Projection methods in conic optimization

Didier Henrion, Jérôme Malick

Projection methods in conic optimization

Didier Henrion[1,2] Jérôme Malick[3]

Abstract

There exist efficient algorithms to project a point onto the intersection of a convex cone and an affine subspace. Those conic projections are in turn the work-horse of a range of algorithms in conic optimization, having a variety of applications in science, finance and engineering. This chapter reviews some of these algorithms, emphasizing the so-called regularization algorithms for linear conic optimization, and applications in polynomial optimization. This is a presentation of the material of several recent research articles; we aim here at clarifying the ideas, presenting them in a general framework, and pointing out important techniques.

1 Introduction, motivations, outline

1.0.1 Projection onto semidefinite positive matrices

Consider the space $\mathcal{S}_n$ of symmetric n-by-n matrices, equipped with the norm associated to the usual inner product

$$\langle X, Y\rangle = \sum_{i,j=1}^{n} X_{ij}Y_{ij} = \text{trace}(X^\top Y).$$

The subset $\mathcal{S}_n^+$ made of positive semidefinite matrices forms a closed convex cone of $\mathcal{S}_n$. A general result for closed convex sets yields that we can project onto $\mathcal{S}_n^+$: given $C \in \mathcal{S}_n$, there exists an unique element of $\mathcal{S}_n^+$ (called the projection of C onto $\mathcal{S}_n^+$ and denoted by $\text{Proj}_{\mathcal{S}_n^+}(C)$) such that

$$\|\,\text{Proj}_{\mathcal{S}_n^+}(C) - C\| = \min_{X\in\mathcal{S}_n^+} \|X - C\|.$$

It turns out that we also have an explicit expression of this projection, through the spectral decomposition of C. Consider indeed the decomposition

$$C = U\,\text{Diag}(\lambda_1,\ldots,\lambda_n)U^\top$$

[1]CNRS; LAAS; 7 avenue du colonel Roche, F-31077 Toulouse, France; Université de Toulouse; UPS, INSA, INP, ISAE; LAAS; F-31077 Toulouse, France.

[2]Faculty of Electrical Engineering, Czech Technical University in Prague, Technická 4, CZ-16626 Prague, Czech Republic.

[3]CNRS; LJK; Grenoble, France.

where $\lambda_1 \geq \cdots \geq \lambda_n$ are the eigenvalues of C and U is a corresponding orthonormal matrix of eigenvectors of C; then the projection of C onto $\mathcal{S}_n^+$ is

$$\mathrm{Proj}_{\mathcal{S}_n^+}(C) = U \,\mathrm{Diag}\left(\max(0,\lambda_1),\ldots,\max(0,\lambda_n)\right)U^\top. \tag{1}$$

This result was noticed early by statisticians [SA79] (see also [Hig88]), and since then this projection has been widely used. We notice that this result generalizes nicely for "spectral sets"; see [LM08]. Note also that the numerical cost of computing this projection is essentially that of computing the spectral decomposition of C, the matrix to project.

The developments of this chapter show that more sophisticated projections onto subsets of $\mathcal{S}_n^+$ are also computable using standard tools of numerical optimization. More specifically, the subsets that we consider are intersections of the cone $\mathcal{S}_n^+$ with a polyhedron (defined as affine equalities and inequalities). Though the projection onto those intersections is not explicit anymore, we still have efficient algorithms to compute them, even for large-scale problems.

1.0.2 Projection onto correlation matrices

The most famous example of such projections is the projection onto the set of correlation matrices (that are the real symmetric positive semidefinite matrices with ones on the diagonal). It is common to be faced with a matrix that is supposed to be a correlation matrix but for a variety of reasons is not. For example, estimating correlation matrices when data come from different time frequencies may lead to a non-positive semidefinite matrix. Another example is stress-testing in finance: a practitioner may wish to explore the effect on a portfolio of assigning certain correlations differently from the historical estimates, but this operation can destroy the semidefiniteness of the matrix.

These important practical questions have led to much interest in the problem of computing the nearest correlation matrix to a given a matrix C (see e.g. [Hig02], [Mal04], [QS06] and [BH08]). This problem is simply formulated as the projection of C onto correlation matrices

$$\left\{\begin{array}{ll} \min & \frac{1}{2}\|X-C\|^2 \\ & X_{ii} = 1, \quad i = 1,\ldots,n \\ & X \succeq 0. \end{array}\right. \tag{2}$$

The methods reviewed in this chapter apply to solving this problem in particular. The point is that this problem (and variants of it) can now be solved efficiently (for sizes up to $n = 5000$; the only limitation on a standard computer is the memory constraints).

1.0.3 Conic projection problem

The general problem that we first focus on in this chapter is the following. In the space $\mathbb{R}^n$ equipped with the standard inner product, we want to compute the projection of a point $c \in \mathbb{R}^n$ onto the intersection $\mathcal{K} \cap \mathcal{P}$ where

- $\mathcal{K}$ is a closed convex cone in $\mathbb{R}^n$ (that we further assume to have full dimension in $\mathbb{R}^n$; that is, $\operatorname{int}\mathcal{K} \neq \emptyset$),
- $\mathcal{P}$ is a convex polyhedron defined by affine (in)equalities

$$\mathcal{P} := \left\{ x \in \mathbb{R}^n : \; a_i^\top x = (\text{or} \leq)\; b_i, \quad i = 1, \ldots, m \right\}.$$

We suppose moreover that the intersection $\mathcal{K} \cap \mathcal{P}$ is nonempty, so that the projection onto the closed convex set $\mathcal{K} \cap \mathcal{P}$ exists and is unique (see e.g. [HUL93]).

The fact that $\mathcal{P}$ is defined by both equalities and inequalities does not really matter in our developments. To simplify presentation, one may take only equalities, so that $\mathcal{P}$ is an affine subspace. We prefer to keep the above loose notation with both equalities and inequalities, because it is closer to projection problems arising in practice, and because it does not impact the basics of projection algorithms. Adding positive slack variables for the inequalities allows us to reformulate the problem as a projection onto the intersection of an affine space with a cone of the form $\mathcal{K} \times (\mathbb{R}_+)^{m_\mathrm{I}}$.

We note that in general one can project onto a polyhedron $\mathcal{P}$. For the case when there are only (independent) equalities in the definition (i.e. if $\mathcal{P} = \mathcal{A}$ is an affine subspace of the equation $Ax = b$ with a full-rank matrix A), we have the explicit expression of the projection of x

$$\operatorname{Proj}_{\mathcal{A}}(x) = x - A^\top [AA^\top]^{-1}(Ax - b). \tag{3}$$

For a general polyhedron $\mathcal{P}$, we still can compute the projection $\operatorname{Proj}_{\mathcal{P}}(x)$ efficiently using quadratic programming solvers (see e.g. [NW99] or [BGLS03]). In this chapter, we make the practical assumption that it is also easy to project onto $\mathcal{K}$. Recall from above that we have an easy-to-compute expression of the projection for $\mathcal{K} = \mathcal{S}_n^+$; it turns out to be also the case for the second-order cone (or Lorentz cone)

$$\mathcal{L}_n := \{ x \in \mathbb{R}^n : \; \|(x_1, \ldots, x_{n-1})\| \leq x_n \}.$$

Though it is easy to project onto $\mathcal{P}$ and also onto $\mathcal{K}$ by assumption, the projection onto the intersection $\mathcal{P} \cap \mathcal{K}$ can still be challenging. The difficulty comes from the presence of both (affine and conic) constraints at the same time. We will see in Section 2 that many numerical methods to compute the projection onto the intersection use combinations of projections onto $\mathcal{P}$ and $\mathcal{K}$ separately.

The geometrical projection problem has an obvious analytical formulation as a least-squares problem, namely minimizing the (squared) norm subject to the conic constraints and the affine constraints:

$$\begin{cases} \min & \frac{1}{2}\|x - c\|^2 \\ & x \in \mathcal{P} \cap \mathcal{K}. \end{cases} \tag{4}$$

For example, an important subclass of such problems are semidefinite least-squares problems (i.e. when $\mathcal{K} = \mathcal{S}_n^+$):

$$\begin{cases} \min & \frac{1}{2}\|X - C\|^2 \\ & \langle A_i, X \rangle = (\text{or} \leq)\; b_i, \quad i = 1, \ldots, m \\ & X \succeq 0, \end{cases} \tag{5}$$

for $C, A_i \in \mathcal{S}_n$. For example, the nearest correlation matrix problem (2) is an instance of this later class. Notice finally that problems (4) and (5) coincide formally with $x \in \mathbb{R}^{n^2}$ collecting the rows of $X \in \mathcal{S}_n$. This also explains the slight abuse of notation when writing $\mathcal{K} = \mathcal{S}_n^+$. We use this ambiguity $x \leftrightarrow X$ in particular in Section 4 to ease presentation of relaxations of polynomial optimization problems.

1.0.4 Linear conic optimization problem

The second development of this chapter is about a more standard topic: solving linear conic optimization problems. With the above notation, these problems can be expressed as

$$\begin{cases} \min & c^\top x \\ & x \in \mathcal{P} \cap \mathcal{K}. \end{cases} \tag{6}$$

As presented in this handbook and as well as in the first handbook [SVW00], linear conic programming has been a very active field of research spurred by many applications and by the development of efficient methods.

In this chapter, we explain how conic projections can be used to develop a new family of algorithms for solving linear conic problems. In fact, the projection problem (4) can be written as a linear problem of the form (6) and then can be solved using usual conic programming solvers (we come back to this at the beginning of Section 2 and we explain why it is not a good idea to do so). However we will also show the other way around: Section 3.1 explains that one can also solve the linear conic problem (6) by solving projection problems (4), more precisely with a succession of (truncated) projection-like problems. So-called regularization methods are presented, discussed and illustrated on solving semidefinite relaxations of combinatorial optimization and polynomial optimization problems having many constraints.

1.0.5 Polynomial optimization

Over the last decade, semidefinite programming has been used in polynomial optimization, namely for deciding whether a multivariate real polynomial is nonnegative, or, more generally, to minimize a polynomial on a semialgebraic set (a set described by polynomial inequalities and equations). A hierarchy of embedded linear semidefinite relaxations (of the form (6)) can be constructed to generate a monotone sequence of bounds on the global minimum of a polynomial optimization problem. Asymptotic convergence of the sequence to the global minimum can be guaranteed under mild assumptions, and numerical linear algebra can be used to detect global optimality and extract global minimizers. The theory is surveyed in [Lau09] and [Las09]; the potential applications are numerous (see e.g. in control theory [HG05] or signal processing [Dum07]).

Section 4 reports numerical experiments showing that regularization algorithms based on projections outperform classical primal-dual interior-point algorithms for solving semidefinite relaxations arising when deciding whether a polynomial is nonnegative, and for globally minimizing a polynomial.

1.0.6 Objectives and outline of this chapter

This chapter focuses on projection problems that have a simple geometric appeal as well as important applications in engineering. We give references to some of these applications, and we give emphasis on polynomial optimization.

The first goal of this chapter is to sketch the different approaches to solve (4). Section 2 is devoted to this review, with an emplasis on dual methods (in Section 2.2). The bottomline is that as soon as we can easily project onto $\mathcal{K}$ (we have in mind $\mathcal{L}_n$ and $\mathcal{S}_n^+$ as well as direct products of these), we have efficient algorithms to project onto the intersection $\mathcal{K} \cap \mathcal{P}$.

The second goal of this chapter is to explain how to use these conic projections to build a family of "regularization" methods for linear conic programming. The approach uses standard optimization techniques (proximal algorithms and augmented Lagrangian methods) and has been recently developed for the case $\mathcal{K} = \mathcal{S}_n^+$. Section 3 presents it in a general framework and underlines the role of conic projections. The final section presents some numerical experiments with regularization methods on polynomial optimization problems, showing the interest of the approach in that context.

This chapter is meant to be an elementary presentation of parts of the material of several papers; among those, our main references are [Mal04], [QS06], [MPRW09], [HM11], [ZST10] and [Nie09]. We aim at clarifying the ideas, presenting them in a general framework, unifying notation, and most of all, pointing out what makes things work. To this purpose, we have to elude some technical points; in particular, we discuss algorithms, but we do not to give convergence results. We try to give precise references throughout the text on these lacking points.

2 Conic projections: algorithms and applications

This section reviews the methods for solving the conic projection problem (4), presenting them in chronological order. We sketch the main ideas and give references; we do not get into much details. Discussions about convergence issues and numerical comparisons are beyond the scope of this section.

Beside interior-point methods, the basic idea of all the approaches is to somehow separate the two constraint-sets $\mathcal{K}$ and $\mathcal{P}$ and to use the projections onto them successively: this is obvious for alternating projections and alternating directions methods; it is also the case for dual methods (we focus on this latter method in Section 2.2). The point is that we can solve the conic projection problem (4) efficiently (by dual algorithms in particular).

To simplify presentation, we stick here with the projection problem (4), but the approach generalizes in two directions. First, we could replace the cone $\mathcal{K}$ by any closed convex set: in this more general case, the developments are similar, with slightly more complicated expressions of dual objects (a related work is [MU88]). Second, we could consider problems with strongly convex quadratic

objective functions, such as

$$\begin{cases} \min \quad (x-c)^\top Q(x-c) + d^\top x \\ \quad x \in \mathcal{K} \cap \mathcal{P} \end{cases} \tag{7}$$

with Q positive definite. Such problems can be phrased as projection problems with respect to $\|x\|_Q = \sqrt{x^\top Q\, x}$ the norm associated to Q. The practical technical assumption is then that one can project onto $\mathcal{K}$ with respect to $\|\cdot\|_Q$ (which is not easy in general).

2.1 Computing conic projections

2.1.1 Using linear conic programming

A tempting method to solve (4) is to cast this projection problem as a usual linear conic programming problem, so that we can use the powerful tools developed for this case. There are several ways to do so; a simple one consists in pushing down the objective function with an additional variable t: (4) is indeed equivalent to linear conic program

$$\begin{cases} \min \quad t \\ \quad x \in \mathcal{P} \\ \quad x - c = z \\ \quad (x,(z,t)) \in \mathcal{K} \times \mathcal{L}_{n+1} \end{cases}$$

where the variable $z \in \mathbb{R}^n$ is then introduced to express the additional second-order cone constraint appearing in the constraints. This problem can be readily given to usual conic solvers, for example interior-points methods, like SeDuMi [Stu99] or `SDPT3` [TTT03] under Matlab. Unfortunately, adding (z,t) makes the computational cost and memory space needed by a standard primal-dual interior-point method increase, and numerical testing confirms that the method is not viable in general (as mentioned e.g. in [Hig02],[Toh08]).

We note furthermore that the projection problem (4) is a quadratic conic programming problem, hence a special case of nonlinear conic optimization problems. We could solve (4) by algorithms and software devoted to nonlinear conic optimization problems such as the penalization method of [KS03]. However those methods would not use the special structure of (4), and as the above approach by linear conic programming, they would be efficient only for small-size projection problems. The projection problems are important enough to design algorithms specifically to them, as presented in the sequel. Note that we are not aware of a tailored penalization algorithm for (4).

2.1.2 Alternating projections

The alternating projection method is an intuitive algorithmic scheme to find a point in the intersection of two sets: it consists in projecting the initial point onto the first set, then projecting the new point onto the second set, and then

projecting again the new point onto the first and keep on projecting alternatively. In other words, it consists in repeating:

$$\begin{cases} x_{k+1} = \mathrm{Proj}_{\mathcal{K}}(y_k) \\ y_{k+1} = \mathrm{Proj}_{\mathcal{P}}(x_{k+1}) \end{cases} \tag{8}$$

If the two sets have a "regular" intersection, this algorithm converges linearly to a point in $\mathcal{P} \cap \mathcal{K}$ and we know the speed of convergence (for two convex sets, see e.g. [Deu01]; for the general case, see the local result of [LLM09]).

We can modify this simple alternating projection scheme by adding a correction step (called Dykstra's correction [Dyk83]) at each iteration (8)

$$\begin{cases} x_{k+1} = \mathrm{Proj}_{\mathcal{K}}(z_k) \\ y_{k+1} = \mathrm{Proj}_{\mathcal{P}}(x_{k+1}) \\ z_{k+1} = z_k - (x_{k+1} - y_{k+1}). \end{cases} \tag{9}$$

This modification ensures the convergence of the sequence $(x_k)_k$ to the projection $\mathrm{Proj}_{\mathcal{K}\cap\mathcal{P}}(c)$ – and not only to a point in the intersection $\mathcal{K} \cap \mathcal{P}$. This approach was proposed by [Hig02] for the nearest correlation matrix problem (2). It generalizes to (4) since it is easy to project onto $\mathcal{P}$ and we assume that it is the same for $\mathcal{K}$. We will see that dual methods and alternating direction methods can be interpreted as variants of this basic geometrical method.

2.1.3 Dual methods

The conic programming problem (4) looks more complicated than a usual conic programming problem with linear function instead of a norm as objective function. It turns out that the strong convexity of the objective function provides nice properties to the dual problem that can then be solved efficiently.

The dual approach was proposed for the conic least-squares problem (4) in [Mal04], later revisited by [BX05] for the case of $\mathcal{K} = \mathcal{S}_n^+$, and then enhanced by [QS06] and [BH08] for the projection onto correlation matrices. In the next section, we give more details and more references about this approach.

2.1.4 Interior points

As a convex optimization problem, (4) can be attacked with the interior-point machinery [NN94], assuming that both the cone $\mathcal{K}$ and its polar cone

$$\mathcal{K}^o := \left\{ s \in \mathbb{R}^n : \ s^\top x \leq 0 \text{ for all } x \in \mathcal{K} \right\}$$

are equipped with so-called self-concordant barriers (as is the case for $\mathcal{L}_n, \mathcal{S}_n^+$). The approach consists in solving perturbed optimality conditions of (4). As any projection problem, notice that the optimality condition is

$$\bar{x} \in \mathcal{P} \cap \mathcal{K}, \quad (c - \bar{x})^\top (x - \bar{x}) \leq 0, \quad \text{for all } x \in \mathcal{P} \cap \mathcal{K}.$$

To write down the optimality conditions more concretely, let us make explicit the affine constraints with the help of $A_{\mathrm{E}} \in \mathbb{R}^{n \times m_{\mathrm{E}}}$ and $A_{\mathrm{I}} \in \mathbb{R}^{n \times m_{\mathrm{I}}}$ as

$$\left\{ \begin{array}{ll} \min & \|x - c\|^2 \\ & A_{\mathrm{E}} x = b_{\mathrm{E}},\ A_{\mathrm{I}} x \leq b_{\mathrm{I}} \\ & x \in \mathcal{K}. \end{array} \right. \tag{10}$$

Under a non-degeneracy assumption (e.g. Slater condition, see next section), the optimality conditions of (10) give the complementarity system

$$\left\{ \begin{array}{l} x - c + u + {A_{\mathrm{E}}}^\top y + {A_{\mathrm{I}}}^\top z = 0 \\ A_{\mathrm{E}} x = b_{\mathrm{E}},\ y \in \mathbb{R}^{m_{\mathrm{E}}} \\ A_{\mathrm{I}} x \leq b_{\mathrm{I}},\ z \in \mathbb{R}^{m_{\mathrm{I}}}_+,\ z^\top (A_{\mathrm{I}} x - b_{\mathrm{I}}) = 0 \\ x \in \mathcal{K},\ u \in \mathcal{K}^o,\ u^\top x = 0. \end{array} \right.$$

Roughly speaking, an interior-point approach consists in perturbing the complementary equations above and keeping other equations satisfied. (We will see that the forthcoming dual approach goes exactly the other way around.) A first interior-point method is proposed in [AHTW03] for the nearest correlation matrix problem (2). Interior-point methods for general quadratic SDP are introduced and tested on projection problems (4) in [TTT06] and [Toh08].

2.1.5 Alternating directions

The alternating direction method is a standard method in variational analysis (see e.g. [GM76]), going back to [DR56]. This method was proposed by [HXY09] for solving the semidefinite projection problem (5) and by [SZ10] for more general quadratically constrained quadratic SDP. The idea of the method is to exploit the separable structure of the problem, as follows. Let us duplicate the variables to write the equivalent problem

$$\left\{ \begin{array}{ll} \min & \frac{1}{2}\|x - c\|^2 + \frac{1}{2}\|y - c\|^2 \\ & x = y \\ & x \in \mathcal{K}, \quad y \in \mathcal{P}. \end{array} \right. \tag{11}$$

The alternating direction method applied to (11) gives the following scheme: consider the augmented Lagrangian function

$$L(x, y; z) = \frac{1}{2}\|x - c\|^2 + \frac{1}{2}\|y - c\|^2 - \langle z, x - y \rangle + \frac{\beta}{2}\|x - y\|^2;$$

the minimization of L with respect to primal variables (x, y) is decomposed in two steps, so that an augmented Lagrangian iteration is

$$\left\{ \begin{array}{l} x_{k+1} = \mathrm{argmin}_{x \in \mathcal{K}}\, L(x, y_k, z_k) \\ y_{k+1} = \mathrm{argmin}_{y \in \mathcal{P}}\, L(x_{k+1}, y, z_k) \\ z_{k+1} = z_k - \beta(x_{k+1} - y_{k+1}). \end{array} \right.$$

It is not difficult to prove that the two above minimizations boil down to projections, more specifically

$$x_{k+1} = \mathrm{Proj}_{\mathcal{K}}\Big(\frac{\beta y_k + z_k + c}{1+\beta}\Big), \quad y_{k+1} = \mathrm{Proj}_{\mathcal{A}}\Big(\frac{\beta x_{k+1} - z_k + c}{1+\beta}\Big).$$

Thus the approach alternates projections onto $\mathcal{P}$ and $\mathcal{K}$ to compute the projection onto $\mathcal{K} \cap \mathcal{P}$; it can thus be seen as a modification of the simple alternating projection scheme (8), with the same flavour as Dykstra modification (9).

2.2 More on dual approach

2.2.1 Apply standard machinery

Let us give more details about the dual approach for solving (4). Following [Mal04], we apply the standard mechanism of Lagrangian duality to this problem; we refer to [HUL93, Ch. XII] and [BV04, Ch. 5] for more on this mechanism in general.

Let us consider the more explicit form (10), and denote also by $A := [A_{\mathrm{E}}; A_{\mathrm{I}}]$ and $b := [b_{\mathrm{E}}; b_{\mathrm{I}}]$ the concatenation of the affine constraints. We dualize affine constraints only: introduce the Lagrangian, a function of primal variable $x \in \mathcal{K}$ and dual variable $(y, z) \in \mathbb{R}^{m_{\mathrm{E}}} \times \mathbb{R}^{m_{\mathrm{I}}}_+$

$$L(x; y, z) := \frac{1}{2}\|c - x\|^2 - y^\top (A_{\mathrm{E}} x - b_{\mathrm{E}}) - z^\top (A_{\mathrm{I}} x - b_{\mathrm{I}}), \tag{12}$$

and the corresponding concave dual function

$$\theta(y, z) := \min_{x \in \mathcal{K}} L(x; y, z), \tag{13}$$

which is to be maximized. There is no more affine constraint in the above minimum, and it is easy to prove ([Mal04, Th.3.1]) that the problem corresponds to a projection onto $\mathcal{K}$: there exists a unique point which reaches the above minimum, namely

$$x(y, z) := \mathrm{Proj}_{\mathcal{K}}(c + {A_{\mathrm{E}}}^\top y + {A_{\mathrm{I}}}^\top z), \tag{14}$$

so we have

$$\theta(y, z) = {b_{\mathrm{E}}}^\top y + {b_{\mathrm{I}}}^\top z + \frac{1}{2}(\|c\|^2 - \|x(y, z)\|^2). \tag{15}$$

It is also not difficult to show [Mal04, Th.3.2] that the concave function θ is differentiable on $\mathbb{R}^m$, and that its gradient

$$\nabla\theta(y, z) = -Ax(y, z) + b \tag{16}$$

is Lipschitz continuous. As any function with Lipschitz gradient, θ is twice differentiable almost everywhere, but not everywhere (this basically relies on the differentiability properties of $\mathrm{Proj}_{\mathcal{K}}$; for the case $\mathcal{K} = \mathcal{S}_n^+$, see more in [SS02] and [MS06] among others).

The dual problem is thus

$$\begin{cases} \max \quad \theta(y,z) \\ \quad (y,z) \in \mathbb{R}^{m_{\mathrm{E}}} \times \mathbb{R}^{m_{\mathrm{I}}}_{+}. \end{cases} \tag{17}$$

Strong duality (the optimal values of (10) and (17) coincide) holds under a standard assumption in convex optimization. The so-called (weak) Slater assumption (see e.g. [Ber95], [HUL93]) is in our context:

$$\exists\ \bar{x} \in \mathcal{P} \cap \operatorname{int} \mathcal{K}. \tag{18}$$

In fact, this assumption yields moreover that there exists solutions to (17) (note that the assumption has also a natural geometrical appeal in context of projection methods, see [HM11, Sec. 3]). Finally we get directly the projection from dual solutions: let (y^*, z^*) be a (dual) solution of (17), the (primal) solution x^* of (4) is the associated $x^* = x(y^*, z^*)$ (see [Mal04, Th. 4.1]).

2.2.2 Apply standard algorithms

To compute the projection of c onto $\mathcal{P} \cap \mathcal{K}$, we just have to solve the dual problem (17). Let us have a closer look to this problem: the constraints are simple positivity constraints on the variable corresponding to the dualization of inequality constraints; the dual function is a differentiable concave function with Lipschitz gradient. This regularity has a huge impact in practice: it opens the way for using standard algorithms for nonlinear optimization. Hence we can use any of the following numerical methods to solve (17) (as soon as the software can deal with the constraints $z_i \geq 0$):

1. gradient methods: standard methods [Ber95] or more evolved ones, as e.g. Nesterov's method [Nes05];

2. Newton-like methods: quasi-Newton, limited memory quasi-Newton, inexact Newton, Newton-CG, see textbooks [NW99] and [BGLS03] – with the restriction that θ is not twice differentiable everywhere, so that we have to use the so-called semismooth Newton methods, see [QS93].

For example, [Mal04] uses a quasi-Newton method for solving (5), and [QS06] uses a semismooth inexact Newton method for solving (2). We come back on these two methods in the next section to give more practical details.

We also mention here the so-called inexact smoothing method of [GS09] which consists in writing the optimality conditions of the dual problem (17) as a nonsmooth fixed point problem (and solving it by combining smoothing techniques and an inexact Newton method; see e.g. [NW99]).

The dual problem (17) can thus be attacked with classical tools or more evolved techniques. In practice, the choice of the solving method depends on the structure of the problem and the target level of sophistication.

We call dual projection methods any method using an optimization code for functions with Lipschitz gradient to maximize θ on $\mathbb{R}^{m_{\mathrm{E}}} \times \mathbb{R}^{m_{\mathrm{I}}}_{+}$. Specifically, a

dual projection method generates a maximizing dual sequence $\{y_k, z_k\}_k$ together with the primal sequence $x_k = x(y_k, z_k)$ such that:

$$\theta(y_k, z_k) = b_{\mathrm{E}}{}^{\top} y_k + b_{\mathrm{I}}{}^{\top} z_k + \frac{1}{2}(\|c\|^2 - \|x_k\|^2) \tag{19}$$

$$\nabla\theta(y_k, z_k) = -Ax_k + b. \tag{20}$$

We notice that in our numerical experiments with dual methods, we have observed better behaviour and convergence when the (strong) Slater assumption holds (that is, when (18) holds and moreover A_{E} is full rank).

2.2.3 More algorithmic details (for the case without inequalities)

We detail now further some algorithmic issues. To simplify we focus on the case without inequalities ($m_{\mathrm{I}} = 0$, no dual variables z). Iterations of most algorithms for maximizing θ can be written as

$$y_{k+1} = y_k + \tau_k W_k \nabla\theta(y_k). \tag{21}$$

Note that the usual stopping test of these methods has an intrinsic meaning: a threshold condition on the gradient

$$\|\nabla\theta(y_k)\| = \|Ax_k - b\| \leq \varepsilon \tag{22}$$

controls in fact the primal infeasibility. Among these methods, let us discuss further the three following ones.

Gradient descent with constant step-size. We have a remarkable result: the gradient method in an adapted metric, namely (21) with

$$W_k = [AA^{\top}]^{-1} \qquad \text{and} \qquad \tau_k = 1, \tag{23}$$

corresponds exactly to the alternating projection method (9) (see [Mal04] for a proof in the special case of correlation matrices, and [HM11] for the proof in general). We thus have a (surprizing) dual interpretation of the primal projection method. Using descent schemes more evolved than a simple gradient descent (see below) then leads to (dual) projection methods that can be seen as improvements of the basic alternating projection method.

BFGS Quasi-Newton method. The method is known to be very efficient in general, and have many industrial applications (one of the most striking is in weather forecasting [GL89]). The method can be readily applied to the dual problem, since it requires no more information than (19): W_k is constructed with successive gradients with the BFGS formula and τ_k is well-chosen with a Wolfe line-search (see e.g. [BGLS03]). The initial paper about dual methods [Mal04] proposes to use this method in general and reports very good numerical results on the nearest correlation matrix problem (2). Since then, this dual method has

been used successfully to solve real-life projection problems in numerical finance (among them: the problem of calibrating covariance matrices in robust portfolio selection [Mal04, 5.4]). A simple Matlab implementation has been made publicly available together with [HM11] for pedagogical and diffusion purposes.

Generalized (or semismooth) Newton. A pure Newton method would be to use $\tau_k = 1$ and $W_k = [H_k]^{-1}$ with the Hessian H_k of θ at the current iterate y_k. In practice, an inexact generalized Newton method is used for the following reasons.

As mentioned earlier, θ is differentiable but not twice differentiable (though its gradient is Lipschitz continuous). We can still replace the usual Hessian by a matrix $H_k \in \partial_c^2\theta(y_k)$ the Clarke generalized Hessian of θ at y_k [Cla83]. Computing a matrix in $\partial_c^2\theta(y_k) \subset \mathcal{S}_n^+$ amounts to computing an element of the Clarke generalized Jacobian of the projection onto the cone $\partial_c \mathrm{Proj}_{\mathcal{K}}$ since we have (see [HUSN84])

$$\partial_c^2\theta(y_k) = A\,\partial_c \mathrm{Proj}_{\mathcal{K}}(c + A^\top y_k)A^\top.$$

We can often compute an element of $\partial_c \mathrm{Proj}_{\mathcal{K}}$. For example, we even have an explicit expression of the whole $\partial_c \mathrm{Proj}_{\mathcal{S}_n^+}$ [MS06].

For overall efficiency of the method, the Newton direction d_k is computed by solving the system $H_k d = \nabla\theta(y_k)$ approximately, usually by conjugate gradient (CG) type methods. More precisely, the idea of so-called Newton-CG (also called inexact Newton going back to [DET82]) is to stop the inner iteration of CG when

$$\|H_k d + \nabla\theta(\lambda_k)\| \leq \eta_k \|\nabla\theta(\lambda_k)\| \tag{24}$$

with small η_k (see e.g. [NW99]). Note that preconditioning the Newton system is then crucial for practical efficiency. The nice feature of this algorithm is that H_k has just to be known through products $H_k d$ so that large-scale problems can be handled. In our context, the main work on this method is [QS06] about the nearest correlation matrix problem; we come back to it in the next section.

We finish here with a couple of words about convergence of this Newton dual method. In general (see [QS93]), the two conditions to prove local superlinear convergence are that the minimum is strong (i.e. all elements of the generalized Hessian are positive definite), and the function has some smoothness (namely, the so-called semismoothness). In our situation, the two ingredients implying those conditions are the following ones:

- The intersection has some "nondegeneracy", in the sense of [BS00, 4.172] and [AHO97, Def. 5]. This allows us to prove $\partial_c^2\theta(y_k) \succ 0$ (see e.g. [QS06] for a special case).

- The convex cone $\mathcal{K}$ has some "regularity". An example of sufficient regularity is that $\mathcal{K}$ is a semialgebraic set (i.e. defined by a finite number of polynomial (in)equalities). Indeed for semialgebraic convex sets, the projection $\mathrm{Proj}_{\mathcal{K}}$ and then θ are automatically semismooth [BDL08] (which

is the property needed to apply the convergence results of [QS93]. This is the case for direct products of the cones $\mathcal{L}_n$ and $\mathcal{S}_n^+$ (for which we even have strong semismoothness [SS02] so in fact quadratic convergence).

2.2.4 Illustration on nearest correlation matrix problem

We give a rough idea of the efficiency of the dual approach on the projection problem (2). The first numerical results of [Mal04, Sec. 4] show that the dual approach copes with large-scale problems, in reporting that one can solve in a couple of minutes projection problems of size around one thousand. By using the dual generalized Newton method (instead of quasi-Newton as in [Mal04]), the algorithm of [QS06], improved later by [BH08], gives very nice results in both practice and theory. Nondegeneracy of the constraints and then of the generalized Hessian is proved in [QS06, Prop. 3.6]: as recalled above, this technical point leads to quadratic convergence of the method [QS06, Prop. 5.3].

Today's state of the art is that one can solve nearest correlation matrix problems of big size (say, up to 4000-5000) in a reasonable amount of computing time (say, less than 10 minutes on a standard personal computer). The only limitation seems to be the memory constraint to store and deal with dense large-scale data.

To give a more precise idea, let us report a couple of results from [BH08]. The implementation of their dual algorithm is in Matlab with some external Fortran subroutines (for eigenvalues decomposition in particular). The stopping criterion is set to

$$\|\nabla\theta(y_k)\| \leq 10^{-7}n. \tag{25}$$

We consider the nearest correlation matrix problems for two (non-SDP) matrices with unit diagonal (of size $n_1 = 1399$ and $n_2 = 3120$) provided by a fund management company. The dual method solves them in around 2 and 15 min., respectively, on a very standard machine (see more details in [BH08]).

We finish with a last remark about accuracy. The approximate correlation matrix X that is computed by such a dual method is often just what is needed in practice. It might happen though that a special application requires a perfect correlation matrix – that is, with exactly ones on the diagonal, whereas X satisfies only (by (25))

$$\Big(\sum_{i=1}^{n}(X_{ii}-1)^2\Big)^{-1/2} \leq 10^{-7}n.$$

A simple post-treatment corrects this. Setting diagonal elements to ones may destroys the positiveness, so we apply the usual transformation that computes the associated correlation matrix $\bar{X}$ from a covariance matrix X, namely

$$\bar{X} = D^{-1/2}XD^{-1/2} \qquad \text{and} \qquad D = \operatorname{diag}(X).$$

This operation increases the distance from C; but the error is still under control (by $\varepsilon/(1-\varepsilon)$; see [BH08, Prop. 3.2]).

2.3 Discussion: applications, generalizations

2.3.1 Direct or indirect applications

Conic projection problems with the positive semidefinite cone (like $\mathcal{K} = \mathcal{S}_n^+$, $\mathcal{K} = \mathcal{S}_n^+ \times (\mathbb{R}^+)^p$ or $\mathcal{K} = \mathcal{S}_{n_1}^+ \times \cdots \times \mathcal{S}_{n_p}^+$) are numerous in engineering. Constructing structured semidefinite matrices, for example, are naturally modeled this way. Such problems naturally appear in finance for constructing structured covariance matrices (as a calibration step before simulations); they also appear in many other fields, such as in control (e.g. [LJ09]), in numerical algebra (e.g. [AH07]), or in optics (e.g. [NWV08]), to name a few of them.

Conic projections also appear as inner subproblems within more involved optimization problems. Solving efficiently these inner problems is often the key to numerical efficiency of the overall approach. Let us give some examples.

- *Linear conic programming.* So-called regularization methods for solving (6) use the conic projection problem as an inner subproblem; these methods are studied in Section 3.

- *Weighted projections.* For given weights $H_{ij} \geq 0$, consider the semidefinite projection (5) with a different objective function

$$\left\{ \begin{array}{ll} \min & \frac{1}{2} \sum_{i,j=1}^n H_{ij}(X_{ij} - C_{ij})^2 \\ & \langle A_i, X \rangle = (\text{or } \leq)\ b_i, \quad i = 1, \ldots, m \\ & X \succeq 0. \end{array} \right.$$

  An augmented Lagrangian approach for this problem [QS10] produces a projection-like inner problem, which is solved by a semismooth Newton method (recall the discussion of the previous section).

- *Low-rank projections.* Consider the semidefinite projection problem (5) with additional rank-constraint

$$\left\{ \begin{array}{ll} \min & \frac{1}{2}\|X - C\|^2 \\ & \langle A_i, X \rangle = (\text{or } \leq)\ b_i, \quad i = 1, \ldots, m \\ & X \succeq 0,\ \operatorname{rank} X = r. \end{array} \right. \tag{26}$$

  This difficult non-convex calibration problem has several applications in finance and insurance industry (e.g. pricing interest rate derivatives for some models; see e.g. [BM06]). Two approaches (by augmented Lagrangian [LQ10] and by penalty techniques [GS10]) have been recently proposed to solve these types of problems; both approaches solve a sequence of projection-like subproblems. The numerical engine is a dual semismooth truncated Newton algorithm for computing projections.

For these applications of conic projections, the techniques and the arguments are often the same, but are redeveloped for each particular projection problem encountered. We hope that the unified view of Section 2 can bring forth the common ground of these methods and to better understand how and why they work well. We finish this section by pointing out an easy geometrical application.

2.3.2 Application for solving conic feasibility problems

The conic feasibility problem consists simply in finding a point x in the intersection $\mathcal{K} \cap \mathcal{P}$. Many engineering problems can be formulated as semidefinite or conic feasibility problems (for example in robust control [BGFB94] where an element in the intersection is a certificate of stability of solutions of differential equations). Section 4.2 focuses on semidefinite feasibility problems arising when testing positivity of polynomials. We refer to the introduction of [HM11] for more examples and references.

A simple and natural technique for solving conic feasibility problems is just to project a (well-chosen) point onto the intersection $\mathcal{K} \cap \mathcal{P}$ (by dual projection methods for example). In [HM11], a comparative study of such a conic projection method with the usual approach using SeDuMi was carried out precisely on polynomial problems. It was shown there that an elementary Matlab implementation can be competitive with a sophisticated primal-dual interior-point implementation. This would even have a better performance if an initial heuristic for finding a good point to project could be determined (the numerical experiments of [HM11, Sec. 6] simply use $c = 0$). An answer to this latter point is provided by the regularization methods of the next section.

3 Projections in regularization methods

We focus in this section on standard linear conic programming. We show that, following classical convex optimization techniques, conic projections can be used to solve linear conic programming problems.

There exist many numerical methods for solving linear conic problem (6) (see the first handbook [SVW00]). But on the other hand, there also exist big conic problems, and especially big SDP problems, that make all the standard methods fail. Relaxations of combinatorial optimization problems and polynomial optimization problems yield indeed challenging problems. This motivates the development of new algorithmic schemes.

The strategy that we present in this section exploits the efficiency of projection methods by developing proximal algorithms for linear conic programming. We generalize the developments of [MPRW09], and give all way long references to related works. As for numerical aspects, the target problems are semidefinite programs with the number of constraints possibly very large (more than 100,000).

3.1 Proximal method for linear conic programming

3.1.1 Apply classical techniques of convex optimization

The proximal algorithm is a classical method of convex optimization and variational analysis: it goes back from the 1970s with premises in [BKL66], the first work [Mar70] and the important reference [Roc76b]. The driving idea of the proximal algorithm is to add quadratic terms to the objective function to

"regularize" the problem (ensuring existence, uniqueness, and stability of solutions). A (primal) proximal method of the linear conic problem (6) goes along the following lines.

Consider the problem with respect to (x, p)

$$\left\{ \begin{array}{ll} \min & c^\top x + \frac{1}{2t}\|x-p\|^2 \\ p \in \mathbb{R}^n, & x \in \mathcal{P} \cap \mathcal{K}. \end{array} \right.$$

By minimizing first with respect to p, we see that this problem is equivalent to the primal linear conic problem (6). We have added to the objective function a quadratic "regularizing" term $\|x-p\|^2$ with the so-called "prox-parameter" t. The idea now is to solve this problem in two steps: first with respect to x, and second to p:

$$\left\{ \begin{array}{ll} \min \\ p \in \mathbb{R}^n \end{array} \quad \left(\begin{array}{ll} \min & c^\top x + \frac{1}{2t}\|x-p\|^2 \\ x \in \mathcal{P} \cap \mathcal{K} \end{array} \right) \right. . \tag{27}$$

The outer problem is thus the minimization with respect to p of the function

$$F(p) := \left\{ \begin{array}{ll} \min & c^\top x + \frac{1}{2t}\|x-p\|^2 \\ x \in \mathcal{P} \cap \mathcal{K} \end{array} \right. \tag{28}$$

which is the result of the inner optimization problem parametrized by p. As such defined, F is the so-called Moreau-Yosida regularization of the function $x \to c^\top x + i_{\mathcal{P}\cap\mathcal{K}}(x)$ the linear objective function plus the indicator function of the intersection (see e.g. [HUL93, Ch.XV.4]).

The connection with the previous developments of this chapter is then obvious: the above inner problem is essentially a projection problem as studied in the previous section (see (7)). The solution of the inner problem (the "projection") is called the proximal point and denoted

$$\mathrm{Prox}(p) := \left\{ \begin{array}{ll} \mathrm{argmin} & c^\top x + \frac{1}{2t}\|x-p\|^2 \\ x \in \mathcal{P} \cap \mathcal{K}. \end{array} \right.$$

Note that, for simplicity, the dependence of F and Prox with respect to t is dropped in notation.

3.1.2 Primal proximal algorithm for conic programming

Applying basic convex analysis properties, it is easy to prove (see e.g. [HUL93, Ch.XV.4]) that the Moreau-Yosida regularization F is convex and differentiable with gradient $\nabla F(p) = (p - \mathrm{Prox}(p))/t$. The optimality condition of the unconstrained minimization of F is then simply

$$\bar{p} = \mathrm{Prox}(\bar{p}). \tag{29}$$

Moreover a fixed-point $\bar{p}$ of the Prox operator is also a solution of the initial linear conic problem (6): observe indeed that $\bar{p}$ is feasible and reaches the optimal value, since

$$\mathrm{val}\,(6) = \min F(p) = F(\bar{p}) = c^\top \bar{p}. \tag{30}$$

A (primal) proximal algorithm for solving (6) then consists of a fixed-point algorithm on (29)

$$p_{k+1} = \text{Prox}(p_k). \tag{31}$$

Since computing $\text{Prox}(p_k)$ corresponds to solving a projection problem, we can use any of the algorithmic schemes described in Section 2.1 to implement (31) inside of this proximal algorithm. We call the proximal method the outer algorithm, and the chosen projection algorithm, the inner algorithm. We study in Section 3 the family of proximal algorithms obtained when dual projection algorithms of Section 2.2 are used as inner algorithms.

As we have an iterative optimization algorithm (inner algorithm) inside of another iterative algorithm (outer algorithm), the question of the stopping tests of the inner algorithm is obviously crucial. For practical efficiency, the inner stopping test should somehow depend on some outer information; we come back later in detail to this important point.

So in fact the iteration (31) is not carried out exactly, and replaced instead by a looser implementable relation

$$\|p_{k+1} - \text{Prox}(p_k)\| \leq \varepsilon_k. \tag{32}$$

Whatever is the inner projection algorithm, we have the general global convergence of the method under the assumption that the sequence of errors ε_k goes rapidly to zero.

Proposition 1 (Global convergence) *Assume that there exist a solution to* (6). *If* $(t_k)_k$ *is bounded away from* 0 *and if the primal proximal algorithm generates a sequence* $(p_k)_k$ *such that*

$$\sum_k \varepsilon_k < +\infty \tag{33}$$

then $(p_k)_k$ *converges to a solution* $\bar{p}$ *of* (6).

Proof: The result is straightforward from the general convergence result of proximal algorithms. As a consequence of (33) and the existence of a solution to (6), the sequence $(p_k)_k$ is bounded and we can apply [Roc76b, Th.1]: $(p_k)_k$ converges to a fixed-point to Prox which is a solution of (6) by (30). □

3.1.3 Dual point of view: augmented Lagrangian

We give here some details about the dual interpretation of the above primal algorithmic approach. It is known indeed that a proximal method for a problem corresponds exactly to an augmented Lagrangian method on its dual; we detail this for our case. To simplify writing duals, we abandon the general formulation (6), and we suppose that there is no affine inequalities (or that there are incorporated with slack variables in $\mathcal{K}$). So we work from now with the standard form of primal and dual linear conic problems

$$\left\{ \begin{array}{l} \min \quad c^\top x \\ \quad Ax = b \\ \quad x \in \mathcal{K} \end{array} \right. \quad \text{and} \quad \left\{ \begin{array}{l} \max \quad b^\top y \\ \quad A^\top y - u - c = 0 \\ \quad u \in \mathcal{K}^o. \end{array} \right. \tag{34}$$

Augmented Lagrangian methods are important classical regularization techniques in convex optimization (see [PT72], [Roc76a] for important earlier references, and [HUL93, Chap.XII] for the connection with usual Lagrangian duality). In our situation, a dual augmented Lagrangian method goes along the following lines. Introduce the augmented Lagrangian function L with parameter $t > 0$, for the dual problem (34):

$$L(y, u; p) := b^\top y - p^\top (A^\top y - u - c) - \frac{t}{2}\|A^\top y - u - c\|^2.$$

Note that this is just the usual Lagrangian for the problem

$$\left\{ \begin{array}{c} \max \quad b^\top y - \frac{t}{2}\|A^\top y - u - c\|^2 \\ A^\top y - u - c = 0,\ u \in \mathcal{K}^o, \end{array} \right. \tag{35}$$

that is the dual problem with an additional redundant quadratic term in the objective. The convex (bi)dual function is then defined as

$$\Theta(p) := \max_{y \in \mathbb{R}^m, u \in \mathcal{K}^o} L(y, u; p). \tag{36}$$

The bridge between the primal proximal method and the dual augmented Lagrangian is set in the next proposition, formalizing a well-known result.

Proposition 2 *With notation above, we have $\Theta(p) = F(p)$ for $p \in \mathbb{R}^n$.*

Proof: Just apply [HUL93, XII.5.2.3]: the augmented Lagrangian function $\Theta(p)$ is the Moreau-Yosida of the usual dual function, which is here

$$c^\top p + i_{\{Ax=b\} \cap \mathcal{K}}(p) = \max_{y, u \in \mathcal{K}^o} b^\top y - p^\top (A^\top y - u - c).$$

This is exactly $F(p)$ defined by (28) (in the case when $\mathcal{P}$ is just the affine subspace of equation $Ax = b$).□

The primal regularization by proximal approach and the dual augmented Lagrangian regularization thus correspond exactly to the same quadratic regularization process viewed either on the primal problem or on the dual (34).

The developments of this section share similar properties with other augmented Lagrangian-type approaches for conic programming, among them: a primal augmented Lagrangian in [BV06], a primal-dual augmented Lagrangian in [JR08] and a penalized augmented Lagrangian in [KS07].

3.2 Regularization methods for linear conic programming

In this section we give more details on primal proximal algorithms (or dual augmented Lagrangian algorithms) that use dual projection methods as inner algorithms to carry out the proximal iteration (32). This family of algorithms is introduced for the case $\mathcal{K} = \mathcal{S}_n^+$ in [MPRW09]. They are called regularization algorithms (rather than proximal algorithms, which would focus on the primal point of view only); we keep this terminology here. This section is more technical and could be skipped at a first reading.

Regularization algorithms for conic programming specialize on three points:

1. the dual projection algorithm to compute $\mathrm{Prox}(x_k)$,

2. the rule to stop this inner algorithm,

3. the rule to update the prox-parameter t_k.

The third point is an inherent difficulty of any practical implementation of proximal methods (e.g. bundle methods, see [CL93]). We are not aware of general techniques to tackle it. So we focus here on the first two points.

3.2.1 Dual projection methods as inner algorithms

We could use any dual projection algorithm of Section 2.2 to solve

$$\left\{ \begin{array}{ll} \min & c^\top x + \frac{1}{2t}\|x-p\|^2 \\ & Ax = b,\ x \in \mathcal{K}. \end{array} \right. \tag{37}$$

Embedded in a proximal scheme, a dual projection algorithm would lead to the forthcoming overall algorithm for solving linear conic problems (34).

Note first that equations (14) and (15) for the projection-like problem (37) become respectively

$$x(y) = \mathrm{Proj}_{\mathcal{K}}\left(p + t(A^\top y - c)\right) \tag{38}$$

$$\theta(y) = b^\top y + \frac{1}{2t}(\|p\|^2 - \|x(y)\|^2). \tag{39}$$

We use the (slightly loose) formulation (21) of the iteration of dual projection methods to write a general regularization algorithm. We index the outer iterations by k and the inner ones by ℓ.

Algorithm 1 (Regularization methods)
Outer loop on k stopped when $\|p_{k+1} - p_k\|$ small:
Inner loop on ℓ stopped when $\|Ax_\ell - b\|$ small enough:
Compute $x_\ell = \mathrm{Proj}_{\mathcal{K}}(p_k + t_k(A^\top y_\ell - c))$ and $g_\ell = b - Ax_\ell$
Update $y_{\ell+1} = y_\ell + \tau_\ell W_\ell g_\ell$ with appropriate τ_ℓ and W_ℓ
end (inner loop)
Update $p_{k+1} = x_\ell$ (and t_k)
end (outer loop)

We discuss several points about the above conceptual algorithm.

- *Memory.* An important feature of regularization methods is the rather low memory requirement. The intrinsic operations of the algorithm are basically the projection onto the cone and the multiplications by A and $A^\top$. If the data has some structure, those multiplications could be performed efficiently (without constructing matrices). Moreover for maximizing θ (that is, essentially, implementing $y_{\ell+1} = y_\ell + \tau_\ell W_\ell g_\ell$), we could use algorithms of smooth unconstrained optimization adapted to large-scale problems and then requiring low-memory (as limited memory BGFS or

Newton-CG, see e.g. [NW99] and [BGLS03]). We come back to this point later when discussing numerical issues. Roughly speaking, the point is the following: the computer memory can be used for storing problem data and the computation does not require much more extra memory.

- *Inner restarting.* At the outer iteration k, the inner projection algorithm can be initialized with the best y_ℓ of the previous iteration $k-1$. This has an intuitive appeal, so that in practice, ℓ keeps increasing over the outer iterations. (Note also that the historical information on gradients may be carried out from iteration $k-1$ to k as well.)

- *Dual variable u.* It is known that for any $x \in \mathbb{R}^n$, the projection onto the polar cone $\mathrm{Proj}_{\mathcal{K}^o}(x)$ is given by $\mathrm{Proj}_{\mathcal{K}}(x) + \mathrm{Proj}_{\mathcal{K}^o}(x) = x$ (together with $\mathrm{Proj}_{\mathcal{K}}(x)^\top \mathrm{Proj}_{\mathcal{K}^o}(x) = 0$, see [HUL93, III.3.2.5]). When computing x_ℓ, we thus get automatically

$$u_\ell = \mathrm{Proj}_{\mathcal{K}^o}(p_k + t_k(A^\top y_\ell - c))/t_k$$

  and it holds

$$p_k + t_k(A^\top y_\ell - c) = t_k u_\ell + x_\ell. \tag{40}$$

- *Dual outer iterates.* At the end of outer iteration k, we set (with a slight abuse of notation) $y_{k+1} = y_\ell$ and $u_{k+1} = y_\ell$ for ℓ the final iteration of inner algorithm. Thus we have a sequence of primal-dual outer iterates $(p_k, y_k, u_k) \in \mathcal{K} \times \mathbb{R}^n \times \mathcal{K}^o$. Under some technical assumptions, we can prove a convergence result of the same vein as Proposition 1: any accumulation point of the sequence (p_k, y_k, u_k) is a primal-dual solution of (10) (see e.g. Theorem 4.5 of [MPRW09] for a proof when $\mathcal{K} = \mathcal{S}_n^+$).

- *Outer stopping test.* We have already noticed in (22) that the natural stopping test of dual projection algorithms controls primal infeasibility $\|Ax_\ell - b\|$. Interpreted as a fixed point iteration (31), the natural stopping of the proximal algorithm is $\|p_{k+1} - p_k\|$; it turns out that this can interpreted as controlling dual infeasibility. Note indeed that (40) yields

$$p_k + t_k(A^\top y_k - c) = t_k u_k + p_k$$

  and then we have

$$\|p_{k+1} - p_k\| = t_k \|A^\top y_k - u_k - c\|.$$

- *Normal to interior-point methods.* By construction, conic feasibility $p \in \mathcal{K}$ (and $u \in \mathcal{K}^o$) and complementary $x^\top u = 0$ are ensured throughout the algorithm, while primal-dual feasibilities are obtained asymptotically. In contrast, recall that basic interior-point methods maintain primal and

dual feasibility and the conic feasibility and work to reach complementarity. Note also that, regularization algorithms give solutions that are usually on the boundary of the cone $\mathcal{K}$, since the primal iterates are constructed by projections onto $\mathcal{K}$. In contrast again, basic interior-points give solutions as inside of the cone as possible. In a sense, regularization methods are then "normal" to interior point methods.

- *Overall stopping test.* We have seen above that the natural outer and inner stopping rules of the regularization algorithm have a practical interpretation as dual and primal infeasibilities. Since complementary and conic feasibility are ensured by construction, the natural stopping test of the overall algorithm is

$$\max\left\{\|Ap_k - b\|, \|A^\top y_k - u_k - c\|\right\}. \tag{41}$$

  In practice, one should divide moreover the two infeasibilities by some constant quantities to get homogeneous ratios.

3.2.2 Stopping inner iterations

For which inner iteration ℓ can we set $p_{k+1} = x_\ell$ to proceed with the outer iteration? Since we have a loop inside of another, the rule to terminate the inner projection algorithm is indeed an important technical point for regularization algorithms. We discuss three strategies to set up inner stopping rules.

Solving approximately the inner problem. The usual stopping inner rule in proximal methods is to stop inner iterations when the current inner iterate x_ℓ is close to the proximal point $\mathrm{Prox}(p_k)$. Doing this, the regularization algorithm approximates at best the conceptual proximal algorithm (which requires to solve the inner problem exactly), so that we keep convergence properties (as in Proposition 1 for instance).

This is the strategy followed by the regularization method of [ZST10] for $\mathcal{K} = \mathcal{S}_n^+$. This paper adopts the dual point of view (augmented Lagrangian) and uses semismooth Newton as dual projection algorithm for inner iterations (remember Section 2.2). The regularization method thus combines the usual stopping strategy and an efficient inner algorithm (supporting large-scale problems); it gives excellent numerical results on various semidefinite programming test-problems (see the last section of [ZST10]). Under nondegeneracy assumptions, we have moreover proofs of global and local convergences of the inner algorithm as well as the overall regularization method.

Only one inner iteration; interpretation as saddle-point. An opposite strategy is to do only one inner iteration per outer iteration. This cautious strategy is motivated by the following remark. Let us come back to the proximal formulation (27) of the linear conic problem. Under Slater assumption (18), the inner projection problem can be replaced by its dual (recall (15), (17) and

(38)), so that the primal and dual conic problems (34) have the saddle-point formulation

$$\left\{ \begin{array}{l} \min \\ \quad p \in \mathbb{R}^n \end{array} \quad \left(\begin{array}{l} \max \quad b^\top y - \frac{1}{2t}(\|p\|^2 - \|\operatorname{Proj}_{\mathcal{K}}(p + t(A^\top y - c)\|^2) \\ \quad y \in \mathbb{R}^m \end{array} \right) . \right.$$

With this point of view on the process, the choice of inner stopping conditions appears indeed to be crucial, because the inner and outer loops are antagonistic, as the first minimizes and the second maximizes. The idea of the "Arrow–Hurwicz" approach (see, for instance, [AHU59]) is essentially to proceed with gradient-like iterations with respect to each variable successively.

This is the strategy of the simple regularization method presented in [MPRW09, Sec. 5.1]. Doing one inner iteration of (21) with $W_k = [AA^\top]$ and $\tau_k = 1/t_k$ allows to simplify the regularization algorithm to just one loop with

$$\begin{aligned} p_{k+1} &= \operatorname{Proj}_{\mathcal{K}}(p_k + t_k(A^\top y_\ell - c)) \quad \text{(with } u_{k+1} \text{ as by-product)} \quad (42) \\ y_{k+1} &= y_k + [AA^\top]^{-1}(b - Ap_k)/t_k. \quad (43) \end{aligned}$$

We retrieve algorithm 5.1 of [MPRW09] by using the proof of proposition 3.4 in there. This simple regularization algorithm has also an interpretation as an alternating direction method, see the chapter of this book devoted to them.

In practice, it is important to note that $AA^\top$ and its Cholesky factorization can be computed only once at the beginning of the algorithm. Even if this is an expensive task in general for problems that have many unstructured constraints (so that $AA^\top$ is big and unstructured), there exists some cases when $AA^\top$ is very simple, so that solving the system (43) is cheap. This is the case in particular when $AA^\top$ is diagonal, as for SDP relaxations of max-cut problem, or k-max-cut problem [GW95], frequency assignment problems, see [BMZ01, (5)], max-stable problem, see more below, and polynomial minimization problems, see forthcoming Section 4.

Something in-between. An attractive option is to find something in-between the previous two extreme strategies. Explicit rules for the management of ε_k in (32) should be given, and for numerical efficiency they should be given online. Using off-line rules independent of the function is interesting in theory since it allows to get proof of linear convergence. Following usual paradigms of numerical optimization, it would be possible to do better as soon as practical efficiency in concerned. An appropriate stopping rule still has to be found and studied; this is actually a general question and we are not aware of efficient techniques.

Note finally that the practical implementation of the regularization method of [ZST10] does indeed something in-between: the simple scheme with one inner gradient iteration (second strategy) is used as a preprocessing phase before switching to making inner Newton iterations (first strategy). See more about this on numerical illustrations of the method in Section 4. A stopping test along the above lines would further enhance the numerical performance of the implementation.

3.2.3 Illustration: Computing Lovász theta number

We finish this section with a numerical illustration (borrowed from [MPRW09]) of the performance of regularization methods on a classical combinatorial optimization problem.

Lovász [Lov79] proved the celebrated "sandwich" theorem in graph theory: the stable number $\alpha(G)$ of a graph G and the chromatic number $\chi(\bar{G})$ of its complementary graph $\bar{G}$ are separated

$$\alpha(G) \leq \vartheta(G) \leq \chi(\bar{G})$$

by the optimal value of an SDP problem

$$\vartheta(G) = \left\{ \begin{array}{ll} \max & \langle \mathbf{1}_{n\times n}, X\rangle \\ & X_{ij} = 0, \quad \text{when } (i,j) \text{ is an edge of } G \\ & \text{trace } X = 1, \quad X \succeq 0. \end{array} \right. \tag{44}$$

As expected, it can be shown that this SDP problem is a formulation of the SDP relaxation of the max-stable problem. The stable number $\alpha(G)$ and the chromatic number $\chi(\bar{G})$ are both NP-hard to compute and even hard to approximate, so that the tractable $\vartheta(G)$ gives interesting information about G.

Some graphs from the DIMACS collection [JT96] are very challenging instances for computing $\vartheta(G)$. Those graphs have many edges and also many edges on the complementary, so that makes them the most difficult for standard methods (as noticed in [DR07]). On the other hand, the structure of problem (44) is very favorable for regularization methods and in particular for the simple one of [MPRW09, Sec. 5.1] which is essentially (42)-(43). Observe indeed that the affine constraints (44) is given by "orthogonal" matrices (i.e. $\langle A_j, A_i\rangle = 0$), such that the matrix $AA^\top$ is diagonal. For illustration, the next table reports some of the bounds that were computed for the first time in [MPRW09].

graph name	n	m	$\vartheta(G)$
brock400-1	400	59723	10.388
keller5	776	225990	31.000
brock800-1	800	207505	19.233
p-hat500-1	500	31569	58.036
p-hat1000-3	1000	371746	18.23

For more examples, see [MPRW09, Sec. 5.4] and [ZST10, Sec. 6.3].

4 Applications to polynomial optimization

In this section, we illustrate the regularization methods for solving linear semidefinite optimization problems in the context of polynomial optimization. We collect numerical experiments showing that regularization algorithms can be considered as an alternative to standard methods for deciding whether a polynomial is non-negative (Section 4.2) and for globally minimizing a polynomial

(Section 4.3). The point is thus the same as in [Nie09] which reports extensive numerical results using regularization methods for solving various large-scale polynomial optimization problems.

We aim at giving here a methodology: our main focus is the generality of the approach and the reproducibility of experiments and results. We explain how to generate the test problems, we use public-domain implementations of the algorithms with default parameter tunings and with no attempt to adapt them to each particular problem instances (contrary to [Nie09]). We do not carry out a comprehensive benchmarking with all methods and solvers; we just compare a widely used implementation of interior-point algorithm [Stu99] with two recent implementations of regularization methods (the basic one of [MPRW09, Sec. 5.1], and the more sophisticated one of [ZST10]).

4.1 Sum-of-squares, SDP and software

We briefly introduce in this section the notions and notation of polynomial optimization that we need. We refer to the recent surveys [Lau09] and [Las09] and to the other chapters of this book for more on this topic.

Consider a multivariate polynomial of total degree $2d$

$$v \in \mathbb{R}^N \longmapsto p(v) = \sum_{|\alpha| \leq 2d} p_\alpha v^\alpha. \tag{45}$$

We use here the multi-index notation $v^\alpha = v_1^{\alpha_1} \cdots v_N^{\alpha_N}$ where $\alpha \in \mathbb{N}^N$ runs over all nonnegative integer vectors of sum $|\alpha| = \alpha_1 + \cdots + \alpha_N \leq 2d$. We say that $p(v)$ is a sum-of-squares (SOS) of polynomials if one could find polynomials $q_k(v)$ such that

$$p(v) = \sum_k {q_k}^2(v). \tag{46}$$

It can be shown that finding such polynomials $q_k(v)$ amounts to a semidefinite feasibility problem. More specifically, if $\pi(v)$ denotes a vector of basis of polynomials of total degree less than or equal to d, finding an SOS decomposition (46) amounts to finding a so-called Gram matrix $X \in \mathbb{R}^{n \times n}$ such that

$$p(v) = \pi(v)^\top X \pi(v) \qquad \text{and} \qquad X \in \mathcal{S}_n^+. \tag{47}$$

The set of SOS polynomials has thus a SDP representation of the form

$$Ax = b, \quad x \in \mathcal{K} \tag{48}$$

where $\mathcal{K} = \mathcal{S}_n^+$, $A \in \mathbb{R}^{m \times n^2}$ is a linear operator depending only on the choice of basis $\pi(v)$, and $b \in \mathbb{R}^m$ is a vector depending on $p(v)$. For example if the vector of basis polynomials $\pi(v) = [x^\alpha]_{|\alpha| \leq d}$ contains monomials x^α indexed by $\alpha \in \mathbb{N}^n$, then identifying powers of v in relation (47) yields

$$p_\alpha = \langle A_\alpha, \pi(v)\pi(v)^\top \rangle, \quad \text{for all } \alpha$$

where matrix A_α selects monomials x^α in rank-one matrix $\pi(v)\pi(v)^\top$. More specifically, the entry in A_α with row index β and column index γ is equal to one if $\beta + \gamma = \alpha$ and zero otherwise. In problem (48), the row indexed by α in matrix A collects entries of matrix A_α, and the row indexed by α in vector b is equal to p_α. Note that

$$n = \begin{pmatrix} N+d \\ N \end{pmatrix} \qquad \text{and} \qquad m = \begin{pmatrix} N+2d \\ N \end{pmatrix},$$

so that the sizes of SDP problems grow quickly with the degree and the number of variables.

The important remark is that this type of constraints are favorable to regularization methods: $AA^\top$ is always diagonal indeed. To see this, let α, β denote the row and column indices in matrix $AA^\top$. By construction, the entry (α, β) in $AA^\top$ is equal to $\langle A_\alpha, A_\beta \rangle$: if $\alpha = \beta$, this is equal to the number of non-zero entries in matrix A_α, otherwise, this is zero. Since it is important for numerical efficiency, we formalize the previous remark in a proposition.

Proposition 3 (Orthogonality of constraints) *Let A be the matrix in SOS semidefinite problem (48). Then $AA^\top$ is diagonal with integer entries.*

Polynomial optimization problems that we consider in the next two sections are difficult to tackle directly but admit standard SOS relaxations involving constraints sets (48). In practice, an SOS relaxation approach boils down to solving linear semidefinite problems of the form

$$\left\{ \begin{array}{ll} \min & c^\top x \\ & Ax = b,\ x \in \mathcal{S}_n^+ \end{array} \right. \tag{49}$$

where and $c \in \mathbb{R}^{n^2}$, $b \in \mathbb{R}^m$, $A \in \mathbb{R}^{m \times n^2}$, and vector x collects entries of matrix $X \in \mathbb{R}^{n \times n}$. For solving problem (49), we use the three following the public-domain Matlab implementations:

1. SeDuMi1.3 implementing the primal-dual interior-point algorithm of [Stu99] (available on `sedumi.ie.lehigh.edu`)
2. MPRW a version of the basic regularization method of [MPRW09, Sec. 5.1] (available on `www.math.uni-klu.ac.at/or/Software/mprw2.m`)
3. SDPNAL0.1 the regularization method of [ZST10] (available on `www.math.nus.edu.sg/~mattohkc/SDPNAL.html`)

Our goal here is just to show that the regularization methods are interesting in this context. We simply use default parameter tunings of the algorithms, with no attempt to adapt them to each particular problem instances contrary to in [Nie09]. With `K.s=n`, the calling sequences of the three Matlab functions `SeDuMi`, `MPRW` and `SDPNAL` for solving (49) are thus as follows:

```
pars = []; pars.tol = 1e-9;
[x,y] = sedumi(A,b,c,K,pars);
X = reshape(x,K.s,K.s);

tol = 1e-9; C = reshape(c,K.s,K.s);
[X,y] = mprw(A,b,C,1e6,1,tol);

opts = []; opts.tol = 1e-9;
[blk,At,C,B] = read_sedumi(A,b,c,K);
[obj,X,y] = sdpnal(blk,At,C,B,opts);
X = X{1};
```

Experiments are carried out with Matlab 7.7 running on a Linux PC with Intel Xeon CPU W3520 2.67Ghz using 64 bit arithmetic and 8GB RAM. Computation times are given in seconds, with two significant digits only (since our objective is not a comprehensive accurate benchmarking of codes).

Similarly to [Nie09], we will see that, due to lower memory requirements, regularization methods can solve larger polynomial optimization problems than classical interior-point methods with the above setting.

A last note about tolerance. The tolerance parameters `tol` for the three solvers are set to 10^{-9} for all the numerical experiments (except otherwise stated). Notice though that the meaning of the tolerance parameter is not the same for two types of algorithms. With regularization methods, the relative accuracy measured in terms of primal and residuals (remember (41)) is easily controlled. We stress that lower requirements on the relative accuracy could result in a significant saving of computational time, and this could be useful when solving approximately large-scale problems with MPRW and SDPNAL (see some examples in [Nie09]). In contrast, we observe (as expected) that the iteration count of SeDuMi does not depend significantly on the expected accuracy, measured in terms of duality gap. Most of the computational time is spent to generate an approximately feasible primal-dual pair with relatively small duality gap, and only a few more iterations are required to refine the accuracy below 10^{-9}.

4.2 Testing positivity of polynomials

We focus in this section on the very first problem of polynomial optimization: we would like to know whether a polynomial (45) is positive

$$p(v) \geq 0, \quad \text{for all } v \in \mathbb{R}^N. \tag{50}$$

In general this is a difficult problem for which no polynomial-time algorithm is known. It can be relaxed to the easier problem of testing if p could be expressed as an SOS (46). Whenever it holds, then obviously condition (50) is satisfied. The converse is not true in general if $N \geq 2$ and $d \geq 3$, and there are explicit counter-examples; the simplest of them (the Motzkin polynomial) is studied below.

4.2.1 Random full-rank polynomial SOS problems

We consider random polynomial SOS problems which are constructed so that there is a full-rank orthogonal Gram matrix X (an interior point) solving problem (47). We use GloptiPoly 3 (see [Las09]) to generate matrix A and vector b as follows:

```
N = 5; % number of variables
d = 3; % half degree
mpol('v',N,1); % variables
P = msdp(min((v'*v)^d); % construct A matrix
[A,b,c,K] = msedumi(P); % retrieve A and K in SeDuMi format
A = [c';-A]; % constant term and sign change
c = zeros(size(A,2),1); % no objective function
X = orth(randn(K.s)); % random Gram matrix
b = A*X(:); % corresponding right handside vector
```

On Table 1 we report execution times (in seconds) required by SeDuMi, MPRW and SDPNAL to solve problem (48) for $d = 3$ (degree six polynomials) and $N = 5, \ldots, 12$. We also indicate the size n of matrix X and the number m of constraints (row dimension of matrix A). We observe that SeDuMi is largely outperformed by MPRW and SDPNAL. We also observe that MPRW is about 4 times slower than SDPNAL, but this is not surprising as MPRW is a simple prototype (without comments and printing instructions it is about 50 lines of interpreted Matlab), whereas SDPNAL is a much more sophisticated package heavily relying on the efficient data handling and numerical linear algebra routines of the SDPT3 package. We also recall that SDPNAL makes several iterations of MPRW as preprocessing.

N	n	m	SeDuMi	MPRW	SDPNAL
5	56	462	0.29	0.03	0.05
6	84	924	0.92	0.05	0.07
7	120	1716	4.8	0.13	0.10
8	165	3003	25	0.35	0.16
9	220	5005	110	0.66	0.25
10	286	8008	410	1.3	0.43
11	364	12376	1500	3.0	0.73
12	455	18564	> 3600	5.0	1.3

Table 1: Comparative execution times for SOS problems.

4.2.2 Random low-rank polynomial SOS problems

We consider random polynomial SOS problems which are constructed so that there is a rank-one Gram matrix X solving problem (48). For such problems, it is unlikely that there is an interior point x solving problem (48), and indeed

SeDuMi does not find a full-rank solution. We use the same code as above, replacing the instruction `X = orth(randn(K.s));` with the instructions

```
X = orth(randn(K.s,1));
X = X*X';
```

We report execution times (in seconds) in Table 2. In comparison with the problems with interior points of Table 1, we observe that all the solvers experience convergence issues. However, there is still a considerable improvement in terms of efficiency brought by regularization methods, though Slater's qualification constraint cannot be invoked to guarantee convergence.

N	n	m	SeDuMi	MPRW	SDPNAL
5	56	462	0.83	0.20	0.21
6	84	924	0.85	0.32	0.28
7	120	1716	16	2.1	0.51
8	165	3003	61	4.8	0.98
9	220	5005	330	12	1.2
10	286	8008	1300	24	2.5
11	364	12376	> 3600	50	3.5
12	455	18564	> 3600	110	6.6

Table 2: Comparative execution times for low-rank SOS problems.

4.2.3 Motzkin's polynomial

We study a well-known bivariate ($N = 2$) polynomial of sixth degree ($d = 3$) which is non-negative but cannot be written as a polynomial SOS, namely Motzkin's polynomial

$$p_0(v) = 1 + v_1^2 v_2^2 (v_1^2 + v_2^2 - 3)$$

see [Lau09] or [Las09]. This polynomial achieves its minimum zero at the four points $v_1 = \pm 1$, $v_2 = \pm 1$. In a basis of monomials of degree up to 3 there is no Gram matrix X solving problem (48). However, it was observed in [HL05] and later on shown theoretically in [Las06] that the perturbed polynomial

$$p_0(v) + \varepsilon p_1(v)$$

can be represented as a polynomial SOS (with full-rank Gram matrix) provided the degree of the perturbation polynomial $p_1(v)$ is high enough, inversely proportional to scalar $\varepsilon > 0$. In some sense, this can be interpreted as a regularization procedure as in [HM11]. Practically speaking, since semidefinite programming solvers use inexact operations (floating point arithmetic), it is not necessary to perturb explicitly the data. It is enough to choose a basis $\pi(v)$ of sufficiently high degree $d > 3$ in relation (47), and higher-order perturbations are automatically introduced by the algorithm.

We use the following GloptiPoly3 instructions to generate data A, b for increasing values of d:

```
d = 8; % half degree
mpol v1 v2
p = 1+v1^2*v2^2*(v1^2+v2^2-3);
P = msdp(min(p),d);
[A,b,c,K,b0] = msedumi(P);
A = [c';-A];
b = [-b0;-b];
c = zeros(size(A,2),1);
```

For this problem, we set `tol=1e-6` for the three solvers. When $d = 3, 4, 5, 6$, SeDuMi takes less than 0.1 seconds to detect that problem (48) is infeasible, and it provides a Farkas dual certificate vector $y \in -\mathcal{K}$ such that $b^\top y = 1$. When $d = 7, 8, 9, 10$, SeDuMi takes less than 0.5 seconds to return a vector x such that the primal residual $\|Ax - b\|_2/\|b\|_2$ is less than 10^{-9} and the dual objective function $b^\top y$ is less than 10^{-9} in absolute value.

The behavior of SDPNAL and MPRW is more erratic, and convergence issues occur, as shown by the execution times (in seconds) of Table 3. For $d = 3, 4, 5$, MPRW stops after 10^6 iterations, as there is no mechanism to detect infeasibility in this prototype software.

d	time	$\|Ax - b\|_2/\|b\|_2$	$b^\top y$
3	-	-	-
4	-	-	-
5	-	-	-
6	15	$6.20 \cdot 10^{-6}$	$1.12 \cdot 10^{-6}$
7	25	$6.03 \cdot 10^{-7}$	$6.81 \cdot 10^{-7}$
8	26	$5.80 \cdot 10^{-6}$	$-4.08 \cdot 10^{-7}$
9	34	$1.01 \cdot 10^{-6}$	$-1.45 \cdot 10^{-7}$
10	75	$5.42 \cdot 10^{-7}$	$-1.58 \cdot 10^{-7}$

d	time	$\|Ax - b\|_2/\|b\|_2$	$b^\top y$
3	5.1	$4.28 \cdot 10^{-3}$	33.4
4	9.2	$1.56 \cdot 10^{-4}$	0.832
5	3.5	$4.59 \cdot 10^{-6}$	$4.37 \cdot 10^{-5}$
6	4.6	$6.33 \cdot 10^{-6}$	$1.05 \cdot 10^{-6}$
7	5.7	$8.95 \cdot 10^{-6}$	$3.86 \cdot 10^{-7}$
8	5.9	$2.79 \cdot 10^{-6}$	$-3.46 \cdot 10^{-7}$
9	7.9	$2.54 \cdot 10^{-6}$	$-3.25 \cdot 10^{-7}$
10	8.8	$1.88 \cdot 10^{-6}$	$-1.34 \cdot 10^{-7}$

Table 3: Behavior of MPRW (left) and SDPNAL (right) for Motzkin's polynomial.

4.2.4 Regularization vs projection

Though it solves linear semidefinite problems, using regularization techniques somehow generalizes and enhances the idea [HM11] to using projection methods

directly for SOS feasibility problems. With this approach indeed, a question is to find a good point to project; taking systematically the zero matrix gives interesting results but could be greatly enhanced. Regularization methods provide a numerical solution to this: doing a sequence of truncated projections allows to keep the projection idea while getting rid of the question of the initial point to project. The behaviour of SDPNAL is interesting with this respect: it does first a preprocessing of several alternating direction iterations to get a meaningful point, then follows by projection-like iterations. In practice, we observe usually a very few iterations, and often one. For example, to decide whether the (admittedly trivial) polynomial $p(v) = \sum_{i=1}^{10} v_i^{10}$ is SOS, the SDP problem dimensions are $n = 3003$ and $m = 184756$, and after 90 seconds and only one projection-like iteration, SDPNAL provides a vector x satisfying $\|Ax - b\|_2/\|b\|_2 \approx 1.4 \cdot 10^{-10}$.

4.3 Unconstrained polynomial minimization

In this section we study global minimization problems

$$p^* = \min_{v \in \mathbb{R}^N} p(v) \tag{51}$$

where $p(v)$ is a given polynomial. For this problem, a semidefinite relaxation readily follows from the observation that

$$\begin{array}{rcll} p^* & = & \max_{\underline{p}} & \underline{p} \\ & & \text{s.t.} & p(v) - \underline{p} \geq 0, \quad \forall v \in \mathbb{R}^N \end{array}$$

and by relaxing the above non-negativity constraint by the semidefinite programming constraint that polynomial $p(v) - \underline{p}$ is SOS, see [Lau09] and [Las09].

4.3.1 Random polynomial minimization problems

We generate well-behaved instances of unconstrained polynomial minimization problems (51) with

$$p(v) = p_0(v) + \sum_{i=1}^{N} v_i^{2d}$$

where $p_0(v)$ is a random polynomial of total degree strictly less than $2d$. The leading term $\sum_{i=1}^{N} v_i^{2d}$ ensures coercivity of $p(v)$ and hence existence of a global minimum in (51). We use the following GloptiPoly 3 script to generate our examples:

```
N = 10;
mpol('v',N,1);
b = mmon(v,0,2*d-1); % degree up to 2d-1
p0 = randn(1,length(b)); p0 = p0/norm(p0);
p = p0*b + sum(mmon(v,d).^2);
P = msdp(min(p));
[A,b,c,K] = msedumi(P);
```

In Table 4 we report comparative execution times (in seconds) for $d = 2$ and various values of N, for solving the semidefinite relaxation. It turns out that for these generic problems, we observe that global optimality is always certified with a rank-one moment matrix [HL05]. Both MPRW and SDPNAL largely outperform SeDuMi on these examples.

N	n	m	SeDuMi	MPRW	SDPNAL
5	21	126	0.09	0.05	0.18
6	28	209	0.11	0.07	0.18
7	36	329	0.24	0.12	0.20
8	45	494	0.36	0.19	0.22
9	55	714	0.77	0.28	0.26
10	66	1000	1.9	0.45	0.29
11	78	1364	5.0	0.78	0.36
12	91	1819	11	1.1	0.41
13	105	2379	20	1.6	0.47
14	120	3059	42	2.3	0.65
15	136	3875	74	3.0	0.68

Table 4: Comparative execution times for semidefinite relaxations of random polynomial minimization problems.

4.3.2 A structured example

Consider the problem studied in [Nie09, Example 3.5], that is (51) with

$$p(v) = \sum_{i=1}^{N} \left(1 - \sum_{j=1}^{i} (v_j + v_j^2)\right)^2 + \left(1 - \sum_{j=1}^{N} (v_j + v_j^3)\right)^2 .$$

We solve the semidefinite relaxation for increasing values of N. We collect comparative execution times on Table 5 for this example. For example, when $N = 10$, SDPNAL resp. MPRW returns a point x such that $\|Ax - b\|/\|b\|$ is equal to $1.4 \cdot 10^{-9}$ resp. $2.6 \cdot 10^{-10}$ and the minimum eigenvalue of X is equal to zero to machine precision.

We observe again a considerable improvement in terms of performance brought by regularization methods in comparison with a classical interior-point method. For larger instances, most of the computation time of SeDuMi is spent for memory swapping when constructing and handling large matrices. We refer to the recent extensive numerical work of [Nie09] for various structured problems.

Acknowledgements

The work of the first author was partly supported by Research Programme MSM6840770038 of the Czech Ministry of Education and by Project 103/10/0628 of the Grant Agency of the Czech Republic.

N	n	m	SeDuMi	MPRW	SDPNAL
5	56	461	0.50	0.54	0.60
6	84	923	2.5	1.2	1.2
7	120	1715	14	5.0	2.6
8	165	3002	92	19	6.7
9	220	5004	410	65	22
10	286	8007	1800	200	71
11	364	12375	7162	490	150
12	455	18563	> 7200	1500	530
13	560	27131	> 7200	3500	2300
14	680	38760	> 7200	> 7200	9900

Table 5: Comparative execution times for semidefinite relaxations of a larger polynomial minimization problem.

References

[AH07] S. Al-Homidan. Solving Hankel matrix approximation problem using semidefinite programming. *Journal of Computational and Applied Mathematics*, 202(2):304 – 314, 2007.

[AHO97] F. Alizadeh, J.-P. Haeberly, and M. Overton. Complementarity and nondegeneracy in semidefinite programming. *Math. Program.*, 77(2):111–128, 1997.

[AHTW03] M. F. Anjos, N. J. Higham, P. L. Takouda, and H. Wolkowicz. A semidefinite programming approach for the nearest correlation matrix problem. Technical report, Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario, N2L 3G1 Canada, September 2003.

[AHU59] K. Arrow, L. Hurwicz, and H. Uzawa. *Studies in Linear and Nonlinear Programming*. Stanford University Press, 1959.

[BDL08] J. Bolte, A. Daniilidis, and A. Lewis. Tame functions are semismooth. *Math. Program.*, 117(1):5–19, 2008.

[Ber95] D.P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1995.

[BGFB94] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*. Studies in Applied Mathematics. SIAM, 1994.

[BGLS03] J.F. Bonnans, J.Ch. Gilbert, C. Lemaréchal, and C. Sagastizábal. *Numerical Optimization*. Springer Verlag, 2003.

[BH08] R. Borsdorf and N. Higham. A preconditionned Newton algorithm for the nearest correlation matrix. *Submitted*, 2008.

[BKL66] R. Bellman, R. Kalaba, and J. Lockett. *Numerical Inversion of the Laplace Transform*. Elsevier, 1966.

[BM06] D. Brigo and F. Mercurio. *Interest rate models: theory and practice*. Springer-Verlag, 2006.

[BMZ01] S. Burer, R. Monteiro, and Y. Zhang. A computational study of a gradient-based log-barrier algorithm for a class of large-scale sdps. In *Mathematical Programming Series B*, pages 359–379, 2001.

[BS00] J.F. Bonnans and A. Shapiro. *Perturbation Analysis of Optimization Problems*. Springer Verlag, 2000.

[BV04] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[BV06] Samuel Burer and Dieter Vandenbussche. Solving lift-and-project relaxations of binary integer programs. *SIAM Journal on Optimization*, 16:726–750, 2006.

[BX05] S. Boyd and L. Xiao. Least-squares covariance matrix adjustment. *SIAM Journal on Matrix Analysis and Applications*, 27(2), 2005.

[CL93] R. Correa and C. Lemaréchal. Convergence of some algorithms for convex minimization. *Mathematical Programming*, 62(2):261–275, 1993.

[Cla83] F.H. Clarke. *Optimization and Nonsmooth Analysis*. Wiley, 1983; reprinted by SIAM, 1983.

[DET82] R. Dembo, S. Eisenstat, and T.Steihaug. Inexact Newton methods. *SIAM Journal on Numerical Analysis*, 19(2), 1982.

[Deu01] F. Deutsch. *Best Approximation in Inner Product Spaces*. Springer, New York, 2001.

[DR56] J. Douglas and H.H. Rachford. On the numerical solution of heat conduction problems in two and three space variables. *Trans. Amer. Math. Soc.*, 82:421–439, 1956.

[DR07] I. Dukanovic and F. Rendl. Semidefinite programming relaxations for graph coloring and maximal clique problems. *Mathematical Programming*, 109:345–365, 2007.

[Dum07] B. Dumitrescu. *Positive Trigonometric Polynomials and Signal Processing Applications*. Springer Verlag, 2007.

[Dyk83] R.L. Dykstra. An algorithm for restricted least-square regression. *Journal of the American Statistical Association*, 78:837–842, 1983.

[GL89] J.Ch. Gilbert and C. Lemaréchal. Some numerical experiments with variable-storage quasi-Newton algorithms. *Mathematical Programming*, 45:407–435, 1989.

[GM76] D. Gabay and B. Mercier. A dual algorithm for the solution of nonlinear variational problems via finite-element approximations. *Computers and Mathematics with Applications*, 2, 1976.

[GS09] Y. Gao and D. Sun. Calibrating least squares semidefinite programming with equality and inequality constraints. *SIAM J. Matrix Anal. Appl.*, 31(3):1432–1457, 2009.

[GS10] Y. Gao and D. Sun. A majorized penalty approach for calibrating rank constrained correlation matrix problems. Technical report, Univ. of Singapore, 2010.

[GW95] M. Goemans and D. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 6:1115–1145, 1995.

[HG05] D. Henrion and A. Garulli. *Positive polynomials in control*. LNCIS, Springer Verlag, 2005.

[Hig88] N. Higham. Computing a nearest symmetric positive semidefinite matrix. *Linear Algebra and its Applications*, 103:103–118, 1988.

[Hig02] N. Higham. Computing a nearest symmetric correlation matrix — a problem from finance. *IMA Journal of Numerical Analysis*, 22(3):329–343, 2002.

[HL05] D. Henrion and J.-B. Lasserre. Detecting global optimality and extracting solutions in gloptipoly. In D. Henrion and A. Garulli, editors, *Positive polynomials in control*, LNCIS. Springer, 2005.

[HM11] D. Henrion and J. Malick. Projection methods for conic feasibility problems; application to sum-of-squares decompositions. *Optimization Methods and Software*, 26(1):23–46, 2011.

[HUL93] J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex Analysis and Minimization Algorithms*. Springer Verlag, Heidelberg, 1993. Two volumes.

[HUSN84] J.-B. Hiriart-Urruty, J.J. Strodiot, and H.V. Nguyen. Generalized hessian matrix and second-order optimality conditions for problems with $C^{1,1}$ data. *Applied Mathematics and Optimization*, 11:43–56, 1984.

[HXY09] B. He, M. Xu, and X. Yuan. Solving large-scale least-squares covariance matrix problems by alternating direction methods. *Technical report*, 2009.

[JR08] F. Jarre and F. Rendl. An augmented primal-dual method for linear conic problems. *SIAM Journal on Optimization*, 2008.

[JT96] D.J. Johnson and M.A. Trick, editors. *Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge, Workshop, October 11-13, 1993*. American Mathematical Society, Boston, MA, USA, 1996.

[KS03] M. Kočara and M. Stingl. PENNON: a code for convex nonlinear and semidefinite programming. *Optimisation Methods and Sofware*, 18(3):317–333, 2003.

[KS07] M. Kočvara and M. Stingl. On the solution of large-scale sdp problems by the modified barrier method using iterative solvers. *Math. Program.*, 109(2):413–444, 2007.

[Las06] J.-B. Lasserre. A sum of squares approximation of nonnegative polynomials. *SIAM J. Optim.*, 16:751 – 765, 2006.

[Las09] J.B. Lasserre. *Moments, Positive Polynomials and Their Applications*. Imperial College Press, 2009.

[Lau09] M. Laurent. Sums of squares, moment matrices and optimization over polynomials. In M. Putinar and S. Sullivant, editors, *Emerging Applications of Algebraic Geometry*, IMA Volumes in Mathematics and its Applications. Springer, 2009.

[LJ09] F. Lin and M. R. Jovanović. Least-squares approximation of structured covariances. *IEEE Trans. Automat. Control*, 54(7):1643–1648, July 2009.

[LLM09] A. Lewis, D. Luke, and J. Malick. Local linear convergence for alternating and averaged nonconvex projections. *Foundations of Computational Mathematics*, 9:485–513, 2009.

[LM08] A. Lewis and J. Malick. Alternating projections on manifolds. *Mathematics of Operations Research*, 33(1):216–234, 2008.

[Lov79] L. Lovász. On the Shannon capacity of a graph. *IEEE Transactions on Information Theory*, IT 25:1–7, 1979.

[LQ10] Q. Li and H. Qi. A sequential semismooth Newton method method for the nearest low-rank correlation matrix problem. Technical report, 2010.

[Mal04] J. Malick. A dual approach to semidefinite least-squares problems. *SIAM Journal on Matrix Analysis and Applications*, 26, Number 1:272–284, 2004.

[Mar70] B. Martinet. Régularisation d'inéquations variationnelles par approximations successives. *Revue Française d'Informatique et Recherche Opérationnelle*, R-3:154–179, 1970.

[MPRW09] J. Malick, J. Povh, F. Rendl, and A. Wiegele. Regularization methods for semidefinite programming. *SIAM Journal on Optimization*, 20(1):336–356, 2009.

[MS06] J. Malick and H. Sendov. Clarke generalized Jacobian of the projection onto the cone of positive semidefinite matrices. *Set-Valued Analysis*, 14(3):273–293, 2006.

[MU88] C.A. Micchelli and F.I. Utretas. Smoothing and interpolation in a convex subset of an Hilbert space. *SIAM J. Sci. Stat. Comput.*, 9:728–746, 1988.

[Nes05] Y. Nesterov. Smooth minimization of non-smooth functions. *Math. Program.*, 103(1):127–152, 2005.

[Nie09] J. Nie. Regularization methods for sum of squares relaxations in large scale polynomial optimization. Technical report, ArXiv, 2009.

[NN94] Y. Nesterov and A.S. Nemirovski. *Interior-Point Polynomial Algorithms in Convex Programming*. Number 13 in SIAM Studies in Applied Mathematics. SIAM, Philadelphia, 1994.

[NW99] J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer Verlag, New York, 1999.

[NWV08] M. Nouralishahi, C. Wu, and L. Vandenberghe. Model calibration for optical lithography via semidefinite programming. *Optimization and Engineering*, 9:19–35, 2008.

[PT72] B.T. Polyak and N.V. Tretjakov. On an iterative method for linear programming and its economical interpretations. *Èkonom. i Mat. Methody*, 8:740–751, 1972.

[QS93] L.Q. Qi and J. Sun. A nonsmooth version of Newton's method. *Mathematical Programming*, 58(3):353–367, 1993.

[QS06] H. Qi and D. Sun. Quadratic convergence and numerical experiments of Newton's method for computing the nearest correlation matrix. *SIAM Journal on Matrix Analysis and Applications*, 28:360–385, 2006.

[QS10] H. Qi and D. Sun. An augmented Lagrangian dual approach for the h-weighted nearest correlation matrix problem. *IMA Journal of Numerical Analysis*, 2010.

[Roc76a] R.T. Rockafellar. Augmented Lagrangians and applications of the proximal point algorithm in convex programming. *Mathematics of Operations Research*, 1:97–116, 1976.

[Roc76b] R.T. Rockafellar. Monotone operators and the proximal point algorithm. *SIAM Journal on Control and Optimization*, 14:877–898, 1976.

[SA79] N. Schwertman and D. Allen. Smoothing an indefinite variance-covariance matrix. *Journal of Statistical Computation and Simulation*, 9:183–194, 1979.

[SS02] D. Sun and J. Sun. Semismooth matrix valued functions. *Mathematics of Operations Research*, 57, 2002.

[Stu99] J.F Sturm. Using Sedumi 1.02, a Matlab toolbox for optimization over symmetric cones. *Optmization Methods and Softwares*, 11-12:625–653, 1999.

[SVW00] R. Saigal, L. Vandenberghe, and H. Wolkowicz. *Handbook of Semidefinite-Programming*. Kluwer, 2000.

[SZ10] J. Sun and S. Zhang. A modified alternating direction method for convex quadratically constrained quadratic semidefinite programs. *European Journal of Operational Research*, 207(3):1210 – 1220, 2010.

[Toh08] K. Toh. An inexact primal-dual path following algorithm for convex quadratic SDP. *Math. Program.*, 112(1):221–254, 2008.

[TTT03] R.H Tutuncu, K.C. Toh, and M.J. Todd. Solving semidefinite-quadratic-linear programs using SDPT3. *Mathematical Programming B*, 95:189–217, 2003.

[TTT06] R.H Tutuncu, K.C. Toh, and M.J. Todd. Inexact primal-dual path-following algorithms for a special class of convex quadratic SDP and related problems. *Pacific Journal on Optimization*, 2006.

[ZST10] X. Zhao, D. Sun, and K. Toh. A Newton-CG augmented lagrangian method for semidefinite programming. *SIAM J. Optim*, 20(4), 2010.