/\sqrt{\nu})$ so that $\bar{x}^+ \in \bar{\mathcal{K}}$ and $\bar{s}^+ \in \bar{\mathcal{K}}^*$.*

*Proof See appendix C.3.*

**Lemma 5** *Assume $z \in \mathcal{N}(\eta)$. If $\eta \leq 1/6$, then we can choose $\alpha = \Omega(1/\sqrt{\nu})$ so that $z^+ \in \mathcal{N}(2\eta)$.*

*Proof* See appendix C.4.

### 4.4 Correction phase

Given some point $z^+ = (\bar{x}^+, y^+, \bar{s}^+) \in \mathcal{N}(2\eta)$, the goal of the correction phase is to find a new point $z = (\bar{x}, y, \bar{s})$ which is close to the central path and satisfy the same linear constraints as $z^+$. That is, we want to find $z$ so that $\|\psi(\bar{x}, \bar{s}, \mu(z))\|^*_{\bar{x}} \leq \eta\mu(z)$ and $G(y; \bar{x}) - (0; \bar{s}) = G(y^+; \bar{x}^+) - (0; \bar{s}^+)$. We therefore apply Newton's method to the equations

$$\begin{aligned} G(y; \bar{x}) - (0; \bar{s}) &= G(y^+; \bar{x}^+) - (0; \bar{s}^+) \\ \psi(\bar{x}, \bar{s}, \mu(z)) &= 0 \end{aligned} \tag{12}$$

The Newton step for these of equations is the solution $\delta_z := (\delta_{\bar{x}}, \delta_y, \delta_{\bar{s}})$ to the following linear system of equations:

$$G(\delta_y; \delta_{\bar{x}}) - (0; \delta_{\bar{s}}) = 0 \tag{13}$$

$$\delta_{\bar{s}} + \mu(z) H_{\bar{x}} \delta_{\bar{x}} = -\psi(\bar{x}, \bar{s}, \mu(z)) \tag{14}$$

We then solve (13)-(14) and starting from $z = z^+$, we apply

$$z := z + \hat{\alpha}\delta_z \tag{15}$$

repeatedly until $\|\psi(\bar{x}, \bar{s}, \mu(z))\| \leq \eta\mu(z)$.

The following Lemma shows that this process terminates quickly.

**Lemma 6** *If $\eta \leq 1/6$, then the correction process* (15) *terminates in at most two steps.*

*Proof* See appendix D.

### 4.5 Convergence and complexity of algorithm

It is evident that our algorithm is a nonsymmetric conic generalization of the simplified LP homogeneous and self-dual model [29]. Similarly to [29], let us write $\theta^{k+1} = (1 - \alpha^k)\theta^k$ and $\theta^0 = 1$, where $\alpha^k$ is the step length taken in the prediction step in the $k$'th iteration. From (13), we see that the linear residuals do not change during the correction phase. Thus, a useful result from [29] applies also to our algorithm:

**Algorithm 1** Nonsymmetric Predictor-Corrector Algorithm

**Input:** Barrier function $F$, $\eta \leq 1/6$, and initial point $z \in \mathcal{F} \cap \mathcal{N}(\eta)$.
  $\hat{\alpha} := 1/84$
  **Repeat**
    Set $\mu := \mu(z)$
    **Stopping**
      If stopping criteria satisfied: **terminate**.
    **Prediction**
      Solve (10)–(11) for $d_z$
      Choose largest $\alpha$ so that $z + \alpha d_z \in \mathcal{F} \cap \mathcal{N}(2\eta)$
      Set $z := z + \alpha d_z$ and $\mu = \mu(z)$.
    **Correction**
      Solve (13)–(14) for $\delta_z$
      Set $z := z + \hat{\alpha}\delta_z$
      Solve (13)–(14) for $\delta_z$
      Set $z := z + \hat{\alpha}\delta_z$
  **End**

**Lemma 7** *Algorithm 1 generates iterates $z^k = (x^k, \tau^k, y^k, s^k, \kappa^k)$, $k = 0, 1, \ldots$ that satisfy*

$$\bar{\nu}\left(\mu^k/\theta^k + \theta^k\mu^0\right) = (s^k)^T x^0 + (x^k)^T s^0 + \kappa^k\tau^0 + \tau^k\kappa^0 \tag{16}$$

*Proof* See Lemma 4, page 57 in [29].

This lemma implies that if $\mu^k$ and $\theta^k$ decrease at the same rate, then (16) functions as a normalizing constraint — i.e. all the iterates remain bounded. This is readily seen: The left-hand side of (16) remains bounded and since each term on the right-hand side is non-negative, each term must remain individually bounded. In particular as $\mu^k$ decreases to zero, at least one of $\tau^k$ and $\kappa^k$ will go to zero while the other will remain non-negative and bounded above.

The following theorem will now finish our analysis.

**Theorem 1** *Algorithm 1 terminates with a point $z = (\bar{x}, y, \bar{s})$ that satisfies*

$$\mu(z) \leq \epsilon\mu(z^0) \quad \textit{and} \quad \|G(y;\bar{x}) - (0;\bar{s})\| \leq \epsilon\|G(y^0;\bar{x}^0) - (0;\bar{s}^0)\|$$

*in no more than $\mathcal{O}\left(\sqrt{\nu}\log(1/\epsilon)\right)$ iterations.*

*Proof* See appendix E.

It is important to note that this theorem alone does not guarantee that we have recovered a sufficiently accurate solution (or infeasibility certificate) to the original problem (PD), only to (HSD). From the proofs of Theorem 1 and Lemma 3 it follows, however, that $\mu^k$ and $\theta^k$ decrease at the same rate. Therefore, Lemma 3 guarantees that $\tau^k$ and $\kappa^k$ both remain bounded and if the final point $z$ has one of $\tau$ or $\kappa$ large enough a sufficiently accurate solution or infeasibility certificate for (PD) has been determined. For further details regarding the exact accuracy obtained and an explanation of all different types of feasibility cases, the reader is again referred to [11,20]. For us, importance is

**Algorithm 2** Aggressive step implementation

**Input:** Barrier function $F$, $0 < \eta \leq \beta < 1$, and initial point $z \in \mathcal{F} \cap \mathcal{N}(\eta)$.
**Repeat**
Set $\mu := \mu(z)$
**Stopping**
If stopping criteria satisfied: **terminate**.
**Prediction**
Solve (10)–(11) for $d_z$
Choose largest $\alpha$ so that $z + \alpha d_z \in \mathcal{F} \cap \mathcal{N}(\beta)$
Set $z := z + \alpha d_z$ and $\mu = \mu(z)$.
**Correction**
**Repeat**
Solve (13)–(14) for $\delta_z$
Choose $\hat{\alpha}$ to approximately minimize $\|\psi\|_{\bar{x}}^*$ along $\delta_z$
Set $z := z + \hat{\alpha}\delta_z$
**Until** $z \in \mathcal{F} \cap \mathcal{N}(\eta)$.
**End**

placed on the ability to practically distinguish these cases and what we mean by "sufficiently close" is precisely stated in Section 5.4.

In this section, we have emphasized only the asymptotic behavior of our algorithm. In several places, it may be possible to improve the constants in the leading terms but as the above analysis serves only to demonstrate asymptotic worst-case behavior, this is of minor importance.

## 5 Implementation

In order for an interior-point method to be practical and competitive, the implementation must deviate somewhat from the pure theoretical algorithm. In this section, we describe how such an efficient algorithm can be implemented.

Our implementation is outlined in Algorithm 2. As is common practice in implementations of interior-point methods, we allow for a much longer prediction step, for example $\beta \geq 0.80$. This leads to faster convergence once we get close to the optimal point. Indeed we do observe what appears to be superlinear convergence in this region.

It should be noted, however, that we can no longer be certain that two correction steps will be enough to reach a sufficiently centered point. Therefore, we continue taking correction steps until the centrality condition $\|\psi\|_{\bar{x}}^* \leq \eta\mu$ is satisfied. As the computational experiments later show, for the problems we have solved, rarely more than one or two correction steps are needed. We can further reduce the cost of the correction phase by using quasi-Newton updating as we explain in the next section.

### 5.1 Quasi-Newton updating in the correction phase

Solving either for a prediction or a correction step requires the factorization of the sparse $n \times n$ matrix $H_{\bar{x}}$ and of the possibly sparse $m \times m$ matrix

$Q = AH_{\bar{x}}^{-1}A^T$. To reduce the total number of factorizations needed in the correction phase, we suggest taking $J$ quasi-Newton steps for each normal correction step.

Let us show how this can be done computationally efficient without destroying sparsity in the KKT-system, which is an essential requirement in practical applications.

Let $B$ and $M$ denote the current quasi-Newton approximation of the *inverses* of $H$ and $Q$ respectively. Conceptually, we update $B$ to $B^+$ using BFGS updating (see e.g. [22]), a rank-2 updating scheme: $B^+ = B + k^{(v)}vv^T + k^{(w)}ww^T$. In order to keep the ability to exploit sparsity of $A$ and $Q$, we do not actually store $B$ or $M$ but simply the Cholesky factors of the most recent $H$ and $Q$ and the sequence of BFGS update vectors. More specifically, for $q \leq J$, let $B^{(q)}$ be the $q$'th update of $H^{-1}$, i.e.

$$B^{(q)} = C^{-1}C^{-T} + \Psi\Lambda\Psi^T$$

where $\Psi = [v^{(1)}, \ldots, v^{(q)}, w^{(1)}, \ldots, w^{(q)}], \Lambda = \operatorname{diag}(k_1^{(v)}, \ldots, k_q^{(v)}, k_1^{(w)}, \ldots, k_q^{(w)})$. Then we compute products such as $B^{(q)}r$ by means of

$$B^{(q)}r = C^{-1}(C^{-T}r) + \Psi\left(\Lambda(\Psi^T r)\right).$$

For $M$, the situation is similar:

$$\begin{aligned} M^{(q)} &= \left(AB^{(q)}A^T\right)^{-1} \\ &= \left(A(H^{-1} + \Psi\Lambda\Psi^T)A^T\right)^{-1} \\ &= \left(Q + \Phi\Lambda\Phi^T\right)^{-1} \end{aligned}$$

where $\Phi = A\Psi$. By the Sherman-Morrison-Woodbury formula, we get

$$M^{(q)} = Q^{-1} - Q^{-1}\Phi\left(\Lambda^{-1} + \Phi^T Q^{-1}\Phi\right)^{-1}\Phi^T Q^{-1}.$$

We can thus compute products like $M^{(q)}r$ by

$$\begin{aligned} M^{(q)}r &= Q^{-1}\left(I - \Phi\left(\Lambda^{-1} + \Phi^T Q^{-1}\Phi\right)^{-1}\Phi^T Q^{-1}\right)r \\ &= D^{-1}D^{-T}\left(r - \Phi\left(\Lambda^{-1} + \Phi^T D^{-1}D^{-T}\Phi\right)^{-1}\Phi^T D^{-1}D^{-T}r\right) \end{aligned}$$

where we remark that 1) only two columns are added to $\Phi$ in each iteration so that only two new back-substitutions in the operation $D^{-T}\Phi$ are needed, 2) $\Lambda$ is diagonal and thus cheap to invert and 3) the matrix $\left(\Lambda^{-1} + \Phi^T D^{-1}D^{-T}\Phi\right)$ is only of size $2q \times 2q$ and is therefore also cheap to invert.

We then alternate between taking $J$ BFGS steps and one full Newton correction step, starting with BFGS steps and terminate when $\|\psi\|_{\bar{x}}^* \leq \eta\mu$. The resulting BFGS search direction is a descent direction for the function $\|\psi\|_{\bar{x}}^*$, so by using a backtracking line search along these directions, we can not make the objective worse by proceeding in this way. On the other hand, we have no theoretical guarantee that BFGS steps improve the objective value. However,

as the computational experiments will demonstrate, it is often the case that enough centrality can be achieved after just a few BFGS steps.

The norm $\|v\|_{\bar{x}}^*$ is computed as $(v^T H_{\bar{x}}^{-1} v)^{1/2}$. Computing this number requires the evaluation and factorization of $H_{\bar{x}}$. But since $H_{\bar{x}}$ is block-diagonal, this operation is cheap.

We finally remark that whether or not it is beneficial to take BFGS steps, and if it is, how many should be taken, depends on the cost of building and Cholesky factorizing $AH_{\bar{x}}^{-1}A^T$ relative to the cost of subsequent back-substitutions, of which the needed amount is increased if BFGS steps are used. This ratio depends on the dimension and sparsity pattern of $A$ — quantities about which we know nothing beforehand. However, since the dimension and sparsity pattern of $AH_{\bar{x}}^{-1}A^T$ do not vary with $\bar{x}$, it is possible to determine this ratio at initialization time. Thus we can determine an upper bound on $J$ before the main loop of the algorithm.

### 5.2 Higher order predictor direction

It is well known that the Mehrotra second order correction [12] term significantly improves performance of interior-point methods for symmetric cones. This technique is used in virtually all competitive industry standard interior-point implementations solving self-scaled problems. Mehrotra's second order correction generalizes nicely to self-scaled conic problem by use of the Jordan product that can be defined on such cones, see e.g. [2]. For non-symmetric cones, this generalization seems to no longer be possible. Hoping to achieve a similar improvement in performance, we suggest instead to compute a higher order prediction step as described in the following.

Let us denote the central path point with complementarity gap $\mu$ by $z(\mu)$, which corresponds to $\mu = \gamma\mu^0$ in equations (7)–(8). By an appropriate definition of a matrix $K(z)$ and a vector $u(z)$, dependent on the current iterate $z = (\bar{x}, y, \bar{s})$, it is clear that the equations (10)–(11) defining $d_z$ can be written

$$K(z)d_z(\mu) = u(z) \quad \text{or} \quad d_z(\mu) = K(z)^{-1}u(z) =: f(z).$$

The central path is thus the solution of the ordinary differential equation defined by $d_z(\mu) = f(z)$. A step in the predictor direction, i.e. the direction $d_z$, is then the same as taking one Euler step for this ODE. We can obtain a direction that contains, for example, second order information by computing a stage-2 Runge-Kutta direction $d_2$, remembering that each evaluation of $f$ requires solving a system of the type $Kd_z = u$. Such a direction is defined by

$$d_2 = h\left(1 - \frac{1}{2\theta}\right) f(z) + h\frac{1}{2\theta} f(\zeta)$$
$$\zeta = (\zeta_{\bar{x}}, \zeta_y, \zeta_{\bar{s}}) = z(\mu) + \theta h f(z)$$

where $h$ is the stepsize possible in the direction $f(z)$ and $\theta \in (0, 1]$ is a parameter. The choices $\theta = 1/2$ and $\theta = 1$ correspond to the classical midpoint and trapezoidal rules respectively [6].

Our experience shows that this approach reduces the *total* number of iterations as well as the number of factorizations needed to reach an optimal solution, even though two factorizations are needed to compute $d_2$.

We can, however, restrict ourselves to just one factorization by using in place of $H_{\zeta_{\bar{x}}}$ the BFGS update of $H_{\bar{x}}$. In section 5.1, we showed how to implement such a procedure efficiently.

### 5.3 Initial point

The initial point $z^0 = (\bar{x}^0, y^0, \bar{s}^0)$ is required to satisfy $z^0 \in \mathcal{F} \cap \mathcal{N}(\eta)$. We therefore choose some $\bar{x}^0 \in \bar{\mathcal{K}}^\circ$ and set $\bar{s}^0 = -g_{\bar{x}^0}$. We then get $\bar{\nu}\mu(z^0) = (\bar{x}^0)^T \bar{s}^0 = -(\bar{x}^0)^T g_{\bar{x}^0} \overset{(20)}{=} \bar{\nu}$ and hence $\mu(z^0) = 1$. Therefore, this $z^0$ is exactly on the central path, i.e. $z^0 \in \mathcal{N}(0) \subset \mathcal{N}(\eta)$.

### 5.4 Termination

A point $(\bar{x}, y, \bar{s}) = (x, \tau, y, s, \kappa)$ that satisfies the bounds in Theorem 1 solves to $\epsilon$-accuracy the homogeneous model (HSD). However, we are interested in either a certificate of infeasibility or a solution of (PD). Therefore, we need to use stopping criteria able to detect one of these two situations. Consider the following inequalities:

$$\|Ax - \tau b\|_\infty \leq \epsilon \cdot \max\{1, \|[A, b]\|_\infty\} \tag{P}$$
$$\|A^T y + s - c\tau\|_\infty \leq \epsilon \cdot \max\{1, \left\|\left[A^T, I, -c\right]\right\|_\infty\} \tag{D}$$
$$\left|-c^T x + b^T y - \kappa\right| \leq \epsilon \cdot \max\{1, \|\left[-c^T, b^T, 1\right]\|_\infty\} \tag{G}$$
$$\left|c^T x/\tau - b^T y/\tau\right| \leq \epsilon \cdot \left(1 + \left|b^T y/\tau\right|\right) \tag{A}$$
$$\tau \leq \epsilon \cdot 10^{-2} \cdot \max\{1, \kappa\} \tag{T}$$
$$\tau \leq \epsilon \cdot 10^{-2} \cdot \min\{1, \kappa\} \tag{K}$$
$$\mu \leq \epsilon \cdot 10^{-2} \cdot \mu^0 \tag{M}$$

We then terminate and conclude as follows:

| | | |
|---|---|---|
| (OPT) | (P) $\wedge$ (D) $\wedge$ (A) $\Rightarrow$ | Feas. and approx. optimal solution found |
| (INFEAS) | (P) $\wedge$ (D) $\wedge$ (G) $\wedge$ (T) $\Rightarrow$ | Problem nearly primal or dual infeasible |
| (ILLP) | (K) $\wedge$ (M) $\Rightarrow$ | Problem deemed ill-posed |

In case (OPT), the approximately optimal solution $(x, y, s)/\tau$ is returned. If we find (INFEAS), the problem is deemed dual infeasible if $c^T x < 0$ and primal infeasible if $b^T y > 0$. The number $\epsilon > 0$ is a user-specified tolerance.

## 6 Computational experiments

In this section we present results from running our algorithm, which we will denote by NPC, on different test problems. We first introduce the nonsymmetric cones needed for our test problems and then present the test problems. Finally, we include tables with numerical results and discussion.

For all test problems that we consider, $\mathcal{K}$ will have the form $\mathcal{K} = \mathcal{K}_1 \times \cdots \times \mathcal{K}_K$ where each $\mathcal{K}_j$ is either a *three*-dimensional proper cone or $\mathbb{R}_+$. This limitation to cones of such low dimension implies simple expressions for the barrier function and its gradient and Hessian. As we shall see, it does impose any restrictions on which problems can be formulated. See also [16] for further discussion on this topic.

The notation used in this section is independent of previous sections.

### 6.1 Two three-dimensional nonsymmetric cones

In the rest of this paper, we will be considering problems involving the following two nonsymmetric convex cones, both three dimensional.

The three-dimensional *exponential cone* is defined by

$$\mathcal{K}_{\exp} = \text{closure}\,\{(x_1; x_2; x_3) \in \mathbb{R} \times \mathbb{R}_+ \times \mathbb{R}_{++} : \exp(x_1/x_3) \leq x_2/x_3\}$$

for which we are using the barrier function

$$F_{\exp}(x) = -\log(x_3 \log(x_2/x_3) - x_1) - \log x_2 - \log x_3$$

with barrier parameter $\nu = 3$.

The three-dimensional *power cone* is defined by

$$\mathcal{K}_\alpha = \left\{(x_1; x_2; x_3) \in \mathbb{R} \times \mathbb{R}^2_+ : |x_1| \leq x_2^\alpha x_3^{1-\alpha}\right\}$$

where $\alpha \in [0, 1]$ is a parameter. Notice that $\mathcal{K}_{1/2}$ is the standard rotated quadratic cone. For all other $\alpha \in (0, 1)$, $\mathcal{K}_\alpha$ is not symmetric. In [7], it was proved that the function

$$F_\alpha(x) = -\log(x_2^{2\alpha} x_3^{2-2\alpha} - x_1^2) - (1-\alpha)\log x_2 - \alpha \log x_3$$

is a logarithmically homogeneous self-concordant barrier with parameter $\nu = 3$ for $\mathcal{K}_\alpha$. It is this barrier function we are using in our experiments. Nesterov proposed in [16] a barrier function for the three-dimensional power cone with parameter $\nu = 4$. Our computational experience shows that $F_\alpha$ is better in practice which is in accordance with theory.

### 6.2 Test problems

In this section, $e$ will denote the vector of all ones. The dimension of $e$ will be clear from the context.

*6.2.1 p-cone problem*

Given $A \in \mathbb{R}^{M\times N}$ and $b \in \mathbb{R}^M$, the $p$-cone problem is the problem

$$\min_x \quad \|x\|_p, \quad \text{s.t. } Ax = b.$$

In [15], it is shown that this is equivalent to

$$\begin{aligned} \min_{x,y,t} \ t, \qquad \text{s.t.} \quad & Ax = b, \ \ e^T y = t \\ & (x_j; y_j; t) \in K_{(1/p)}, \quad j = 1, \dots, M. \end{aligned}$$

*6.2.2 Facility location problem*

Given $M$ points (locations) in $\mathbb{R}^N$: $C^{(j)}, j = 1, \dots, M$, we want to find the point $z$ with the minimal sum of weighted distances to the locations $C^{(j)}$ measured in $p_j$-norms, $p_j \geq 1$. That is

$$\min_z \quad \sum_{j=1}^{M} a_j \|z - C^{(j)}\|_{p_j} \tag{17}$$

where $a_j \geq 0$ are the weights. We can then formulate (17) in conic form:

$$\begin{aligned} \min_{z^+,z^-,v,w,u} \quad & \sum_{j=1}^{M} a_j u_1^{(j)} \\ \text{s.t.} \quad & v^{(j)} = z^+ - z^- - C^{(j)} && j = 1, \dots, M \\ & e^T w^{(j)} = u_1^{(j)}, \ \ u_1^{(j)} = u_2^{(j)} = \cdots = u_N^{(j)} && j = 1, \dots, M \\ & (v_i^{(j)}; w_i^{(j)}; u_i^{(j)}) \in \mathcal{K}_{1/p_j} && j = 1, \dots, M, \ i = 1, \dots, N \\ & z^+ \geq 0, \ \ z^- \geq 0 \end{aligned}$$

*6.2.3 Geometric programming*

This is a problem of the type

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & f^{(0)}(\boldsymbol{x}) \\ \text{s.t.} \quad & g^{(j)}(\boldsymbol{x}) = 1, \quad j = 1, \dots, M \\ & f^{(j)}(\boldsymbol{x}) \leq 1, \quad j = 1, \dots, P \end{aligned}$$

where $g^{(j)}$ are monomials and $f^{(j)}$ are posynomials. Using the notation $\boldsymbol{x}^{\boldsymbol{v}} := \prod_{i=1}^n x_i^{v_i}$ where each $x_i > 0$, they can be writting

$$g(\boldsymbol{x}) = k_j \boldsymbol{x}^{\boldsymbol{b}^{(j)}}, \quad f^{(j)}(\boldsymbol{x}) = \sum_{i=1}^{N_j} d_i \boldsymbol{x}^{\boldsymbol{a}_i^{(j)}}.$$

With the $j$'th posynomial $f^{(j)}$, we then associate

– the matrix $\boldsymbol{A}^{(j)} := \left(\boldsymbol{a}_1^{(j)}, \boldsymbol{a}_2^{(j)}, \ldots, \boldsymbol{a}_{N_j}^{(j)}\right)^T \in \mathbb{R}^{N_j \times N}$,
– the vector $\boldsymbol{d}^{(j)} = (d_1^{(j)}, \ldots, d_{N_j}^{(j)})^T \in \mathbb{R}^{N_j \times 1}$ and
– the vector $\boldsymbol{c}^{(j)} = \log(\boldsymbol{d}^{(j)}) = (\log(d_1), \ldots, \log(d_{N_j}))^T \in \mathbb{R}^{N_j \times 1}$

Similarly, we associate with the $j$'th monomial $g^{(j)}$

– the vector $\boldsymbol{b}^{(j)}$, the scalar $k^{(j)}$, the scalar $h^{(j)} = \log(k^{(j)})$.

Using the change of variables $u_i = \log(x_i) \Leftrightarrow x_i = \exp(u_i)$ for all $i$, we can write the problem in conic form:

$$
\begin{aligned}
\min_{\boldsymbol{u}_+, \boldsymbol{u}_-, \boldsymbol{w}, \boldsymbol{v}, \boldsymbol{y}, t^{(0)}} \quad & t^{(0)} \\
\text{s.t.:} \quad & \boldsymbol{B}(\boldsymbol{u}_+ - \boldsymbol{u}_-) + \boldsymbol{h} = 0 \\
& \boldsymbol{w}^{(j)} = \boldsymbol{A}^{(j)}(\boldsymbol{u}_+ - \boldsymbol{u}_-) + \boldsymbol{c}^{(j)} && j = 0, \ldots, P \\
& \boldsymbol{e}^T \boldsymbol{v}^{(j)} = t^{(j)}, \quad \boldsymbol{y}^{(j)} = \boldsymbol{e} && j = 0, \ldots, P \\
& \boldsymbol{u}_+, \boldsymbol{u}_-, t^{(0)} \geq 0 \\
& \left(w_i^{(j)}; v_i^{(j)}; y_i^{(j)}\right) \in K_{\exp} && j = 0, \ldots, P, \ i = 1, \ldots, N_j
\end{aligned}
$$

where $\boldsymbol{h} = (h^{(1)}, \ldots, h^{(M)})^T \in \mathbb{R}^{M \times 1}$ and $\boldsymbol{B} = \left(\boldsymbol{b}^{(1)}, \ldots, \boldsymbol{b}^{(M)}\right)^T \in \mathbb{R}^{M \times N}$.

*6.2.4 Entropy maximization*

Given $A \in \mathbb{R}^{M \times N}$, $b \in \mathbb{R}^M$ and $d \in \mathbb{R}_+^N$, the entropy maximization problem is

$$
\begin{aligned}
\min_x \quad & \sum_{j=1}^{N} d_j x_j \log x_j \\
\text{s.t.} \quad & Ax = b \\
& x_j \geq 0, \quad j = 1, \ldots, N
\end{aligned}
$$

which can be formulated as

$$
\begin{aligned}
\min_{x,u} -d^T u, \qquad \text{s.t.} \quad & Ax = b, \ v = e \\
& (u_j; v_j; x_j) \in \mathcal{K}_{\exp}, \ \ j = 1, \ldots, N.
\end{aligned}
$$

## 6.3 Computational results

The remaining tables in this section show the number of iterations (it), the *total* number of factorizations made (ch), the average number of *full* correction steps per iteration (ce) and the termination status (st). `opt` means that an optimal solution was found and `ipr/idu` means a primal/dual infeasibility certificate was found. For all computational experiments, we used the parameters displayed in Table 1.

**Table 1** *Parameters used in computational experiments.*

| Parameter | $J$ | $\theta$ | $\eta$ | $\beta$ | $\epsilon$ |
|---|---|---|---|---|---|
| Value | 3 | 0.70 | 0.50 | 0.80 | $10^{-6}$ |

For entropy maximization problems and geometric programs, we compare our algorithm to the purpose-built solvers in MOSEK [13]. For $p$-cone problems, we compare our algorithm to `SeDuMi` (see [24]) when called through `CVX` (see [8]). We intentionally compare only the number of Cholesky factorizations performed by each algorithm. This is to eliminate from the comparisons the CPU-time consumed by software overhead. Therefore, it is reasonable to measure only the dominating operations, i.e. the Cholesky factorizations.

### *6.3.1 p-cone problems*

Table 2 shows results from solving a series of $p$-cone problems. The data $A$ and $b$ are from the NETLIB collection of linear programs. We see that NPC

**Table 2** *Computational results for p-cone problems. Data $A \in \mathbb{R}^{M\times N}$ and b from* NETLIB. *sp(A) denotes the sparsity of A. This table contains only a part of the instances tested. Full results can be found in the electronic supplements.*

| *Problem* | | NPC | | | | | | CVX/SeDuMi | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| name & size | $p$ | $m$ | $n$ | it | ch | ce | st | $m$ | $n$ | ch | st |
| `bandm` | 1.13 | 777 | 1416 | 9 | 19 | 1.1 | opt | 6913 | 14632 | 21 | opt |
| $M = 305$ | 1.57 | 777 | 1416 | 11 | 23 | 1.1 | opt | 8801 | 18408 | 26 | opt |
| $N = 472$ | 2.09 | 777 | 1416 | 14 | 29 | 1.1 | opt | 9745 | 20296 | 27 | opt |
| sp$(A) = 1.73\%$ | 4.71 | 777 | 1416 | 23 | 37 | 0.6 | opt | 10689 | 22184 | 26 | opt |
| | 7.39 | 777 | 1416 | 24 | 43 | 0.8 | opt | 11633 | 24072 | 26 | opt |
| `blend` | 1.13 | 188 | 342 | 9 | 19 | 1.1 | opt | 1670 | 3534 | 21 | opt |
| $M = 74$ | 1.57 | 188 | 342 | 9 | 20 | 1.2 | opt | 2126 | 4446 | 22 | opt |
| $N = 114$ | 2.09 | 188 | 342 | 9 | 16 | 0.8 | opt | 2354 | 4902 | 20 | opt |
| sp$(A) = 6.19\%$ | 4.71 | 188 | 342 | 11 | 19 | 0.7 | opt | 2582 | 5358 | 20 | opt |
| | 7.39 | 188 | 342 | 13 | 21 | 0.6 | opt | 2810 | 5814 | 21 | opt |
| | | | | | ⋮ | *More results online.* | | | | | |
| `stocfor1` | 1.13 | 282 | 495 | 9 | 16 | 0.8 | opt | 2427 | 5115 | 19 | opt |
| $M = 117$ | 1.57 | 282 | 495 | 8 | 17 | 1.1 | opt | 3087 | 6435 | 20 | opt |
| $N = 165$ | 2.09 | 282 | 495 | 9 | 19 | 1.1 | opt | 3417 | 7095 | 22 | opt |
| sp$(A) = 2.60\%$ | 4.71 | 282 | 495 | 18 | 30 | 0.7 | opt | 3747 | 7755 | 25 | opt |
| | 7.39 | 282 | 495 | 22 | 29 | 0.3 | opt | 4077 | 8415 | 26 | opt |

performs very well compared to `SeDuMi`. `CVX` solves the problem by approximating the original $p$-cone problem by an approximately equivalent self-scaled problem. The resulting self-scaled problem is then solved using `SeDuMi`. As discussed in the introduction, this modelling of a nonsymmetric problem by

symmetric cones requires the introduction of extra variables and constraints. The table shows for each of the two solution methods, the number of rows $m$ and columns $n$ of the final linear constraint matrix (corresponding to $A$ in (PD)). These results clearly demonstrate the advantage of modelling this inherently nonsymmetric problem (the $p$-norm is not a self-dual norm when $p \neq 2$) directly by using a nonsymmetric cone. As seen from the table, the size of the problem built by `CVX` is much greater, in some instances by as much as 17 times, than the size of the problem solved by NPC. Notice also that the latter problem, unlike the first, is independent of $p$.

In terms of iterations, NPC uses about 40% less than `SeDuMi`. The total number of factorizations for the two methods is about the same. However, as described above, `SeDuMi` factorizes much larger matrices. Therefore we may conclude for these problems, that the direct modelling method coupled with a nonsymmetric solver like NPC is clearly superior to `CVX/SeDuMi`.

### *6.3.2 Facility location problems*

Table 3 shows the performances of our algorithm when run on random instances of the facility location problem. For each pair $(N, M)$, we generated 10 instances each with $C^{(j)}$ chosen at random from the standard normal distribution. For each instance, $M$ different $p_j$ were chosen as the maximum of 1.0 and a sample from a normal distribution with mean 2.0 and variance 0.25. The $a_j$ were chosen randomly from a uniform distribution on $[0, 1]$. The column labelled $\bar{p}$ shows the number $M^{-1} \sum_{j=1}^{M} p_j$ averaged over the 10 instances. This number should be close to 2.0.

We see that our algorithm uses in the region 10–20 iterations and the number of Cholesky factorizations never exceeds 32. On average slightly more than 0.50 full centering steps are needed in each iteration. These results can be loosely compared with the computational results in [7, Table 4.1, page 142]. There, a dual variant of the algorithm of [16] is used to solve the same kind of problem. Overall, our algorithm performs better, both in terms of iterations and factorizations.

### *6.3.3 Geometric programs*

Table 4 shows results from applying our algorithms to a set of geometric programs supplied to us by MOSEK. The column labelled `dod` denotes the *degree of difficulty* of the problem [5]. For a particular problem instance $j$, let $I_j^{\mathcal{A}}$ and $C_j^{\mathcal{A}}$ be the number of iterations and Cholesky factorization respectively used by algorithm $\mathcal{A}$ to solve instance $j$ and let us define the ratio of sums $\mathcal{S} = (\sum_j C_j^{\text{NPC}})/(\sum_j C_j^{\text{NPC}})$. Further let $\mathcal{R}_j^{\text{it}} = I_j^{\text{NPC}}/I_j^{\text{MOSEK}}$ and $\mathcal{R}_j^{\text{ch}} = C_j^{\text{NPC}}/C_j^{\text{MOSEK}}$. If we let an overbar denote arithmetic mean and a tilde denote geometric mean over all $j$, we then find

$$(S,\ \overline{\mathcal{R}^{\text{it}}},\ \overline{\mathcal{R}^{\text{ch}}},\ \widetilde{\mathcal{R}^{\text{it}}},\ \widetilde{\mathcal{R}^{\text{ch}}}) = (1.3,\ 1.1,\ 1.9,\ 0.94,\ 1.7).$$

**Table 3** *Results for facility location problems. The algorithm always terminated after reaching optimality as all problem instances were feasible by construction. This table contains only a part of the instances tested. Full results can be found in the electronic supplements.*

| *Problem* | | | | NPC | | |
|---|---|---|---|---|---|---|
| $N$ | $M$ | $\nu$ | $\bar{p}$ | it | ch | ce |
| 3 | 4 | 44 | 2.03 | 11.1 | 18.2 | 0.65 |
| 10 | 4 | 128 | 2.07 | 13.2 | 20.1 | 0.54 |
| 3 | 20 | 220 | 2.09 | 17.1 | 27.5 | 0.64 |
| 19 | 4 | 236 | 2.00 | 13.8 | 21.0 | 0.54 |
| | | ⋮ | *More results online.* | | | |
| 10 | 12 | 384 | 2.06 | 16.0 | 25.1 | 0.58 |
| 32 | 4 | 392 | 2.03 | 13.4 | 20.9 | 0.56 |
| 10 | 20 | 640 | 1.99 | 18.7 | 30.5 | 0.66 |
| 19 | 20 | 1180 | 2.01 | 19.7 | 30.5 | 0.60 |
| 32 | 20 | 1960 | 1.98 | 17.7 | 31.5 | 0.79 |

**Table 4** *Results for geometric programs. This table contains only a part of the instances tested. Full results can be found in the electronic supplements.*

| *Problem* | | | NPC | | | | mskgpopt | |
|---|---|---|---|---|---|---|---|---|
| name | $n$ | dod | it | ch | ce | st | ch | st |
| beck751 | 7 | 10 | 16 | 30 | 0.9 | opt | 18 | opt |
| beck753 | 7 | 10 | 13 | 27 | 1.1 | opt | 10 | opt |
| car | 37 | 104 | 15 | 28 | 0.9 | opt | 46 | opt |
| demb761 | 11 | 19 | 12 | 22 | 0.8 | ipr | 10 | opt |
| | | | ⋮ | *More results online.* | | | | |
| demb781 | 2 | 1 | 7 | 10 | 0.4 | opt | 7 | opt |
| fang88 | 11 | 16 | 9 | 18 | 1.0 | opt | 11 | opt |
| jha88 | 30 | 274 | 17 | 34 | 1.0 | opt | 13 | opt |
| mra01 | 61 | 844 | 16 | 30 | 0.9 | opt | 58 | opt |
| mra02 | 126 | 3494 | 30 | 57 | 0.9 | opt | 53 | opt |
| rijc786 | 8 | 3 | 9 | 16 | 0.8 | opt | 6 | opt |
| rijc787 | 7 | 40 | 12 | 23 | 0.9 | opt | 36 | opt |

For these problems we therefore conclude that our algorithm performs somewhat inferiorly to Mosek, using less iterations but cumulatively 30% more Cholesky factorization than Mosek.

### *6.3.4 Entropy problems*

Table 5 shows results from solving a set of real-world entropy problems supplied to us by Mosek. Generally the problems have many variables compared to the number of constraints resulting in a very "fat" constraint matrix $A$. For these problems we compare our algorithms to the commercial solver from

**Table 5** *Computational results for entropy problems. This table contains only a part of the instances tested. Full results can be found in the electronic supplements.*

| *Problem* | | | NPC | | | | mskenopt | |
|---|---|---|---|---|---|---|---|---|
| name | $N$ | $M$ | it | ch | ce | st | ch | st |
| prob | 17 | 15 | 9 | 15 | 0.7 | opt | 8 | opt |
| prob2 | 18 | 14 | 9 | 18 | 1.0 | opt | 8 | opt |
| ento46 | 130 | 21 | 25 | 50 | 1.0 | opt | 42 | opt |
| ento22 | 794 | 28 | 28 | 60 | 1.1 | ipr | 14 | ipr |
| ento21 | 931 | 28 | 55 | 112 | 1.0 | ipr | 18 | ipr |
| a_tb | 1127 | 25 | 38 | 87 | 1.3 | opt | 97 | opt |
| ento23 | 1563 | 28 | 34 | 73 | 1.1 | ipr | 14 | ipr |
| | | | ⋮ | *More results online.* | | | | |
| a_35 | 4333 | 37 | 43 | 90 | 1.1 | ipr | 18 | ipr |
| a_24 | 5162 | 37 | 36 | 90 | 1.5 | ipr | 23 | ipr |
| ento3 | 5172 | 28 | 49 | 126 | 1.6 | opt | 146 | opt |
| ento50 | 5172 | 28 | 49 | 126 | 1.6 | opt | 146 | opt |
| a_46 | 9455 | 37 | 40 | 102 | 1.6 | ipr | 20 | ipr |
| a_56 | 9702 | 37 | 65 | 158 | 1.4 | opt | 123 | opt |
| ento25 | 10142 | 28 | 116 | 250 | 1.2 | opt | 149 | opt |
| entodif | 12691 | 40 | 50 | 130 | 1.6 | opt | 155 | opt |
| ento48 | 15364 | 31 | 16 | 52 | 2.2 | opt | 47 | opt |

Mosek, which solves the monotone complementarity problem [3] corresponding to the entropy problem.

We see that, except for a few of the problems, our algorithm compares somewhat unfavourable to Mosek. With the notation defined in Section 6.3.3, we find

$$(S,\ \overline{\mathcal{R}^{\mathrm{it}}},\ \overline{\mathcal{R}^{\mathrm{ch}}},\ \widetilde{\mathcal{R}^{\mathrm{it}}},\ \widetilde{\mathcal{R}^{\mathrm{ch}}}) = (1.6,\ 1.2,\ 2.8,\ 0.93,\ 2.1).$$

That is, although npc uses fewer iterations, it uses cumulatively about 60% more Cholesky factorizations to solve the entire set of problems when compared to Mosek.

We remark that the solvers from Mosek for entropy problems and geometric programs are two *different* solvers, each purpose-built to solve those particular problems and not modelled as conic optimization problems. Our algorithm, on the other hand, uses a purely conic formulation and thus is a much more general purpose algorithm. We use no particular tuning of parameters to particular problems. From simple experiments we know that tuning the parameters $\eta$ and $\beta$ for each type of problem, we could improve the computational performance of our algorithm. However, since we believe in the importance of practical applicability across various problem types, we choose to fix the parameters and instead let our algorithm enjoy a very high degree of versatility. In that light, and considering the fact that Mosek is an industry-grade implementation, we believe our algorithm compares very well.

## 7 Conclusions

In this paper, we have presented a homogeneous primal-dual interior-point algorithm for nonsymmetric convex conic optimization. Unlike previous work solving the homogenized convex conic problem, our algorithm makes use *only* of the primal barrier function thus making the algorithm widely applicable. We have proven the standard $\mathcal{O}(\sqrt{\nu}\log(1/\epsilon))$ worst-case complexity result. Inspired by techniques known to significantly improve efficiency of algorithms for self-scaled cones, we have developed techniques similar in purpose but for the non-symmetric case. These include quasi-Newton updating to reduce computational load and a Runge-Kutta type second order search direction, which is new in this context. We demonstrated how to efficiently implement these techniques without loosing the ability to exploit sparsity in the data matrix $A$. Finally we have presented extensive computational results that indicate the algorithm works well in practice.

By inspecting the tables in Section 6.3, we see that

- The performance of the algorithm depends a lot on the type of problem.
- For the $p$-cone problems, our algorithm superior in performance to `SeDuMi` called via `CVX`. These experiments clearly show the potential advantage of directly modelling nonsymmetric problems by using nonsymmetric cones.
- For the facility location problems, our algorithm compares favorably to an algorithm [7], which is a dual variant of the one presented in [16].
- For geometric programs, our algorithm compares somewhat unfavourable to Mosek.
- For entropy maximization problems, our algorithm again compares somewhat unfavourable to Mosek.

The computational results comparing our algorithm to Mosek should, however, be seen in the light of the comments in Section 6.3.4 on page 23.

Comparing the kind of algorithm we have presented with a primal-dual IPM for self-scaled cones, we see that the major difference is the need for a separate correction phase. Nesterov remarks in [16] that this process can be seen as the process of finding a scaling point, i.e. a point $w$ such that $x = \nabla^2 F(w)s$. It seems reasonable that this is a more complex problem when the cone is not symmetric. We can not compute it analytically, so we need an iterative procedure.

This difference is interesting theoretically as well as practically. For the problems we have considered, the centering problem certainly is a relatively easy problem compared to the full problem, in the sense that we do not need a very *accurately* centered point. We have seen in the experiments with our algorithm that rarely more a couple of correction steps are needed, some or all of which may be comparably inexpensive quasi-Newton steps.

**Acknowledgements** The authors thank Erling D. Andersen and Joachim Dahl of Mosek ApS for lots of insights and for supplying us with test problems for the geometric programs and the entropy problems. The authors also thank the reviewers for many helpful comments.

## A Properties of the barrier function

Here we list some properties of logarithmically homogeneous self-concordant barriers (LHSCB) that we use in this paper. Many more properties and proofs can be found in [18,19].

Let $\mathcal{K}^\circ$ denote the interior of $\mathcal{K}$. We assume that $F : \mathcal{K}^\circ \mapsto \mathbb{R}$ is a LHSCB for $\mathcal{K}$ with barrier parameter $\nu$. This means that for all $x \in \mathcal{K}^\circ$ and $t > 0$,

$$F(tx) = F(x) - \nu \log t.$$

It follows that the conjugate of $F$, denoted $F^*$ and defined for $s \in (\mathcal{K}^*)^\circ$ by

$$F^*(s) = \sup_{x \in \mathcal{K}} \{-s^T x - F(x)\}$$

is a LHSCB for the dual cone $\mathcal{K}^*$. Similarly to the notation used in [18,19], we write the local Hessian norms on $\mathcal{K}$ and $\mathcal{K}^*$ as:

$$\begin{aligned} \|g\|_x &= \|H_x^{1/2} g\|, \quad \text{for } x \in \mathcal{K}^\circ \\ \|h\|_s^* &= \|(H_s^*)^{1/2} g\|, \quad \text{for } s \in (\mathcal{K}^*)^\circ \\ \|h\|_x^* &= \|H_x^{-1/2} h\|, \quad \text{for } x \in (\mathcal{K})^\circ, \end{aligned}$$

where $H_s^* = \nabla^2 F^*(s)$. Notice the different definitions of $\|\cdot\|_y^*$ depending on whether $y$ is in $\mathcal{K}$ or $\mathcal{K}^*$. Using this convention and that $-g_x \in (\mathcal{K}^*)^\circ$ and $H_{-g_x}^* = H_x^{-1}$, we see that

$$\|s\|_{-g_x}^* = \|(H_{-g_x}^*)^{-1/2} s\| = \|H_x^{1/2} s\| = \|s\|_x^*. \tag{18}$$

For $x \in \mathcal{K}^\circ$, $F$ satisfies

$$H_x x = -g_x \tag{19}$$

$$x^T g_x = -\nu \tag{20}$$

$$\|x\|_x^2 = \nu. \tag{21}$$

The *Dikin ellipsoids* are feasible [4]. That is:

$$x \in \mathcal{K}^\circ \quad \Rightarrow \quad W(x) = \{u, \|u - x\|_x \le 1\} \subseteq \mathcal{K} \tag{22}$$

$$s \in (\mathcal{K}^*)^\circ \quad \Rightarrow \quad W^*(s) = \{h, \|h - s\|_s^* \le 1\} \subseteq K^*. \tag{23}$$

## B The homogeneous and self-dual model

### B.1 Optimality and infeasibility certificate

Let $G$ be defined by (5) and notice that $G$ is skew-symmetric: $G = -G^T$.

1. Observe that we can write (HSD) as $G(y; x; \tau)^T - (0; s; \kappa)^T = 0$. Premultiplying this equation by $(y; x; \tau)^T$ gives $x^T s + \tau\kappa = 0$.

2. $\tau > 0$ implies $\kappa = 0$ and hence $b^T(y/\tau) - c^T(x/\tau) = 0$ and therefore $x^T s = 0$. Dividing the two first linear feasibility equations of (HSD) by $\tau$, we obtain the linear feasibility equations of (1). Thus $(x, y, s)/\tau$ is optimal for (PD).
3. If $\kappa > 0$ then $\tau = 0$ so $Ax = 0$ and $A^T y + s = 0$. Further $c^T x - b^T y = -\kappa < 0$ so not both $c^T x$ and $-b^T y$ can be non-negative. Assume $-b^T x < 0$. If (PD) is primal-feasible then there exists $\bar{x} \in \mathcal{K}$ such that $A\bar{x} = b$. But then $0 > -b^T y = -\bar{x}^T A^T y = \bar{x}^T s \geq 0$, a contradiction. We can argue similarly if $c^T x < 0$,

and this completes the proof of Lemma 1.

### B.2 Self-duality

The dual of (HSD) problem is

$$\max_{\hat{y}_1, \hat{y}_1, \hat{y}_1, \hat{y}_1, \hat{y}_1, \hat{s}} \quad 0$$

$$\text{s.t.} \quad \begin{pmatrix} A^T & 0 & -c \\ 0 & c^T & -b^T \\ 0 & -I & 0 \\ -1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \hat{y}_3 \end{pmatrix} + \begin{pmatrix} \hat{s}_1 \\ \hat{s}_2 \\ \hat{s}_3 \\ \hat{s}_4 \end{pmatrix} = 0 \tag{24}$$

$$-A\hat{y}_2 + b\hat{y}_3 = 0 \tag{25}$$

$$\hat{s} \in (\mathcal{K} \times \mathbb{R}_+ \times \mathcal{K}^* \times \mathbb{R}_+)^*, \quad \hat{y} \text{ free.} \tag{26}$$

After a few eliminations, we see that (24)–(26) are equivalent to

$$\begin{aligned} A\hat{s}_3 \; -b\hat{s}_4 \qquad &= 0 \\ -A^T\hat{y}_1 \qquad +c\hat{s}_4 \; -\hat{s}_1 &= 0 \\ b^T\hat{y}_1 \; -c^T\hat{s}_3 \qquad -\hat{s}_2 &= 0 \\ (\hat{s}_3, \hat{s}_4) \in \mathcal{K} \times \mathbb{R}_+, \;\; (\hat{s}_1, \hat{s}_2) \in \mathcal{K}^* \times \mathbb{R}_+, \;\; \hat{y}_1 &\in \mathbb{R}^m. \end{aligned} \tag{27}$$

Through the following identification of variables

$$\hat{s}_1 \sim s, \quad \hat{s}_2 \sim \kappa, \quad \hat{s}_3 \sim x, \quad \hat{s}_4 \sim \tau, \quad \hat{y}_1 \sim y,$$

it is clear that the constraints (27) are equivalent to those of the problem (HSD). Since the objective function in both problems is constant zero, the two problems are identical and this proves Lemma 2.

## C Prediction

The direction $d_z$ is defined by

$$G(d_y; d_{\bar{x}}) - (0; d_{\bar{s}}) = -\left(G(y; \bar{x}) - (0; \bar{s})\right) \tag{28}$$

$$d_{\bar{s}} + \mu H_{\bar{x}} d_{\bar{x}} = -\bar{s} \tag{29}$$

### C.1 Reduction of residuals

We first show:

$$1. \quad \bar{s}^T d_{\bar{x}} + \bar{x}^T d_{\bar{s}} + \bar{x}^T \bar{s} = \psi(z)^T d_{\bar{x}} \tag{30}$$

$$2. \quad (\bar{x} + d_{\bar{x}})^T (\bar{s} + d_{\bar{s}}) = 0 \tag{31}$$

$$3. \quad d_{\bar{x}}^T d_{\bar{s}} = -\psi(z)^T d_{\bar{x}}. \tag{32}$$

1. We get $\bar{s}^T d_{\bar{x}} + \bar{x}^T d_{\bar{s}} + \bar{x}^T \bar{s} \overset{(29)}{=} \bar{s}^T d_{\bar{x}} + \bar{x}^T(-\bar{s} - \mu H_{\bar{x}} d_{\bar{x}}) + \bar{x}^T \bar{s}$, which, after reduction, gives $d_{\bar{x}}^T(\bar{s} - \mu H_{\bar{x}} \bar{x}) = \psi(z)^T d_{\bar{x}}$.
2. Equation (28) is equivalent to $G(y + d_y; \bar{x} + d_{\bar{x}}) - (0; \bar{s} + d_{\bar{s}}) = 0$. Premultiplying this equation by $(y + d_y, \bar{x} + d_{\bar{x}})$ gives (31).
3. Follows from expanding (31) and using (30).

Now the lemma follows readily: We simply note that the first equation follows directly from elementary linear algebra. To show the second:

$$\begin{aligned}
\bar{\nu}\mu(z^+) &= (\bar{x} + \alpha d_{\bar{x}})^T (\bar{s} + \alpha d_{\bar{s}}) \\
&= \bar{x}^T \bar{s} + \alpha(\bar{s}^T d_{\bar{x}} + \bar{x}^T d_{\bar{s}}) + \alpha^2 d_{\bar{x}}^T d_{\bar{s}} \\
&\overset{(30)-(32)}{=} \bar{x}^T \bar{s} + \alpha(-\bar{x}^T \bar{s} + \psi(z)^T d_{\bar{x}}) + \alpha^2(-\psi(z)^T d_{\bar{x}}) \\
&= (1-\alpha)\bar{x}^T \bar{s} + \alpha(1-\alpha)\psi(z)^T d_{\bar{x}}
\end{aligned}$$

which after division by $\bar{\nu}$ proves Lemma 3.

### C.2 Bounds on $\bar{s}$, $d_{\bar{s}}$ and $d_{\bar{x}}$

Assume $\|\psi\|_{\bar{x}}^* \leq \eta\mu$. By definition, $\psi = \bar{s} - \mu H_{\bar{x}} \bar{x}$, which after left-multiplication by $H_{\bar{x}}^{-1/2}$, taking norms and squaring both sides gives

$$\begin{aligned}
(\|\bar{s}\|_{\bar{x}}^*)^2 &= (\|\psi\|_{\bar{x}}^*)^2 + \mu^2 \|\bar{x}\|_{\bar{x}}^2 + 2\mu \bar{x}^T \psi \\
&= (\|\psi\|_{\bar{x}}^*)^2 + 2 + \mu^2 \bar{\nu} \leq \mu^2(\bar{\nu} + \eta^2) \\
\|\bar{s}\|_{\bar{x}}^* &\leq \mu\sqrt{\eta^2 + \bar{\nu}}
\end{aligned} \tag{33}$$

where we used (21) and $\bar{x}^T \psi = 0$.

This bound allows us to obtain bounds on $d_{\bar{x}}$ and $d_{\bar{s}}$: Left-multiplying (29) by $H_{\bar{x}}^{-1/2}$, taking norms and squaring both sides gives

$$\begin{aligned}
(\|d_{\bar{s}}\|_{\bar{x}}^*)^2 + \mu^2 \|d_{\bar{x}}\|_{\bar{x}}^2 = (\|\bar{s}\|_{\bar{x}}^*)^2 - 2\mu d_{\bar{x}}^T d_{\bar{s}} &\overset{(32)}{=} (\|\bar{s}\|_{\bar{x}}^*)^2 + 2\mu d_{\bar{x}}^T \psi \\
&\leq (\|\bar{s}\|_{\bar{x}}^*)^2 + 2\mu \|d_{\bar{x}}\|_{\bar{x}} \|\psi\|_{\bar{x}}^*
\end{aligned}$$

by the Cauchy-Schwarz inequality. Therefore: $\mu^2 \|d_{\bar{x}}\|_{\bar{x}}^2 \leq (\|\bar{s}\|_{\bar{x}}^*)^2 + 2\mu \|d_{\bar{x}}\|_{\bar{x}} \|\psi\|_{\bar{x}}^*$. Now subtracting $2\mu \|d_{\bar{x}}\|_{\bar{x}} \|\psi\|_{\bar{x}}^*$ and adding $(\|\psi\|_{\bar{x}}^*)^2$ to both sides, we get

$$(\mu \|d_{\bar{x}}\|_{\bar{x}} - \|\psi\|_{\bar{x}}^*)^2 \leq (\|\bar{s}\|_{\bar{x}}^*)^2 + (\|\psi\|_{\bar{x}}^*)^2$$

or

$$\begin{aligned}\|d_{\bar{x}}\|_{\bar{x}} &\leq \mu^{-1}\left(\|\psi\|^*_{\bar{x}} + \sqrt{(\|\bar{s}\|^*_{\bar{x}})^2 + (\|\psi\|^*_{\bar{x}})^2}\right)\\ &\leq \mu^{-1}(\eta\mu + \sqrt{\mu^2(\eta^2+\bar{\nu}) + \eta^2\mu^2}) = \eta + \sqrt{\eta^2+\bar{\nu}} =: k_{\bar{x}}. \end{aligned} \tag{34}$$

For $d_{\bar{s}}$, we similarly have

$$\begin{aligned}(\|d_{\bar{s}}\|^*_{\bar{x}})^2 &\leq (\|\bar{s}\|^*_{\bar{x}})^2 + 2\mu\|d_{\bar{x}}\|_{\bar{x}}\|d_{\bar{s}}\|^*_{\bar{x}}\\ (\|d_{\bar{s}}\|^*_{\bar{x}} - \mu\|d_{\bar{x}}\|_{\bar{x}})^2 &\leq (\|\bar{s}\|^*_{\bar{x}})^2 + \mu^2\|d_{\bar{x}}\|^2_{\bar{x}}\\ \|d_{\bar{s}}\|^*_{\bar{x}} &\leq k_{\bar{x}}\mu + \sqrt{\mu^2(\eta^2+\bar{\nu}) + k^2_{\bar{x}}\mu^2} = k_{\bar{s}}\mu \end{aligned} \tag{35}$$

where $k_{\bar{s}} := k_{\bar{x}} + \sqrt{(\eta^2+\bar{\nu}) + k^2_{\bar{x}}}$.

C.3 Feasibility of $z^+$.

Define $\alpha_1 := k^{-1}_{\bar{x}} = \Omega(1/\sqrt{\bar{\nu}})$. Then for any $\alpha \leq \alpha_1$, we have

$$\|\bar{x} - (\bar{x} + \alpha d_{\bar{x}})\|_{\bar{x}} = \alpha\|d_{\bar{x}}\|_{\bar{x}} \overset{(34)}{\leq} \alpha k_{\bar{x}} \leq 1$$

and so from (22), we conclude $\bar{x} + \alpha d_{\bar{x}} = \bar{x}^+ \in \bar{\mathcal{K}}$.

Now, define $\alpha_2 := (1-\eta)k^{-1}_{\bar{s}} = \Omega(1/\sqrt{\bar{\nu}})$. Then for $\alpha \leq \alpha_2$, we have

$$\begin{aligned}\mu^{-1}\|\bar{s}^+ + \mu g_{\bar{x}}\|^*_{-g_{\bar{x}}} &= \mu^{-1}\|\bar{s} + \alpha d_{\bar{s}} + \mu g_{\bar{x}}\|^*_{-g_{\bar{x}}} = \mu^{-1}\|\psi + \alpha d_{\bar{s}}\|^*_{-g_{\bar{x}}}\\ &\overset{(18)}{\leq} \mu^{-1}\|\psi\|^*_{\bar{x}} + \mu^{-1}\alpha\|d_{\bar{s}}\|^*_{\bar{x}} \overset{(35)}{\leq} \eta + \alpha k_{\bar{s}} \leq 1. \end{aligned}$$

Since $-g_{\bar{x}} \in \bar{\mathcal{K}}^*$, we have by (23) that $\mu^{-1}\bar{s}^+ \in \bar{\mathcal{K}}^*$ and therefore $\bar{s}^+ \in \bar{\mathcal{K}}^*$. Therefore, Lemma 4 holds with $\alpha = \min\{\alpha_1, \alpha_2\} = \Omega(1/\sqrt{\bar{\nu}}) = \Omega(1/\sqrt{\nu})$.

C.4 Bound on $\psi^+$.

First recall the definition (6): $\psi(\bar{x}, \bar{s}, t) = \bar{s} + tg_{\bar{x}}$. Now consider for a fixed $v_0$ the function

$$\Phi_t(\bar{x}) = \bar{x}^T v_0 + tF(\bar{x})$$

which is self-concordant with respect to $\bar{x}$. Define its Newton step by $n_t(\bar{x}) := -\nabla^2\Phi_t(\bar{x})^{-1}\nabla\Phi_t(\bar{x})$. Define also $q = \|n_{t_2}(\bar{x})\|_{\bar{x}}$. From the general theory of self-concordant functions, the following inequality holds. If $q \leq 1$, then

$$\|n_{t_2}(\bar{x}_2)\|_{\bar{x}_2} \leq \left(\frac{q}{1-q}\right)^2. \tag{36}$$

For a proof of this relation, see e.g. Theorem 2.2.4 in [23]. With $v_0 = \bar{s}^+$, $t_2 = \mu^+$ and $\bar{x}_2 = \bar{x}^+$, the inequality (36) is

$$\|\psi^+\|^*_{\bar{x}^+} \leq \mu^+ \left(\frac{q}{1-q}\right)^2. \tag{37}$$

where $\mu^+ q = \|H_{\bar{x}}^{-1}(\bar{s}^+ + \mu^+ g_{\bar{x}})\|_{\bar{x}} = \|\bar{s}^+ + \mu^+ g_{\bar{x}}\|^*_{\bar{x}}$. From Lemma 3 and (34):

$$\begin{aligned} |\mu - \mu^+| &= |-\alpha\mu + \alpha(1-\alpha)\bar{\nu}^{-1}\psi^T d_{\bar{x}}| \\ &\leq \mu\alpha\left(1 + (1-\alpha)\eta k_{\bar{x}}\bar{\nu}^{-1}\right). \end{aligned} \tag{38}$$

By the assumption $\|\psi\|^*_{\bar{x}} \leq \eta\mu$ combined with (34), we have $\psi^T d_{\bar{x}} \geq -\eta k_{\bar{x}}\mu$. Therefore

$$\begin{aligned} \mu^+ &= (1-\alpha)\mu + \alpha(1-\alpha)\bar{\nu}^{-1}\psi^T d_{\bar{x}} \\ &\geq \mu(1-\alpha)(1-\alpha\eta k_{\bar{x}}\bar{\nu}^{-1}) \\ \mu/\mu^+ &\leq (1-\alpha)^{-1}(1-\alpha\eta k_{\bar{x}}\bar{\nu}^{-1})^{-1} \end{aligned} \tag{39}$$

Let us now obtain a bound on $q$.

$$\begin{aligned} \mu^+ q &= \|\bar{s}^+ + \mu^+ g_{\bar{x}}\|^*_{\bar{x}} = \|\psi - (\mu - \mu^+)g_{\bar{x}} + \alpha d_{\bar{s}}\|^*_{\bar{x}} \\ &\leq \|\psi\|^*_{\bar{x}} + |\mu - \mu^+|\|g_{\bar{x}}\|^*_{\bar{x}} + \alpha\|d_{\bar{s}}\|^*_{\bar{x}} \\ &\leq \eta\mu + \mu\alpha\left(1 + (1-\alpha)\eta k_{\bar{x}}\bar{\nu}^{-1}\right)\sqrt{\bar{\nu}} + \alpha k_{\bar{s}}\mu \\ &\leq \mu\left(\eta + \alpha k_{\bar{s}} + \alpha(1 + (1-\alpha)\bar{\nu}^{-1}\eta k_{\bar{x}})\sqrt{\bar{\nu}}\right) \\ q &\leq (\mu/\mu^+)(\eta + \alpha(\sqrt{\bar{\nu}} + k_{\bar{s}} + \eta k_{\bar{x}})) \\ &\leq (1-\alpha)^{-1}(1-\alpha\eta k_{\bar{x}}\bar{\nu}^{-1})^{-1}(\eta + \alpha(\sqrt{\bar{\nu}} + k_{\bar{s}} + \eta k_{\bar{x}})) \end{aligned} \tag{40}$$

where we used (35), (38), (39) and the assumption $\|\psi\|^*_{\bar{x}} \leq \eta\mu$. Now the reader can verify that for $\eta \leq 1/6$ and $\bar{\nu} \geq 2$, we have the implication

$$\alpha \leq \alpha_3 := \frac{1}{11\sqrt{\bar{\nu}}} = \Omega(1/\sqrt{\bar{\nu}}) \quad \Rightarrow \quad q^2/(1-q)^2 \leq 2\eta \leq 1/3 \tag{41}$$

which also implies $q < 1$. Now by (37), we see that (41) implies $\|\psi^+\|^*_{\bar{x}^+} \leq 2\eta\mu^+$ and hence $z^+ \in \mathcal{N}(2\eta)$ which finishes the proof of Lemma 5.

## D Correction phase

Assume $\|\psi(\bar{x}, \bar{s}, \mu)\|^*_{\bar{x}} \leq \beta\mu$ where $\mu := \mu(z)$ with $z = (\bar{x}, y, \bar{s})$. The equations defining the correction step $(\delta_{\bar{x}}, \delta_y, \delta_{\bar{s}})$ are

$$G(\delta_y; \delta_{\bar{x}}) - (0; \delta_{\bar{s}}) = 0 \tag{42}$$

$$\delta_{\bar{s}} + \mu H_{\bar{x}}\delta_{\bar{x}} = -\psi(\bar{x}, \bar{s}, \mu) \tag{43}$$

and the next point is then $(\bar{x}^+, y^+, \bar{s}^+) := (\bar{x}, y, \bar{s}) + \hat{\alpha}(\delta_{\bar{x}}, \delta_y, \delta_{\bar{s}})$. Left-multiplying (42) by $(\delta_y, \delta_{\bar{x}})^T$, we get $\delta_{\bar{x}}^T \delta_{\bar{s}} = 0$. From (43), we then have

$$(\|\delta_{\bar{s}}\|_{\bar{x}}^*)^2,\ \mu^2\|\delta_{\bar{x}}\|_{\bar{x}}^2 \leq (\|\delta_{\bar{s}}\|_{\bar{x}}^*)^2 + \mu^2\|\delta_{\bar{x}}\|_{\bar{x}}^2 = (\|\psi(\bar{x}, \bar{s}, \mu)\|_{\bar{x}}^*)^2 \leq \beta^2\mu^2$$

and therefore

$$\|\delta_{\bar{x}}\|_{\bar{x}} \leq \beta, \quad \|\delta_{\bar{s}}\|_{\bar{x}}^* \leq \beta\mu. \tag{44}$$

From (43), we also have

$$\begin{aligned} \|\psi(\bar{x}, \bar{s}, \mu) + \hat{\alpha}\delta_{\bar{s}}\|_{\bar{x}}^* &= \|(1-\hat{\alpha})\psi(\bar{x}, \bar{s}, \mu) + \hat{\alpha}\mu H_{\bar{x}}\delta_{\bar{x}}\|_{\bar{x}}^* \\ &\leq (1-\hat{\alpha})\|\psi(\bar{x}, \bar{s}, \mu)\|_{\bar{x}}^* + \hat{\alpha}\mu\|\delta_{\bar{x}}\|_{\bar{x}} \\ &\leq (1-\hat{\alpha})\beta\mu + \hat{\alpha}\mu\beta = \beta\mu \end{aligned} \tag{45}$$

Where we used (44). Now define $q = (\mu^+)^{-1}\|\bar{s}^+ + \mu^+ g_{\bar{x}}\|_{\bar{x}}^*$. Then estimating similarly to (40), we get

$$\begin{aligned} \mu^+ q &\leq \|\psi(\bar{x}, \bar{s}, \mu) + (\mu^+ - \mu)g_{\bar{x}} + \hat{\alpha}\delta_{\bar{s}}\|_{\bar{x}}^* \\ &\leq \beta\mu(1 + \hat{\alpha}(\beta\bar{\nu}^{-1/2} + 1)) \end{aligned}$$

and similarly to the computation in (39), we therefore find

$$\mu/\mu^+ \leq (1 - \hat{\alpha}\bar{\nu}^{-1}\beta^2)^{-1}$$

so that altogether

$$q \leq \beta(1 - \hat{\alpha}\bar{\nu}^{-1}\beta^2)^{-1}(1 + \hat{\alpha}(\beta\bar{\nu}^{-1/2} + 1)). \tag{46}$$

Now we can apply the theorem (36) with $v_0 = \bar{s}^+$, $t = \mu$ and $\bar{x}_2 = \bar{x}^+$:

$$\|\psi(\bar{x}^+, \bar{s}^+, \mu^+)\|_{\bar{x}^+}^* \leq \mu^+ \left(\frac{q}{1-q}\right)^2 \tag{47}$$

The reader can verify that for $\hat{\alpha} \leq 1/84$, $\bar{\nu} \geq 2$, $\beta \leq 2\eta \leq 1/3$, the bound (46) implies that when recursively using (47) *twice*, we obtain

$$\|\psi(\bar{x}^+, \bar{s}^+, \mu^+)\|_{\bar{x}^+}^* \leq \frac{1}{2}\beta \leq \eta$$

and therefore $z^+ \in \mathcal{N}(\eta)$ which proves Lemma 6.

## E Algorithm complexity

From Lemma 3, we have that the linear residuals $G(y;\bar{x})-(0;\bar{s})$ are reduced by a factor $(1-\alpha)$ in each iteration. Since we can always take $\alpha = \Omega(1/\sqrt{\bar{\nu}})$, we see that $G(y;\bar{x})-(0;\bar{s})$ decreases geometrically with a rate of $(1-\Omega(1/\sqrt{\bar{\nu}}))$ which implies that

$$\|G(y;\bar{x})-(0;\bar{s})\| \leq \epsilon \|G(y^0;\bar{x}^0)-(0;\bar{s}^0)\|$$

in $\mathcal{O}(\sqrt{\bar{\nu}}\log(1/\epsilon)) = \mathcal{O}(\sqrt{\nu}\log(1/\epsilon))$ iterations.

To see that the same holds for $\mu(z)$, let us briefly use the following notation: $z$ is the starting point, $z^+$ is the point after prediction and $z^{(j)}$ is the point after applying $j$ correction steps starting in $z^+$. Then from Lemma 3 and (34), we have

$$\begin{aligned}\mu(z^+) &\leq (1-\alpha)\mu(z) + \alpha(1-\alpha)\bar{\nu}^{-1}\mu\eta k_{\bar{x}} \\ &\leq \mu(z)(1-\alpha)(1+\alpha\eta k_{\bar{x}}\bar{\nu}^{-1}) \\ &= \mu(z)(1-\Omega(1/\sqrt{\bar{\nu}})) \end{aligned} \tag{48}$$

Since $\delta_{\bar{x}}^T\delta_{\bar{s}} = 0$, we see from (43) that

$$(\bar{x}^+)^T\delta_{\bar{s}} = \mu(z^+)\delta_{\bar{x}}^T g_{\bar{x}^+} = \delta_{\bar{x}}^T\psi(\bar{x}^+,\bar{s}^+,\mu(z^+)) - \delta_{\bar{x}}^T\bar{s}^+ \tag{49}$$

Therefore

$$\begin{aligned}\bar{\nu}\mu(z^{(1)}) &= (\bar{x}^+ + \hat{\alpha}\delta_{\bar{x}})^T(\bar{s}^+ + \hat{\alpha}\delta_{\bar{s}}) \overset{(49)}{=} (\bar{x}^+)^T(\bar{s}^+) + \hat{\alpha}\delta_{\bar{x}}^T\psi(\bar{x}^+,\bar{s}^+,\mu(z^+)) \\ &\leq \bar{\nu}\mu(z^+) + \hat{\alpha}\beta^2\mu(z^+) \\ &= \bar{\nu}\mu(z^+)(1+\hat{\alpha}\beta^2\bar{\nu}^{-1})\end{aligned}$$

and hence

$$\begin{aligned}\mu(z^{(2)}) &\leq \mu(z^+)(1+\hat{\alpha}\beta^2\bar{\nu}^{-1})^2 \\ &\overset{(48)}{\leq} \mu(z)(1-\Omega(1/\sqrt{\bar{\nu}}))(1+\hat{\alpha}\beta^2\bar{\nu}^{-1})^2 \\ &= \mu(z)(1-\Omega(1/\sqrt{\bar{\nu}}))\end{aligned}$$

which shows that also $\mu(z)$ is decreased geometrically with a rate of $(1-\Omega(1/\sqrt{\bar{\nu}}))$. Therefore $\mu(z) \leq \epsilon\mu(z^0)$ in $\mathcal{O}(\sqrt{\bar{\nu}}\log(1/\epsilon)) = \mathcal{O}(\sqrt{\nu}\log(1/\epsilon))$ iterations, finishing the proof of Theorem 1.

## References

1. Andersen, E. D., Andersen, K. D.: The MOSEK interior point optimization for linear programming: an implementation of the homogeneous algorithm. In: Frenk, H., Roos, K., Terlaky, T., Zhang, S.: High Performance Optimization, 197–232. Kluwer, Boston (1999).
2. Andersen, E. D., Roos, C., Terlaky, T.: On implementing a primal-dual interior-point method for conic quadratic optimization. Math. Program. 95(2), 249–277 (2003).

3. Andersen, E. D., Ye, Y.: On a homogeneous algorithm for the monotone complementarity problem. Math. Program. 84(2), 375–399 (1999).
4. Ben-Tal, A., Nemirovski, A. S.: Lectures on Modern Convex Optimization: Analysis, Algorithms and Engineering Applications. SIAM, Philadelphia (2001).
5. Boyd, S., Kim, S. J., Vandenberghe, L., Hassibi, A.: A Tutorial on Geometric Programming. Optim. Eng. 8, 67–127 (2007).
6. Butcher, J. C.: Numerical Methods for Ordinary Differential Equations. Wiley, 2nd edition (2008).
7. Chares, P. R.: Cones and Interior-Point Algorithms for Structured Convex Optimization involving Powers and Exponentials. PhD thesis, Uni. Catholique de Louvain, 2009.
8. Grant, M., Boyd, S.: CVX: Matlab software for disciplined convex programming, version 1.21. `http://cvxr.com/cvx`, October 2010.
9. Güler, O.: Barrier Functions in Interior Point Methods. Math. Oper. Res. 21, 860–885 (1996).
10. Karmarkar, N.: A new polynomial-time algorithm for linear programming. Combinatorica 4, 373–395 (1984).
11. Luo, Z. Q., Sturm, J. F., Zhang, S.: Conic convex programming and self-dual embedding. Optim. Method. Softw. 14, 169–218 (2000).
12. Mehrotra, S.: On the Implementation of a Primal-Dual Interior Point Method. SIAM J. Optim. 2, 575–601 (1992).
13. MOSEK Optimization Software: Developed by MOSEK ApS. See `www.mosek.com`.
14. Nemirovski, A. S., Todd, M. J.: Interior-point methods for optimization. Acta Numerica 17, 191–234 (2008).
15. Nesterov, Y. E.: Constructing self-concordant barriers for convex cones. CORE Discussion Paper (2006/30).
16. Nesterov, Y. E.: Towards Nonsymmetric Conic Optimization. Optim. Method. Softw. 27, 893–917 (2012).
17. Nesterov, Y. E., Nemirovski, A. S.: Interior-Point Polynomial Algorithms in Convex Programming. SIAM (1994).
18. Nesterov, Y. E., Todd, M. J.: Self-Scaled Barriers and Interior-Point Methods for Convex Programming. Math. Oper. Res. 22, 1–42 (1997).
19. Nesterov, Y. E., Todd, M. J.: Primal-Dual Interior-Point Methods for Self-Scaled Cones. SIAM J. Optim. 8, 324–364 (1998).
20. Nesterov, Y. E., Todd, M. J., Ye, Y.: Infeasible-Start Primal-Dual Methods and Infeasibility Detectors for Nonlinear Programming Problems. Math. Program. 84, 227–267 (1999).
21. Netlib repository: Collection of linear programs. See `www.netlib.org/lp/`.
22. Nocedal, J., Wright, S. J.: Numerical Optimization. Springer, 2nd edition (2006).
23. Renegar, J.: A Mathematical View of Interior-Point Methods in Convex Optimization. SIAM, Philadelphia (1987).
24. Sturm, J. F.: Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. Optim. Method. Softw. 12, 625–653 (1999).
25. Sturm, J. F.: Implementation of Interior Point Methods for Mixed Semidefinite and Second Order Cone Optimization Problems. Optim. Method. Softw. 17, 1105–1154 (2002).
26. Tuncel, L.: Primal-Dual Symmetry and Scale Invariance of Interior-Point Algorithms for Convex Optimization. Math. Oper. Res. 23, 708–718 (1998).
27. Tuncel, L.: Generalization Of Primal-Dual Interior-Point Methods To Convex Optimization Problems In Conic Form. Found. Comput. Math. 1, 229–254 (2001).
28. Wright, S. J.: Primal-Dual Interior-Point Methods. SIAM (1997).
29. Xu, X., Hung, P. F., Ye, Y.: A simplified homogeneous and self-dual linear programming algorithm and its implementation. Ann. Oper. Res. 62, 151–171 (1996).
30. Xue, G. Ye, Y.: An Efficient Algorithm for Minimizing a Sum of $p$-Norms. SIAM J. Optimiz. 10, 551–579 (1999).
31. Ye, Y.: Interior Point Algorithms: Theory and Analysis. Wiley (1997).

# Electronic supplements: Complete Numerical Results

## 1. $p$-cone Problems

**Table 6** *Computational results for p-cone problems. Data $A \in \mathbb{R}^{M \times N}$ and b from* NETLIB. *sp(A) denotes the sparsity of A.*

| *Problem* | | NPC | | | | | | CVX/SeDuMi | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| name & size | $p$ | $m$ | $n$ | it | ch | ce | st | $m$ | $n$ | ch | st |
| bandm | 1.13 | 777 | 1416 | 9 | 19 | 1.1 | opt | 6913 | 14632 | 21 | opt |
| $M = 305$ | 1.57 | 777 | 1416 | 11 | 23 | 1.1 | opt | 8801 | 18408 | 26 | opt |
| $N = 472$ | 2.09 | 777 | 1416 | 14 | 29 | 1.1 | opt | 9745 | 20296 | 27 | opt |
| sp$(A) = 1.73\%$ | 4.71 | 777 | 1416 | 23 | 37 | 0.6 | opt | 10689 | 22184 | 26 | opt |
| | 7.39 | 777 | 1416 | 24 | 43 | 0.8 | opt | 11633 | 24072 | 26 | opt |
| blend | 1.13 | 188 | 342 | 9 | 19 | 1.1 | opt | 1670 | 3534 | 21 | opt |
| $M = 74$ | 1.57 | 188 | 342 | 9 | 20 | 1.2 | opt | 2126 | 4446 | 22 | opt |
| $N = 114$ | 2.09 | 188 | 342 | 9 | 16 | 0.8 | opt | 2354 | 4902 | 20 | opt |
| sp$(A) = 6.19\%$ | 4.71 | 188 | 342 | 11 | 19 | 0.7 | opt | 2582 | 5358 | 20 | opt |
| | 7.39 | 188 | 342 | 13 | 21 | 0.6 | opt | 2810 | 5814 | 21 | opt |
| bore3d | 1.13 | 565 | 1002 | 7 | 8 | 0.1 | opt | 4907 | 10354 | 6 | opt |
| $M = 231$ | 1.57 | 565 | 1002 | 7 | 8 | 0.1 | opt | 6243 | 13026 | 6 | opt |
| $N = 334$ | 2.09 | 565 | 1002 | 7 | 8 | 0.1 | opt | 6911 | 14362 | 6 | opt |
| sp$(A) = 1.87\%$ | 4.71 | 565 | 1002 | 7 | 8 | 0.1 | opt | 7579 | 15698 | 6 | opt |
| | 7.39 | 565 | 1002 | 7 | 8 | 0.1 | opt | 8247 | 17034 | 6 | opt |
| scagr25 | 1.13 | 1142 | 2013 | 11 | 18 | 0.6 | opt | 9865 | 20801 | 21 | opt |
| $M = 471$ | 1.57 | 1142 | 2013 | 10 | 21 | 1.1 | opt | 12549 | 26169 | 21 | opt |
| $N = 671$ | 2.09 | 1142 | 2013 | 11 | 21 | 0.9 | opt | 13891 | 28853 | 20 | opt |
| sp$(A) = 0.55\%$ | 4.71 | 1142 | 2013 | 10 | 16 | 0.6 | opt | 15233 | 31537 | 19 | opt |
| | 7.39 | 1142 | 2013 | 10 | 21 | 1.1 | opt | 16575 | 34221 | 17 | opt |
| sctap1 | 1.13 | 960 | 1980 | 10 | 21 | 1.1 | opt | 9540 | 20460 | 22 | opt |
| $M = 300$ | 1.57 | 960 | 1980 | 10 | 20 | 1.0 | opt | 12180 | 25740 | 25 | opt |
| $N = 660$ | 2.09 | 960 | 1980 | 9 | 22 | 1.4 | opt | 13500 | 28380 | 21 | opt |
| sp$(A) = 0.95\%$ | 4.71 | 960 | 1980 | 9 | 23 | 1.6 | opt | 14820 | 31020 | 18 | opt |
| | 7.39 | 960 | 1980 | 9 | 20 | 1.2 | opt | 16140 | 33660 | 21 | opt |
| share1b | 1.13 | 370 | 759 | 10 | 21 | 1.1 | opt | 3659 | 7843 | 21 | opt |
| $M = 117$ | 1.57 | 370 | 759 | 12 | 20 | 0.7 | opt | 4671 | 9867 | 26 | opt |
| $N = 253$ | 2.09 | 370 | 759 | 13 | 24 | 0.8 | opt | 5177 | 10879 | 24 | opt |
| sp$(A) = 3.98\%$ | 4.71 | 370 | 759 | 13 | 23 | 0.8 | opt | 5683 | 11891 | 23 | opt |
| | 7.39 | 370 | 759 | 13 | 24 | 0.8 | opt | 6189 | 12903 | 24 | opt |
| share2b | 1.13 | 258 | 486 | 9 | 20 | 1.2 | opt | 2364 | 5022 | 19 | opt |
| $M = 96$ | 1.57 | 258 | 486 | 9 | 18 | 1.0 | opt | 3012 | 6318 | 20 | opt |
| $N = 162$ | 2.09 | 258 | 486 | 9 | 16 | 0.8 | opt | 3336 | 6966 | 20 | opt |
| sp$(A) = 5.00\%$ | 4.71 | 258 | 486 | 11 | 22 | 1.0 | opt | 3660 | 7614 | 20 | opt |
| | 7.39 | 258 | 486 | 11 | 20 | 0.8 | opt | 3984 | 8262 | 20 | opt |
| stocfor1 | 1.13 | 282 | 495 | 9 | 16 | 0.8 | opt | 2427 | 5115 | 19 | opt |
| $M = 117$ | 1.57 | 282 | 495 | 8 | 17 | 1.1 | opt | 3087 | 6435 | 20 | opt |
| $N = 165$ | 2.09 | 282 | 495 | 9 | 19 | 1.1 | opt | 3417 | 7095 | 22 | opt |
| sp$(A) = 2.60\%$ | 4.71 | 282 | 495 | 18 | 30 | 0.7 | opt | 3747 | 7755 | 25 | opt |
| | 7.39 | 282 | 495 | 22 | 29 | 0.3 | opt | 4077 | 8415 | 26 | opt |

# Electronic supplements: Complete Numerical Results

## 2. Facility Location problems

**Table 7** *Results for facility location problems. The algorithm always terminated after reaching optimality as all problem instances were feasible by construction.*

| *Problem* | | | | NPC | | |
|---|---|---|---|---|---|---|
| $N$ | $M$ | $\nu$ | $\bar{p}$ | it | ch | ce |
| 3 | 4 | 44 | 2.03 | 11.1 | 18.2 | 0.65 |
| 3 | 8 | 88 | 1.96 | 14.1 | 22.3 | 0.61 |
| 10 | 4 | 128 | 2.07 | 13.2 | 20.1 | 0.54 |
| 3 | 12 | 132 | 1.93 | 15.3 | 23.4 | 0.56 |
| 3 | 20 | 220 | 2.09 | 17.1 | 27.5 | 0.64 |
| 19 | 4 | 236 | 2.00 | 13.8 | 21.0 | 0.54 |
| 10 | 8 | 256 | 1.98 | 15.6 | 23.4 | 0.51 |
| 10 | 12 | 384 | 2.06 | 16.0 | 25.1 | 0.58 |
| 32 | 4 | 392 | 2.03 | 13.4 | 20.9 | 0.56 |
| 19 | 8 | 472 | 1.98 | 15.2 | 23.1 | 0.53 |
| 10 | 20 | 640 | 1.99 | 18.7 | 30.5 | 0.66 |
| 19 | 12 | 708 | 1.99 | 15.3 | 25.9 | 0.70 |
| 32 | 8 | 784 | 2.04 | 14.0 | 23.3 | 0.67 |
| 32 | 12 | 1176 | 2.05 | 16.4 | 27.0 | 0.65 |
| 19 | 20 | 1180 | 2.01 | 19.7 | 30.5 | 0.60 |
| 32 | 20 | 1960 | 1.98 | 17.7 | 31.5 | 0.79 |

# Electronic supplements: Complete Numerical Results

## 3. Geometric Programs

**Table 8** *Results for geometric programs.*

| *Problem* | | | NPC | | | | mskgpopt | |
|---|---|---|---|---|---|---|---|---|
| name | $n$ | dod | it | ch | ce | st | ch | st |
| beck751 | 7 | 10 | 16 | 30 | 0.9 | opt | 18 | opt |
| beck752 | 7 | 10 | 15 | 29 | 0.9 | opt | 28 | opt |
| beck753 | 7 | 10 | 13 | 27 | 1.1 | opt | 10 | opt |
| bss2 | 2 | 1 | 9 | 13 | 0.4 | opt | 5 | opt |
| car | 37 | 104 | 15 | 28 | 0.9 | opt | 46 | opt |
| demb761 | 11 | 19 | 12 | 22 | 0.8 | ipr | 10 | opt |
| demb762 | 11 | 19 | 9 | 19 | 1.1 | opt | 11 | opt |
| demb763 | 11 | 19 | 10 | 20 | 1.0 | opt | 11 | opt |
| demb781 | 2 | 1 | 7 | 10 | 0.4 | opt | 7 | opt |
| fang88 | 11 | 16 | 9 | 18 | 1.0 | opt | 11 | opt |
| fiac81a | 22 | 50 | 11 | 22 | 1.0 | opt | 16 | opt |
| fiac81b | 10 | 9 | 12 | 21 | 0.8 | ipr | 10 | opt |
| gptest | 4 | 1 | 8 | 12 | 0.5 | opt | 5 | opt |
| jha88 | 30 | 274 | 17 | 34 | 1.0 | opt | 13 | opt |
| mra01 | 61 | 844 | 16 | 30 | 0.9 | opt | 58 | opt |
| mra02 | 126 | 3494 | 30 | 57 | 0.9 | opt | 53 | opt |
| rijc781 | 4 | 1 | 8 | 12 | 0.5 | opt | 5 | opt |
| rijc782 | 3 | 5 | 10 | 18 | 0.8 | opt | 8 | opt |
| rijc783 | 4 | 7 | 12 | 23 | 0.9 | opt | 7 | opt |
| rijc784 | 4 | 3 | 13 | 19 | 0.5 | rnd | 6 | opt |
| rijc785 | 8 | 3 | 9 | 16 | 0.8 | opt | 9 | opt |
| rijc786 | 8 | 3 | 9 | 16 | 0.8 | opt | 6 | opt |
| rijc787 | 7 | 40 | 12 | 23 | 0.9 | opt | 36 | opt |

# Electronic supplements: Complete Numerical Results

## 4. Entropy Problems

**Table 9** *Computational results for entropy problems.*

| *Problem* | | | NPC | | | | mskenopt | |
|---|---|---|---|---|---|---|---|---|
| name | $N$ | $M$ | it | ch | ce | st | ch | st |
| prob | 17 | 15 | 9 | 15 | 0.7 | opt | 8 | opt |
| prob2 | 18 | 14 | 9 | 18 | 1.0 | opt | 8 | opt |
| ento46 | 130 | 21 | 25 | 50 | 1.0 | opt | 42 | opt |
| ento47 | 255 | 21 | 23 | 49 | 1.1 | opt | 54 | opt |
| ento28 | 740 | 16 | 38 | 78 | 1.1 | opt | 63 | opt |
| ento30 | 740 | 21 | 38 | 84 | 1.2 | opt | 55 | opt |
| ento31 | 740 | 21 | 38 | 84 | 1.2 | opt | 55 | opt |
| ento22 | 794 | 28 | 28 | 60 | 1.1 | ipr | 14 | ipr |
| ento21 | 931 | 28 | 55 | 112 | 1.0 | ipr | 18 | ipr |
| a_tb | 1127 | 25 | 38 | 87 | 1.3 | opt | 97 | opt |
| ento23 | 1563 | 28 | 34 | 73 | 1.1 | ipr | 14 | ipr |
| ento20 | 1886 | 28 | 41 | 94 | 1.3 | opt | 21 | ipr |
| a_12 | 2183 | 37 | 46 | 104 | 1.3 | opt | 47 | opt |
| ento12 | 2183 | 37 | 26 | 60 | 1.3 | ipr | 13 | ipr |
| a_13 | 3120 | 37 | 44 | 99 | 1.2 | ipr | 20 | ipr |
| a_23 | 3301 | 37 | 31 | 86 | 1.8 | ipr | 20 | ipr |
| a_34 | 3905 | 37 | 36 | 83 | 1.3 | ipr | 17 | ipr |
| a_14 | 3986 | 37 | 47 | 109 | 1.3 | ipr | 20 | ukn |
| a_35 | 4333 | 37 | 43 | 90 | 1.1 | ipr | 18 | ipr |
| a_bd | 4695 | 26 | 44 | 102 | 1.3 | opt | 78 | opt |
| ento2 | 4695 | 26 | 44 | 102 | 1.3 | opt | 78 | opt |
| a_24 | 5162 | 37 | 36 | 90 | 1.5 | ipr | 23 | ipr |
| ento3 | 5172 | 28 | 49 | 126 | 1.6 | opt | 146 | opt |
| ento50 | 5172 | 28 | 49 | 126 | 1.6 | opt | 146 | opt |
| a_15 | 5668 | 37 | 84 | 176 | 1.1 | opt | 34 | ipr |
| a_25 | 6196 | 37 | 61 | 137 | 1.2 | opt | 122 | opt |
| a_36 | 7497 | 37 | 40 | 99 | 1.5 | ipr | 18 | ipr |
| a_45 | 7667 | 37 | 54 | 120 | 1.2 | opt | 23 | ipr |
| ento26 | 7915 | 28 | 43 | 107 | 1.5 | opt | 131 | opt |
| a_16 | 8528 | 37 | 89 | 204 | 1.3 | opt | 135 | opt |
| a_26 | 9035 | 37 | 39 | 108 | 1.8 | opt | 113 | opt |
| ento45 | 9108 | 37 | 51 | 128 | 1.5 | opt | 149 | opt |
| a_46 | 9455 | 37 | 40 | 102 | 1.6 | ipr | 20 | ipr |
| a_56 | 9702 | 37 | 65 | 158 | 1.4 | opt | 123 | opt |
| ento25 | 10142 | 28 | 116 | 250 | 1.2 | opt | 149 | opt |
| entodif | 12691 | 40 | 50 | 130 | 1.6 | opt | 155 | opt |
| ento48 | 15364 | 31 | 16 | 52 | 2.2 | opt | 47 | opt |

Appendix B

# Paper: Warmstarting the Homogeneous and Self-Dual Interior-Point Method for Linear and Conic Quadratic Problems

DOI: 10.1007/s12532-012-0046-z

Available from:

http://link.springer.com/article/10.1007%2Fs12532-012-0046-z

Math. Prog. Comp.
DOI 10.1007/s12532-012-0046-z

FULL LENGTH PAPER

# Warmstarting the homogeneous and self-dual interior point method for linear and conic quadratic problems

**Anders Skajaa · Erling D. Andersen · Yinyu Ye**

Received: 17 November 2011 / Accepted: 28 July 2012


**Abstract** We present two strategies for warmstarting primal-dual interior point methods for the homogeneous self-dual model when applied to mixed linear and quadratic conic optimization problems. Common to both strategies is their use of only the *final* (optimal) iterate of the initial problem and their negligible computational cost. This is a major advantage when compared to previously suggested strategies that require a pool of iterates from the solution process of the initial problem. Consequently our strategies are better suited for users who use optimization algorithms as black-box routines which usually only output the final solution. Our two strategies differ in that one assumes knowledge only of the final *primal* solution while the other assumes the availability of both primal *and dual* solutions. We analyze the strategies and deduce conditions under which they result in improved theoretical worst-case complexity. We present extensive computational results showing work reductions when warmstarting compared to coldstarting in the range 30–75 % depending on the problem class and magnitude of the problem perturbation. The computational experiments thus substantiate that the warmstarting strategies are useful in practice.

A. Skajaa (✉)
Department of Informatics and Mathematical Modelling,
Technical University of Denmark, 2800 Kgs. Lyngby, Denmark
e-mail: andsk@imm.dtu.dk

E. D. Andersen
MOSEK ApS, Fruebjergvej 3, Box 16, 2100 Copenhagen, Denmark
e-mail: e.d.andersen@mosek.com

Y. Ye
Department of Management Science and Engineering, Stanford University,
Stanford, CA 94305-4121, USA
e-mail: yinyu-ye@stanford.edu

Published online: 19 August 2012

**Keywords** Warmstart · Interior point method · Homogeneous model · Conic programming

**Mathematics Subject Classification** 90C25 · 90C51 · 90C05 · 90C20

## 1 Introduction

The problem of *warmstarting* an optimization algorithm occurs when one needs to solve a sequence of different but presumably related optimization problems. Let $x^*$ denote the solution to an optimization problem $\mathcal{P}$. The aim is then to use the information contained in $x^*$ to initialize the optimization algorithm at a particularly good (or *warm*) point when solving $\hat{\mathcal{P}}$, a related but different problem. Hopefully this will enable us to solve $\hat{\mathcal{P}}$ using less computational effort than had we not known or used $x^*$.

It is widely perceived that it is hard to warmstart interior point methods (IPM). The main reason is that if the solution $x^*$ of $\mathcal{P}$ is on the boundary of the feasible region, then $x^*$ is also close to the boundary for $\hat{\mathcal{P}}$ but not well-centered. At an iterate that is close to the boundary but not well-centered, IPMs generally behave badly producing either ill conditioned linear systems or search directions that allow only tiny step sizes. For that reason, progress towards the solution of $\hat{\mathcal{P}}$ is very slow and often it would have been better to simply coldstart the IPM. For the problem classes usually considered (this work included) $x^*$ is effectively always on the boundary of $\mathcal{P}$.

Different warmstarting strategies for IPMs have previously been studied in e.g. [6–8,10–12,30], most often for the case of Linear Programming (LP). Common to several of these approaches is the requirement of more information from the solution process of $\mathcal{P}$ than just the final solution $x^*$. In both [11] and [30], for example, a pool of primal and dual (non-final) iterates from the solution process of $\mathcal{P}$ is required. Other approaches include (a) further perturbing $\hat{\mathcal{P}}$ to move the boundary and in that way avoid tiny stepsizes [14] and (b) allowing decreasing infeasibility of nonnegativity constraints yielding an "exterior point" method, see e.g. [21]. Computational results from several of the above references are generally positive in that they obtain reductions in the number of interior point iterations on the order of about 50 % when perturbations are not too large. A problem often incurred, however, is a relatively costly procedure to compute the warm point. This is in particular seen in the comparisons of different warmstarting schemes in [12]. Very recently, a warm-starting method based on a *slack-approach* was introduced in [8]. Extra artificial variables are introduced to avoid any of the two above mentioned drawbacks and the method exhibits promising numerical results. For further information about previous work on warmstarting IPMs, see the thorough overview in [8].

The contribution of the present paper is to introduce two warmstart strategies that use *only the final* optimal iterate of the solution of $\mathcal{P}$ and has low computational complexity. One of the strategies, $\mathrm{W_P}$, uses only the primal optimal solution $x^*$ while the other, $\mathrm{W_{PD}}$, uses the primal $x^*$ and the dual optimal solution $(y^*, s^*)$ of $\mathcal{P}$. There are several reasons motivating these schemes. Firstly, optimization software is often used as black-box subroutines that output only *final* iterates. Hence intermediate non-optimal iterates or internal algorithmic variables may not be available at all. In such a situation,

both strategies are useful. Secondly, sometimes just one optimization problem is to be solved, but a user with technical insight into the particular problem may know a good guess for the optimal primal solution. This information should be possible to utilize without requiring a guess for the dual solution as well. In this situation, the strategy $\mathrm{W_P}$ is useful.

It seems sensible to be modest in our expectations about the gains from warmstarting an IPM. Let the linear program $\{\min_x c^T x, \text{ s.t. } Ax = b, x \geq 0\}$ be denoted by $\mathrm{LP}(A, b, c)$ and let $x^*$ be its optimal primal solution. Megiddo [15] showed in 1991 that the existence of a strongly polynomial time algorithm for {given $x^*$, solve $\mathrm{LP}(A, b, c)$} would imply the existence of a strongly polynomial time algorithm for {solve $\mathrm{LP}(A, b, c)$}. Here "solve" means (a) finding an optimal solution *and* a certificate of optimality or (b) certify that no such solution exists. Thus even *checking* whether a given point is primal optimal (even if the point actually is a primal optimal solution) is likely to be as hard as simply solving the problem from scratch.

In this paper we consider general convex conic optimization problems of the form

$$\begin{array}{ll} \min_x & c^T x \\ \text{s.t.} & Ax = b \\ & x \in \mathcal{K} \end{array} \tag{1}$$

where $x, c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$ and $\mathcal{K} \subseteq \mathbb{R}^n$ is a proper cone of the form

$$\mathcal{K} = \mathbb{R}^{n_\ell}_+ \times \mathcal{K}_{\mathrm{q}}^{(q_1)} \times \cdots \times \mathcal{K}_{\mathrm{q}}^{(q_{n_q})}. \tag{2}$$

Here, $\mathbb{R}^{n_\ell}_+$ denotes the positive orthant of dimension $n_\ell$ and $\mathcal{K}_q^{(k)}$ denotes the standard quadratic cone (or the Lorentz cone) of dimension $k$ defined by

$$\mathcal{K}_{\mathrm{q}}^{(k)} = \left\{ x \in \mathbb{R}^k : x_1 \geq \|(x_2, \ldots, x_k)\|_2 \right\} \tag{3}$$

We are further assuming that $m \leq n$ and that $A$ has full row-rank. We have $n = n_\ell + \sum_{j=1}^{n_q} q_j$ and we will be using the notation $\nu = n_\ell + n_q$. Notice that if $q_j = 0$ for all $j$, the problem (1) reduces to an LP in standard form. A problem of the kind (1) is defined by the data $A$, $b$ and $c$ along with the cone $\mathcal{K}$ defined by $n_\ell$ and $q_1, \ldots, q_{n_q}$. We will only consider cases where the cone in $\mathcal{P}$ is identical to the cone in $\hat{\mathcal{P}}$ so we need only consider changes in $A$, $b$ and $c$.

We begin by presenting the Homogeneous and Self-Dual (HSD) model for (1) and its dual in Sect. 2. We then analyze our warmstarting strategies from a theoretical worst-case complexity viewpoint in Sect. 3. In Sect. 4, we describe our IPM. Readers familiar with the standard concepts of homogeneous primal-dual interior point methods for mixed linear and quadratic cones can safely skip Sect. 4. Finally, in Sect. 5, we present extensive computational results and conclude with directions for future work in Sect. 6.

## 2 Homogeneous self-dual model

Convex optimization has a very strong duality theory that connects the primal problem (1) to its *dual*, see e.g. [18]. Because of the strong relations between these two problems, modern interior point methods solve simultaneously the two problems making use of information from one to help progress in the other.

$$\text{PRIMAL}\left\{\begin{array}{ll}\min_x & c^T x\\ \text{s.t.} & Ax = b\\ & x \in \mathcal{K}\end{array}\right. \qquad \text{DUAL}\left\{\begin{array}{ll}\max_{y,s} & b^T y\\ \text{s.t.} & A^T y + s = c\\ & s \in \mathcal{K}^*,\ \ y \in \mathbb{R}^m\end{array}\right. \tag{4}$$

Here, $\mathcal{K}^*$ denotes the *dual cone* of $\mathcal{K}$, but when $\mathcal{K}$ is of the form (2), we have $\mathcal{K} = \mathcal{K}^*$. Therefore we can employ the very efficient primal-dual symmetric interior point methods [19,20]. However, instead of solving (4) directly, we aim to solve the *homogeneous and self-dual* model [28] of problems (4). This problem is slightly larger than the original problem, but in our situation there are enough benefits to offset the modest extra cost incurred. We present this model for the case that $\mathcal{K} = \mathbb{R}^n$, i.e. for linear programming.

For brevity, we will write $z = (x, \tau, y, s, \kappa) \in S := \mathbb{R}^n_+ \times \mathbb{R}_+ \times \mathbb{R}^m \times \mathbb{R}^n_+ \times \mathbb{R}_+$ and we introduce

$$\begin{aligned} r_p(z) &= Ax - b\tau\\ r_d(z) &= -A^T y - s + c\tau\\ r_g(z) &= -c^T x + b^T y - \kappa\\ \mu(z) &= (x^T s + \tau\kappa)/(\nu + 1). \end{aligned}$$

Now let $z^0 = (x^0, \tau^0, y^0, s^0, \kappa^0) \in S$ be some initial point. Assume $\theta$ is a scalar variable. We then consider the problem

$$\left\{\begin{array}{lllllll} \min_{(z,\theta)} & \theta\mu(z^0) & & & & & \\ \text{s.t.} & & & Ax & -b\tau & & = \theta\, r_p(z^0)\\ & -A^T y & & & +c\tau & -s & = \theta\, r_d(z^0)\\ & b^T y & & -c^T x & & -\kappa & = \theta\, r_g(z^0)\\ & r_p(z^0)^T y & & -r_d(z^0)^T x & +r_g(z^0)\tau & & = \mu(z^0)\\ & & & (x,\tau) \geq 0, & (s,\kappa) \geq 0, & (y,\theta) \text{ free.} & \end{array}\right. \tag{5}$$

The following lemma explains the advantages of solving (5) instead of (4):

**Lemma 1** *Assume $(z, \theta)$ is a solution of (5). Then $\theta = 0$ and*

(i) *if $\tau > 0$ then $(x, y, s)/\tau$ is optimal for (4);*
(ii) *if $\kappa > 0$ then, one or both of $b^T y > 0$ and $c^T x < 0$ hold. If the first holds, then (4) is primal infeasible. If the second holds, then (4) is dual infeasible.*

So any solution to (5) with $\tau + \kappa > 0$ provides either an optimal solution to our original problems (4) or a certificate of infeasibility of (one of) the original problems. See [13,25,29] for a proof and further details.

Advantages of using the HSD-model thus include the ability to detect infeasibilities in the problem and particularly the ease of finding a suitable starting point will be of importance to us later. This latter property also eliminates the need for a Phase I procedure in the interior point method.

## 3 Warmstarting

Since (5) is a linear program, we may solve it using any algorithm for linear programming that generates a solution with $\tau + \kappa > 0$. The point $z^0$ used in generating the HSD-model is by construction a feasible point for the problem, so we can use a *feasible-start* IPM initialized in $z^0$ to solve the problem. To obtain the best known complexity, we could, for example, apply the Mizuno-Todd-Ye feasible predictor-corrector interior point algorithm [17]. If we initialize this algorithm in $z^0$, the worst-case iteration complexity to obtain a solution or an infeasibility-certificate to (4) at the tolerance $\epsilon$ (see Sect. 4.3) is given by

$$\mathcal{O}\left(\sqrt{n}\log\left(\Psi(z^0)/\epsilon\right)\right), \quad \text{where } \Psi(z) = \max\left\{\mu(z), \|r_p(z)\|, \|r_d(z)\|\right\}, \tag{6}$$

see particularly [29, pp. 169]. $\|\cdot\|$ denotes some norm. In practice when solving the HSD-model, one usually initializes the algorithm from the point $\mathrm{C} := (e, 1, 0, e, 1)$. Here $e$ denotes the vector of all ones of length $n$ and 0 denotes a zero-vector of length $m$. We will refer to starting from this point as a *cold start*. To obtain a better worst-case complexity, we would need to initialize the algorithm in a point $z^0$ satisfying $\Psi(z^0) < \Psi(\mathrm{C})$, which is certainly satisfied if

$$\mu(z^0) < \mu(\mathrm{C}), \quad \|r_p(z^0)\| < \|r_p(\mathrm{C})\|, \quad \|r_d(z^0)\| < \|r_d(\mathrm{C})\|. \tag{7}$$

For the above complexity result to hold, the initial point $z^0$ must lie in the central-path neighborhood $\mathcal{N}_2(\eta)$, defined by

$$\mathcal{N}_2(\eta) = \{z \in S : \ \|(x \circ s, \tau\kappa) - \mu(e, 1)\|_2 \leq \eta\mu\}, \quad \text{for } \eta \in (0, 1). \tag{8}$$

where $\circ$ denotes *elementwise* product of vectors of equal length. That is, $(v \circ w)_i = v_i w_i$ for all $i$. Since $\mu(\mathrm{C}) = 1$, we clearly have $\mathrm{C} \in \mathcal{N}_2(\eta)$, but we must generally make sure that our initial point is in $\mathcal{N}_2(\eta)$.

### 3.1 Warm starting points

Now let $x^*$ be the primal optimal solution and $(y^*, s^*)$ the dual optimal solution of a linear program $\mathcal{P}$. Further let $\lambda \in [0, 1)$ and $\mu^0 > 0$ be (user chosen) parameters. We propose the following two starting points for the initialization of a related but different linear program $\hat{\mathcal{P}}$:

$$(\mathrm{W_P})\ \begin{cases} x^0 = \lambda x^* + (1-\lambda)e \\ s^0 = \mu^0 (x^0)^{-1} \\ y^0 = 0 \\ \tau^0 = 1 \\ \kappa^0 = \mu^0 \end{cases} \qquad (\mathrm{W_{PD}})\ \begin{cases} x^0 = \lambda x^* + (1-\lambda)e \\ s^0 = \lambda s^* + (1-\lambda)e \\ y^0 = \lambda y^* \\ \tau^0 = 1 \\ \kappa^0 = (x^0)^T s^0 / n \end{cases} \tag{9}$$

Here, $(x^0)^{-1}$ denotes the *elementwise* reciprocal of $x^0$. Much computational experience [1,2,23] indicates that the starting point C seems to work well for the initialization of an interior point method to solve the HSD-model. We can view the starting point $\mathrm{W_{PD}}$ as a convex combination of $(x^*, y^*, s^*)$ and C. Thus, hopefully, $\mathrm{W_{PD}}$ is a point closer (in some sense) to the solution of $\hat{\mathcal{P}}$, but incorporation of $(1-\lambda)\mathrm{C}$ introduces enough centrality to avoid tiny step sizes. The point $\mathrm{W_P}$ is identical to $\mathrm{W_{PD}}$ for the primal variable, but, as we restrict ourselves to using *only primal information*, we cannot do the same for the dual variables. Instead we choose $s^0$ so that the point is perfectly centered and has a prescribed duality gap, namely $\mu^0$.

Since strategy $\mathrm{W_P}$ uses only the primal solution $x^*$, it is especially suited for situations where just one optimization problem is to be solved, but the user may have a qualified guess at a point close to the primal optimal solution or for some other reason, only the primal optimal solution to $\mathcal{P}$ is available. The strategy $\mathrm{W_{PD}}$ uses the full primal-dual solution $(x^*, y^*, s^*)$ and is hence suited for the situation where a sequence of optimization problems is to be solved and a black-box routine for solving (4) that outputs $(x^*, y^*, s^*)$ is used internally as a part of a larger program. We will see several examples of both in Sect. 5.

Our goal in the rest of this section is, for each of the two starting points, to deduce conditions under which they satisfy (7). We remark that (7) are sufficient conditions for $\Psi(z^0) < \Psi(\mathrm{C})$ but not necessary since no attention is paid to which term in (6) is the dominating one.

### 3.2 Comparison of primal and dual residuals

We first introduce some notation consistent with that of [8]. The original LP instance $\mathcal{P}$ consists of the data triplet $d^\circ = (A^\circ, b^\circ, c^\circ)$. We will denote the perturbation by $\Delta d = (\Delta A, \Delta b, \Delta c)$ so that the perturbed problem $\hat{\mathcal{P}}$ has data

$$d = (A, b, c) = d^\circ + \Delta d = (A^\circ, b^\circ, c^\circ) + (\Delta A, \Delta b, \Delta c).$$

As in [8], we will measure the relative magnitude of perturbation between the two problems by the quantities $(\alpha, \alpha', \beta, \gamma)$, defined via:

$$\begin{aligned} \|\Delta A\| &\le \alpha \|A^\circ\| \\ \|\Delta A^T\| &\le \alpha' \|A^{\circ T}\| \\ \|\Delta b\| &\le \beta \|b^\circ\| \\ \|\Delta c\| &\le \gamma \|c^\circ\|. \end{aligned}$$

We are now ready to present three lemmas that, for both $\mathrm{W_P}$ and $\mathrm{W_{PD}}$, state when their primal and dual residuals are smaller than those of $\mathrm{C}$.

Notice that $r_p(\mathrm{W_P}) = r_p(\mathrm{W_{PD}})$ so the lemma below applies to both points:

**Lemma 2** *Define* $\delta_p = \max\{(\|x^*\| + \|e\|)\alpha,\ 2\beta\}$. *If*

$$\delta_p \leq \frac{\|A^\circ e - b^\circ\|}{\|A^\circ\| + \|b^\circ\|}$$

*then*

$$\|r_p(\mathrm{W_P})\| = \|r_p(\mathrm{W_{PD}})\| \leq \|r_p(\mathrm{C})\|.$$

*Proof* If $\lambda = 0$, $r_p(\mathrm{W_P}) = r_p(\mathrm{C})$ so the statement olds trivially. For $\lambda \in (0, 1)$,

$$\begin{aligned} r_p(\mathrm{W_P}) &= Ax^0 - b\tau^0 \\ &= \lambda(Ax^* - b) + (1-\lambda)(Ae - b) \\ &= \lambda(\Delta Ax^* - \Delta b) + (1-\lambda) r_p(\mathrm{C}). \end{aligned}$$

Therefore,

$$\|r_p(\mathrm{W_P})\| \leq \lambda(\alpha\|A^\circ\|\|x^*\| + \beta\|b^\circ\|) + (1-\lambda)\|r_p(\mathrm{C})\|.$$

Similarly,

$$\begin{aligned} \|r_p(\mathrm{C})\| &= \|Ae - b\| = \|A^\circ e - b^\circ + \Delta Ae - \Delta b\| \\ &\geq \|A^\circ e - b^\circ\| - (\alpha\|A^\circ\|\|e\| + \beta\|b^\circ\|) \end{aligned}$$

and therefore,

$$\begin{aligned} \|r_p(\mathrm{C})\| - \|r_p(\mathrm{W_P})\| &\geq \|r_p(\mathrm{C})\| - \lambda(\alpha\|A^\circ\|\|x^*\| + \beta\|b^\circ\|) - (1-\lambda)\|r_p(\mathrm{C})\| \\ &= \lambda\|r_p(\mathrm{C})\| - \lambda(\alpha\|A^\circ\|\|x^*\| + \beta\|b^\circ\|) \qquad \Rightarrow \\ \frac{1}{\lambda}\left(\|r_p(\mathrm{C})\| - \|r_p(\mathrm{W_P})\|\right) &\geq \|r_p(\mathrm{C})\| - (\alpha\|A^\circ\|\|x^*\| + \beta\|b^\circ\|) \\ &\geq \|A^\circ e - b^\circ\| - (\alpha\|A^\circ\|\|e\| + \beta\|b^\circ\|) \\ &\quad -(\alpha\|A^\circ\|\|x^*\| + \beta\|b^\circ\|) \\ &= \|A^\circ e - b^\circ\| - (\|x^*\| + \|e\|)\alpha\|A^\circ\| - 2\beta\|b^\circ\| \\ &\geq \|A^\circ e - b^\circ\| - \delta_p\left(\|A^\circ\| + \|b^\circ\|\right). \end{aligned} \tag{10}$$

The statement then follows after a rearrangement of (10) $\geq 0$. □

We remark that the preceding lemma is very similar in nature to the ones found in [8, sec. 3.1].

The dual parts of $\mathrm{W_P}$ and $\mathrm{W_{PD}}$ are not identical. Let us begin with $\mathrm{W_P}$:

**Lemma 3** *Define* $\psi = \|e\|^{-1}(\|c-e\| - \|c\|)$. *If*

$$\psi \geq \mu^0(1-\lambda)^{-1}$$

*then*

$$\|r_d(\mathrm{W_P})\| \leq \|r_d(\mathrm{C})\|.$$

*Proof* We have $\|r_d(\mathrm{C})\| = \|c-e\|$ and

$$\begin{aligned}\|r_d(\mathrm{W_P})\| &= \|c - \mu^0(x^0)^{-1}\| \\ &\leq \|c\| + \mu^0\|(x^0)^{-1}\| \\ &\leq \|c\| + \mu^0(1-\lambda)^{-1}\|e\|\end{aligned}$$

The statement follows by using this latter inequality to show that $\|r_d(\mathrm{C})\| - \|r_d(\mathrm{W_P})\| \geq 0$ by simple rearrangement. □

The statement for $r_d(\mathrm{W_{PD}})$ is very similar to the one in Lemma 2:

**Lemma 4** *Define* $\delta_d = \max\left\{\alpha'\|y^*\|,\ 2\gamma\right\}$. *If*

$$\delta_d \leq \frac{\|c^\circ - e\|}{\|A^{\circ T}\| + \|c^\circ\|}$$

*then*

$$\|r_d(\mathrm{W_{PD}})\| \leq \|r_d(\mathrm{C})\|.$$

*Proof* If $\lambda = 0$, $r_d(\mathrm{W_{PD}}) = r_d(\mathrm{C})$, so the statement holds trivially. For $\lambda \in (0,1)$, we get from manipulations similar to those in Lemma 2 that

$$\begin{aligned} r_d(\mathrm{W_{PD}}) &= \lambda(-\Delta A^T y^* + \Delta c) + (1-\lambda)r_d(\mathrm{C}) \qquad \Rightarrow \\ \|r_d(\mathrm{W_{PD}})\| &\leq \lambda(\alpha'\|A^{\circ T}\|\|y^*\| + \gamma\|c^\circ\|) + (1-\lambda)\|r_d(\mathrm{C})\|.\end{aligned}$$

Similarly, $\|r_d(\mathrm{C})\| = \|c^\circ + \Delta c - e\| \geq \|c^\circ - e\| - \gamma\|c^\circ\|$. Therefore,

$$\begin{aligned} \|r_d(\mathrm{C})\| - \|r_d(\mathrm{W_{PD}})\| &\geq \lambda\|r_d(\mathrm{C})\| - \lambda(\alpha'\|A^{\circ T}\|\|y^*\| + \gamma\|c^\circ\|) \qquad \Rightarrow \\ \frac{1}{\lambda}\left(\|r_d(\mathrm{C})\| - \|r_d(\mathrm{W_{PD}})\|\right) &\geq \|r_d(\mathrm{C})\| - \left(\alpha'\|A^{\circ T}\|\|y^*\| + \gamma\|c^\circ\|\right) \\ &\geq \|c^\circ - e\| - \gamma\|c^\circ\| - (\alpha'\|A^{\circ T}\|\|y^*\| + \gamma\|c^\circ\|) \\ &= \|c^\circ - e\| - \alpha'\|A^{\circ T}\|\|y^*\| - 2\gamma\|c^\circ\| \\ &\geq \|c^\circ - e\| - \delta_d\left(\|A^{\circ T}\| + \|c^\circ\|\right) \end{aligned} \tag{11}$$

The statement then follows after a rearrangement of (11) $\geq 0$. □

The three preceding lemmas state conditions under which the primal and dual residuals, for each of the two points, are smaller in norm than those of C. Combined with the lemmas in the following section, this will allow us to present conditions under which we obtain an improved worst-case complexity.

### 3.3 Comparison of centrality and complementarity gap

We also need results about the centrality and initial penalty value $\mu(z^0)$ of our warm points.

We start with $\mathrm{W_P}$, for which the situation is particularly simple: We have $\mu(\mathrm{W_P}) = \mu^0$ directly from the definition of $\mathrm{W_P}$. So for a better initial complementarity gap than the cold start, we must choose $\mu^0 \leq 1 = \mu(\mathrm{C})$. Now let us apply this to Lemma 3: Assume that $\psi \geq 0$. The condition in Lemma 3 states

$$\begin{aligned} \mu^0(1-\lambda)^{-1} &\leq \psi && \Leftrightarrow \\ (1-\lambda) &\geq \mu^0/\psi && \Leftrightarrow \\ \lambda &\leq 1-\mu^0/\psi. \end{aligned} \tag{12}$$

Since we must take $\lambda \in [0, 1)$, (12) implies that we must require $\mu^0 \leq \psi$. We remark that the condition of Lemma 3 can only be satisfied if $\psi > 0$, which is a rather strong requirement. Notice also that $\mathrm{W_P}$ is perfectly centered, so it automatically satisfies any neighborhood requirement imposed by the algorithm.

For $\mathrm{W_{PD}}$, the situation is more complicated: Define

$$\xi = e^T(x^* + s^*)/n.$$

We can then state the following lemma which expresses when the initial complementarity of $\mathrm{W_{PD}}$ is smaller than that of C:

**Lemma 5** *If*

$$\begin{aligned} &\xi \in (0, 2] \quad \textit{or} \\ &\xi > 2 \quad \textit{and} \quad \lambda \geq 1 - \frac{1}{\xi - 1} \end{aligned} \tag{13}$$

*then*

$$\mu(\mathrm{W_{PD}}) \leq 1.$$

*Proof* We have

$$\begin{aligned} x^0 \circ s^0 &= (\lambda x^* + (1-\lambda)e) \circ (\lambda s^* + (1-\lambda)e) \\ &= \lambda^2(x^* \circ s^*) + (1-\lambda)^2 e + \lambda(1-\lambda)(x^* + s^*) \\ &= \lambda(1-\lambda)(x^* + s^*) + (1-\lambda)^2 e \end{aligned} \tag{14}$$

where we used that $x^* \circ s^* = 0$. Therefore

$$\begin{aligned}
\mu(\mathrm{w}_{\mathrm{PD}}) &= \frac{(x^0)^T s^0 + \tau^0 \kappa^0}{n+1} = \frac{(x^0)^T s^0 + \frac{(x^0)^T s^0}{n}}{n+1} = \frac{(x^0)^T s^0}{n} \\
&= \frac{1}{n} e^T (x^0 \circ s^0) = \lambda(1-\lambda)\xi + (1-\lambda)^2 \qquad (15) \\
\mu(\mathrm{w}_{\mathrm{PD}}) &\leq 1 \quad \Leftrightarrow \\
\lambda(1-\xi) &\leq 2 - \xi \qquad (16)
\end{aligned}$$

Clearly, (16) holds for $\xi \in [0, 2]$ because $\lambda \in (0, 1)$. If $\xi > 2$, then (16) is equivalent to $\lambda \geq (2-\xi)/(1-\xi) = 1 - 1/(\xi - 1)$. □

Lemma 5 imposes a *lower* bound on $\lambda$ when $\xi > 2$. Notice that as $\xi \to \infty$, the lower bound approaches 1, collapsing the width of the interval for $\lambda$ to zero, because $\lambda \in [0, 1]$.

The situation for $\mathrm{w}_{\mathrm{PD}}$ is further complicated by the fact that it, unlike $\mathrm{w}_{\mathrm{P}}$, is not necessarily in $\mathcal{N}_2(\eta)$. Let us define the quantity

$$\pi = \|\xi^{-1}(x^* + s^*) - e\|_2.$$

The following lemma gives conditions under which $\mathrm{w}_{\mathrm{PD}}$ is sufficiently central.

**Lemma 6** *If*

$$\lambda\xi(\pi - \eta) \leq \eta(1-\lambda) \qquad (17)$$

*then*

$$\mathrm{w}_{\mathrm{PD}} \in \mathcal{N}_2(\eta).$$

*Proof* First notice that $\tau^0\kappa^0 - \mu(\mathrm{w}_{\mathrm{PD}}) = 0$ so this term does not contribute in the norm in (8). Now from (14) and (15) we obtain

$$\begin{aligned}
x^0 \circ s^0 - \mu(\mathrm{w}_{\mathrm{PD}})e &= \lambda(1-\lambda)(x^* + s^*) + (1-\lambda)^2 e \qquad (18) \\
&\quad -\lambda(1-\lambda)\xi e - (1-\lambda)^2 e \\
&= \lambda(1-\lambda)\left(x^* + s^* - \xi e\right) \Rightarrow \\
\|(x^0 \circ s^0, \tau^0\kappa^0) - \mu(e, 1)\| &= \lambda(1-\lambda)\xi\pi
\end{aligned}$$

Therefore using (17):

$$\begin{aligned}
\lambda\xi(\pi - \eta) &\leq \eta(1-\lambda) && \Rightarrow \\
\lambda\xi\pi &\leq \eta(\lambda\xi + (1-\lambda)) && \Rightarrow \\
\lambda(1-\lambda)\xi\pi &\leq \eta(\lambda(1-\lambda)\xi + (1-\lambda)^2) && \Rightarrow \\
\|(x^0 \circ s^0, \tau\kappa) - \mu(e, 1)\| &\leq \eta\mu(\mathrm{w}_{\mathrm{PD}})
\end{aligned}$$

which is the statement. □

We now have a lower bound (13) and an upper bound (17) on $\lambda$ so we can determine conditions under which there is a non-empty interval for $\lambda$ which will imply that $\mathrm{w}_{\mathrm{PD}}$ is sufficiently central and simultaneously has smaller initial complementary gap than $\mathrm{c}$:

**Lemma 7** *Define the following quantities:*

$$q = \frac{\eta}{\xi\pi + \eta(1-\xi)}, \quad \xi_1 = \frac{\xi - 1}{\xi}, \quad \xi_2 = \frac{(\xi-1)^2}{\xi(\xi-2)}, \quad \xi_3 = \xi_1/\xi_2 = \frac{\xi-2}{\xi-1}.$$

*We can then distinguish the following cases, all of which have the same conclusion, which is stated afterwards:*

$$\begin{array}{rll}
1: & \textit{Assume } 0 < \xi \leq 1. & \textit{If } \lambda \in (0, q), \\
2(a): & \textit{Assume } 1 < \xi \leq 2 \ \textit{ and } \pi \leq \eta\xi_1. & \textit{If } \lambda \in (0, 1), \\
2(b): & \textit{Assume } 1 < \xi \leq 2 \ \textit{ and } \pi > \eta\xi_1. & \textit{If } \lambda \in (0, q), \\
3(a): & \textit{Assume } \xi > 2 \ \textit{ and } \pi \leq \eta\xi_1. & \textit{If } \lambda \in (\xi_3, 1), \\
3(b): & \textit{Assume } \xi > 2 \ \textit{ and } \eta\xi_1 < \pi \leq \eta. & \textit{If } \lambda \in (\xi_3, 1), \\
3(c): & \textit{Assume } \xi > 2 \ \textit{ and } \eta < \pi < \eta\xi_2. & \textit{If } \lambda \in (\xi_3, q),
\end{array}$$

*then*

$$\mu(\mathrm{w}_{\mathrm{PD}}) \leq 1 \quad \textit{and} \quad \mathrm{w}_{\mathrm{PD}} \in \mathcal{N}_2(\eta).$$

*Proof* First notice that if $\xi \leq 1$, then Lemma 5 imposes no restriction on $\lambda$, so the lower bound on $\lambda$ is 0. If $\xi \leq 1$, then $1 - \xi \geq 0$ so (17) can be written (after some simple manipulation) as $\lambda \leq q$.

If $1 < \xi \leq 2$ then the lower bound on $\lambda$ is still 0, for the same reason as above. However, (17) may now be written

$$\lambda\,[\xi\pi + \eta(1-\xi)] \leq \eta. \tag{19}$$

The expression in the hard brackets might be negative, which happens if $\pi \leq \eta(\xi - 1)/\xi = \eta\xi_1$. In this case, the condition (19) turns into $\lambda \geq q$, but then $q < 0$, so this is already satisfied for $\lambda \geq 0$. Thus if $\pi \leq \eta\xi_1$, we can allow $\lambda \in (0, 1)$. If on the other hand $\pi > \eta\xi_1$, the expression in the hard brackets of (19) is positive, and we can write it simply as $\lambda \leq q$.

If $\xi > 2$, Lemma 5 requires $\lambda \geq (\xi - 2)/(\xi - 1) = \xi_3$ while Lemma 6 only imposes an upper bound on $\lambda$ if $\pi > \eta\xi_1$. In this case, the two lemmas require $\lambda \in (\xi_3, q)$, which is only a non-empty interval if $q > \xi_3$. This latter inequality holds precisely when $\pi < \eta\xi_2$. This accounts for all cases. □

### 3.4 Summary

Using all of the Lemmas 2–7, we can now summarize the conditions under which we get better worst-case complexity for each of the two points. We begin with $\mathrm{w}_{\mathrm{P}}$:

**Proposition 1** *If*

1. $\delta_p := \max\{(\|x^*\| + \|e\|)\alpha,\ 2\beta\} \leq (\|A^\circ\| + \|b^\circ\|)^{-1}\|A^\circ e - b^\circ\|$
2. $\|c - e\| \geq \|c\|$
3. *we choose* $\mu^0 \in (0, \psi)$ *and finally*
4. *we choose* $\lambda \in (0, 1 - \mu^0/\psi)$

*then starting in* $\mathrm{W_P}$ *results in a better worst-case complexity than a coldstart.*

Similarly for $\mathrm{W_{PD}}$:

**Proposition 2** *If*

1. $\delta_p := \max\{(\|x^*\| + \|e\|)\alpha,\ 2\beta\} \leq (\|A^\circ\| + \|b^\circ\|)^{-1}\|A^\circ e - b^\circ\|$
2. $\delta_d := \max\left\{\alpha'\|y^*\|,\ 2\gamma\right\} \leq (\|A^{\circ T}\| + \|c^\circ\|)^{-1}\|c^\circ - e\|$ *and*
3. *the conditions of one of the six cases of Lemma 7 are satisfied,*

*then starting in* $\mathrm{W_{PD}}$ *results in a better worst-case complexity than a coldstart.*

Thus we have established sufficient conditions under which we have improved worst-case complexity by warmstarting. We are, however, aware of the apparent gap between IPM complexity theory and state-of-the-art implementations, which in most cases perform much better than the worst case complexity estimates. Indeed, the algorithm described in the following sections is in practice usually superior to the predictor-corrector algorithm for which we have just derived complexity estimates relating warmstarts to coldstarts. It is therefore more fruitful to think of the results above as *conceptual* and purely theoretical justifications. That is, these statements should be seen as an attempt to show the existence of conditions under which the warmstarting strategies imply improved worst-case performance for the best-known algorithm in terms of theoretical complexity. However whether the warmstart strategies are effective in practice for the *practically* best-known algorithm shall be determined via computational experiments. For that reason, we devote the rest of the paper to such experiments. In the following section we present the actual algorithm used in experiments. Then, we show a series of computational evidences supporting the effectiveness of the warmstart strategies in Sect. 5.

## 4 Symmetric primal-dual interior point algorithm

To carry out numerical experiments, we have implemented in MATLAB a symmetric primal-dual interior point method called CCOPT. It uses the Nesterov-Todd scaling and Mehrotra's second order correction. Following [2], we give in this section a brief overview of the algorithm. We consider first the case of linear programming, i.e. $\mathcal{K} = \mathbb{R}^n_+$, and then show how we handle the more general quadratic cones (3). A reader familiar with the standard ideas in this algorithm can safely skip this entire section. We use our own implementation instead of other public domain software because it is then easier to modify, control and monitor the algorithm. We remark that all of our source code is publicly available[1] and a reader can therefore reproduce and verify any of the following computational experiments.

[1] http://www2.imm.dtu.dk/~andsk/files/warmstart/downloadcode.html.

### 4.1 Simplified homogeneous self-dual model

Instead of solving (5), our algorithm solves a slightly simpler version known as the *simplified* HSD-model [27]:

$$Ax - b\tau = 0 \tag{20}$$
$$-A^T y - s + c\tau = 0 \tag{21}$$
$$-c^T x + b^T y - \kappa = 0 \tag{22}$$
$$x \geq 0,\ s \geq 0,\ y \in \mathbb{R}^m,\ \tau \geq 0,\ \kappa \geq 0 \tag{23}$$

The HSD-model (5) and the simplified HSD-model (20)–(23) are closely related. See [25,27] for results in this direction. The important points are that we retain the ability to detect infeasibility and our warmstarting strategies are still valid.

### 4.2 Algorithm for linear programming

Assume $z^0 = (x^0, \tau^0, y^0, s^0, \kappa^0) \in \mathcal{K} \times \mathbb{R}_+ \times \mathbb{R}^m \times \mathcal{K} \times \mathbb{R}_+$ is the initial point and $\mu^0 = \mu(z^0)$ its complementarity gap. We then define the central path, parametrized by $\rho \in [0, 1]$, for (20)–(23) by

$$Ax - b\tau = \rho(Ax^0 - b\tau^0) \tag{24}$$
$$-A^T y - s + c\tau = \rho(-A^T y^0 - s^0 + c\tau^0) \tag{25}$$
$$-c^T x + b^T y - \kappa = \rho(-c^T x^0 + b^T y^0 - \kappa^0) \tag{26}$$
$$x \circ s = \rho\mu^0 e \tag{27}$$
$$\tau\kappa = \rho\mu^0 \tag{28}$$

The idea of a primal-dual interior point algorithm for the simplified HSD-model is to loosely track the central path (24)–(28) towards a solution of (20)–(23). Notice that (24)–(26) are the feasibility equations while (27)–(28) are relaxed complementarity conditions. As $\rho \to 0$, we are guided towards an optimal solution for (20)–(23).

In each iteration we compute the direction $(d_x, d_\tau, d_y, d_s, d_\kappa)$ which is the solution to the system of linear equations (29)–(33):

$$Ad_x - bd_\tau = (\sigma - 1)(Ax - b\tau) \tag{29}$$
$$-A^T d_y - d_s + cd_\tau = (\sigma - 1)(-A^T y - s + c\tau) \tag{30}$$
$$-c^T d_x + b^T d_y - d_\kappa = (\sigma - 1)(-c^T x + b^T y - \kappa) \tag{31}$$
$$\tau d_\kappa + \kappa d_\tau = -\tau\kappa + \sigma\mu - d_{\tau\kappa} \tag{32}$$
$$x \circ d_s + s \circ d_x = -x \circ s + \sigma\mu e - d_{xs} \tag{33}$$

where $(x, \tau, y, s, \kappa)$ is the current iterate and $\mu$ its duality gap. The numbers $\sigma$ and $d_{\tau\kappa}$ and the vector $d_{xs}$ are computed by first solving (29)–(33) with $\sigma = d_{\tau\kappa} = 0$ and $d_{xs} = 0$. Let us denote the solution to this (pure Newton) system $(\hat{d}_x, \hat{d}_\tau, \hat{d}_y, \hat{d}_s, \hat{d}_\kappa)$.

We then compute

$$\hat{\alpha} = \max_\alpha \left\{ \alpha : (x, \tau, y, s, \kappa) + \alpha(\hat{d}_x, \hat{d}_\tau, \hat{d}_y, \hat{d}_s, \hat{d}_\kappa) \geq 0 \right\} \tag{34}$$

and set

$$\sigma = (1 - \hat{\alpha}) \min \left( 0.5, (1 - \hat{\alpha})^2 \right) \tag{35}$$

The *Mehrotra second order correctors* [16] $d_{xs}$ and $d_{\tau\kappa}$ are computed by

$$d_{\tau\kappa} = \hat{d}_\tau \hat{d}_\kappa \quad \text{and} \quad d_{xs} = \hat{d}_x \circ \hat{d}_s \tag{36}$$

After computing $\sigma$, $d_{\tau\kappa}$ and $d_{xs}$ by (35)–(36) we compute the final search direction by solving (29)–(33) again but with a now altered right hand side. The iterate is then updated by taking a step of length $\alpha$ in this direction: $(x, \tau, y, s, \kappa) := (x, \tau, y, s, \kappa) + \alpha(d_x, d_\tau, d_y, d_s, d_\kappa)$. It should be stressed that only the right hand side changes so the factorization from the first solve can be used again. The step size $\alpha$ is chosen to be maximal under the conditions that the iterate stays feasible in the cone and that the iterates stay within a certain neighborhood of the central-path. See e.g. [29, pp. 128] for several reasonable definitions of such a neighborhood.

### 4.3 Termination

Assume $(x, \tau, y, s, \kappa)$ is the current iterate and consider the following inequalities:

$$\|Ax - \tau b\|_\infty \leq \epsilon \cdot \max\{1, \|[A, b]\|_\infty\} \tag{P}$$

$$\|A^T y + s - c\tau\|_\infty \leq \epsilon \cdot \max\left\{1, \left\|\left[A^T, I, -c\right]\right\|_\infty\right\} \tag{D}$$

$$\left|-c^T x + b^T y - \kappa\right| \leq \epsilon \cdot \max\left\{1, \|\left[-c^T, b^T, 1\right]\|_\infty\right\} \tag{G}$$

$$\left|c^T x/\tau - b^T y/\tau\right| \leq \epsilon \cdot \left(1 + \left|b^T y/\tau\right|\right) \tag{A}$$

$$\tau \leq \epsilon \cdot 10^{-2} \cdot \max\{1, \kappa\} \tag{T}$$

$$\tau \leq \epsilon \cdot 10^{-2} \cdot \min\{1, \kappa\} \tag{K}$$

$$\mu \leq \epsilon \cdot 10^{-2} \cdot \mu^0 \tag{M}$$

We then terminate and conclude as follows:

| | | |
|---|---|---|
| (OPT) | (P) $\wedge$ (D) $\wedge$ (A) | $\Rightarrow$ Feasible and optimal solution found |
| (INFEAS) | (P) $\wedge$ (D) $\wedge$ (G) $\wedge$ (T) | $\Rightarrow$ Problem primal or dual infeasible |
| (ILLP) | (K) $\wedge$ (M) | $\Rightarrow$ Problem ill-posed |

In case (OPT), the optimal solution $(x, y, s)/\tau$ is returned. If we find (INFEAS), the problem is dual infeasible if $c^T x < 0$ and primal infeasible if $b^T y > 0$. The number $\epsilon > 0$ is a user-specified tolerance.

### 4.4 Generalization to quadratic cones

In order to handle the more general quadratic cones alongside the positive orthant, it is necessary to modify only a few steps in the algorithm in Sect. 4.2. Notationally, this is facilitated by generalizing the product $\circ$ as follows (see e.g. [23] for many more details). First define

$$e_+^k := (1, 1, \ldots, 1)^T \in \mathbb{R}^k$$
$$e_{\mathrm{q}}^k := (1, 0, \ldots, 0)^T \in \mathbb{R}^k$$

and for $x \in \mathbb{R}^k$:

$$\mathrm{mat}_+(x) := \mathrm{diag}(x) \in \mathbb{R}^{k\times k}$$
$$\mathrm{mat}_{\mathrm{q}}(x) := \begin{pmatrix} x_1 & x_{2:k}^T \\ x_{2:k} & x_1 I_{k-1} \end{pmatrix} \in \mathbb{R}^{k\times k}$$

For an $x \in \mathbb{R}_+^{n_\ell} \times \Pi_{j=1}^{n_q} \mathcal{K}_{\mathrm{q}}^{(q_j)}$ partitioned by $x = (x_+, x_{\mathrm{q}}^{(1)}, \ldots, x_{\mathrm{q}}^{(n_q)})$ we then define

$$\mathrm{mat}(x) = \mathrm{mat}_+(x_+) \oplus \mathrm{mat}_{\mathrm{q}}(x_{\mathrm{q}}^{(1)}) \oplus \cdots \oplus \mathrm{mat}_{\mathrm{q}}(x_{\mathrm{q}}^{(n_q)}) \,. \tag{37}$$

where $\oplus$ denotes direct matrix sum. So $\mathrm{mat}(x)$ is a block-diagonal matrix, where the blocks are the individual terms of the right-hand-side of (37). Similarly, we re-define $e := (e_+^{n_\ell}, e_{\mathrm{q}}^{q_1}, \ldots, e_{\mathrm{q}}^{q_{n_q}})$. If $y \in \mathcal{K}$ is partitioned in the same manner as $x$, we finally re-define $\circ$ by

$$x \circ y := \mathrm{mat}(x)\, y$$

and the inverse

$$x^{-1} := \mathrm{mat}(x)^{-1} e.$$

It is easy to see that $x \circ x^{-1} = x^{-1} \circ x = e$.

When applying the algorithm to problems with mixed linear and quadratic cones, the search direction is instead the solution to the linear equations (29)–(32) and the equation

$$\Psi B^{-1} d_s + \Psi B d_x = -\psi \circ \psi + \sigma \mu e - d_{xs}. \tag{38}$$

Here we have introduced the notation $\Psi := \mathrm{mat}(\psi)$ and $\psi = Bx$, where $B$ is a so called *scaling matrix*, chosen to ensure the primal-dual symmetry of the algorithm (see e.g. [24] for more details). Several different choices for $B$ exist but in this algorithm we use the particularly interesting *Nesterov-Todd scaling* [19,20], determined such that $B$ satisfies $Bx = B^{-1}s$. This scaling matrix has proven very efficient in practice [2,23]. The numbers $\sigma$ and $d_{\tau\kappa}$ are determined as in Sect. 4.2, but now $d_{xs}$ is computed by

$$d_{xs} = (B\hat{d}_x) \circ (B^{-1}\hat{d}_s). \tag{39}$$

We remark that all operations involving $B$ can be carried out in $\mathcal{O}(n)$ floating point operations. Thus for example computing $Bx$ or $B^{-1}\hat{d}_s$ is negligible in terms of computational effort. See [2] for more details. The termination criteria are unchanged.

### 4.5 Modelling free variables

Some of the problems in Sect. 5 contain unrestricted (free) variables. Our algorithm handles a free variable $x_f \in \mathbb{R}^{n_f}$ by introducing the extra variable $t$ and adding another standard quadratic cone constraint $t \geq \|x_f\|_2$. The entry in $c$ corresponding to $t$ is set to zero. See [3] for a discussion of this approach.

### 4.6 Solving the linear systems

In each iteration of the homogeneous and self-dual interior point method, linear systems of the type (29)–(33) need to be solved. This system can be solved by block-reducing it to obtain the *normal equations*, a system of the form $ADA^T v = r$ where $D$ is diagonal and strictly positive and $v$ is the unknown, see e.g. [1] for details. The matrix $ADA^T$ is symmetric and positive definite, so we solve the equation using Cholesky factorization.

The matrix $ADA^T$ becomes increasingly ill-conditioned as an optimal point is approached. For this reason, special handling of the factorization is usually employed as the optimal point is approached [26]. In our MATLAB-implementation of CCOPT, we switch from the standard `chol` to `cholinc` if numerical problems are encountered with `chol`. Essentially `cholinc` perturbs small pivots during the Cholesky factorization as is common practice, so the performance penalty is insignificant. This approach often enables us to obtain a higher accuracy of the solution than had we not switched to `cholinc`.

## 5 Numerical results

In this section we present a series of computational results that support the effectiveness of our warmstarting strategies. We first describe the general methodology of our testing and then we present results for linear programs and for mixed linear and quadratic conic problems.

### 5.1 General methodology

When conducting numerical experiments with CCOPT cold- and warmstarted, we use the following procedure. We first solve $\mathcal{P}$ using CCOPT and store the solution $(x^*, y^*, s^*)$. We then perturb $\mathcal{P}$ to obtain the new problem $\hat{\mathcal{P}}$—how we perturb depends on the type of problem and is described in each subsection below. We then solve $\hat{\mathcal{P}}$ using CCOPT coldstarted, denoted CCOPT(C) and CCOPT warmstarted using

just $x^*$ or $(x^*, y^*, s^*)$ in the computation of the warm point, denoted CCOPT($\mathrm{W_P}$) and CCOPT($\mathrm{W_{PD}}$) respectively. For each warmstart, we use the measure

$$\mathcal{R} = \frac{\text{\#Iterations to solve } \hat{\mathcal{P}} \text{ warmstarted}}{\text{\#Iterations to solve } \hat{\mathcal{P}} \text{ coldstarted}}$$

to quantify the gain from warmstarting. If $\mathcal{R} < 1$ the warmstarted run was more efficient than the coldstarted and vice versa. For an entire set of problems $\mathcal{P}_1, \ldots, \mathcal{P}_K$, we define $\mathcal{G}$, the geometric mean of $\mathcal{R}_1, \ldots, \mathcal{R}_K$, i.e.

$$\mathcal{G} = \sqrt[K]{\mathcal{R}_1 \ldots \mathcal{R}_K}$$

Further, we use the following rules:

1. If the solution status of $\mathcal{P}$ was different from that of $\hat{\mathcal{P}}$, the problem was discarded. By solution status we mean either OPT, INFEAS or ILLP—see Sect. 4.3.
2. If $\mathcal{P}$ was primal or dual infeasible, the problem was discarded. In this case there is no reason to expect the final iterate of the algorithm to contain any valuable information for the solution of $\hat{\mathcal{P}}$.
3. If $\hat{\mathcal{P}}$ was solved completely by the *presolve procedures* described in [4], the problem was discarded. In this case, the number of main interior point iterations can be considered zero, making comparison meaningless. This happened only rarely for problems from the NETLIB-LP test set. We used MOSEK[2] to carry out this test.

For linear programs, we have tested our warmstarting strategies both when the solution $(x^*, y^*, s^*)$ to $\mathcal{P}$ was generated by a coldstarted run of CCOPT and when it was generated by a simplex method,[3] which, unlike the IPM, always returns a vertex (basic) solution. The warmstart strategies $\mathrm{W_P}$ and $\mathrm{W_{PD}}$ performed equally well for both cases. This suggests that the IPM is capable of truly using the *information* contained in the solution of $\mathcal{P}$, regardless of whether the final solution is an interior optimal or vertex solution and that the effectiveness of warmstart is not a result of some special "IPM property" of the specific solution produced by CCOPT.

### 5.2 The parameters $\lambda$ and $\mu^0$

In all the following experiments, except the one presented in Sect. 5.6, we use $\lambda = 0.99$ and $\mu^0 = 1 - \lambda = 0.01$. There is no theoretically well-justified reason for this choice. It is a heuristic choice motivated by numerical experience. The experiment in Sect. 5.6 investigates the dependence on the parameter $\lambda$ while using $\mu^0 = 1 - \lambda$. The experiment shows that particularly the performance of $\mathrm{W_P}$ is somewhat sensitive to the choice of $\lambda$. Therefore, it is an interesting topic of future interest to devise an adaptive method to choose the parameters $\lambda$ and $\mu^0$. In the present work, however, we use the static value of $\lambda = 0.99$ (except in Sect. 5.6) and always set $\mu^0 = 1 - \lambda$.

[2] See http://www.mosek.com.

[3] We used the simplex solver in MOSEK.

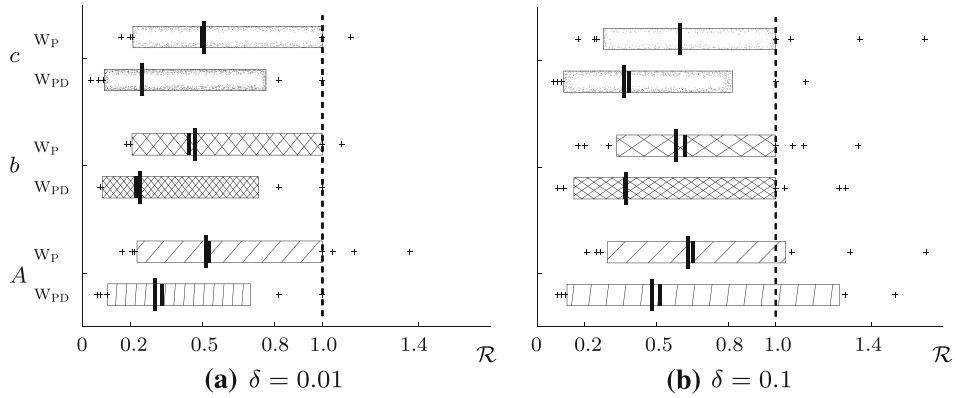



**Fig. 1** Results from the NETLIB LP test set with $\lambda = 0.99$ and $\mu^0 = 0.01$. The box contains 90 % of the problems, pluses are the remaining 10 %. The *dashed line* is $\mathcal{R} = 1.0$. The *largest solid line* is the geometric mean and the *smaller solid line* is the median. The accuracy used was $\epsilon = 10^{-6}$ (cf. Sect. 4.3). See text for further explanation of this figure

## 5.3 Netlib linear programs

In this section we present results from running our warmstarted algorithm on the linear programs in the NETLIB[4] collection of test problems. We perturb the original problem in a manner similar to the one introduced in [6] and reused in [11]: Let $v$ be a vector we want to perturb randomly (think of either $b, c$ or the vector of nonzeros of $A$). Assume $v$ has $M$ elements. An element in $v$ is changed if a [0, 1]-uniform randomly chosen number is less than $\min\{0.1, 20/M\}$.

Thus on average, we change 10 % but at most 20 elements of $v$. An element $v_i$ is changed by setting

$$v_i := \begin{cases} \delta r & \text{if } |v_i| \leq 10^{-6} \\ (1+\delta r)v_i & \text{otherwise} \end{cases}$$

where $r$ is a number chosen randomly from a uniform distribution on $[-1, 1]$. The scalar $\delta$ is a parameter that controls the perturbation magnitude.

We present results for the three cases where $v$ is either $b, c$ or the nonzeros of $A$. Figure 1 shows the value of $\mathcal{R}$ found for each problem in the test set. This was done for all three types of perturbations and for two values of $\delta$. We observe that at these levels of $\delta$, the gain in warmstarting using either strategy is significant. Overall, we see a reduction in the geometric mean of the number of iterations ranging from 34 to 52 % when comparing CCOPT(C) to CCOPT($W_P$) and 50–75 % for CCOPT($W_{PD}$). Usually about one in four problems were discarded because of rules 1–3, Sect. 5.1. Clearly the gain is smaller for the larger value of $\delta$, compare Fig. 1a, b. Figure 2 shows the relation between the magnitude of the perturbation $\delta$ and reduction in the geometric mean of number of iterations. As expected, we clearly observe that the reduction depends crucially on $\delta$. The size of the reduction is significant as long as $\delta$ is small enough. It

[4] http://www.netlib.org/lp/data/.

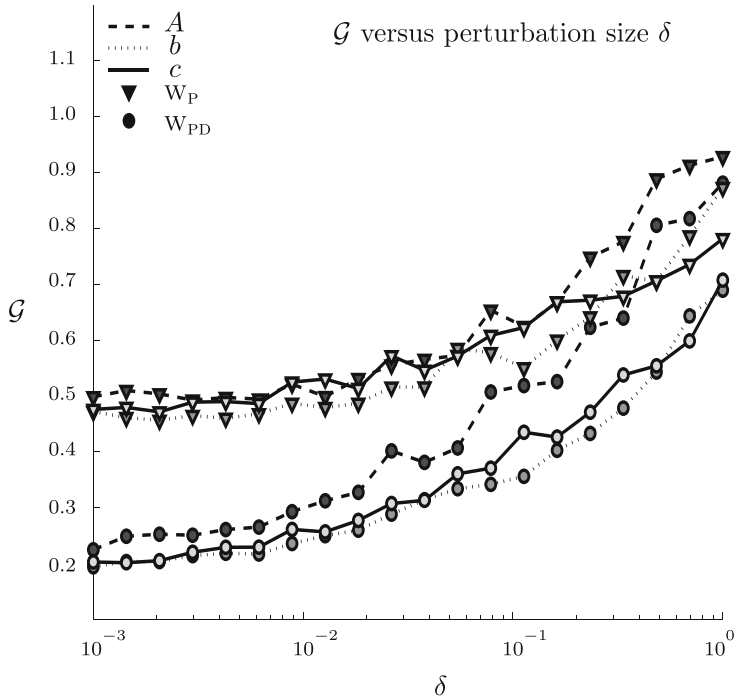



**Fig. 2** Results from the NETLIB-LP test set with $\lambda = 0.99$ and $\mu^0 = 0.01$ and varying $\delta$. Each data point in the figure corresponds to solving the entire NETLIB-LP test set with the problem-perturbation specified in the legend for a certain value of $\delta$. All problems were solved to the accuracy $\epsilon = 10^{-6}$ (cf. Sect. 4.3). See text for further explanation of this figure

is apparent that $\mathrm{W_{PD}}$ is consistently better than $\mathrm{W_P}$. This is of course reasonable since $\mathrm{W_{PD}}$ uses more information from the solution of $\mathcal{P}$ than $\mathrm{W_P}$. Notice, however, that the gap between $\mathrm{W_P}$ and $\mathrm{W_{PD}}$ narrows as $\delta$ grows. This too is reasonable, because as the problem is perturbed more, the information from the primal or the dual points can no longer be expected to be good. Thus both behave more and more like a coldstart.

## 5.4 Efficient frontier computation

An obvious candidate problem on which a warmstarting strategy should be employed is that of computing the efficient frontier in the Markowitz portfolio selection setting. The presentation here follows that of [5].

Assume that $r \in \mathbb{R}^n$ is a multivariate random variable modelling the return of $n$ different assets. Assume further that the mean vector $\mu_r$ and covariance matrix $\Sigma_r$ are known. If our initial holding in asset $j$ is $w_j^0$ and we invest $x_j$, the portfolio after the investment period is $w^0 + x$ and thus the expected return of the investment is $r^T(w^0 + x)$. The risk of the investment is defined as the variance of the return of the investment, namely $(w^0 + x)^T \Sigma_r (w^0 + x) = \|R(w^0 + x)\|_2^2$ where $R$ is a factor in the $QR$-factorization of $\Sigma = QR$. In the classical Markowitz portfolio selection problem, one seeks to minimize risk while fixing a certain return $t$. That is, we solve

$$\begin{array}{rl} \min_x & \|R(w^0+x)\|_2 \\ \text{s.t.} & \bar{r}^T(w^0+x) \quad = t \\ & e^T x \quad = 0 \\ & w^0 + x \quad \geq 0 \end{array} \tag{40}$$

Here, $\bar{r}$ denotes the mean of observed historical realizations of $r$ and $R$ is the triangular factor from the $QR$-factorization of $\bar{X} = (N-1)^{-1/2}(X - e\bar{r}^T)$ where $X \in \mathbb{R}^{N\times n}$ contains the returns of each asset over time. Notice that $\bar{X}$ is a scaled zero-mean version of the observed data in $X$. We do not allow short-selling, so we also impose the constraint $w^0 + x \geq 0$. The problem (40) can be reformulated in conic form as

$$\begin{array}{rl} \min_{z,f,g} & f \\ \text{s.t.} & \bar{r}^T z = t \\ & Rz = g \\ & e^T z = e^T w^0 \\ & f \geq \|g\|_2 \\ & z \geq 0 \end{array} \tag{41}$$

and it is this version that we are solving using CCOPT. The solution $x$ is then obtained via $z = x + w^0$. Let $f(t)$ denote the optimal value of (41) for a requested return of $t$. The set of points $(t, f(t))$ for $t \in [0, \max(\bar{r})]$ is called the *efficient frontier*. To compute this curve, we must solve a sequence of problems of the type (41) where only $t$ varies from problem to problem—thus this entire computation is very well suited for a warmstarting scheme: Compute the optimal solution of (41) for the first value of $t$ and compute a warm starting point using this solution as described in Sect. 3.1. Then solve (41) for the next value of $t$, initializing the algorithm in the warm starting point. We can then repeat this process for all following values of $t$ using the solution of (41) for the previous value of $t$ to compute a warm starting point for the next problem.

We use as the data matrix $X$ the historically observed data from $N$ daily prices for the 500 stocks in the S&P500 stock index.[5] With $N = 800$, (41) is a problem of the type (1) with $A \in \mathbb{R}^{502\times 1{,}002}$ and $\text{nnz}(A) = 126{,}750$. The results are shown in Table 1. We see that the work is reduced by about 25 % when using $\text{W}_\text{P}$ and by about 60 % if we use $\text{W}_\text{PD}$.

### 5.5 Frequent robust portfolio rebalancing

The Markowitz portfolio selection problem presented in the previous section can be further generalized by assuming that the data $X$ are uncertain but belong to known uncertainty sets. The *robust* portfolio selection problem consists in choosing the best possible portfolio while assuming that the worst case scenario within the uncertainty sets is realized. The optimal such portfolio is the solution of a second order cone program (SOCP). For a complete description of the model, see [9]—here we omit a detailed description of the model as it is not the primary interest of this paper.

[5] See e.g. http://www.standardandpoors.com/indices/main/en/us.

**Table 1** Results from solving a series of Markowitz portfolio optimization problems, combined comprising an efficient frontier

| CCOPT(C) | | | CCOPT($W_P$) | | CCOPT($W_{PD}$) | |
|---|---|---|---|---|---|---|
| $t$ | $f(t)$ | Iters | Iters | $\mathcal{R}$ | Iters | $\mathcal{R}$ |
| 1.00000 | 0.0042 | 14 | 14 | 1.00 | 14 | 1.00 |
| 1.00013 | 0.0037 | 16 | 16 | 1.00 | 8 | 0.50 |
| 1.00027 | 0.0038 | 14 | 13 | 0.93 | 8 | 0.57 |
| 1.00040 | 0.0042 | 14 | 12 | 0.86 | 7 | 0.50 |
| 1.00053 | 0.0050 | 16 | 14 | 0.88 | 6 | 0.38 |
| 1.00067 | 0.0058 | 15 | 13 | 0.87 | 6 | 0.40 |
| 1.00080 | 0.0068 | 14 | 14 | 1.00 | 7 | 0.50 |
| 1.00093 | 0.0078 | 14 | 12 | 0.86 | 7 | 0.50 |
| 1.00107 | 0.0089 | 14 | 12 | 0.86 | 6 | 0.43 |
| 1.00120 | 0.0101 | 19 | 11 | 0.58 | 6 | 0.32 |
| 1.00133 | 0.0114 | 16 | 12 | 0.75 | 6 | 0.38 |
| 1.00147 | 0.0127 | 14 | 11 | 0.79 | 5 | 0.36 |
| 1.00160 | 0.0141 | 14 | 10 | 0.71 | 6 | 0.43 |
| 1.00173 | 0.0158 | 19 | 9 | 0.47 | 6 | 0.32 |
| 1.00187 | 0.0177 | 15 | 10 | 0.67 | 5 | 0.33 |
| 1.00200 | 0.0197 | 14 | 9 | 0.64 | 5 | 0.36 |
| 1.00213 | 0.0219 | 14 | 10 | 0.71 | 5 | 0.36 |
| 1.00227 | 0.0242 | 14 | 8 | 0.57 | 5 | 0.36 |
| 1.00240 | 0.0265 | 13 | 10 | 0.77 | 4 | 0.31 |
| 1.00253 | 0.0289 | 14 | 9 | 0.64 | 4 | 0.29 |
| 1.00267 | 0.0313 | 11 | 9 | 0.82 | 4 | 0.36 |
| 1.00280 | 0.0338 | 12 | 10 | 0.83 | 5 | 0.42 |
| 1.00293 | 0.0363 | 12 | 8 | 0.67 | 4 | 0.33 |
| 1.00307 | 0.0388 | 12 | 8 | 0.67 | 5 | 0.42 |
| 1.00320 | 0.0414 | 12 | 8 | 0.67 | 5 | 0.42 |
| | $\mathcal{G}$ | 14.1 | 10.7 | 0.76 | 5.7 | 0.41 |

We used $\lambda = 0.99$ and $\mu^0 = 0.01$. The third column shows the number of iterations spent solving the problem using CCOPT from a coldstart. The two column blocks to the right show the performance of $W_P$ and $W_{PD}$. All problems were solved to the accuracy $\epsilon = 10^{-6}$ (cf. Sect. 4.3)

Instead, we focus on the following situation. On a certain trading day, we can estimate the return and variance of each asset and their uncertainty sets from historical data, for example from the past $H$ trading days. This is done as in Sect. 5.4 (see [9] for estimation of the uncertainty sets). We can then compute a robust portfolio by solving the corresponding SOCP. A number of trading days later (say, $k$ days), we repeat this procedure, estimating the relevant parameters over an equally long backwards time horizon, which is now shifted by $k$ days. If $k$ is small compared to $H$, the new estimates of the parameters are likely to be only slightly different from the previous ones. Therefore we can compute a warm starting point using the solution of the previous problem. This procedure can then be repeated.

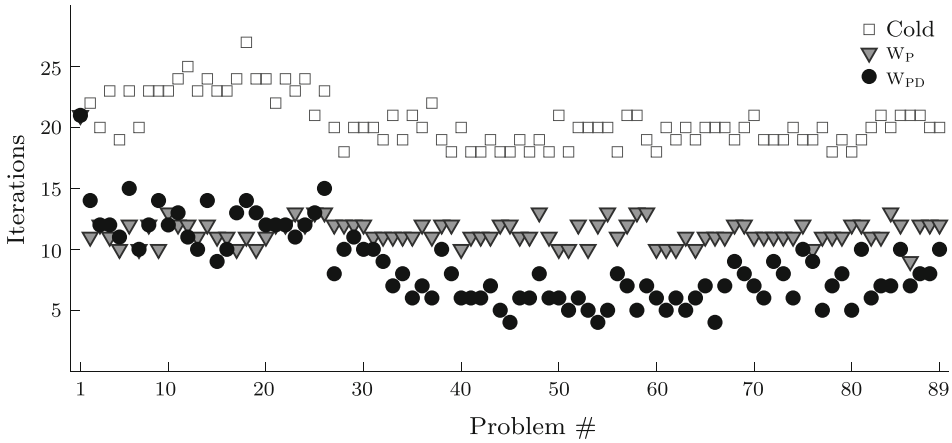



**Fig. 3** Results from the portfolio rebalancing problems. The problem set contains 89 problems. The figure shows the number of iterations spent solving each problem from a cold start (*squares*), the point $w_P$ (*triangles*) and the point $w_{PD}$ (*circles*) computed using the solution of the previous problem. We used $\lambda = 0.99$ and $\mu^0 = 0.01$ for all problems

To facilitate future research in the field of warmstarting optimization algorithms for SOCPs, we have generated a sequence of such problems using data from 2761 consecutive trading days from the stocks in the S&P500 stock index. Starting on day number 1,001, we estimated returns and uncertainty sets over the past $H = 1{,}000$ days and repeated this procedure for every $k = 20$ trading days. The result is a problem set consisting of 89 neighboring SOCPs each with 2,531 variables and 1,527 linear constraints, of which only two do not come from the introduction of slack variables. Of the 2,531 variables, 2,005 are non-negative and the rest belong to quadratic cones of dimensions 3, 21 and 502. The problems are stored in SeDuMi format (see [22]) in the MATLAB binary `.mat`-format and they are publicly available.[6]

Figure 3 shows the performance of $w_P$ and $w_{PD}$ on this set of problems. We see that each problem is usually solved in about 20 iterations by CCOPT when started from a coldstart. Using warmstart from $w_P$ reduces the number of iterations to about 10–13. Warmstarting from $w_{PD}$ reduces the number even further to the range 4–15 iterations. The quantity $\mathcal{G}$ (defined in Sect. 5.1) for $w_P$ and $w_{PD}$ was 0.5590 and 0.3985 respectively. We can conclude that for these problems, our warmstarting strategies are highly effective.

### 5.6 Minimal norm vector in convex hull

In certain algorithms called *bundle methods* employed particularly in the field of nonsmooth optimization, a series of vectors (gradients at the iterates) are stored (in a *bundle*) and used in computing the next search direction and sometimes used to check stopping criteria. If the current bundle contains $g_1, \ldots, g_k \in \mathbb{R}^n$, usually we will have $k \ll n$. At every iteration of these algorithms, the vector with minimal norm in the

[6] http://www2.imm.dtu.dk/~andsk/files/warmstart/robpfrebalancing_probs.html.

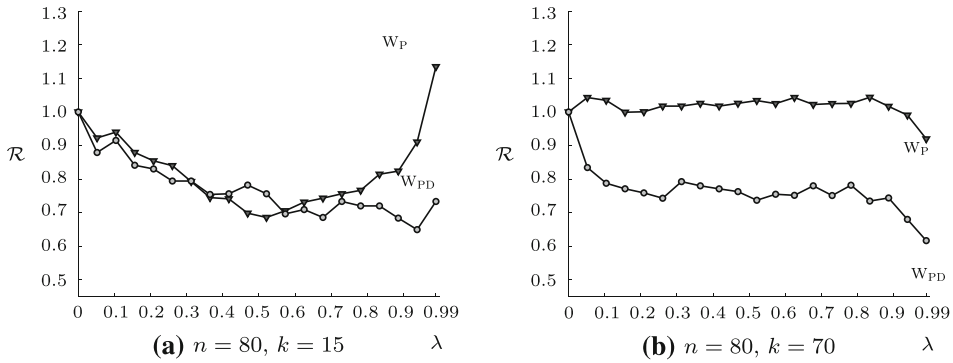



**Fig. 4** Results from solving (42). Geometric means of $\mathcal{R}$ over 10 random instances are shown. We used the tolerance $\epsilon = 10^{-6}$ (cf. Sect. 4.3) and always used $\mu^0 = 1 - \lambda$. *Triangles* denote $\mathrm{W_P}$, *circles* denote $\mathrm{W_{PD}}$

convex hull of the vectors $g_1, \ldots, g_k$ is needed. At the end of each iteration, the bundle is updated, for example by removing one vector and replacing it by another one. We thus get a sequence of related optimization problems to solve—hence another suitable candidate for a warmstarting strategy.

Let $G \in \mathbb{R}^{n \times k}$ be a matrix with $g_1, \ldots, g_k$ in the columns. The problem of finding the minimal norm vector in the convex hull of $g_1, \ldots, g_k$ can be formulated as

$$\begin{cases} \min_x \ \|Gx\|_2 \\ \quad \text{s.t.} \quad e^T x = 1 \\ \qquad\qquad x \geq 0 \end{cases} \quad \text{or} \quad \begin{cases} \min_{(x,t,y)} \quad t \\ \qquad \text{s.t.} \ \ Gx = y \\ \qquad\quad e^T x = 1 \\ \qquad\qquad x \geq 0 \\ \qquad\qquad t \geq \|y\|_2 \end{cases} \tag{42}$$

The formulation on the right is in the standard conic form (1). If $x^*$ solves this problem then $Gx^*$ is the vector we seek. Using the notation of (1), we see that modifying $G$ corresponds to changing the constraint matrix $A$ of the problem. We experiment numerically with this problem by first generating $G \in \mathbb{R}^{n \times k}$ randomly from a $[-1, 1]$-uniform distribution and then solving the problem coldstarted—the solution is used in computing the warm points for the modified problem. We then change one entire column of $G$ to a vector in $\mathbb{R}^n$ randomly chosen from the $[-1, 1]$-uniform distribution. The new problem is then solved both cold- and warmstarted for 20 equidistantly distributed $\lambda \in [0, 0.99]$. All this is done for 10 random instances, for $n = 80$ and two values of $k$. The results (geometric means over the 10 random instances) are shown in Fig. 4. We clearly see, particularly for $\mathrm{W_P}$, that the best value of $\lambda$ depends on the problem (in this case on $k$). Again $\mathrm{W_{PD}}$ consistently performs better than $\mathrm{W_P}$, producing improvements in the range 20–40 % depending on problem and $\lambda$.

## 6 Conclusion and future work

In this paper, we have presented two new warmstarting strategies particularly well suited for homogeneous interior point methods to solve convex conic optimization

problems involving linear and quadratic cones. We have analyzed them and given conditions under which each of them results in improved performance over a standard coldstart. In contrast to several previous warmstarting strategies, one of our strategies uses only the primal optimal point of the previous problem to solve the next. The other strategy uses only the primal and dual optimal solution but no intermediate iterates. This is significant in that it allows users of black-box optimization algorithms to apply our warmstarting strategy as part of a larger program where a series of related optimization problems are subproblems that need to be solved. A further benefit of our strategies is that they cost virtually nothing to compute.

We have presented extensive computational experiments with our warmstarting strategies showing work reductions in the range of 30–75 %. Thus the strategies are very effective in practice. This was shown both for linear programming problems and quadratic programming problems, which we formulated as general mixed linear and quadratic cone problems.

Our results apply to an interior point method used to solve the homogeneous model. It is an interesting question whether the presented warmstarting strategies would work equally well when used in a primal-dual interior point method applied to solve the original primal-dual pair of conic programs.

Using the general convex conic format, we expect to be able to easily generalize our warmstarting strategies to the context of semidefinite programming. This step simply involves the already known generalization of the Jordan product $\circ$ to the cone of symmetric and semidefinite matrices, similar to what was done in Sect. 4.4 for the quadratic cones. For that reason, we expect our strategies to also be useful in algorithms for solving combinatorial optimization problems. Here, problems are often reduced to solving a series of related simpler continuous problems such as linear programs, quadratic programs or semidefinite programs. Thus warmstarting is an obvious idea to improve computational performance. In this situation, the number of variables in $\mathcal{P}$ and $\hat{\mathcal{P}}$ may be different. In case it increases, we can add in components from the standard cold starting point $\mathrm{C}$ in appropriate places. If the number of variables on the other hand decreases, we simply drop those variables from the warm starting point.

## References


1. Andersen, E.D., Andersen, K.D. : The MOSEK interior point optimization for linear programming: an implementation of the homogeneous algorithm. In: Frenk, H., Roos, K., Terlaky, T., Zhang, S. (eds.) High Performance Optimization, pp. 197–232. Kluwer, Dordrecht (1999)
2. Andersen, E.D., Roos, C., Terlaky, T.: On implementing a primal-dual interior-point method for conic quadratic optimization. Math. Program. **95**(2), 249–277 (2003)
3. Andersen, E.D.: Handling free variables in primal-dual interior-point methods using a quadratic cone. Available from http://www.mendeley.com/c/4812865462/p/11467401/andersen-2002-handling-free-variables-in-methods-using-a-quadratic-cone/ (2002)
4. Andersen, E.D., Andersen, K.D.: Presolving in linear programming. Math. Program. **71**(2), 221–245 (1995)
5. Andersen, E.D., Dahl, J., Friberg, H.A.: Markowitz portfolio optimization using MOSEK. MOSEK Technical report: TR-2009-2 (2011)
6. Benson, H.Y., Shanno, D.F.: An exact primal-dual penalty method approach to warmstarting interior-point methods for linear programming. Comput. Optim. Appl. **38**, 371–399 (2007)

7. Colombo, M., Gondzio, J., Grothey, A.: A warm-start approach for large-scale stochastic linear programs. Math. Program. **127**(2), 371–397 (2011)
8. Engau, A., Anjos, M.F., Vannelli, A.: On interior-point warmstarts for linear and combinatorial optimization. SIAM J. Optim. **20**(4), 1828–1861 (2010)
9. Goldfarb, D., Iyengar, G.: Robust portfolio selection problems. Math. Oper. Res. **28**(1), 1–38 (2003)
10. Gondzio, J., Grothey, A.: Reoptimization with the primal-dual interior point method. SIAM J. Optim. **13**, 842–864 (2002)
11. Gondzio, J., Grothey, A.: A new unblocking technique to warmstart interior point methods based on sensitivity analysis. SIAM J. Optim. **19**(3), 1184–1210 (2008)
12. John, E., Yildirim, E.A.: Implementation of warm-start strategies in interior-point methods for linear programming in fixed dimension. Comput. Optim. Appl. **41**, 151–183 (2008)
13. Luo, Z.Q., Sturm, J.F., Zhang, S.: Conic convex programming and self-dual embedding. Optim. Methods Softw. **14**(3), 169–218 (2000)
14. Lustig, I.J., Marsten, R.E., Shanno, D.F.: Interior point methods for linear programming: computational state of the art. ORSA J. Comput. **6**(1), 1–14 (1994)
15. Megiddo, N.: On finding primal- and dual-optimal bases. ORSA J. Comput. **3**(1), 63–65 (1991)
16. Mehrotra, S.: On the implementation of a primal-dual interior point method. SIAM J. Optim. **2**(4), 575–601 (1992)
17. Mizuno, S., Todd, M.J., Ye, Y.: On adaptive-step primal-dual interior-point algorithms for linear programming. Math. Oper. Res. **18**(4), 964–981 (1993)
18. Nesterov, Y.E., Nemirovski, A.S.: Interior-Point Polynomial Algorithms in Convex Programming. SIAM, Philadelphia, PA (1994)
19. Nesterov, Y.E., Todd, M.J.: Self-scaled barriers and interior-point methods for convex programming. Math. Oper. Res. **22**(1), 1–42 (1997)
20. Nesterov, Y.E., Todd, M.J.: Primal-dual interior-point methods for self-scaled cones. SIAM J. Optim. **8**(2), 324–364 (1998)
21. Polyak, R.: Modified barrier functions (theory and methods). Math. Program. **54**, 177–222 (1992)
22. Sturm, J.F.: Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. Optim. Methods Softw. **12**, 625–653 (1999)
23. Sturm, J.F.: Implementation of interior point methods for mixed semidefinite and second order cone optimization problems. Optim. Methods Softw. **17**(6), 1105–1154 (2002)
24. Tuncel, L.: Primal-dual symmetry and scale invariance of interior-point algorithms for convex optimization. Math. Oper. Res. **23**(3), 708–718 (1998)
25. Wright, S.J.: Primal-Dual Interior-Point Methods . SIAM, Philadelphia, PA (1987)
26. Wright, S.J.: Modified cholesky factorizations in interior-point algorithms for linear programming. SIAM J. Optim. **9**(4), 1159–1191 (1999)
27. Xu, X., Hung, P.F., Ye, Y.: A simplified homogeneous and self-dual linear programming algorithm and its implementation. Ann. Oper. Res. **62**(1), 151–171 (1996)
28. Ye, Y., Todd, M.J., Mizuno, S.: An O(sqrt(n)L)-iteration homogeneous and self-dual linear programming algorithm. Math. Oper. Res. **19**(1), 53–67 (1994)
29. Ye, Y.: Interior Point Algorithms: Theory and Analysis. Wiley-Interscience, New York (1997)
30. Yildirim, E.A., Wright, S.J.: Warm-start strategies in interior-point methods for linear programming. SIAM J. Optim. **12**(3), 782–810 (2002)

# Bibliography


[1] Andersen, E. D., Andersen, K. D.: The MOSEK interior point optimization for linear programming: an implementation of the homogeneous algorithm. In: Frenk, H., Roos, K., Terlaky, T., Zhang, S.: High Performance Optimization, 197–232. Kluwer, Boston (1999).

[2] Andersen, E. D., Roos, C., Terlaky, T.: On implementing a primal-dual interior-point method for conic quadratic optimization. Math. Program. 95(2), 249–277 (2003).

[3] Andersen, E. D., Ye, Y.: On a homogeneous algorithm for the monotone complementarity problem. Math. Program. 84(2), 375–399 (1999).

[4] Andersen, K. D.: A modified Schur complement method for handling dense columns in interior-point methods for linear programming. ACM Trans. Math. Software 22(3), 348–356 (1996).

[5] Andersen, M. S., Dahl, J., Vandenberghe, L.: Logarithmic barriers for sparse matrix cones. Optimization Methods and Software 28(3), 396–423 (2012).

[6] Arnold, E., Neupert, J., Sawodny, O., Schneider, K.: Trajectory tracking for boom cranes based on nonlinear control and optimal trajectory generation. In Proc. IEEE International Conference on Control Applications —, 1444–1449 (2007).

[7] Ben-Tal, A., Nemirovski, A. S.: Lectures on Modern Convex Optimization: Analysis, Algorithms and Engineering Applications. SIAM, Philadelphia (2001).

[8] Benson, H.Y., Shanno, D.F.: An exact primal-dual penalty method approach to warmstarting interior-point methods for linear programming. Comput. Optim. Appl. 38, 371–399 (2007).

[9] Bertsimas, D., Tsitsiklis, J. N.: Introduction to Linear Optimization. Athena Scientific (1997).

[10] Boyd, S., Kim, S. J., Vandenberghe, L., Hassibi, A.: A Tutorial on Geometric Programming. Optim. Eng. 8, 67–127 (2007).

[11] Butcher, J. C.: Numerical Methods for Ordinary Differential Equations. Wiley, 2nd edition (2008).

[12] Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press (2004).

[13] Chares, P. R.: Cones and Interior-Point Algorithms for Structured Convex Optimization involving Powers and Exponentials. PhD thesis, Uni. Catholique de Louvain, 2009.

[14] Colombo, M., Gondzio, J., Grothey, A.: A warm-start approach for large-scale stochastic linear programs. Math. Program. 127(2), 371–397 (2011).

[15] El-Bakry, A. S., Tapia, R.A., Tsuchiya, T., Zhang, Y.: On the formulation and theory of the Newton interior-point method for nonlinear programming. J. Optim. Theory Appl. 89(3), 507–541 (1996).

[16] Engau, A., Anjos, M.F., Vannelli, A.: On Interior-Point warmstarts for linear and combinatorial optimization. SIAM J. Optim. 20(4), 1828–1861 (2010).

[17] Ferreau, H. J.: An Online Active Set Strategy for Fast Solution of Parametric Quadratic Programs with Applications to Predictive Engine Control. MSc thesis, Uni. of Heidelberg, 2006.

[18] Ferreau, H.J., Bock, H.G., Diehl, M.: An online active set strategy to overcome the limitations of explicit MPC. International Journal of Robust and Nonlinear Control 18(8), 816–830 (2008).

[19] Fukuda, M., Braams, B. J., Nakata, M., Overton, M. L., Percus, J. K., Yamashita, M., Zhao, Z.: Large-Scale Semidefinite Programs in Electronic Structure Calculation. Math. Program. 109(2–3), 553–580 (2007).

[20] George, A., Liu, J. W. H.: The evolution of the minimum degree ordering algorithm. SIAM Rev. 31, 1–19 (1989).

[21] Glineur, F.: Topics in Convex Optimization: Interior-Point Methods, Conic Duality and Approximations. PhD thesis, Faculté Polytechnique de Mons.

[22] Gondzio, J., Grothey, A.: Reoptimization with the Primal-Dual interior point method. SIAM J. Optim. 13, 842–864 (2002).

[23] Gondzio, J., Grothey, A.: A new unblocking technique to warmstart interior point methods based on sensitivity analysis. SIAM J. Optim. 19(3), 1184–1210 (2008).

[24] Grant, M., Boyd, S.: CVX: Matlab software for disciplined convex programming, version 1.21. http://cvxr.com/cvx, October 2010.

[25] Güler, O.: Barrier Functions in Interior Point Methods. Math. Oper. Res. 21, 860–885 (1996).

[26] Hovgaard, T. G., Edlund, K., Jørgensen, J. B.: The Potential of Economic MPC for Power Management. In Proc. of 49th IEEE Conference on Decision and Control, 2010 —, 7533–7538 (2010).

[27] John, E., Yildirim, E. A.: Implementation of warm-start strategies in interior-point methods for linear programming in fixed dimension. Comput. Optim. Appl. 41, 151–183 (2008).

[28] Karmarkar, N.: A new polynomial-time algorithm for linear programming. Combinatorica 4, 373–395 (1984).

[29] Luenberger, D. G., Ye, Y.: Linear and Nonlinear Programming. Springer (2008).

[30] Luo, Z. Q., Sturm, J. F., Zhang, S.: Conic convex programming and self-dual embedding. Optim. Method. Softw. 14, 169–218 (2000).

[31] Lustig, I.J., Marsten, R.E., Shanno, D.F.: Interior point methods for linear programming: Computational state of the art. ORSA J. Comput. 6(1), 1–14 (1994).

[32] Mattingley, J., Boyd, S.: CVXGEN: A Code Generator for Embedded Convex Optimization. Optimization and Engineering 13(1), 1–27 (2012).

[33] Megiddo, N.: On finding primal- and dual-optimal bases. ORSA J. Comput. 3(1), 63–65 (1991).

[34] Mehrotra, S.: On the Implementation of a Primal-Dual Interior Point Method. SIAM J. Optim. 2, 575–601 (1992).

[35] Mizuno, S., Todd, M. J., Ye, Y.: On adaptive-step primal-dual interior-point algorithms for linear programming. Math. Oper. Res. 18, 964–981 (1993).

[36] MOSEK Optimization Software: Developed by MOSEK ApS. See www.mosek.com.

[37] Nemirovski, A. S., Todd, M. J.: Interior-point methods for optimization. Acta Numerica 17, 191–234 (2008).

[38] Nesterov, Yu. E.: Constructing self-concordant barriers for convex cones. CORE Discussion Paper (2006/30).

[39] Nesterov, Yu. E.: Towards Nonsymmetric Conic Optimization. Optim. Method. Softw. 27, 893–917 (2012).

[40] Nesterov, Yu. E., Nemirovski, A. S.: Interior-Point Polynomial Algorithms in Convex Programming. SIAM (1994).

[41] Nesterov, Yu. E., Todd, M. J.: Self-Scaled Barriers and Interior-Point Methods for Convex Programming. Math. Oper. Res. 22, 1–42 (1997).

[42] Nesterov, Yu. E., Todd, M. J.: Primal-Dual Interior-Point Methods for Self-Scaled Cones. SIAM J. Optim. 8, 324–364 (1998).

[43] Nesterov, Yu. E., Todd, M. J., Ye, Y.: Infeasible-Start Primal-Dual Methods and Infeasibility Detectors for Nonlinear Programming Problems. Math. Program. 84, 227–267 (1999).

[44] Netlib repository: Collection of linear programs. See www.netlib.org/lp/.

[45] Nocedal, J., Wright, S. J.: Numerical Optimization. Springer, 2nd edition (2006).

[46] Online QP Benchmark Collection: Developed by Diehl, M. and Ferreau, H.J.. See http://www.kuleuven.be/optec/software/onlineQP.

[47] Pannocchia, G., Rawlings, J. B., Wright, S.: Fast, large-scale model predictive control by partial enumeration. Automatica 43(5), 852–860 (2007).

[48] Polyak, R.: Modified barrier functions (theory and methods). Math. Program. 54, 177–222 (1992).

[49] Renegar, J.: A Mathematical View of Interior-Point Methods in Convex Optimization. SIAM, Philadelphia (1987).

[50] Skajaa, A., Andersen, E. D., Ye, Y.: Warmstarting the homogeneous and self-dual interior point method for linear and conic quadratic problems. Math. Program. Comp. 5(1), 1–25 (2013).

[51] Skajaa, A., Ye, Y.: A Homogeneous Interior-Point Algorithm for Nonsymmetric Convex Conic Optimization. Submitted to Math. Program., 2013.

[52] Sturm, J. F.: Primal-dual interior-point approach to semidefinite programming. PhD thesis, Erasmus Universiteit Rotterdam, 1997.

[53] Sturm, J. F.: Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. Optim. Method. Softw. 12, 625–653 (1999).

[54] Sturm, J. F.: Implementation of Interior Point Methods for Mixed Semidefinite and Second Order Cone Optimization Problems. Optim. Method. Softw. 17, 1105–1154 (2002).

[55] Tuncel, L.: Primal-Dual Symmetry and Scale Invariance of Interior-Point Algorithms for Convex Optimization. Math. Oper. Res. 23, 708–718 (1998).

[56] Tuncel, L.: Generalization Of Primal-Dual Interior-Point Methods To Convex Optimization Problems In Conic Form. Found. Comput. Math. 1, 229–254 (2001).

[57] Wirsching, L., Bock, H. G., Diehl, M.: Fast NMPC of a chain of masses connected by springs. In Proc. of the IEEE International Conference on Control Applications, Munich —, 591–596 (2006).

[58] Wright, S. J.: Primal-Dual Interior-Point Methods. SIAM (1987).

[59] Wu, Q., Nielsen, A. H., Østergaard, J., Cha, S. T., Marra, F., Chen, Y., Træholt, C.: Driving Pattern Analysis for Electric Vehicle (EV) Grid Integration Study. In Proc.: Innovative Smart Grid Technologies Conference Europe (ISGT Europe), 2010 IEEE PES —, 1–6 (2010).

[60] Xu, X., Hung, P. F., Ye, Y.: A simplified homogeneous and self-dual linear programming algorithm and its implementation. Ann. Oper. Res. 62, 151–171 (1996).

[61] Xue, G. Ye, Y.: An Efficient Algorithm for Minimizing a Sum of $p$-Norms. SIAM J. Optimiz. 10, 551–579 (1999).

[62] Ye, Y.: Interior Point Algorithms: Theory and Analysis. Wiley (1997).

[63] Yildirim, E.A., Wright, S.J.: Warm-start strategies in interior-point methods for linear programming. SIAM J. Optim. 12(3), 782–810 (2002).