Advances in Convex Optimization: Conic Programming

Arkadi Nemirovski*

Abstract. During the last two decades, major developments in Convex Optimization were focusing on *Conic Programming*, primarily, on Linear, Conic Quadratic and Semidefinite optimization. Conic Programming allows to reveal rich structure which usually is possessed by a convex program and to exploit this structure in order to process the program efficiently. In the paper, we overview the major components of the resulting theory (conic duality and primal-dual interior point polynomial time algorithms), outline the extremely rich "expressive abilities" of Conic Quadratic and Semidefinite Programming and discuss a number of instructive applications.

Mathematics Subject Classification (2000). Primary 90C22,90C25,90C51,90C90; Secondary 49N15.

Keywords. Convex, conic and semidefinite programming, duality, polynomial time algorithms.

1. Conic Programming – motivation

1.1. Efficiency issues in Mathematical Programming. Mathematical Programming is about solving *optimization programs* of the form

$$\mathrm{Opt} = \min_{x \in \mathbf{R}^n} \{f_0(x) : f_i(x) \le 0,\ i = 1, ..., m\}, \tag{1}$$

where the *objective* $f_0(\cdot)$ and the *constraints* $f_i(\cdot)$ are given functions on $\mathbf{R}^n$, usually assumed to be smooth (at least C^1). The corresponding body of knowledge, in its mathematical aspects (that is, aside of modelling and implementation issues), focuses on characterization of optimal solutions (necessary/sufficient optimality conditions), their sensitivity to program's data and on developing and theoretical investigation of computational algorithms aimed at building approximate solutions. It should be stressed that the ultimate goal is to *find* a solution rather than to *prove* its existence, uniqueness, etc. As a matter of fact, the situations where a "closed analytic form solution" is available are rare exceptions, this is why the focus is on algorithms capable to approximate optimal solutions within (any) desired accuracy. Moreover, what matters is not just the convergence of

*ISYE, Georgia Institute of Technology, and Technion – Israel Institute of Technology.

an algorithm, but its *efficiency* – a "reasonable" dependence of the computational effort on the required accuracy. The emphasis on the algorithmic and efficiency aspects is what makes the major difference between Mathematical Programming (MP) and its "pure" counterparts, like Calculus of Variations, and gives the MP theory its specific flavour.

Mathematical Programming arose in early 1950's as a natural extension of Linear Programming (LP). The latter, invented around 1947 by George Dantzig, focuses on optimization programs with linear objective and constraints. Significant post-war demand on techniques for modelling and processing logistic and planning problems, nice duality theory allowing, among other things, for far-reaching interpretations in Economics, excellent practical performance of the famous LP Simplex algorithm (Dantzig 1947), availability of new computational devices – computers – all these factors were crucial for emerging and rapid development of theory and algorithms for LP and MP. Eventually, in these developments the problem of *theoretically efficient solvability* of various classes of MP programs was addressed. This problem was posed first for Linear Programming, and the corresponding story is an excellent example of the "specific flavour" of MP we have mentioned. At the first glance, the problem of finding a solution to a system of m linear inequalities with n real unknowns (this is what LP is about) seems completely trivial – invoking the standard characterization of extreme points of polyhedral sets, it reduces to solving at most $\binom{m}{n}$ systems of linear equations. The difficulty is that the number of systems blows up exponentially with the sizes m, n of the system, which makes this naive approach completely impractical already with m, n like few tens. Dantzig's Simplex algorithm inspects the systems in a "smart" order, which in practice allows to arrive at a solution pretty fast; however, in the worst case, exponentially many systems should be inspected, so that the Simplex method is *not* theoretically efficient. Developing theoretically efficient algorithms for LP is an extremely challenging and deep problem, and it took over 20 years to find the first such algorithm (Khachiyan, 1978). Note that all known efficient LP algorithms do something completely different from inspecting the above systems of linear equations.

Informally, the question we are interested in is: given a class of MP programs, does it admit an efficient solution algorithm, and the first issue here is what does efficiency mean. In Combinatorial Complexity Theory (CCT), there are good reasons to qualify a 'fully finite" algorithm (e.g., Turing Machine) converting finite binary words – data of instances of a discrete problem – into solutions to the instances as *efficient*, if the conversion time is polynomial in the bit length of the input (see, e.g., [21]). For problems with continuous data and decision variables, similar reasons lead to the concept of a Real Arithmetic polynomial time algorithm as follows ([8]; cf. [12]). Consider a *generic* MP problem $\mathcal{P}$ – a family of instances of the form (1), with instance p specified within $\mathcal{P}$ by its *data vector* $\mathrm{Data}(p) \in \mathbf{R}^{N(p)}$.

E.g., LP can be naturally considered as a generic problem, with the data vector $\mathrm{Data}(p)$ of an LP program p defined as follows: the first 2 entries are the numbers $m = m(p)$ of constraints and $n = n(p)$ of variables, and the remaining $(m(p)+1)(n(p)+1)-1$ entries

are the vectors of coefficients of the objective and constraints stacked atop each other into a single column.

A *solution algorithm* $\mathcal{B}$ for $\mathcal{P}$ is a code for an idealized Real Arithmetic computer capable to store real numbers and to perform exact Real Arithmetic operations (a.o.) – four arithmetic operations, comparisons and computations of univariate elementary functions, like $\sqrt{\cdot}$, $\exp\{\cdot\}$, etc. Loaded with this code and an input comprised of $\mathrm{Data}(p)$, $p \in \mathcal{P}$, and $\epsilon > 0$, the computer should terminate in a finite number $N(p,\epsilon)$ of operations and output either an ϵ-solution to p – a vector $x_\epsilon \in \mathbf{R}^{n(p)}$ such that $f_i^{(p)}(x_\epsilon) \le \epsilon$, $i = 1, ..., m(p)$, and

$$f_0^{(p)}(x_\epsilon) \le \epsilon + \mathrm{Opt}(p),\ \mathrm{Opt}(p) = \inf_x\{f_0^{(p)}(x) : f_i^{(p)}(x) \le 0,\ i = 1, ..., m(p)\}$$

(here $n(p)$ is number of variables in p, $f_0^{(p)}, ..., f_{m(p)}^{(p)}$ are the objective and the constraints of p), or a correct claim that p is infeasible (i.e., $\mathrm{Opt}(p) = +\infty$), or a correct claim that p is unbounded (i.e., $\mathrm{Opt}(p) = -\infty$). A solution method $\mathcal{B}$ is *polynomial time* ("computationally efficient"), if $N(p,\epsilon)$ is bounded by a polynomial in the *size* $\mathrm{Size}(p) = \dim \mathrm{Data}(p)$ of the instance and the *number of accuracy digits* in an ϵ-solution defined as

$$\mathrm{Digits}(p,\epsilon) = \log\left([\mathrm{Size}(p) + \|\mathrm{Data}(p)\|_1 + \epsilon^2]/\epsilon\right) = (1 + o(1))\log(1/\epsilon),\ \epsilon \to 0.$$

Finally, generic problem $\mathcal{P}$ is called *polynomially solvable* ("computationally tractable"), if it admits a polynomial time solution algorithm.

The standard informal interpretation of polynomiality is that with solution time fixed, a 10-fold progress in computer's performance allows for both (a) a *constant factor* progress in the sizes of instances which can be solved to a given accuracy, and (b) a *constant factor* progress in the number of accuracy digits to which an instance of a given size can be solved. (a), (b) compare favourably polynomial time algorithms with methods suffering from "curse of dimensionality" ($N(p,\epsilon)$ can grow with $\mathrm{Size}(p)$ as $\exp\{\mathrm{Size}(p)\}$), same as with sublinearly converging methods ($\mathcal{N}(p,\epsilon)$ can grow as $1/\epsilon^c$, $c > 0$, when $\epsilon \to 0$).

1.2. Convex programming – solvable case of MP.

At early 1980's it became clear that what forms the "efficiently solvable case" in MP, is *Convex Programming* – programs (1) with convex objective and constraints. Specifically, it was proved that *generic convex problems, under mild computability and boundedness assumptions, are polynomially solvable*. In contrast to this, *no efficient algorithms for typical generic non-convex problems are known, and there are strong reasons to believe that no such algorithms exist* (e.g., programs with quadratic objective and constraints are not polynomially solvable unless P=NP).

Here is a basic example of a "convex programming solvability statement" (cf. [8, Theorem 5.3.1]):

Theorem 1.1. *A generic MP problem $\mathcal{P}$ with convex instances is polynomially solvable, provided it possesses the following properties:*

(i) [polynomial computability] *There exists an algorithm $\mathcal{O}$ which, given on input* $\mathrm{Data}(p)$*, a point $x \in \mathbf{R}^{n(p)}$ and a tolerance $\delta > 0$, computes in polynomial in* $\mathrm{Size}(p)$ *and* $\mathrm{Digits}(p,\delta)$ *number of a.o. δ-accurate approximations to the values and subgradients of*

the objective and the constraints of p at x;

(ii) [polynomial growth] $\max\limits_{0\le i\le m(p)} |f_i^{(p)}(x)| \le (\chi[\mathrm{Size}(p) + \|\mathrm{Data}(p)\|_1 + \|x\|_1])^{\chi\,\mathrm{Size}^\chi(p)}$ *for all x (here and below χ is an instance-independent constant);*

(iii) [polynomial boundedness of feasible sets] *If x is feasible for an instance p, then* $\|x\|_1 \le (\chi[\mathrm{Size}(p) + \|\mathrm{Data}(p)\|_1])^{\chi\,\mathrm{Size}^\chi(p)}$.

Note that in fact (i) can be weakened to the possibility to compute in polynomial in $\mathrm{Size}(p)$ and $\mathrm{Digits}(p,\epsilon)$ time δ-approximations solely to the values of the objective and the constraints at x.

Polynomial time solvability results like the one stated by Theorem 1.1 are based upon existence of linearly converging *black box oriented* methods for solving general-type convex programs, i.e., methods which work solely with local information on the program – the values and the subgradients of $f_0, f_1, ...f_m$ at successively generated *search points*, with no direct access to program's data. Historically, the first method of this type was the Ellipsoid algorithm proposed independently in [43] and [65] (for detailed study of the algorithm and its theoretical consequences, see [26]), and the corresponding result is as follows.

Theorem 1.2. *Let* (1) *be a convex program with n variables, and let the feasible set $X = \{x : f_i(x) \le 0, i \ge 1\}$ be contained in the ball $B = \{x : \|x\|_2 \le R\}$ and contain a ball of radius $r > 0$, with R, r known. Assume that we have an access to*

- *a first order oracle $\mathcal{O}$ capable to compute the value $f_0(x)$ and a subgradient $f_0'(x)$ at every given point $x \in B$;*
- *a separation oracle $\mathcal{S}$ which, given on input a point $x \in B$, reports whether $x \in X$, and if it is not the case, returns a linear form which separates x and X.*

In this environment, certain explicit algorithm (the Ellipsoid method) finds, for every accuracy $\epsilon > 0$, a feasible ϵ-solution to (1) *at the cost of no more than*

$$N(\epsilon) \le \mathrm{Ceil}\left(2n^2\left[\log(R/r) + \ln\left(1 + \mathrm{Var}_B(f_0)/\epsilon\right)\right]\right) + 1, \quad \mathrm{Var}_B(f_0) = \max_B f_0 - \min_B f_0$$

subsequent calls to $\mathcal{O}$ and $\mathcal{S}$ plus $O(1)n^2$ a.o. per call to process oracle's answer.

In spite of their crucial role in demonstrating polynomial solvability of Convex Programming, black-box-oriented techniques like the Ellipsoid method are too slow to be of actual practical interest; indeed, for all known methods of this type, the computational effort per accuracy digit grows with the design dimension n *at least* as $O(n^4)$, which, in reality, makes it problematic to process already medium-scale (few hundreds of variables) convex programs. This contrast between theoretical properties and practical performance is especially sharp in LP: on the theoretical side, the Ellipsoid method allowed to resolve affirmatively the long-standing problem of whether LP with rational data admits a CCT-polynomial time solution algorithm (Khachiyan [34], 1979), while practical performance of the method in LP is incomparably worse than the one of the "theoretically bad" (with exponential in the size of an LP program worst-case complexity) Simplex method. Comparing extremely powerful in practice Simplex method with its black-box-oriented rivals, it is easy to guess from where the weakness of the rivals comes: the Simplex method fully utilizes the rich structure of an LP program and works directly on program's

data, which is not the case with "nearly blind" black-box-oriented algorithms. Note, however, that while in reality a convex program usually has a lot of structure (otherwise, how could we know that the program is convex?), the standard way to think about nonlinear convex programs, suggested by representation (1), made it extremely difficult to reveal and to utilize this structure. In retrospect, tremendous developments in Convex Programming during what is called "Interior Point Revolution" (started in 1984 when Karmarkar [33] invented his famous practically efficient polynomial time algorithm for LP) were mainly focused on finding and utilizing novel "structure revealing" representations of convex programs, most notably, in the form of *conic programs*.

Remark 1.3. A "classically oriented" mathematician might be surprised by the attention we pay to representation issues. Indeed, finally an optimization problem is to minimize a function over a set; why should we bother about representations of these, completely transparent by themselves, entities? The answer is, that an algorithm cannot work with abstract entities, it can work only with their representations, and different representations of the same entities may be of completely different "algorithmic value".

1.3. Conic programs.

When passing from a Linear Programming program

$$\min_x \left\{c^T x : Ax - b \geq 0\right\} \tag{2}$$

to its nonlinear extensions, the most natural way is the one used in MP – to replace the linear objective $c^T x$ and left hand sides $[Ax-b]_i$ in the constraints with nonlinear ones, thus arriving at (1). As far as Convex Programming is concerned, there is an equivalent, less trivial and, as it turns out, much more convenient way to introduce nonlinearity, namely, replacing the standard coordinate-wise vector inequality

$$a \geq b \Leftrightarrow a - b \geq 0 \Leftrightarrow a - b \in \mathbf{R}^m_+ = \{y \in \mathbf{R}^m : y_i \geq 0,\ i = 1, ..., m\}$$

with another "good" vector inequality given by a subset $K \subset \mathbf{R}^n$ according to

$$a \geq_K b \Leftrightarrow a - b \geq_K 0 \Leftrightarrow a - b \in K.$$

The evident necessary and sufficient condition for the resulting binary relation to be a partial order compatible with linear operations on $\mathbf{R}^m$ is for $K \subset \mathbf{R}^m$ to be a nonempty convex pointed ($K \cap (-K) = \{0\}$) cone. From the analytical perspective, it makes sense also to require from K to be closed with a nonempty interior. Given a *regular* (convex, pointed, closed and with a nonempty interior) cone K, we define a *conic program on* K as the optimization program

$$\min_x \left\{c^T x : \underbrace{Ax - b \geq_K 0}_{\Leftrightarrow Ax-b\in K}\right\} \tag{CP}$$

Preliminary observations about this representation are that (a) in (CP), it is easy to distinguish between the "structure" (given by the cone K) and the data (c, A, b),

and (b) independently of values of the data, (CP) is a problem of optimizing a linear objective over a convex set, and thus is a convex problem (thus, the convexity in (CP) is "built in", while in (1) it should be "added from outside"). At the same time, it is easily seen that every convex program can be represented in the form of (CP). By themselves, (a), (b) do not say much in favour of conic representation of a convex program as compared to its MP form – a general-type convex cone is not a "better structured" entity than a general-type convex function. The crucial advantage of conic representation is that it possesses outstanding "unifying abilities" – as a matter of fact, just 3 families of cones allow to represent an extremely wide spectrum of convex programs. These 3 families are

$\mathcal{LP}$: nonnegative orthants $\mathbf{R}^m_+$ giving rise to LP programs (2),

$\mathcal{CQP}$: finite direct products of Lorentz cones $\mathbf{L}^{k+1} = \{(y,t) \in \mathbf{R}^{k+1} = \mathbf{R}^k \times \mathbf{R} : t \geq \|y\|_2\}$; the MP form (1) of the resulting *conic quadratic* programs is

$$\min_x \left\{ c^T x : \|A_i x - b_i\|_2 \leq c_i^T x - d_i,\ i = 1, ..., m \right\}, \tag{3}$$

where A_i, b_i, c_i, d_i are matrices and vectors of appropriate dimensions. A constraint $\|Ax - b\|_2 \leq c^T x - d$ is called a CQI (Conic Quadratic Inequality);

$\mathcal{SDP}$: direct products of semidefinite cones $\mathbf{S}^m_+$. $\mathbf{S}^m_+$ is the cone of positive semidefinite (psd) matrices in the Euclidean space $\mathbf{S}^m$ of real symmetric $m \times m$ matrices equipped with the Frobenius inner product $\langle A, B \rangle = \mathrm{Tr}(AB)$. The resulting *semidefinite* programs (sdp's) are of the form

$$\min_x \left\{ c^T x : \mathcal{A}_i x - B^i \equiv x_1 A^i_1 + ... + x_n A^i_n - B^i \succeq 0,\ i = 1, ..., m \right\}, \tag{4}$$

where $A^i_j, B^i \in \mathbf{S}^{k_i}$ and $A \succeq B$ means that $A - B$ is psd. A constraint $\sum_i x_i A_i - B \succeq 0$ is called LMI (Linear Matrix Inequality).

It is immediately seen that $\mathcal{LP} \subset \mathcal{CQP} \subset \mathcal{SDP}$; indeed, a linear inequality is a particular case of CQI, which, in turn, is a particular case of LMI due to $(y,t) \in \mathbf{L}^{k+1}$ iff $\left[\begin{array}{cc} t & y^T \\ y & tI_k \end{array} \right] \succeq 0$.

In the sequel, we intend to overview the main elements of the theory of Conic Programming, specifically, (1) duality, (2) interior point polynomial time algorithms, and (3) "expressive abilities" and applications.

2. Conic Duality

Duality in Optimization stems from the desire to find a systematic way to bound from below the optimal value of a minimization problem; it turns out that a good answer to this question is crucial for building optimality conditions, solution algorithms, etc. As applied to MP, the standard *Lagrangian duality* associates with (1) the Lagrange function $L(x, \lambda) = f_0(x) + \sum_{i=1}^m \lambda_i f_i(x)$ and observes that whenever $\lambda \geq 0$, one has $L_*(\lambda) = \inf_x L(x, \lambda) \leq \mathrm{Opt}$; thus, we get a family $L_*(\lambda)$, $\lambda \geq 0$, of (computable under favourable circumstances) lower bounds on Opt and can now associate with (1) the *dual problem* $\max_{\lambda \geq 0} L_*(\lambda)$ of finding the best lower

bound available from the outlined mechanism. *When (1) is convex* and satisfies mild additional assumptions[1)], the dual problem is solvable with the optimal value Opt ("strong duality"). In the case of (CP), essentially the same Lagrange recipe results in the dual problem of the form

$$\max_{\lambda} \left\{ b^T\lambda : A^T\lambda = c, \lambda \geq_{K_*} 0 \right\}, \tag{D}$$

where $K_* = \{\lambda : \lambda^T\xi \geq 0 \, \forall \xi \in K\}$ is the cone dual to K (and thus regular along with K). (D) again is a conic problem, and in fact the duality is fully symmetric. Indeed, assuming the columns of A to be linearly independent (which in fact does not restrict generality), we can pass in (CP) from original variables x to *primal slack* $\xi = Ax - b$, ending up with equivalent *primal* reformulation

$$\min_{\xi} \left\{ d^T\xi : \xi \in [\mathcal{L} - b] \cap K \right\} \quad \left[\mathcal{L} = \mathrm{Im}A, \; d : A^T d = c \right] \tag{Pr}$$

of (CP) as a problem of minimizing a linear objective over the intersection of a cone and an affine plane. Observe that the linear constraints in (D) read $A^T\lambda = c = A^T d$, or, equivalently, $\lambda \in \mathrm{Ker}\, A^T + d = \mathcal{L}^\perp + d$. Thus, (D) is the problem

$$\max_{\lambda} \left\{ b^T\lambda : \lambda \in [\mathcal{L}^\perp + c] \cap K_* \right\} \tag{Dl}$$

of the geometry completely similar to the one of (Pr). Moreover, $(\mathcal{L}^\perp)^\perp = \mathcal{L}$ and $(K_*)_* = K$, so that the problem dual to (Dl) is exactly (Pr) – the duality indeed is symmetric. The "notational difference" between (CP) and (D) comes from the fact that in (CP) we represent a geometric entity (affine subspace $\mathcal{L} - b$) as the image of an affine embedding, while in (D) a similar entity is represented as the solution set of a system of linear equations. The relations between the primal and the dual problem are summarized in the following theorem (see, e.g., [45, 1, 46]):

Theorem 2.1. *Assuming A in* (CP) *is of full column rank, the following is true:*

(i) *The duality is symmetric:* (D) *is a conic problem, and the conic dual to* (D) *is (equivalent to)* (CP)*;*

(ii) [weak duality] $\mathrm{Opt(D)} \leq \mathrm{Opt(CP)}$*;*

(iii) [strong duality] *If one of the programs* (CP), (D) *is bounded and strictly feasible* (i.e., the corresponding affine plane intersects the interior of the associated cone)*, then the other is solvable and* $\mathrm{Opt(CP)} = \mathrm{Opt(D)}$*. If both* (CP), (D) *are strictly feasible, then both programs are solvable and* $\mathrm{Opt(CP)} = \mathrm{Opt(D)}$*;*

(iv) [optimality conditions] *Assume that both* (CP), (D) *are strictly feasible. Then a pair (x, λ) of feasible solutions to the problem is comprised of optimal solutions iff $c^T x = b^T\lambda$ ("zero duality gap"), same as iff $\lambda^T[Ax - b] = 0$ ("complementary slackness").*

It is highly instructive to compare Lagrange duality with its "particular case" – conic duality. The general Lagrange duality is "asymmetric" (in general, $\mathcal{L}_*(\cdot)$

[1)] The standard assumptions are the existence of a feasible solution where all nonlinear constraints are satisfied as strict inequalities and below boundedness of the objective on the feasible set.

"does not remember" the underlying program (1)) and usually results in *implicitly given* dual problem. In contrast to this, Conic duality is "fully algorithmic" – building a dual problem is a simple purely mechanical process – and completely symmetric. As it turns out, these advantages make Conic duality an extremely useful tool in processing, analytical as well as computational, of conic programs.

Finally, conic duality looks completely similar to the standard LP duality, with the only exception that in the LP case strong duality is ensured by mere feasibility and not a strict one. For "good" cones, like those associated with $\mathcal{CQP}$ and $\mathcal{SDP}$, there exist more advanced (still "fully algorithmic", although non-symmetric) version of duality [59] which is free of this shortcoming – whenever the primal optimal value is finite, the dual is solvable with the same optimal value.

While Conic Programming and Conic duality as "well-established" research subjects arose in early 1990's, initially due to the desire to extend Karmarkar's polynomial time LP algorithm to the non-polyhedral case, there are wonderful earlier examples of using what is now called Semidefinite duality (by Lovasz [40] in connection with his famous θ-function, by A. Shapiro [64] in connection with certain statistical problems, and perhaps more...)

In hindsight, Conic Duality Theorem can be traced to at least as early as 1958, namely, to written by L. Hurwicz chapter 4 of [5] (see [5], Corollary IV.3). Unfortunately, this paper, aimed at infinite-dimensional extensions of the optimality conditions in the "usual" (finite-dimensional) Convex Programming, overlooks the symmetric conic duality itself – the latter is, essentially, a finite-dimensional phenomenon. To the best of our knowledge, the remarkable paper in question made no "observable" impact on the development of Mathematical Programming and now is nearly completely forgotten.

3. Interior point polynomial time methods in Conic Programming

Conic programs are convex, and thus the issue of their polynomial time solvability can be resolved via the general results presented in Section 1.2; modulo minor technicalities, in order to ensure polynomial time solvability of a generic conic problem, it suffices for the associated cones to be "computationally tractable" (to admit polynomial time membership/separation oracles), and for the instances – to include upper bounds on the norms of candidate solutions. For example, $\mathcal{LP}/\mathcal{CQP}/\mathcal{SDP}$ problems *with bounds* $\|x\|_\infty \equiv \max_i |x_i| \le R$ *on variables* [2)], are polynomially solvable. The point, however, is that *conic programs associated with "good" cones admit much faster polynomial time algorithms than those coming from black-box-oriented techniques like the Ellipsoid method.* The first "really fast" (and completely non-traditional) polynomial time algorithm for LP was discovered by Karmarkar in 1984 [33]; the subsequent intensive research in the emerging area of *interior point* (IP) *polynomial time algorithms* for LP resulted, among

[2)] These bounds clearly do not affect the possibility to represent a problem as an $\mathcal{LP}/\mathcal{CQP}/\mathcal{SDP}$.

other things, in developing much more transparent (and theoretically even more efficient) than Karmarkar's method polynomial time algorithms for LP (Renegar, 1986 [60]; Gonzaga [24]) and brought the area in a position where non-polyhedral extensions became possible. These extensions, primarily due to Yu. Nesterov, led to a general theory of polynomial time IP methods in Convex Programming [46]. We start with outlining the basic elements of the theory and then will overview briefly the current state of this area.

3.1. Basics on IP methods. The starting point in all IP constructions is a well-known *interior penalty scheme* for solving a convex program

$$\min_{x\in X} c^T x \tag{P}$$

where $X \subset \mathbf{R}^n$, $\operatorname{int} X \neq \emptyset$, is closed and convex. The scheme, going back to Fiacco and McCormic [19], is to equip X with an *interior penalty* – a smooth and strictly convex function $F(x) : \operatorname{int} X \to \mathbf{R}$ which blows up to ∞ along every sequence $x_i \in \operatorname{int} X$ converging to a boundary point of X – and to associate with (P) a parametric optimization problem

$$x_*(t) = \operatorname{argmin}_{x\in\operatorname{int} X} F_t(x), \quad F_t(x) = tc^T x + F(x).$$

Under mild assumptions, the *central path* $x_*(t)$ is well-defined for $t > 0$ and converges to the optimal set of (P) as $t \to \infty$. In the interior penalty scheme, one "traces" this path by generating iterates (x_i, t_i) such that $x_i - x_*(t_i) \to 0$ and $t_i \to \infty$ as $i \to \infty$. Specifically, given (x_i, t_i) with x_i "close" to $x_*(t_i)$, one increases somehow t_i, thus getting t_{i+1}, and then applies a whatever method of unconstrained minimization to the function $t_{i+1}c^T x + F(x)$, starting the method at x_i, to get a "tight" approximation x_{i+1} to the minimizer $x_*(t_{i+1})$ of this function. A standard "working horse" here is the Newton method, commonly believed to be the fastest method for unconstrained minimization.

Self-concordance. The traditional theory of the Newton method (and all other methods of unconstrained minimization, for this matter) did not suggest polynomiality of the outlined path-following scheme, vice versa, it predicted inevitable slowing down of the path-tracing process. It turned out that there exist interior penalty functions – *self-concordant barriers* – which do allow for polynomial time path-tracing, and that the associated *self-concordant-based* theory of IP methods [46] allows to explain all IP methods previously developed for LP and to extend them onto non-polyhedral convex case. Specifically, an interior penalty function F is called a *ϑ-self-concordant barrier* (ϑ-s.c.b.) for X, if F is C^3 and convex on $\operatorname{int} X$ and satisfies, for all $x \in \operatorname{int} X$, $h \in \mathbf{R}^n$, the differential inequalities

$$\begin{array}{lll} (a) & |D^3F(x)[h,h,h]| \le 2\left(D^2F(x)[h,h]\right)^{3/2} & \text{[self-concordance]} \\ (b) & |DF(x)[h]| \le \sqrt{\vartheta}\left(D^2F(x)[h,h]\right)^{1/2} & \text{[barrier quantification]} \end{array} \tag{5}$$

which can be interpreted as follows: the convex function F at a point defines a "local" Euclidean norm $\|h\|_{x,F} = \sqrt{D^2F(x)[h,h]}$ on $\mathbf{R}^n$; self-concordance (5.a)

means that the Hessian of F is Lipschitz continuous, with constant 2, w.r.t. to the corresponding local metric, while (5.b) says that F itself is Lipschitz continuous, with constant $\sqrt{\vartheta}$, in this metric.

Self-concordance-based path-tracing. It turns out that self-concordant functions – those satisfying (5.a) alone – are extremely well suited for Newton minimization, which ensures nice properties of the "centering" $x_i \mapsto x_{i+1}$ in the path-tracing, while (5.b) is responsible for the possibility to increase penalty parameter t at a constant rate, thus avoiding slowing down. The bottom line is as follows: assume that X does not contain lines, the level sets $\{x \in X : c^T x \le a\}$ are bounded, and that F is ϑ-s.c.b. Then $x_*(t)$ is well-defined and $\nabla^2 F(x) \succ 0$ on $\operatorname{int} X$, so that the *Newton decrement* $\lambda(x,t) = \sqrt{\nabla F_t^T(x)[\nabla^2 F_t(x)]^{-1}\nabla F_t(x)}$ *of* F_t *at* $x \in \operatorname{int} X$ is well-defined; note that $x_*(t)$ is characterized by $\lambda(x,t) = 0$. Let us say that x is close to $x_*(t)$, if $\lambda(x,t) \le 0.1$. Given a starting point (x_0, t_0) with $t_0 > 0$ and x_0 close to $x_*(t_0)$, consider the path-tracing scheme

$$\left[\begin{array}{c} t_i \\ x_i \end{array}\right] \mapsto \left[\begin{array}{l} t_{i+1} = (1 + 0.1\vartheta^{-1/2})t_i \\ x_{i+1} = x_i - \frac{1}{1+\lambda(x_i,t_{i+1})}[\nabla^2 F_{t_{i+1}}(x_i)]^{-1}\nabla F_{t_{i+1}}(x_i) \end{array}\right]. \qquad (6)$$

Then the process is well-defined, ensures closeness of x_i and $x_*(t_i)$ and the relation

$$c^T x_i - \min_X c^T x \le 2\vartheta/t_i \le 2\exp\{-0.05i\vartheta^{-1/2}\}\vartheta t_0^{-1}.$$

Thus, the path-tracing scheme (6) with a *single* Newton-like step per updating the penalty converges linearly, and it takes $O(\sqrt{\theta})$ iterations to get an extra accuracy digit. Of course, to run the outlined scheme, we should once come close to the central path; this can be done by applying the same path-tracing technique to an appropriate auxiliary problem. It follows that *if we are smart enough to equip the feasible domains of instances* (written in the form of (P)) *of a generic convex problem* $\mathcal{P}$ *with self-concordant barriers computable, along with their first and second order derivatives, in time polynomial in the size of an instance, and the parameters* ϑ *of these barriers are bounded by a polynomial of instance's size, then the outlined path-tracing scheme provides a polynomial time algorithm for* $\mathcal{P}$.

As about being "smart enough" to equip generic convex problems with "good" s.c.b.'s, the situation is as follows. *In principle*, every closed convex set $X \subset \mathbf{R}^n$, $\operatorname{int} X \ne \emptyset$, admits an $O(1)n$-s.c.b.; assuming w.l.o.g. that X does not contain lines, this *universal barrier* is $F(x) = O(1)\ln \operatorname{mes}_n(P_x)$, where $P_x = \{y : y^T(z-x) \le 1 \forall z \in X\}$ is the polar of X w.r.t. x [46]. In the case when X is a cone, this construction results in the logarithm of the *characteristic function* $F(x) = \int_{K_*} \exp\{-x^T y\}dy$ of the cone K_* [27]. This existence theorem has restricted algorithmic content, since the universal barrier rare is efficiently computable. There exists, however, a simple "calculus" of s.c.b.'s [46] which shows that basic convexity-preserving operations with sets, e.g., taking intersections, direct products, affine images and inverse affine images (as well as taking inverse images under specific nonlinear mappings, most notably a kind of Siegel domain construction) can be equipped with simple rules which allow to combine s.c.b.'s for

the operands into an s.c.b. for the resulting set. E.g., summing up ϑ_i-s.c.b.'s for sets X_i, $i = 1, ..., m$, we get a $(\sum_i \vartheta_i)$-s.c.b. for the intersection of the sets; superposition $F(Ax+b)$ of a ϑ-s.c.b. F with affine mapping is ϑ-s.c.b. for the inverse image of the domain of F under the mapping, etc. These and more advanced "calculus rules" allow to build "from scratch" (from the only observation that the function $-\ln t$ is 1-s.c.b. for $\mathbf{R}_+$) explicit efficiently computable s.c.b.'s with moderate values of the parameter for a wide variety of interesting convex sets. This family includes the cones underlying $\mathcal{LP}$, $\mathcal{CQP}$, $\mathcal{SDP}$, $\|\cdot\|_p$-cones $\{(x,t) : \|x\|_p \leq t\}$, feasible sets of Geometric Programming programs, intersections, direct products, affine and inverse affine images of the above, and much more. The related *empirical* observation is that all generic polynomially solvable convex problems arising in applications admit "good" explicit s.c.b.'s, and that the outlined scheme is the source of the best known so far complexity bounds and polynomial time algorithms for these generic problems.

Aside of their role in constructing polynomial time algorithms, s.c.b.'s seem to be pretty interesting entities by their own right – their properties are closely related to the geometry of their domains. E.g., a closed convex domain X not containing lines is bounded if and only if (any, and then all) s.c.b. F for X attains it minimum on $\operatorname{int} X$, let the (automatically unique) minimizer be $\bar{x}$. When it is the case, the *Dikin ellipsoid* $D_{\bar{x}} = \{x : \|x - \bar{x}\|_{\bar{x},F} \leq 1\}$ of F gives an $O(\vartheta)$-*rounding* of X: $D_{\bar{x}} \subset X \subset \{x : \|x - \bar{x}\|_{\bar{x},F} \leq \vartheta + 2\sqrt{\vartheta}\}$ ([46, 32]; note that $D_{\bar{x}} \subset X$ for all $\bar{x} \in \operatorname{int} X$, not only when $\bar{x}$ is a minimizer of F [46]). This combines with elementary facts of barrier calculus to imply, e.g., the following: *if the intersection of m ellipsoids in $\mathbf{R}^n$ has a nonempty interior, it is in-between two efficiently computable concentric similar ellipsoids with similarity ratio not exceeding $m + 2\sqrt{m}$.* We are not aware of a direct proof of this, useful in some applications, geometric fact.

Path-tracing and Riemannian geometry. An s.c.b. F defines a Riemannian structure on its domain $X^o = \{x : F(x) < \infty\}$, the metric tensor being $\nabla^2 F(x)$. Various *short step* interior point methods associated with F can be described as follows: in order to solve (P), the method generates a sequence of points $x_i \in X^o$ such that (a) the Riemannian distance from x_i to x_{i+1} does not exceed an absolute constant, say, 0.2, and (b) the shift $x_{i+1} - x_i$ is defined solely in the "local" terms – in terms of the objective c and the quantities $F(x_j), \nabla F(x_j), \nabla^2 F(x_j)$, $0 \leq j \leq i$ (cf. (6)). The complexity of such a method is quantified by the number of steps required to reach the set $X^o_\epsilon = \{x \in X^o : c^T x - \inf_{x' \in X^o} c^T x' \leq \epsilon\}$ of ϵ-solutions to (P). Clearly, the complexity of a short step method is below bounded by the Riemannian distance from x_0 to X^o_ϵ. Ideally, we would like to have the complexity within an absolute constant factor of this "ultimate" lower bound; it, however, is unclear how to build such a method – how to describe the shortest path from x_0 to X^o_ϵ in local terms? (cf. climbing a mountain in a fog and without map). It can be proved [55] that the short step path-following method (6) is not that far from being ideal: its performance is within the factor $O(1)\vartheta^{1/4}$ of the lower bound, provided that X^o is bounded and x_0 is close to $\operatorname{argmin} F$.

3.2. Interior point methods in Conic Programming.

Barriers especially well-suited for conic problems are *ϑ-logarithmically homogeneous* s.c.b.'s,

that is, C^3 convex functions $F : \operatorname{int} K \to \mathbf{R}$, K being a regular cone, satisfying (5.a) and the identity $F(tx) = F(x) - \vartheta \ln t$, $t > 0$. This implies that F is a ϑ-s.c.b. for K and that the *conjugate barrier* $F_*(y) = \max_x \left[-y^T x - F(x)\right]$ is a ϑ-logarithmically homogeneous s.c.b. for K_*; the mappings $x \mapsto -\nabla F(x)$, $y \mapsto -\nabla F_*(y)$ turn out to be inverse to each other one-to-one correspondences between $\operatorname{int} K$ and $\operatorname{int} K_*$. Given a pair of strictly feasible primal-dual conic problems (Pr), (Dl) and a pair of conjugate to each other ϑ-logarithmically homogeneous s.c.b.'s F, F_* for the cones K, K_*, one can develop *primal-dual* interior point methods simultaneously solving (Pr), (Dl). The most popular *primal-dual path-following scheme* is as follows. When both (Pr) and (Dl) are strictly feasible, the corresponding primal and dual central paths $\xi_*(t) = \operatorname{argmin}\left[td^T\xi + F(\xi) : \xi \in \mathcal{L} - b\right]$ and $\lambda_*(t) = \operatorname{argmin}\left[-tb^T\lambda + F_*(\lambda)\right]$ are well-defined and linked to each other: $\lambda_*(t) = -t^{-1}\nabla F(\xi_*(t))$, $\xi_*(t) = -t^{-1}\nabla F_*(\lambda_*(t))$, and one can apply Newton-based path-tracing to the *primal-dual* central path $(\xi_*(t), \lambda_*(t))$, thus solving (Pr) and (Dl) simultaneously. It turns out that processing both problem together has a lot of advantages, allowing, e.g., for

- on-line adjustable "long step" policies [47] which are less conservative than the worst-case-oriented policy (6) and thus exhibit much better practical performance, while still ensuring the theoretical complexity bounds,
- an elegant way ("self-dual embedding", see, e.g., [72, 77, 4, 42, 35, 58]) to initialize the path-tracing even in the case when no feasible solutions to (Pr) and (Dl) are available in advance,
- building certificates of strict (i.e., preserved by small perturbations of the data) primal or dual infeasibility [51] when it is the case, etc.

It should be added that the primal-dual central path $(\xi_*(\cdot), \lambda_*(\cdot))$ is "nearly geodesic" w.r.t. the Riemannian structure on the primal-dual feasible set given by the metric tensor $\nabla^2(F(\xi) + F_*(\lambda))$: the Riemannian length of every segment of the path is within factor $\sqrt{2}$ of the Riemannian distance between the endpoints of the segment [54].

3.3. The case of symmetric cones: LP/CQP/SDP. Interior point constructions achieve maximal flexibility for cones with a lot of symmetries, most notably for *symmetric cones* – those which are homogeneous (i.e., the group of linear authomorphisms of the cone acts transitively on its interior) and self-dual w.r.t. an appropriate Euclidean structure on the embedding space. Classification theory due to Vinberg [71] says that all symmetric cones are direct products of irreducible ones, specifically, Lorentz cones, real semidefinite cones (in particular, nonnegative rays), the cones of Hermitian psd complex matrices, the cones of Hermitian psd quaternion matrices, and, finally, copies of exceptional 27-dimensional octonian cone. The latter 3 cones are cross-sections of the semidefinite cone of "the same" (within factor 4) real dimension and therefore do not add much, as far as the associated families of conic problems are concerned; therefore we lose nearly nothing when focusing on the cones which are direct products of Lorentz and semidefinite cones, thus arriving at problems of minimizing linear objective under a mixture of conic quadratic and LMI constraints (the latter include also linear constraints which are merely 1-dimensional LMI's). Now, we can equip a direct product

$K = K_1 \times ... \times K_m$ of Lorentz and semidefinite cones with the *canonical* logarithmically homogeneous s.c.b. $F(x^1, ..., x^m) = \sum_i F_i(x^i)$, the barriers for the Lorentz factors $K_i = \{x^i = (u_i, s_i) : s_i \geq \|u_i\|_2\}$ being $F_i(x^i) = -\ln(s_i^2 - u_i^T u_i)$ ($\vartheta = 2$) and the barriers for the semidefinite factors $K_i = \mathbf{S}_+^{m_i}$ being $F_i(x^i) = -\ln\det x^i$ ($\vartheta = m_i$). The parameter of the resulting barrier is the sum of those of the components, i.e., it is twice the number of Lorentz factors plus the total row size of the semidefinite ones. The canonical barrier F respects the symmetries $x \mapsto Ax$ of the underlying cone ($F(Ax) = F(x) + \text{const}(A)$) and its self-duality ($F_*(x) = F(x) + \text{const}$) and possesses a number of advanced properties which can be utilized in the interior point algorithms, e.g., in developing long-step policies. Discovery of these properties and the ways to utilize them in the IP context by Nesterov and Todd [48, 49] was one of the most important breakthroughs in developing of IP theory and algorithms.

It should be noted that the theory of IP methods on symmetric cones is one of few (alas!) "avenues of contact" of Convex Optimization and modern Mathematics, specifically, the theory of Euclidean Jordan algebras; the latter turned out to be a natural way to treat the IP methods on symmetric cones, see [62, 63, 17, 18, 29] and references therein. Another such avenue is the theory of hyperbolic polynomials originating from PDEs. Recall that a real homogeneous, of a degree m, polynomial $p(\cdot)$ on $\mathbf{R}^n$ is called *hyperbolic* in a direction d, $p(d) > 0$, if the univariate polynomial $\phi(t) = p(x + td)$ has all its roots real whenever $x \in \mathbf{R}^n$. It is well known that the component of d in the set $p(\cdot) > 0$ is the interior of a closed convex cone K – the *hyperbolicity cone* of p. As discovered in [28], $-\ln\det p(x)$ is an m-logarithmically homogeneous s.c.b. for K with a number of useful properties mimicking those of canonical barriers (the latter are built from logs of specific hyperbolic polynomials $\det(x) : \mathbf{S}^m \to \mathbf{R}$ and $x_{k+1}^2 - \sum_{i=1}^k x_i^2 : \mathbf{R}^{k+1} \to \mathbf{R}$). For further links between convex optimization and hyperbolic polynomials, see [6, 31] and references therein.

We conclude this section with complexity bounds for generic LP/CQP/SDP problems with bounds on variables. For all these problems, the complexity of building ϵ-solution, measured both in the number of IP iterations and in the total number of a.o., is proportional to the required number of accuracy digits $\text{Digits}(\cdot, \epsilon)$, so that we can speak about "number of iterations/a.o. per accuracy digit"; these are the complexity characteristics to be indicated below.

LP: for an LP program $\min_{x \in \mathbf{R}^n} \left\{c^T x : Ax \geq b \in \mathbf{R}^m, \|x\|_\infty \leq R\right\}$, the size is $O(mn)$, and the complexity is $O(1)\sqrt{m+n}$ IP iterations and $O(1)(m+n)^{3/2}n^2$ a.o. per accuracy digit. With smart implementation of subsequent Newton steps ("Karmarkar acceleration"), the # of a.o. per accuracy digit can be reduced to $O(1)[(m+n)n^2 + m^{1.5}n]$ (see [60]); thus, to find an ϵ-solution to an LP with $m \leq O(n^2)$ is, up to factor $\ln(1/\epsilon)$, not more difficult than to find the least squares solution to a system of $m+n$ linear equations with n unknowns;

CQP: for a CQP $\min_{x \in \mathbf{R}^n} \left\{c^T x : \|A_i x - b_i\|_2 \leq c_i^T x - d_i,\ i \leq m,\ \|x\|_2 \leq R\right\}$ with $k_i \times n$ matrices A_i, the size is $O(n\sum_i k_i)$, and the complexity is $O(1)\sqrt{m}$ IP iterations and $O(1)m^{1/2}n\left(mn + n^2 + \sum_i k_i\right)$ a.o. per accuracy digit (provided the matrices $A_i^T A_i$ are computed in advance).

SDP: for an sdp $\min_{x\in\mathbf{R}^n}\left\{c^Tx:\sum_j x_jA_j^i-B^i\succeq 0,\ i\le m,\|x\|_2\le R\right\}$ with $k_i\times k_i$ matrices A_j^i, the size is $O(n\sum_i k_i^2)$, and the complexity is $O(1)\sqrt{\sum_i k_i}$ IP iterations and $O(1)\sqrt{\sum_i k_i}n(n^2+n\sum_i k_i+\sum_i k_i^3)$ a.o. per accuracy digit.

Empirical behaviour of well-implemented IP methods is better than the one predicted by the worst-case theoretical analysis. Theoretically, iteration count in LP and SDP should be $O(\sqrt{m})$, where m is the number of linear constraints in LP and is the total row size of LMIs in SDP. In reality, no essential growth of the iteration count with m is observed, and a high accuracy solutions are found in something like 30-50 iterations. What limits the practical scope of IP methods is the complexity of a single iteration where a Newton-type system of n linear equations with n unknowns is formed and solved, n being the design dimension of the program. With $n\sim 10^4$ and more, solving Newton system in reasonable time is possible only when it is highly sparse; the latter is usually the case with real-world LP's, but typically is not the case with sdp's.

4. Expressive abilities and applications of LP/CQP/SDP

As we have already mentioned, the IP machinery is especially well suited for solving conic problems on symmetric cones, which makes it natural to ask: when a convex program can be reformulated as a conic program on such a cone, specifically, as an LP/CQP/SDP program? Usually, the original formulation of a convex program is in (or can be immediately converted to) the form

$$\min_{x\in X} c^Tx,\quad X=\bigcap_{i=1}^m X_i, \tag{7}$$

where X_i are convex sets, most typically given as the level sets of convex functions: $X_i=\{x:f_i(x)\le 0\}$. What we need are tools to recognize that (7) can be reformulated as, say, an sdp, and to build such a reformulation when possible, and we start with an overview of the corresponding "toolkit".

4.1. Calculus of LP/CQP/SDP-representable sets and functions.

Let $\mathcal{K}$ be a family of regular cones closed w.r.t. passing from a cone to its dual and closed w.r.t. taking direct products; note that these properties are shared by the families $\mathcal{LP}/\mathcal{CQP}/\mathcal{SDP}$. Let us ask ourselves when program (7) can be reformulated as a $\mathcal{K}$-*program* – a conic program on a cone from $\mathcal{K}$. A natural (and somehow tautological) answer is: *it suffices for X to be a $\mathcal{K}$-representable set* ("$\mathcal{K}$-s." for short), meaning that there exists a $\mathcal{K}$-*representation* ("$\mathcal{K}$-r.") of X:

$$X=\{x\in\mathbf{R}^n:\exists u\in\mathbf{R}^m:A(x,u)\succeq_K 0\}, \tag{8}$$

where $K\in\mathcal{K}$ and $A(\cdot)$ is an affine mapping; in other words, X is the projection of the inverse image of K under appropriate affine mapping. Indeed, given representation (8) of X, we can pose (7) as the $\mathcal{K}$-program $\min_{x,u}\left\{c^Tx:A(x,u)\succeq_K 0\right\}$.

It turns out that $\mathcal{K}$-sets admit a kind of simple calculus comprised of "raw materials" (list of "basic" $\mathcal{K}$-s.'s) and "calculus rules" (list of operations preserving $\mathcal{K}$-representability). This calculus is the "toolkit" we are looking for: whenever we see that the set X in question is obtained from "raw materials" via calculus rules, we can be sure that X is a $\mathcal{K}$-s. (and in fact, as we shall see, can point out an explicit $\mathcal{K}$-r. of X, thus converting (7) to an explicit $\mathcal{K}$-program).

Since a convex set often arise as a level set of a convex function, it makes sense to define a *$\mathcal{K}$-representable function* ("$\mathcal{K}$-f." for short) $f : \mathbf{R}^n \to \mathbf{R} \cup \{+\infty\}$ as a function with $\mathcal{K}$-representable epigraph $\mathrm{Epi}\{f\}$. An immediate observation is that *a $\mathcal{K}$-r. of* $\mathrm{Epi}\{f\}$ *induces $\mathcal{K}$-r.'s of all level sets of f.* Indeed,

$$\big(f(x) \le t \Leftrightarrow \exists u : A(x,t,u) \succeq_K 0\big) \Rightarrow \big(f(x) \le a \Leftrightarrow B(x,u) \equiv A(x,a,u) \succeq_K 0\big).$$

4.1.1. "Calculus rules". It turns out (see, e.g., [8]) that all basic convexity-preserving operations with functions/sets preserve $\mathcal{K}$-representability. E.g.,

- A polyhedral set is $\mathcal{K}$-s.
- Finite intersections, arithmetic sums and direct products of $\mathcal{K}$-s.'s are $\mathcal{K}$-s.'s.

While the above statement looks as an existence theorem, it in fact is "fully algorithmic": given $\mathcal{K}$-r.'s for the operands X_i, we can efficiently build a $\mathcal{K}$-r. for the resulting set. E.g., if $X_i = \{x : \exists u^i : A_i(x,u^i) \succeq_{K_i} 0\}$, $i = 1,...,m$, then $\bigcap_{i=1}^m X_i = \{x : \exists u = (u^1,...,u^m) : A(x,u) \equiv (A_1(x,u^1),...,A_m(x,u^m)) \succeq_{K_1\times...\times K_m} 0\}$, and the cone $K = K_1 \times ... \times K_m$ is in $\mathcal{K}$, since this family is closed w.r.t. taking direct products. It should be stressed that *all* calculus rules to follow are equally simple and algorithmic.

- The image and the inverse image of a $\mathcal{K}$-s. under an *affine* mapping is a $\mathcal{K}$-s.
- The *polar cone* $X_* = \{(y,t) : y^T x \le t \forall x \in X\}$ of a $\mathcal{K}$-s. X given by *strictly feasible* $\mathcal{K}$-r. is a $\mathcal{K}$-s. In other words, the support function of X (its epigraph is exactly X_*) is $\mathcal{K}$-f., and thus the polar of $X \ni 0$ (which is the 1-level set of the support function) is $\mathcal{K}$-s.

As an instructive example, let us build a $\mathcal{K}$-r. of X_*. If $X = \{x : \exists u : Ax + Bu + c \succeq_K 0\}$ is a strictly feasible $\mathcal{K}$-r. for X, then $X_* = \{(y,t) : \sup_{x,u}\{y^T x : Ax + Bu + c \succeq_K 0\} \le t\} = \{(y,t) : \min_v \{c^T v : A^T v = -y, B^T v = 0, v \succeq_{K_*} 0\} \le t\} = \{(y,t) : \exists v : A^T v = -y, B^T v = 0, v \succeq_{K_*}, c^T v \le t\}$, with the second "=" in the chain given by conic duality. We see that X_* is the projection onto the (y,t)-space of the set $Y = \{(y,t,v) : A^T v = -y, B^T v = 0, v \succeq_{K_*}, c^T v \le t\}$. Y is a $\mathcal{K}$-s. Indeed, K_* is $\mathcal{K}$-s. since $\mathcal{K}$ is closed w.r.t. passing to dual cones, and Y is the direct product of K_* and a polyhedral set (the space of (y,t)) intersected with another polyhedral set; all these operations preserve $\mathcal{K}$-representability. The same is true for projection, thus X_* is $\mathcal{K}$-s. along with Y.

- Two useful operations with sets – taking closed conic hull $\mathrm{cl}\{(t,x) : t > 0, t^{-1}x \in X\}$ of X and taking the closed convex hull of the union of finitely many convex sets X_i – "nearly preserve" $\mathcal{K}$-representability, meaning that $\mathcal{K}$-r.'s of the operands readily provide a $\mathcal{K}$-r. of a convex set $\widehat{Y}$ which is in-between the "true" result Y of the operation and the closure of Y (in particular, we end up with $\mathcal{K}$-r. of *exactly* Y when Y is closed). As far as the possibility of conic reformulation of a program $\min_{y\in Y} c^T y$ is concerned, a $\mathcal{K}$-r. of a convex set $\widehat{Y}$ which is in-between the true set Y and its closure is, essentially, as good as a $\mathcal{K}$-r. of Y itself.
- Sometimes, getting a $\mathcal{K}$-r. of a result of an operation requires mild regularity

assumptions on the $\mathcal{K}$-r.'s of operands. E.g., let $\{x : \exists u : Ax + Bu + c \geq_K 0\}$ be a $\mathcal{K}$-r. of a convex set X such that $Bu \in K$ implies that $u = 0$. Then the closed conic hull of X and the recessive cone of X are $\mathcal{K}$-s.'s with representations, respectively, $\{x : \exists(u, t \geq 0) : Ax + Bu + tc \geq_K 0\}$ and $\{x : \exists u : Ax + Bu \geq_K 0\}$.

"Functional analogies" of the outlined calculus rules are as follows:

• The maximum, a linear combination with nonnegative coefficients, and a direct sum of finitely many $\mathcal{K}$-f.'s is again a $\mathcal{K}$-f.

• If $f(\cdot)$ is a $\mathcal{K}$-f., then so is $g(y) = f(Ay + b)$.

• If $f(\xi, \eta)$ is a $\mathcal{K}$-f. and the infimum $\phi(\xi) = \inf_\eta f(\xi, \eta)$ is achieved for every ξ for which this infimum is $< +\infty$, then ϕ is a $\mathcal{K}$-f.

• The Legendre transformation ("the conjugate") $f_*(\xi) = \sup_x[\xi^T x - f(x)]$ of a function f with $\mathrm{Epi}\{f\}$ given by a strictly feasible $\mathcal{K}$-r. is a $\mathcal{K}$-f.

• Let f_i, $i = 1, ..., m$, be $\mathcal{K}$-f.'s on $\mathbf{R}^m$ and g be a nondecreasing, w.r.t. the usual partial order $\leq$, $\mathcal{K}$-f. on $\mathbf{R}^m$. Then the superposition $g(f_1(x), ..., f_m(x))$ is again a $\mathcal{K}$-f.

A good news expressed by the above facts is that *with $\mathcal{K}$-r.'s of the involved entities in use, all basic constructions of Convex Analysis* (including "advanced ones", like taking polar, support function, Legendre transformation, etc.) *become "explicitly representable" and thus much better suited for analytical/algorithmic processing than in their original abstract form.*

Equipped with "calculus" of $\mathcal{K}$-representable sets and functions, we are in a position to investigate the "expressive abilities" of LP/CQP/SDP. The situation with LP seems to be clear: $\mathcal{LP}$-s.'s are exactly the polyhedral sets, and $\mathcal{LP}$-f.'s are finite maxima of affine functions. To understand what can be expressed via CQP and SDP, we need to know what are the corresponding "raw materials". These are the issues we are about to address in the next two sections.

4.2. Expressive abilities and applications of CQP. Basic $\mathcal{CQP}$-representable sets and functions include, along with trivial examples, like affine function or the Euclidean norm $f(x) = \|x\|_2$, several less trivial examples, e.g.:

• Convex quadratic forms $f(x) = x^T A^T Ax + b^T x + c$: $\{t \geq f(x) \Leftrightarrow (2(Ax)^T, t - b^T x - c + 1, t - b^T x - c + 1)^T \geq_{\mathbf{L}} 0\}$;

• Univariate power functions $(\max[0, x])^p$, $|x|^p$ and p-norms $\|\cdot\|_p$ on $\mathbf{R}^n$, provided $p \geq 1$ is rational,

• Power monomials $(-\prod_{i=1}^m x_i^{p_i})$ with $x_i \geq 0$ and rational exponentials $p_i \geq 0$ such that $\sum_i p_i \leq 1$ (the latter is necessary and sufficient for the convexity of the monomial), same as monomials $\prod_{i=1}^m x_i^{-p_i}$ with $x_i > 0$ and rational $p_i \geq 0$.

In view of calculus rules, already these "raw materials" allow for CQP reformulations of a wide variety of convex programs, including (but by far not restricted to) convex quadratic quadratically constrained programs.

Example: Truss topology design. A nontrivial example of a $\mathcal{CQP}$-f. is given by $\mathrm{Compl}(t) = \min\{\tau : \begin{bmatrix} \tau & f^T \\ f & A^T \mathrm{Diag}\{t\} A \end{bmatrix} \succeq 0\}$ of *nonnegative* vector variable t. This function is associated with an important application of CQP – *truss topology design*, see [73, Chapter 15] and references therein. In the TTD problem, one is looking for a construction comprised of elastic bars (like railway bridge, electric mast, or Eiffel Tower) most rigid w.r.t. a given set of (non-simultaneous) loading scenarios. The data are given

by a finite 2D or 3D mesh of nodes, where the would-be bars can be linked to each other, boundary conditions restricting virtual displacements of the nodes to given linear subspaces in the embedding physical space, k *loads* – collections of external forces acting at the nodes, and the total weight w of the construction. A design is specified by a collection $t \in \mathbf{R}^n$ of weights of the bars, and its rigidity w.r.t. a load is measured by the *compliance* – the energy capacitated by the construction in the static equilibrium under the load (the less is the compliance, the better). With properly defined matrix A (readily given by the nodal mesh and the boundary conditions) and vector f representing the load, the compliance is exactly Compl$(\cdot)$, so that the TTD problem, in its simplest form, reads

$$\min_{t,\tau}\left\{\tau : \left[\begin{array}{cc} 2\tau & f_i^T \\ f_i & A^T \mathrm{Diag}\{t\}A \end{array}\right] \succeq 0, i = 1, ..., k, t \geq 0, \sum_j t_j \leq w\right\}. \tag{9}$$

The applied importance of this problem stems from the fact that it allows to optimize not only the sizing, but also the topology of truss. To this end one starts with a fine nodal mesh where all pairs of nodes can be linked by bars; in the optimal solution just few of the bars get positive weights, and the solution recovers the (nearly) optimal topology.

As it arises, (9) is an sdp. However, passing to the SDP dual of (9) and more or less straightforwardly processing it, one concludes that the dual is *equivalent* to a CQP program, whence, again applying duality, one gets an equivalent CQP reformulation of (9) and recovers a $\mathcal{CQP}$-r. of the compliance:

$$\tau \geq \mathrm{Compl}(t) \Leftrightarrow \exists(q, r) : A^T q = f, r \geq 0, \sum_i r_i \leq 2\tau, (2q_i, r_i - t_i, r_i + t_i)^T \in \mathbf{L}^3 \, \forall i. \tag{10}$$

This example is very instructive. After the $\mathcal{CQP}$-r. (10) of compliance is guessed, its validity can be easily proved directly. The power of Conic Programming machinery is that there is no necessity to guess (and to the best of our knowledge, (10) never was guessed, in spite of its a posteriori transparent mechanical interpretation) – it can be *computed* in a purely mechanical way. Besides this, the CQP equivalent of the dual to (9) – which again is given by a mechanical computation – is of incomparably smaller design dimension than the original problem. Indeed, to capture the topology design, the mesh cardinality N already in the 2D case should be of order of thousands; when all pair connections are allowed, this results in the design dimension n of (9) of about $N^2/2$, i.e., in the range of millions, which is too much for actual computations. In contrast, the design dimension of the CQP reformulation of the dual to (9) is of order of $kN \ll N^2$, and this is how the TTD problem is actually solved. This example, among many others, shows that Conic Programming is not just a good framework for number crunching; it is a good framework for instructive analytical processing of convex programs.

Example: Robust Linear Programming. In reality, the data c, A, b in an LP program $\min_x \left\{c^T x : Ax \geq b\right\}$ usually are *uncertain* – not known exactly when the program is solved. It turns out that even pretty small from practical viewpoint perturbations of the data, like 0.01%, can make the nominal optimal solution (one corresponding to the nominal data) heavily infeasible and thus practically meaningless. One way to "immunize" solutions against data uncertainty is to assume that the data (c, A, b) are "uncertain-but-bounded", i.e., belong to a given in advance *uncertainty set* $\mathcal{U}$, and to require from candidate solutions to be *robust feasible*, i.e., to satisfy the constraints whatever be a realization of the data from $\mathcal{U}$. Treating in the same worst-case-oriented fashion the objective, one associates with uncertain LP its *Robust Counterpart* (RC) – the problem

$\min_{x,t}\left\{t : c^T x \le t, Ax - b \ge 0 \,\forall (c, A, b) \in \mathcal{U}\right\}$ of minimizing the worst-case value of the objective over robust feasible solutions. While improving significantly "reliability" of resulting decisions in the face of data uncertainty, the RC, as an optimization program, has a drawback: when $\mathcal{U}$ is infinite (which is typical), the RC is a *semi-infinite* (that is, with infinitely many linear constraints) program; programs of this type not necessarily are computationally tractable. Fortunately, in this respect uncertain LP (in contrast to uncertain CQP/SDP) is simple – the RC of an uncertain LP problem is computationally tractable, provided that the (convex) uncertainty set is so. For example, if $\mathcal{K}$ is a family of regular cones closed w.r.t. taking direct product and passing from a cone to its dual, and $\mathcal{U}$ is given by a strictly feasible $\mathcal{K}$-r., the RC can be straightforwardly reformulated as an explicit $\mathcal{K}$-program (an immediate corollary of $\mathcal{K}$-representability of the polar cone of a $\mathcal{K}$-s., see Section 4.1). Now, typical uncertainty sets in uncertain LP are $\mathcal{CQP}$-representable, most notably – intersections of boxes (coming from upper and lower bounds on uncertain coefficients) and ellipsoids (which allow to model reasonably well random uncertainty). As a result, RC's of uncertain LP programs are CQP's.

For other applications of CQP, see [39, 2, 14].

The above suggests that "expressive abilities" of CQP are much stronger, and applications are much wider than those of LP. Surprisingly, the "gap" here is smaller than one could think – *conic quadratic programs with bounds on variables are polynomially reducible to linear programs*. The reduction is given by *fast polyhedral approximation* of Lorentz cones [9]. Specifically, given m-dimensional Lorentz cone $\mathbf{L}^{m+1} = \{(x,t) \in \mathbf{R}^m \times \mathbf{R} : t \ge \|x\|_2\}$ and $\epsilon \in (0, 1/2)$, one can point out an explicit system of $O(m\ln(1/\epsilon))$ linear inequalities $Px + tp + Qu \ge 0$ in original variables x, t and $O(m\ln(1/\epsilon))$ additional variables u such that the projection $\mathbf{P}^m$ of the cone $\{(x,t,u) : Px + tp + Qu \ge 0\}$ onto the space of x,t-variables satisfies $\mathbf{L}^m \subset \mathbf{P}^m \subset \{(x,t) : \|x\|_2 \le (1+\epsilon)t\}$. Exaggerating, we could say that CQP does not exist as an independent entity. It is interesting whether the same is true for SDP; to the best of our knowledge, this question is completely open.

4.3. Expressive abilities and applications of SDP.

4.3.1. Basic $\mathcal{SDP}$-representable sets and functions. As it was already mentioned, the Lorentz cone is a cross-section of the semidefinite one, so that all $\mathcal{CQP}$-representable functions and sets are $\mathcal{SDP}$-representable as well. In fact the expressive abilities of SDP are much wider than those (already pretty rich) of CQP. The essentially new functions/sets we can handle are as follows [8, Section 4.2]:

• *Functions of eigenvalues of symmetric matrices.* For $X \in \mathbf{S}^n$, let $\lambda(X) = (\lambda_1(X), ..., \lambda_n(X))^T$ be the vector of eigenvalues of X taken with their multiplicities in the non-ascending order. We start with observation (see, e.g., [46]) that *the sum of k largest eigenvalues of a symmetric matrix X is an $\mathcal{SDP}$-f.*:

$$t \ge \lambda_1(X) + ... + \lambda_k(X) \Leftrightarrow \exists (Z, s) : 0 \preceq Z, X \preceq Z + sI, t \ge \mathrm{Tr}(Z) + ks.$$

As a result, *whenever $f(\cdot) : \mathbf{R}^n \to \mathbf{R} \cup \{+\infty\}$ is a symmetric w.r.t. permutations of arguments $\mathcal{SDP}$-f., the function $f(\lambda(X)) : \mathbf{S}^n \to \mathbf{R} \cup \{+\infty\}$ is an $\mathcal{SDP}$-f. with $\mathcal{SDP}$-r. readily given by an $\mathcal{SDP}$-r. of f*. In particular, the following functions

on $\mathbf{S}^n$ admit explicit $\mathcal{SDP}$-r.'s: (a) $\sum_{i=1}^k \lambda_i(X)$; (b) $\|X\| = \max_i |\lambda_i(X)|$; (c) $-\det^q(X)$, $X \succeq 0$, $q \leq \frac{1}{n}$ is rational; (d) $\det^{-q}(X)$, $X \succ 0$, $q > 0$ is rational; (e) $\|X\|_p = \left(\sum_{i=1}^m |\lambda_i(X)|^p\right)^{1/p}$, $p \geq 1$ is rational; (f) $\left(\sum_{i=1}^m \max^p[\lambda_i(X), 0]\right)^{1/p}$, $p \geq 1$ is rational.

• *Functions of singular values.* Singular values $\sigma_1(X) \geq \sigma_2(X) \geq ... \geq \sigma_m(X)$ of a rectangular $m \times n$, $m \leq n$, matrix X are closely related to the eigenvalues of the linearly depending on X $(n+m)\times(n+m)$ symmetric matrix $\widehat{X} = \left[\begin{array}{cc} & X \\ X^T & \end{array}\right]$: eigenvalues of $\widehat{X}$ are $\pm\sigma_i(X)$, $i = 1, ..., m$, and $n - m$ zeros. Therefore $\mathcal{SDP}$-r.'s of functions of eigenvalues of symmetric matrices admit "singular value counterparts". E.g., *if $f : \mathbf{R}^m_+ \to \mathbf{R} \cup \{+\infty\}$ is a nondecreasing symmetric w.r.t. permutations of arguments $\mathcal{SDP}$-f., then the function $f(\sigma(X)) : \mathbf{R}^{m\times n} \to \mathbf{R} \cup \{+\infty\}$ is an $\mathcal{SDP}$-f. with $\mathcal{SDP}$-r. readily given by one of f.* In particular, the following functions on $\mathbf{R}^{m\times n}$ admit explicit $\mathcal{SDP}$-r.'s: (a) $\sum_{i=1}^k \sigma_i(X)$; (b) $\|X\| = \max_i \sigma_i(X)$; (c) $\|X\|_p = \left(\sum_{i=1}^m \sigma_i^p(X)\right)^{1/p}$, $p \geq 1$ is rational.

• *Sets of the form $\{X \in \mathbf{S}^n : \lambda(X) \in A\}$, where A is a symmetric w.r.t. permutations of coordinates $\mathcal{SDP}$-s. in $\mathbf{R}^n$*, and their "singular value" analogies $\{X \in \mathbf{R}^{m\times n} : \sigma(X) \in A\}$ with symmetric and monotone ($0 \leq x' \leq x \in A \Rightarrow x' \in A$) $\mathcal{SDP}$-s. A.

• *The set $\{(A, a, \alpha) \in \mathbf{S}^n \times \mathbf{R}^n \times \mathbf{R} : x^T Ax + 2a^T + \alpha \geq 0 \forall (x : x^T Bx + 2b^T x + \beta \geq 0)\}$ of quadratic forms nonnegative on the level set of a given quadratic form which is positive somewhere: $\exists \bar{x}; \bar{x}^T B\bar{x} + 2b^T\bar{x} + \beta > 0$.* By the famous $\mathcal{S}$-Lemma, an $\mathcal{SDP}$-r. of the set in question is $\{(A, a, \alpha) : \exists \lambda \geq 0 : \left[\begin{array}{cc} \alpha & a^T \\ a & A \end{array}\right] \succeq \lambda \left[\begin{array}{cc} \beta & b^T \\ b & B \end{array}\right]\}$.

• *Sets $\mathcal{P}_n(\mathbf{R})$ of (vectors of coefficients of) real algebraic polynomials of degree $\leq n$ which are nonnegative on the entire axis*, same as sets of algebraic polynomials of degree $\leq n$ nonnegative on a given segment or ray, and sets of trigonometric polynomials of degree $\leq n$ nonnegative on a given segment. The same is true for the cones of psd on a given segment matrix-valued univariate algebraic/trigonometric polynomials of degree $\leq n$.

$\mathcal{SDP}$-r. of $\mathcal{P}_n(\mathbf{R})$ is readily given by the following observation [53]: *if $\phi_i(z)$, $1 \leq i \leq m$ are functions on a given set Z, L is the linear space in $\mathbf{R}^Z$ spanned by the functions $\phi_i(\cdot)\phi_j(\cdot)$ and $K \subset L$ is the cone of sums-of-squares (i.e., functions which are sums of squares of linear combinations of ϕ_i), then K is an $\mathcal{SDP}$-s., specifically, the image of $\mathbf{S}^m_+$ under the linear mapping $[X_{ij}]_{i,j=1}^m \mapsto \sum_{i,j} X_{ij}\phi_i(z)\phi_j(z) : \mathbf{S}^m \to L$.* This simple observation underlies recent techniques for testing nonnegativity of a multivariate polynomial on a given domain, see [22, 57, 38] and references therein.

• Some sets given by nonlinear matrix inequalities can be represented by LMI's as well. E.g., $\mathrm{cl}\{X, Y, Z : Y \succ 0, X^T Y^{-1} X \preceq Z\} = \{X, Y, Z : \left[\begin{array}{cc} Z & X^T \\ X & Y \end{array}\right] \succeq 0\}$, and $\mathrm{cl}\{(X, Y) : X \succ 0, Y \preceq (C^T X^{-1} C)^{-1}\} = \{(X, Y) : \exists Z : Y \preceq Z, Z \succeq 0, X \geq CZC^T\}$, provided C is of full column rank.

A seemingly interesting question is to *characterize* $\mathcal{SDP}$-representable sets. Clearly, such a set is convex and semi-algebraic. Is the inverse also true? This question can be relaxed, e.g., to whether an epigraph of convex multivariate polynomial is an $\mathcal{SDP}$-s. (this

indeed is true in the univariate case), or: whether the hyperbolicity cone of a hyperbolic polynomial is an $\mathcal{SDP}$-s. (due to recently proved Lax conjecture [41], this indeed is true for polynomials of 3 variables), etc. This question seems to be completely open.

4.3.2. Applications of SDP. Due to its tremendous expressive abilities and powerful computational tools, SDP has an extremely wide spectrum of applications, including those in Combinatorics (SDP relaxations of difficult problems), Engineering (Robust Control, design of mechanical structures, electrical circuits and arrays of antennae, communications), Signal Processing, design of statistical experiments, etc. Over the last decade, the spectrum of applications of SDP has been constantly growing, and we believe this tendency is to continue in the foreseen future. We are about to overview an instructive sample of SDP applications; for more examples, see [13, 69, 16, 14, 73, Part III].

SDP relaxations of difficult combinatorial problems. The simplest way to derive SDP relaxations of combinatorial problems goes back to N. Shor [66, 67] and is as follows: consider a quadratic quadratically constrained program

$$\mathrm{Opt} = \min_x \{f_0(x) : f_i(x) \le 0, i = 1, ..., m\}, \; f_i(x) = x^T A_i x + 2b_i^T x + c_i, i \ge 0; \quad (11)$$

note that quadratic constraints can easily express combinatorial restrictions (like $x_i^2 - x_i = 0 \Leftrightarrow x_i \in \{0, 1\}$), and let us try to bound the optimal value from below (this is important, e.g., for various branch-and-bound algorithms). To this end, setting $x_+ = (1, x^T)^T$, observe that the objective and the constraints in (11) are linear in the matrix $X(x) = x_+ x_+^T$: $f_i(x) = \mathrm{Tr}(Q_i X(x))$, $i = 0, ..., m$, where $Q_i = \left[\begin{array}{cc} c_i & b_i^T \\ b_i & A_i \end{array}\right]$. When x runs through $\mathbf{R}^n$, $X(x)$ runs through the set cut off the convex set $\{X \succeq 0, X_{11} = 1\}$ by the requirement $\mathrm{Rank}(X) = 1$. Removing this requirement (and thus extending problem's feasible set), we arrive at the sdp

$$\mathrm{Opt(SDP)} = \min_X \{\mathrm{Tr}(Q_0 X) : \mathrm{Tr}(Q_i X) \le 0, \, i = 1, ..., m, X \succeq 0, X_{11} = 1\}; \quad (12)$$

due to the origin of this program, we have $\mathrm{Opt(SDP)} \le \mathrm{Opt}$.

Another way to arrive at (12) is to use Lagrange relaxation: whenever $\lambda \ge 0$, the quantity $L_*(\lambda) \equiv \inf_x \{f_0(x) + \sum_{i=1}^m \lambda_i f_i(x)\}$ is a lower bound on Opt. Maximizing this bound over $\lambda \ge 0$, we again get a lower bound on Opt; at the same time, the fact that all f_i are quadratic makes the program $\max_{\lambda \ge 0} L_*(\lambda)$ an explicit sdp, which is nothing but the semidefinite dual of (12).

Seemingly the first application of SDP in building computable bounds on difficult combinatorial entities is the famous *Lovasz capacity number $\theta(G)$ of a graph G* – a computable upper bound on the stability number of G introduced in [40]. The bound is

$$\begin{array}{l} \theta(G) = \min_{\lambda, X} \{\lambda : \lambda I \succeq X, X_{ij} = 1 \text{ when } (i, j) \text{ is not an arc}\} \\ \quad = \max_Y \left\{\sum_{i,j} Y_{ij} : Y \succeq 0, \mathrm{Tr}(Y) = 1, Y_{ij} = 0 \text{ when } (i, j) \text{ is an arc}\right\}. \end{array} \quad (13)$$

$\theta(G)$ is in-between the stability number $\alpha(G)$ of G and the chromatic number $\xi(\bar{G})$ of the complementary graph: $\alpha(G) \le \theta(G) \le \xi(\bar{G})$ ("Lovasz sandwich theorem"); it coincides with $\alpha(G)$ for perfect graphs, and possesses a number of other interesting and important

properties. In hindsight, $\theta(G)$ can be obtained by Lagrange relaxation from the representation $\alpha(G) = \max_x \left\{\sum_i x_i : x_i^2 - x_i = 0, x_i x_j = 0 \text{ whenever } (i,j) \text{ is an arc}\right\}$. Lovasz capacity is one of the earliest precursors to SDP, and the second equality in (13) found in [40] seems to be the first example of SDP duality.

Yet another way to think about the relaxation (this way *in hindsight* can be traced back to Grothendieck [25], 1953) is to imagine that we are looking for a random vector ξ which *at average* satisfies the constraints of the original problem and minimizes under this restriction the expected value of the objective; X in (12) can be thought of as the covariance matrix $\mathbf{E}\left\{(1,\xi^T)^T(1,\xi^T)\right\}$. The advantage of the latter interpretation is that it suggests a way to produce suboptimal solutions to (11) from the optimal solution X_* to (12). Specifically, given X_*, we can point out (in fact, in many ways) a random vector ξ such that $\mathrm{Cov}(\xi) = X_*$, thus getting a random solution to (11) which *at average* is feasible with expected value of the objective Opt(SDP), i.e., better than the true optimal value Opt. In favourable circumstances it is possible to convert, at a controllable cost in terms of the objective, the "feasible at average" random solution into an actually feasible solution; whenever it is the case, SDP relaxation yields a suboptimal solution to the problem of interest with *known* level of non-optimality. Examples include:

• The famous MAXCUT-related result of Goemans and Williamson [23] stating that SDP relaxation bound in the MAXCUT problem is tight up to factor 1.1382...

In the MAXCUT problem, one is given a graph with arcs assigned nonnegative weights and is looking for a cut (coloring of nodes into two colors) of maximal weight (the total weight of arcs linking nodes of different colors). MAXCUT can be posed in the form of (11), specifically, as

$$\mathrm{Opt} = \max_x \left\{x^T L x : x_i^2 \le 1, i = 1, ..., n\right\} \tag{14}$$

where $L \in \mathbf{S}^n$ is the Laplace matrix of the graph (and thus satisfying $L \succeq 0$, $L_{ij} \le 0$ when $i \ne j$ and $\sum_j L_{ij} = 0$) and n is the number of nodes in the graph. MAXCUT is an NP-complete combinatorial problem; it is known [30] that it is NP-hard to approximate Opt within 4%-accuracy, even when randomized algorithms are allowed. In spite of this "severe computational intractability" of MAXCUT, Goemans and Williamson show that the SDP relaxation of (14) yields a surprisingly tight bound Opt(SDP) on Opt: $1 \le \mathrm{Opt(SDP)}/\mathrm{Opt} \le 1.1382...$ The proof goes as follows: the optimal solution X_* to the SDP relaxation, which is the problem $\max_X \{\mathrm{Tr}(LX) : X \succeq 0, X_{ii} = 1 \forall i\}$, is treated as the covariance matrix of a Gaussian random vector ξ with zero mean; this "feasible at average" random solution ξ is corrected to $(\mathrm{sign}(\xi_1), ..., \mathrm{sign}(\xi_n))$, thus yielding a feasible solution with the expected value of the objective at least Opt(SDP)/1.1382... For extensions and modifications of this result, see [73, Chapter 12] and references therein.

• Nesterov's $\frac{\pi}{2}$ theorem [52] stating that for an *arbitrary* matrix $L \succeq 0$ in (14), the SDP relaxation yields a $\frac{\pi}{2}$-tight upper bound on Opt, and that this remains true when the constraints $x_i^2 \le 1$ in (14) are replaced with an arbitrary system of linear equality and inequality constraints on x_i^2 (for extensions and applications, see [75, 73, Chapter 13]).

One of far-reaching consequences of this fact is a tight efficiently computable upper bound on the (p,r)-norm $\|A\|_{p,r} = \max_x\{\|Ax\|_r : \|x\|_p \le 1\}$. Computing (p,r)-norm is known to be easy only in the cases $p = 1$, $r = \infty$ and $p = r = 2$, and is known to be NP-hard when $p > r$. Nesterov [73, Theorem 13.2.4] shows that when $\infty \ge p \ge 2 \ge r \ge 1$,

the efficiently computable quantity

$$\Psi(A) = \frac{1}{2} \min_{\mu\in\mathbf{R}^n,\nu\in\mathbf{R}^m} \left\{ \|\mu\|_{\frac{p}{p-2}} + \|\nu\|_{\frac{r}{2-r}} : \left[\begin{array}{c|c} \mathrm{Diag}\{\mu\} & A^T \\ \hline A & \mathrm{Diag}\{\nu\} \end{array}\right] \succeq 0 \right\}$$

is an upper bound, tight within the factor $\frac{1}{\frac{2\sqrt{3}}{\pi}-\frac{2}{3}} = 2.2936...$, on $\|A\|_{p,r}$. When $r = 2$ or $p = 2$, the tightness factor can be improved to $\sqrt{\pi/2} = 1.2533...$. Finally, we clearly have $\|A\|_{\infty,1} = \frac{1}{2}\max_{\|z\|_\infty\le 1} z^T \left[\begin{array}{c|c} & A \\ \hline A^T & \end{array}\right] z$; the fact that the sdp relaxation of the latter problem yields a tight, within an absolute constant factor, upper bound on $\|A\|_{\infty,1}$ is nothing but a rephrasing of the Grothendieck inequality discovered as early as in 1953 [25][3)]. In the case in question, the tightness factor of the sdp relaxation bound can be improved to $\frac{\pi}{2\ln(1+\sqrt{2})} \approx 1.7822...$ [37], which is better than Nesterov's "universal" constant 2.2936... For further results on efficient bounding of $\|A\|_{\infty,1}$, see [3].

- "Approximate $\mathcal{S}$-Lemma" [11] stating that the SDP relaxation of the problem $\mathrm{Opt} = \max_x \left\{x^T Lx : x^T Q_i x \le 1,\ i = 1, ..., m\right\}$ with $Q_i \succeq 0$ and possibly indefinite L results in an efficiently computable upper bound Opt(SDP) on Opt which is tight within the factor $O(1)\ln(m+1)$ (and indeed can differ from Opt by factor $O(\ln(m+1))$ even when $L \succeq 0$);
- "Matrix Cube theorem" [10] to be discussed later.

Applications in Structural Design. We have already considered one of these applications – Truss Topology Design, which can be reduced to CQP. SDP offers a natural way to treat Structural Design problems in more complicated settings, most notably in the Free Material Optimization one, see [7, 73, Chapter 15] and references therein. In Free Material Optimization, one seeks for a construction comprised of material distributed over a given 2D/3D domain with varying from point to point mechanical properties and capable to withstand best of all a number of external loads. After Finite Element discretization, the problem reads

$$\min_{\{t_i\}} \left\{ \max_{1\le\ell\le k} \mathrm{Compl}_\ell(t) = \max_{P_\ell v\le p_\ell} \left[f_\ell^t v - v^T A(t) v/2\right] : t_i \succeq 0, \sum_i \mathrm{Tr}(t_i) \le w \right\}, \tag{15}$$

where $A(t) = \sum_{i,s} b_{is} t_i b_{is}^T$ is the *stiffness matrix*, $t_i \in \mathbf{S}^d$ are rigidity tensors of the material in the finite element cells ($d = 3$ for 2D and $d = 6$ for 3D constructions), $P_\ell v \le p_\ell$ are constraints on virtual displacements of the nodes of finite element cells given by rigid obstacles in ℓ-th loading scenario, f_ℓ represents the ℓ-th load, and $\mathrm{Compl}_\ell(t)$ is the corresponding compliance – potential energy capacitated in the construction in the static equilibrium under ℓ-th load. Same as in the TTD case, the main advantage of the FMO model is that it allows to find the topology of the optimal construction; this topology serves as a starting point in actual engineering design restricted to traditional materials and taking into account a lot of complicating details ignored in the FMO model.

[3)] this paper implies, in particular, that the absolute constant $\pi/2$ in Nesterov's $\pi/2$ theorem cannot be improved.

Same as in truss design, compliance in (15) admits an $\mathcal{SDP}$-r., and (15) can be reformulated as the SDP program

$$\min_{t,\tau,\mu}\left\{\tau:\ \begin{array}{c}\left[\begin{array}{cc}2\tau-2p_\ell^T\mu_\ell & \mu_\ell^TP_\ell-f_\ell^T\\ P_\ell^T\mu_\ell-f_\ell & A(t)\end{array}\right]\succeq 0\,\forall \ell\le k\\ t_i\succeq 0\,\forall i,\sum_i\mathrm{Tr}(t_i)\le w,\ \mu_\ell\ge 0\,\forall\ell\le k\end{array}\right\}. \tag{16}$$

Same as in the TTD case, semidefinite duality admits for instructive and better suited for numerical processing equivalent reformulations of (16), in particular, those where the Newton systems to be solved in IP methods are sparse (a rare case in SDP!). As a result, when the number of loading scenarios is small (like 2 or 3), IP methods allow to solve real world FMO problems with design dimension as large as many tens of thousands [36].

Another application of SDP in Structural Design relates to ensuring dynamical stability of a construction, i.e., imposing a lower bound on its eigenfrequencies. This can be modelled by the constraint $A(t)\succeq\omega^2M(t)$, where t is the vector of design parameters, $A(t)$, $M(t)$ are the stiffness and the mass matrices and ω is the desired lower bound. Whenever $A(t)$, $M(t)$ are affine in t (as it is the case for trusses and in FMO), the constraint in question is an LMI.

Control applications. For the time being, the most "mature" applications of SDP are those in Control, where for the last decade or so LMI's became a kind of standard language to pose and to process various control-related problems (see, e.g., [13, 16, 73, Chapter 14] and references therein). To give a flavour of related constructions and results, consider the basic problem of Lyapunov stability analysis/synthesis. The analysis problem is: given an uncertain n-dimensional linear dynamical system

$$\dot{x}(t)=A(t)x(t),\ t\ge 0, \tag{17}$$

where the only restriction on $A(t)$ is to belong, at any time t, to a given "uncertainty set" $\mathcal{U}$, certify systems's *robust stability* – the fact that all trajectories of (all realizations of) the system tend to 0 as $t\to\infty$. In the "certain" case of $A(t)\equiv A$, a necessary and sufficient stability condition is the existence of *Lyapunov Stability Certificate* (LSC) – a matrix $X\succeq I$ and $\alpha>0$ such that $A^TX+XA\preceq-\alpha X$. For an uncertain system, the standard *sufficient* stability condition is the existence of a common LSC (X,α) for all realizations $A\in\mathcal{U}$ of system's matrix; such an LSC implies that $x^T(t)Xx(t)\le\exp\{-\alpha t\}x^T(0)Xx(0)$ for all trajectories, and thus – "stability with the decay rate α". Thus, LSC's of uncertain system are described by the infinite system of matrix inequalities

$$A^TX+XA+\alpha X\preceq 0\,\forall A\in\mathcal{U} \tag{18}$$

in variables $X\succeq I$, $\alpha>0$. To simplify our presentation, we treat below the decay rate α as a given positive constant rather than a variable, which makes (18) an infinite system of LMI's in X. In some cases, this infinite system can be replaced with a finite one, thus allowing for efficient computation of an LSC or detecting that no one exists. The simplest cases of this type are *polytopic uncertainty* $\mathcal{U}=\mathrm{Conv}\{A_1,...,A_N\}$ (the equivalent finite system of LMI's is merely A_i^TX+

$XA_i + \alpha X \preceq 0$, $i = 1, ..., N$) and *norm-bounded uncertainty* $\mathcal{U} = \{A = \bar{A} + B\Delta C : \|\Delta\| \le \rho\}$ (here the LSC's are exactly the X-components of the solutions (X, λ) of the LMI $\left[\begin{array}{c|c} \bar{A}^T X + X\bar{A} + \alpha X + \lambda C^T C & \rho XB \\ \hline \rho B^T X & -\lambda I \end{array}\right] \preceq 0$, see [13]). There are also cases where (18), while being NP-hard, admits provably tight tractable approximation, most notably, the case of "interval uncertainty of level $\rho > 0$":

$$\mathcal{U} = \mathcal{U}_\rho = \{A : |A_{ij} - \bar{A}_{ij}| \le \rho\delta_{ij} \forall i, j\}. \tag{19}$$

In the case of (19), X solves (18) if and only if the image of the cube $\{\zeta \in \mathbf{R}^{n\times n} : \|\zeta\|_\infty \le \rho\}$ under the affine mapping

$$\zeta \mapsto B[X] + \sum_{i,j} \zeta_{ij} B^{ij}[X],\ B[X] = -\bar{A}^T X - X\bar{A} - \alpha X, B^{ij}[X] = -\delta_{ij}\left[e_j e_i^T X + X e_i e_j\right],$$

e_i being the basic orths, belongs to the semidefinite cone. Now, the question whether a "matrix cube" $\mathcal{B}_\rho = \{B + \sum_{\nu=1}^N z_\nu B_\nu : \|z\|_\infty \le \rho\}$ belongs bo the semidefinite cone is NP-hard, unless all matrices B_ν are of rank 1 (already with $\mathrm{Rank}\, B_\nu = 2$, this question is at least as difficult as the MAXCUT problem [10]). There is, however, an evident verifiable *sufficient* condition for the inclusion $\mathcal{B}_\rho \subset \mathbf{S}^n_+$, namely, the existence of matrices $\{X_\nu\}_{\nu=1}^N$ such that $X_\nu \succeq \pm B_\nu$ and $B_0 - \rho\sum_\nu X_\nu \succeq 0$. It turns out ("Matrix Cube theorem" [10]) that this condition is tight, provided that all "edge matrices" B_ν are of low rank; specifically, if the condition is *not* satisfied for certain ρ, then $\mathcal{B}_{\vartheta\rho} \not\subset \mathbf{S}^n_+$, with $\vartheta = \vartheta(\mu) \le \sqrt{\pi\mu/2}$ depending solely on $\mu = \max_\nu \mathrm{Rank}\, B_\nu$; note that $\vartheta(1) = 1$ and $\vartheta(2) = \pi/2$. Thus, our verifiable sufficient condition for the inclusion $\mathcal{B}_\rho \subset \mathbf{S}^n_+$ allows to identify, within factor $\vartheta(\mu)$, the largest ρ for which the inclusion takes place. Now note that when building an LSC for interval uncertainty (19), we seek for X such that a specific matrix cube *with edge matrices $B^{ij}[X]$ of rank 2*, depending on X as on a parameter, belongs to $\mathbf{S}^n_+$. Replacing the latter constraint with the above verifiable sufficient condition with its validity, we end up with a system of LMI's

$$X^{ij} \succeq \pm B^{ij}[X],\ i, j = 1, ..., n, B[X] - \rho\sum_{ij} X^{ij} \succeq 0 \tag{20}$$

in variables (X, X^{ij}) such that the X-part of a feasible solution to this system is feasible for the infinite system of LMI's (18) – (19). The resulting "safe approximation" of (by itself intractable) system (18) – (19) is "tight within the factor $\pi/2$" – whenever (20) is infeasible, so is the system of interest with increased by factor $\pi/2$ uncertainty level. In particular, we can find efficiently a tight, within the factor $\pi/2$, lower bound on the largest possible uncertainty level for which the stability still can be certified by an LSC.

From practical viewpoint, a shortcoming of (20) is large, although polynomial in n, design dimension of this system of LMI's; an $n\times n$ matrix variable per each uncertain entry in the matrix of the original dynamical system is too much... Well, applying semidefinite duality, one can convert (20) into an equivalent system of LMI's in X and just $\approx \frac{3}{2}n^2$ additional scalar variables. Here again we see how useful could be the conic programming machinery in analytical processing of optimization problems.

For the time being, we were focusing on the stability analysis. In the *Lyapunov stability synthesis* problem, one is given a controlled dynamical system

$$\dot{x}(t) = A(t)x(t) + B(t)u(t),\ y(t) = C(t)x(t) \tag{21}$$

where $(A(t), B(t), C(t))$ is known to belong to a given uncertainty set $\mathcal{U}$, and is looking for output-based linear feedback control $u(t) = Ky(t)$ which allows to equip the closed loop system with an LSC (and thus makes it stable). Thus, we look for both a stabilizing feedback *and* a LSC for the closed loop system. The synthesis problem admits a nice LMI-based solution *at least* when the state-based feedback is allowed, i.e., when $C(t) \equiv I$, which we assume from now on. The Lyapunov matrix inequality (18) for the closed loop system reads

$$[A + BK]^T X + X[A + BK] + \alpha X \preceq 0 \ \forall [A, B] \in \mathcal{U}; \tag{22}$$

this is *not* a LMI in our new design variables anymore. However, passing from X, K to the variables $Y = X^{-1}$, $Z = KY$ and multiplying both sides in (22) by Y from the left and from the right, we rewrite (22) as the infinite system of LMI's

$$AY + YA^T + BZ + Z^T B^T + \alpha Y \preceq 0 \ \forall [A, B] \in \mathcal{U}.$$

Same as above, this infinite system of LMI's can be processed efficiently in the case of polytopic or norm-bounded uncertainty, admits tractable tight approximation in the case of interval uncertainty, etc.

Not only stability, but many other "desired properties" of a linear time-invariant dynamical system (passivity, contractiveness, positive realness, nonexpansiveness, etc.) are certified, in a necessary and sufficient fashion, by solutions to appropriate LMI's. Usually, existence of a *common* certificate of this type for all realizations of system's data from a given uncertainty set becomes *sufficient condition* for the associated uncertain system to possess the robust version of the property in question; this explains the unique role played by SDP in Robust Control.

Extremal ellipsoids. In many situations (see, e.g., [15] and references therein), it is natural to approximate convex sets by ellipsoids – the latter are especially easy-to-specify and easy-to-operate convex sets. There are several ways to build ellipsoidal approximations of a convex solid $A \subset \mathbf{R}^n$. When A is given by a membership oracle, one can find a $O(n^{3/2})$-rounding of A, i.e., a pair of concentric similar ellipsoids $E_* \subset A \subset E^*$ with the ratio of linear sizes $O(n^{3/2})$, by a kind of Ellipsoid method [26]. When A admits an explicit ϑ-self-concordant barrier F, one can build a $(\vartheta + 2\sqrt{\vartheta})$-rounding of A by approximating the minimizer of F over A ([13], cf. the end of Section 3.1). SDP enters the game when we are interested to find the "largest" ellipsoid contained in A or the "smallest" ellipsoid containing A. Indeed, representing ellipsoids in $\mathbf{R}^n$ in the form of $E = E(y, Y) = \{x = y + Yu : u^T u \leq 1\}$ with $Y \in \mathbf{S}^n_+$, natural "sizes" of E, like $(\mathrm{mes}_n(E))^{1/n} = c_n(\det(Y))^{1/n}$, or the smallest half-axis $\lambda_{\min}(Y)$, or the sum of k smallest half-axes (i.e., the k smallest eigenvalues of Y), become $\mathcal{SDP}$-representable concave functions of the parameters y, Y of the ellipsoid, so that maximizing such a size over all ellipsoids contained in A becomes a sdp, *provided that the set of parameters y, Y of ellipsoids contained in A is an $\mathcal{SDP}$-s.* Similarly, representing ellipsoids in $\mathbf{R}^n$ in the form of $W = W(z, Z) = \{x : (x - Z^{-1}z)^T Z^2 (x - Z^{-1}z) \leq 1\}$ with $Z \succ 0$, the sizes of W like $(\mathrm{mes}_n(W))^{\frac{1}{2n}} = d_n(\det(Z))^{-1/n}$, or the largest half-axis of W (i.e., $1/\lambda_{\min}(Z)$), or the sum of k largest half-axes of W, become $\mathcal{SDP}$-representable functions of

z, Z, so that minimizing such a size over all ellipsoids containing A again is an sdp, *provided that the set of parameters z, Z of ellipsoids $W(z, Z)$ containing A is an $\mathcal{SDP}$-s.* Thus, when seeking for an extremal ellipsoid inscribed into/circumscribed around a solid $A \subset \mathbf{R}^n$ reduces to finding an $\mathcal{SDP}$-r. for the parameters y, Y, respectively, z, Z of the ellipsoids contained into/containing A. The key positive result here is as follows [13, Section 3.7]: one has $E(y, Y) \subset W(z, Z)$ if and only if there exists λ such that $\left[\begin{array}{c|c|c} I & ZY & Zy - z \\ \hline YZ & \lambda I & \\ \hline y^T Z^T - z^T & & 1 - \lambda \end{array}\right] \succeq 0$, which is an LMI in (y, Y, λ) when z, Z are fixed, and is an LMI in (z, Z, λ) when y, Y are fixed. As a result, the problems of finding (a) the smallest outer ellipsoidal approximation of the union of finitely many ellipsoids, and (b) the largest inner ellipsoidal approximation of the intersection of finitely many ellipsoids can be posed as explicit sdp's. The same is true for the problems of finding the largest inner ellipsoidal approximation of a polytope given by a list of linear inequalities, and finding the smallest outer ellipsoidal approximation of the convex hull of finitely many points. In contrast to this, it is NP-hard to verify that a given ellipsoid is contained in the convex hull of a given finite set of points or contains a polytope given by a list of linear inequalities; thus, the problems of inner ellipsoidal approximation of a convex hull of finite set and outer ellipsoidal approximation of the set of solutions of a finite system of linear inequalities both seem to be computationally intractable.

Concluding remarks. We hope that the outlined constructions and results demonstrate that Convex Programming is not merely something about number crunching; its development, stimulated both by intrinsic reasons and by needs of applications, requires resolving challenging mathematical problems with a specific "operational" flavour (at the end of the day, we want to understand how to build something rather than how something is built) which is a nice complement to typical descriptive flavour of problems arising in purely theoretical areas of Mathematics. The most famous, posed in 1960's and still open, challenge here is to understand whether LP admits a strongly polynomial algorithm (essentially, a Real Arithmetic algorithm solving LP's *exactly* in time polynomial in the dimension of the data). Simplex-type LP algorithms are finite, but no one of them is known to be polynomial (and most are known *not* to be so); all known exact polynomial algorithms for LP work with rational data only, with running time polynomial in the *bit length* of the data, not in the number of data entries. All known in this direction is the existence of a strongly polynomial algorithm for LP with integer *and varying in a once for ever fixed finite range* data in the constraint matrix and real data in the objective and the right hand side (Tardos [68], see also [70]).

The major, in our appreciation, recent challenge comes from the desire to process extremely large-scale (tens and hundreds of thousands of variables) conic quadratic and semidefinite programs arising in some of applications (relaxations of combinatorial problems, structural design, etc.). Problems of huge sizes are beyond the "practical scope" of interior point methods with their Newton-type, and therefore too time consuming in the large scale case, iterations and require essentially different optimization techniques (cf. [56, 44]). Note that in the extremely large

scale case utilizing problem's structure becomes really crucial, which increases the importance of "structure-revealing" Conic Programming formulations of convex programs.

References

[1] Alizadeh, F., Interior point methods in semidefinite programming with applications to combinatorial problems, *SIAM J. Optim.* **5** (1995), 13–51.

[2] Alizadeh, F., Goldfarb, D., Second-order cone programming, *Math. Program.* **95** (2003), 3–51.

[3] Alon, N., Naor, A., Approximating the Cut-Norm via Grothendieck's Inequality. *Proc. of the 36 ACM STOC, Chicago*, ACM Press (2004), 72–80; to appear in *SIAM J. Computing*.

[4] Andersen, E.D., Ye, Y., On a homogeneous algorithm for monotone complemntarity system, *Math. Program.* **84** (1999), 375–399.

[5] Arrow, K.J., Hurwicz, L., Uzawa, H., *Studies in linear and non-linear programming*. Stanford University Press, Stanford, 1958.

[6] Bauschke, H., Güler, O., Lewis, A.S., Sendov, H.S., Hyperbolic polynomials and convex analysis, *Canadian J. of Mathematics* **53** (2001), 470–488.

[7] Ben-Tal, A., Kočvara, M., Nemirovski, A., Zowe, J., Free material design via semidefinite programming. The multiload case with contact conditions, *SIAM J. Optim.* **9** (1999), 813–832.

[8] Ben-Tal, A., Nemirovski, A. *Lectures on Modern Convex Optimization: Analysis, Algorithms and Engineering Applications*. MPS-SIAM Series on Optimization, SIAM, Philadelphia, 2001.

[9] Ben-Tal, A., and Nemirovski, A., On polyhedral approximations of the second-order cone, *Math. of Oper. Res.* **26** (2001), 193–205.

[10] Ben-Tal, A., and Nemirovski, A., On tractable approximations of uncertain linear matrix inequalities affected by interval uncertainty", *SIAM J. Optim.* **v.** 12 (2002), 811–833.

[11] Ben-Tal, A., Nemirovski, A., and Roos, C., Robust solutions of uncertain quadratic and conic-quadratic problems, *SIAM J. Optim.* **13** (2002), 535–560.

[12] Blum, L., Cucker, F., Shub, M., Smale, S., *Complexity and Real Computation*. Springer-Verlag, 1997.

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

[14] Boyd, S., Vandenberghe, L., *Convex Optimization*. Cambridge University Press, 2004.

[15] Chernousko, F.L., *State estimation for dynamic systems*. CRC Press, 1994.

[16] El Ghaoui, L., Niculescu, S.-I. (Eds.), *Advances on Linear Matrix Inequality Methods in Control*. SIAM, Philadelphia, 1999.

[17] Faybusovich, L.. Euclidean Jordan algebras and interior-point algorithms, *Positivity* **1** (1997), 331–357.

[18] Faybusovich, L., Linear system in Jordan algebras and primal-dal interior-point algorithms, *J. of Comput. and Appl. Math.* **86** (1997), 149–175.

[19] Fiacco, A., McCormic, G.P., *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*. J. Wiley & Sons, 1968.

[20] Fu, M., Luo, Z.-Q., Ye, Y., Approximation algorithms for quadratic programming, *J. of Combinatorial Optim.* **2** (1998), 29–50.

[21] Garey, M.R., Johnson, D.S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

[22] Gaterman, K., Parrilo, P.A., Symmetry groups, semidefinite programs, and sums of squares, *J. of Pure and Applied Algebra* **192** (2004), 95–128.

[23] Goemans, M.X., Williamson, D.P., Improved approximation algorithms for maximum cut and satisfability problem using semidefinite programming, *J. of Association for Computing Machinery* **42** (1995), 1115–1145.

[24] Gonzaga, C.C., An algorithm for solving linear programming problems in $O(n^3L)$ operations. In: *Advances in mathematical programming - Interior point and related methods* (edited by N. Megiddo). Springer-Verlag, New York, 1989.

[25] Grothendieck, A., Résuḿe de la théorie ḿetrique des produits tensoriels topologiques, *Bol. Soc. Mat. Sao Paulo* **8** (1953), 179.

[26] Grotschel, M., Lovasz, L., Schrijver, A., *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, Berlin, 1987.

[27] Güler, O., On the self-concordance of the universal barrier function, *SIAM J. Optim.* **7** (1997), 295–303.

[28] Güler, O., Hyperbolic polynomials and interior point methods for convex programming, *Math. of Oper. Res.* **22** (1997), 350–377.

[29] Güler, O., Tunçel, L., Characterization of the barrier parameter of homogeneous convex cones, *Math. Program.* **81** (1998), 55–76.

[30] Håstad, J., Some optimal inapproximability results, *J. of ACM* **48** (2001), 798–859.

[31] Hildebrand, R., An LMI description for the cone of Lorentz-positive maps, http://www.optimization-online.org/DB_HTML/2005/12/1260.html

[32] Jarre, F., Optimal ellipsoidal approximations around the analytic center, *Appl. Math. Optim.* **30** (1994), 15–19.

[33] Karmarkar, N., A new polynomial-time algorithm for linear programming, *Combinatorica* **4** (1984), 373–395.

[34] Khachiyan, L.G., A Polynomial Algorithm in Linear Programming (in Russian), *Doklady Akademii Nauk SSSR* **244** (1979), 1093–1097 [English traslation: *Soviet Mathematics Doklady* **20**, 191–194].

[35] de Klerk, E., Roos, C., Terlaky, T., Initialization in semidefinite programming via a self-dual skew-symmetric embedding, *Oper. Res. Letters* **20** (1997), 213–221.

[36] Koçvara, M., Stingl, M., PENNON – A Code for Convex Nonlinear and Semidefinite Programming, *Optim. Methods and Software* **18** (2003), 317–333.

[37] Krivine, J.L., Sur la constante de Grothendieck, *C. R. Acad. Sci.* Paris Ser. A-B **284** (1977), 445-446.

[38] Lasserre, J.B., Global optimization with polynomials and the problem of moments, *SIAM J. Optim.* **11** (2001), 796–817.

[39] Lobo, M., Vandenberghe, L., Boyd, S., Lebret, H., Applications of second-order cone programming, *Linear Algebra and Appl.* **284** (1998), 193–228.

[40] Lovasz, L., On the Shannon capacity of graphs, *IEEE Trans. on Inform. Theory* **25** (1979), 1–7.

[41] Lewis, A.S., Parillo, P., Ramana, M., The Lax conjecture is true, *Proceedings of the AMS* **133** (2005), 2495–2499.

[42] Luo, Z.-Q., Sturm, J.F., Zhang, S., Coinic convex programming and self-dual embedding, *Optim. Methods and Software* **14** (2000), 169–218.

[43] Nemirovski, A., Yudin, D., Information-based complexity and efficient methods of convex optimization (in Russian), *Ekonomika i Matematicheskie Metody* [English translation: *Matekon*] **12** (1976), 357–379.

[44] Nemirovski, A., Prox-method with rate of convergence $O(1/t)$ for variational inequalities with Lipschitz continuous monotone operators and smooth convex-concave saddle point problems, *SIOPT J. Optim.* **15** (2004), 229–251.

[45] Nesterov, Yu., Nemirovski, A., Conic duality and its applications in Convex Programming, *Optim. Methods and Software* **1** (1992), 95–115.

[46] Nesterov, Yu., Nemirovski, A., *Interior Point Polynomial Time Methods in Convex Programming*. SIAM, Philadelphia, 1994.

[47] Nesterov, Yu., Long-step strategies in interior-point primal-dual methods, *Math. Program.* **76** (1997), 47–94.

[48] Nesterov, Yu., Todd, M. J., Self-scaled barriers and interior-point methods for Convex Programming, *Math. of Oper. Res.* **22** (1997), 1–42.

[49] Nesterov, Yu., Todd, M. J., Primal-dual interior-point methods for self-scaled cones, *SIAM J. Optim.* **8** (1998), 324–364.

[50] Nesterov, Yu., Nemirovski, A., Multiparameter surfaces of analytic centers and long-step path-following interior point methods, *Math. of Oper. Res.* **23** (1998), 1–38.

[51] Nesterov, Yu., Todd, M.J., Ye, Y., Infeasible-start primal-dual methods and infeasibility detectors for nonlinear programming problems, *Math. Program.* **84** (1999), 227–267.

[52] Nesterov, Yu., Semidefinite relaxation and nonconvex quadratic optimization, *Optim. Methods and Software* **9** (1998), 141-160.

[53] Nesterov, Yu., Squared functional systems and optimization problems. In *High Performance Optimization* (edited by H. Frenk, T. Terlaky, Sh. Zhang). Kluwer Academic Publishers, 1999, 405–439.

[54] Nesterov, Yu., Todd, M.J., On the Riemannian geometry defined by self-concordant barriers and interior-point methods, *Found. of Comput. Math.* No. 2 (2002), 333–361.

[55] Nesterov, Yu., Nemirovski, A., Central path and Riemannian distances. Discussion paper 2003/30, CORE, Louvain-la-Neuve, 2003.

[56] Nesterov, Yu., Smooth minimization of non-smooth functions, *Math. Program.* **103** (2005), 127–152.

[57] Parrilo, P.A., Semidefinite programming relaxations for semialgebraic problems, *Math. Program. Series B* **96** (2003), 293–320, 2003.

[58] Potra, F.A., Sheng, R., On homogeneous interior-point algorithms for semidefinite programming, *Optim. Methods and Software* **9** (1998), 161–184.

[59] Ramana, M., An exact duality theory for semidefinite programming and its complexity implications, *Math. Program. Series B* **77** (1997), 129–162.

[60] Renegar, J., A polynomial-time algorithm, based on Newton's method, for linear programming, *Math. Program.* **40** (1988), 59–93.

[61] Renegar, J., *A Mathematical View of Interior-Point Methods in Convex Optimization*. MPS-SIAM Series on Optimization, SIAM, Philadelphia, 2001.

[62] Schmieta, S.H., Alizadeh, F., Associative and Jordan Algebras, and Polynomial Time Interior-Point Algorithms for Symmetric Cones, *Math. of Oper. Res.* **26** (2001), 543–564.

[63] Schmieta, S.H., Alizadeh, F., Extension of primal-dual interior point methods to symmetric cones, *Math. Program.* **96** (2003), 409–438.

[64] Shapiro, A., Extremal problems on the set of nonnegative definite matrices, *Linear Algebra and Appl.* **67** (1985), 7–18.

[65] Shor, N.Z., Cut-off method with space extension in convex programming problems, *Cybernetics* **12** (1977), 94–96.

[66] Shor, N.Z., Class of global minimum bounds of polynomial functions, Cybernetics, **23** (1987), 731–734.

[67] Shor, N.Z., *Nondifferentiable Optimization and Polynomial Problems*. Kluwer Academic Publishers, Dordrecht, 1998.

[68] Tardos, E., A strongly polynomial minimum cost circulation algorithm, *Combinatorica* **5** (1985), 247–256.

[69] Vandenberghe, L., Boyd, S., Applications of semidefinite programming, *Applied Numerical Mathematics* **29** (1999), 283–299.

[70] Vavasis, S., Ye, Y., A primal-dual interior point method whose running time depends only on the constraint matrix, *Math. Program.* **74** (1996), 79–120.

[71] Vinberg, E.B., The theory of homogeneous cones, *Trans. of Moscow Math. Soc.* **12** (1965), 340–403.

[72] Ye, Y., Todd, M.J., Mizuno, S., An $O(\sqrt{nL})$-iteration homogeneous and self-dual linear programming algorithm, *Math. of Oper. Res.* **19** (1994), 53–67.

[73] H. Wolkowicz, R. Saigal, L. Vandenberghe (Eds.), *Handbook of Semidefinite Programming*. Kluwer Academic Publishers, 2000.

[74] Ye, Y., *Interior-Point Algorithms: Theory and Analysis*. Wiley-Interscience Series in Discrete Mathematics and Optimization, John Wiley & Sons, 1997.

[75] Ye, Y., Approximating quadratic programming with bound and quadratic constraints, *Math. Program.* **84** (1999), 219–226.

[76] Ye, Y., Zhang, S., New results on quadratic minimization, *SIAM J. Optim.* **14** (2003), 245–267.

[77] Xu, X., Hung, P.F., Ye, Y., A simplified homogeneous self-dual linear programming algorithm and its implementation, *Annals of Oper. Res.* **62** (1996), 151–171.

ISYE, Georgia Institute of Technology, 765 Ferst Drive, Atlanta GA 30332-0205 USA
E-mail: nemirovs@isye.gatech.edu