Solving natural conic formulations with Hypatia.jl

Chris Coey[1], Lea Kapelevich[1], and Juan Pablo Vielma[2]

[1]Operations Research Center, MIT, Cambridge, MA
[2]Google Research and MIT Sloan School of Management, Cambridge, MA

July 12, 2021

Abstract

Many convex optimization problems can be represented through conic *extended formulations* with auxiliary variables and constraints using only the small number of *standard cones* recognized by advanced conic solvers such as MOSEK 9. Such extended formulations are often significantly larger and more complex than equivalent conic *natural formulations*, which can use a much broader class of *exotic cones*. We define an exotic cone as a proper cone for which we can implement tractable logarithmically homogeneous self-concordant barrier oracles for either the cone or its dual cone. In this paper we introduce Hypatia, a highly-configurable open-source conic primal-dual interior point solver with a generic interface for exotic cones. Hypatia is written in Julia and accessible through JuMP, and currently implements around two dozen useful predefined cones (some with multiple variants). We define some of Hypatia's exotic cones, and for conic constraints over these cones, we analyze techniques for constructing equivalent representations using the standard cones. For optimization problems from a variety of applications, we introduce natural formulations using these exotic cones, and we show that the natural formulations are simpler and lower-dimensional than the equivalent extended formulations. Our computational experiments demonstrate the potential advantages, especially in terms of solve time and memory usage, of solving the natural formulations with Hypatia compared to solving the extended formulations with either Hypatia or MOSEK 9.

Contents

1 Introduction

Any convex optimization problem may be represented as a conic problem that minimizes a linear function over the intersection of an affine subspace with a Cartesian product of primitive proper cones (i.e. irreducible, closed, convex, pointed, and full-dimensional conic sets). An advantage of using conic form is that, under certain conditions, a conic problem has a simple and easily checkable certificate of optimality, primal infeasibility, or dual infeasibility [Permenter et al., 2017]. Although the scope of this paper is limited to conic problems, there are other useful notions of duality that can be leveraged by convex optimization solvers (see for example DDS solver [Karimi and Tunçel, 2019, 2020]).

1.1 Conic interior point methods

Most conic solvers, such as CSDP [Borchers, 1999], CVXOPT [Andersen et al., 2011], ECOS [Serrano, 2015], MOSEK [MOSEK ApS, 2020b], and SDPA [Yamashita et al., 2003], implement primal-dual interior point methods (PDIPMs). Complexity analysis of PDIPMs, which relies on properties of *logarithmically homogeneous self-concordant barrier functions* (LHSCBs; defined in Nesterov and Nemirovskii [1994, Sections 2.3.1 and 2.3.3]), shows they require fewer iterations to converge but exhibit higher per-iteration cost compared to first order conic methods (see O'Donoghue et al. [2016] on SCS solver). Computational evidence accords with this result and demonstrates the superior numerical robustness of PDIPMs.

Historically, PDIPM solvers were based on efficient algorithms specialized for symmetric cones, in particular, the nonnegative, (rotated) second order, and positive semidefinite (PSD) cones. However, many useful non-symmetric conic constraints (such as $u \leq \log(w)$, representable with an exponential cone) are not exactly representable with symmetric cones. Early non-symmetric conic PDIPMs such as Nesterov et al. [1996], Nesterov [2012] had several disadvantages compared to the specialized symmetric methods, for example requiring a strictly feasible initial iterate, the solution of larger linear systems, and conjugate LHSCB oracles. To address these issues, Skajaa and Ye [2015] (henceforth referred to as SY) introduced a PDIPM that requires just a few oracles for the primal cone only: a fixed initial point in the cone's interior, a feasibility test (to determine whether a given point is in the cone's interior), and gradient and Hessian evaluations for an LHSCB.

1.2 Natural and extended formulations

Although advanced conic solvers currently recognize at most only a handful of *standard cones* (nonnegative, second order, rotated second order, positive semidefinite (PSD), and 3-dimensional exponential and power cones), these cones are sufficient for representing many problems of interest [Lubin et al., 2016, MOSEK ApS, 2020b]. Modeling tools such as disciplined convex programming (DCP) packages (see CVX [Grant and Boyd, 2014], CVXPY [Diamond and Boyd, 2016], and Convex.jl [Udell et al., 2014]) and MathOptInterface's bridges [Legat et al., 2020] are designed to facilitate transformations of convex problems into conic problems with standard cones, to enable access to powerful specialized conic solvers. However, for many problems of interest, a representation in terms of standard cones is not the most natural or efficient conic representation.

The process of transforming a general conic problem into a conic *extended formulation* (EF) that uses only standard cones often requires introducing many artificial variables, linear equalities, and/or higher-dimensional conic constraints. For example, in our density estimation example problem in Section 5.6, these dimensions are typically orders of magnitude larger for the EFs than for the NFs. By increasing the size and complexity of problem data, EFs can increase the computational cost of preprocessing/initialization and linear system solving at each iteration. If conic solvers could recognize a much larger class of *exotic* cones, they could directly solve simpler, smaller conic *natural formulations* (NFs).[1] We define an exotic cone as a proper cone for which we can implement a small set of tractable (i.e. fast, numerically stable, analytic) oracles for a logarithmically homogeneous self-concordant barrier for the cone or for its dual cone.

In the particular context of polynomial weighted sum-of-squares (SOS) optimization, Papp and Yildiz [2019] illustrate the potential advantages of using NFs with SOS cones (see Section 4.7) instead of PSD cone EFs, which are much larger. The authors describe tractable LHSCB oracles for dual SOS cones, noting that analytic oracles are not known for primal SOS cones. They show that their SOS NF-based approach has lower theoretical time and space complexity overall compared to a standard EF-based semidefinite programming method. After implementing SY in their MATLAB solver Alfonso [Papp and Yıldız, 2017, 2021], the authors observe improved solve times and scaleability from solving the NFs with Alfonso compared to solving the EFs with MOSEK.

To broaden the computational argument for NFs, in Section 4 we define a variety of exotic cone types (some of which required the development of new LHSCBs; see Coey et al. [2021d]) and describe general techniques for constructing equivalent EFs of NF constraints involving these cones. We analyze how these EF techniques necessarily increase formulation dimensions. We also observe that the EFs are often associated with larger values of the LHSCB *parameter* ν, which impacts the number of iterations $\mathcal{O}(\sqrt{\nu}\log(1/\varepsilon))$ needed in the worst case by an idealized algorithm such as SY to obtain a solution within ε tolerance [Nesterov and Todd, 1997]. However, most performance-oriented conic solver implementations do not directly implement idealized PDIPMs, so the practical impact of the parameter ν on average performance is unclear. Another potential advantage of using the NFs is that converting conic certificates from the space of the EF back into the more meaningful NF space can be complicated. Furthermore, the convergence conditions used by PDIPMs can provide numerical guarantees about conic certificates, but if EF certificates are converted to NF space, the NF certificates might lack such guarantees.

[1] We note that EFs can be beneficial for accelerating outer approximation algorithms for mixed-integer conic optimization, such as the method implemented in Pajarito solver [Coey et al., 2020]. However, folklore says that the EF for the second order cone likely slows down the conic solver, which is why Pajarito manages the EF in the MILP outer approximation model and only solves NFs for the conic subproblems.

1.3 Hypatia: a generic conic solver

We introduce our new open source generic conic PDIPM solver, Hypatia.[2] Hypatia is written in the Julia language [Bezanson et al., 2017] and is accessible through either a flexible, low-level native interface or the open-source modeling tools JuMP [Dunning et al., 2017] and Convex.jl [Udell et al., 2014]. Unlike Alfonso, Hypatia uses a primal-dual form (matching CVXOPT's *cone LP* form [Andersen et al., 2011]) that does not force the user to introduce slack variables, and allows linear operators to be represented with Julia's sparse, dense, or structured abstract matrix types (see Section 3). Hypatia already supports more than two dozen predefined exotic cone types, some of which have multiple variants (for example, real and complex flavors); these cones and associated LHSCBs are listed at Coey et al. [2021a,b].

A key feature of Hypatia is the generic cone interface, which allows defining new proper cones. The interface, like that of Alfonso, requires only the implementation of the few tractable cone oracles needed by SY. However, unlike Alfonso, defining a new cone in Hypatia makes both the cone and its dual cone simultaneously available for use in conic formulations (see Coey et al. [2021c] for details). For many cones of interest, tractable (i.e. fast, numerically stable, and analytic) oracles are only known for either the primal cone or the dual cone but not both. This means Hypatia is able to handle a broader class of conic formulations than SY and Alfonso, which require oracles specifically for all cones in the primal conic formulation. For example, in our portfolio rebalancing example NF in Section 5.1, we have both ℓ_1 norm cone and ℓ_∞ norm cone constraints; we are aware of an LHSCB with analytic oracles for the ℓ_∞ norm cone, but not for its dual cone - the ℓ_1 norm cone (see Section 4.1). Unlike Alfonso, Hypatia's cone interface allows optional specification of additional cone oracles (such as dual cone feasibility tests, inverse Hessians, and third order directional derivatives; see Coey et al. [2021c]), which can improve computational efficiency and numerical performance.

Hypatia's solver interface is also highly extensible. We provide several optional interior point search and stepping procedures, described in Coey et al. [2021c]. For example, while Alfonso alternates between prediction and correction steps, Hypatia's default interior point stepping procedure uses a combined directions method incorporating third-order LHSCB information, inspired by techniques of Andersen et al. [2011], Dahl and Andersen [2021], Domahidi et al. [2013]. Since the per-iteration bottleneck of PDIPMs such as Hypatia's algorithm tends to be solving the large structured linear system for search directions, Hypatia allows the user to choose from several predefined methods (including options for sparse or dense factorization-based solves or linear-operator-based iterative/indirect solves) or to implement their own formulation-specific procedure to leverage additional structure. Unlike Alfonso, Hypatia allows representing and solving conic problems in any real floating point type in Julia, hence it can solve conic problems to arbitrary precision using *BigFloat* types. We do not explore these advanced algorithmic features in this paper, and for our computational experiments, we use a fixed set of default algorithmic options.

1.4 Examples and computational experiments

In Section 5 we present a series of example problems from applications such as matrix completion, experiment design, and smooth density optimization. For these examples, we describe simple NFs using the exotic cones we define in Section 4. Some of these NFs are new and may be valuable to try in real-world applications. We randomly generate NF instances of a wide variety of sizes, construct equivalent EFs using the general EF techniques in Section 4, and observe that the EFs are significantly larger.

Our computational experiments demonstrate significant improvements in solve time and memory overhead

[2]Hypatia is available at github.com/chriscoey/Hypatia.jl under the MIT license; see Coey et al. [2021b] for documentation.

from solving the NFs with Hypatia compared to solving the EFs with Hypatia or MOSEK 9. Our experience also suggests that since EFs are often larger and more complex than NFs, they can be less convenient for the modeler, and noticeably slower and more memory-intensive to construct using JuMP or Hypatia's native interface. For many instances, we could build the NF efficiently, but we hit time or memory limits while constructing the EF.

2 Notation

For sets, cl denotes the closure and int denotes the interior. $\mathbb{R}$ is the scalar reals, $\mathbb{R}_{\geq}$ is the nonnegative reals, and $\mathbb{R}_{>} = \text{int}(\mathbb{R}_{\geq})$ is the positive reals, $\mathbb{R}_{\leq}$ is the nonpositive reals, and $\mathbb{R}_{<} = \text{int}(\mathbb{R}_{\leq})$ is the negative reals. The set of d-dimensional real vectors is $\mathbb{R}^d$, and the set of d_1-by-d_2-dimensional real matrices is $\mathbb{R}^{d_1 \times d_2}$. $\mathbb{S}^d$ is the set of symmetric matrices of side dimension d, $\mathbb{S}^d_{\succeq} \subset \mathbb{S}^d$ is the positive semidefinite matrices, and $\mathbb{S}^d_{\succ} = \text{int}\big(\mathbb{S}^d_{\succeq}\big)$ is the positive definite matrices. For some natural number d, $[\![d]\!]$ is the index set $\{1, 2, \ldots, d\}$.

If a, b, c, d are scalars, vectors, or matrices (of appropriate dimensions), the notation $\left[\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right]$ usually denotes concatenation into a matrix. For a vector or matrix A, the transpose is A'. $I(d)$ is the identity matrix in $\mathbb{R}^{d \times d}$. For dimensions implied by context, 0 may represent vectors or matrices of 0s, and e is a vector of 1s. Diag represents the diagonal matrix of a given vector, and diag represents the diagonal vector of a given square matrix. The inner product of vectors $u, w \in \mathbb{R}^d$ is $u'w = \sum_{i \in [\![d]\!]} u_i w_i$. log is the natural logarithm, $\|\cdot\|_p$ is the ℓ_p norm (for $p \geq 1$) of a vector, det is the determinant of a symmetric matrix, tr is the matrix trace, and $\sigma_i(\cdot)$ is the ith largest singular value of a matrix.

The operator vec maps $\mathbb{R}^{d_1 \times d_2}$ (matrices) to $\mathbb{R}^{d_1 d_2}$ (vectors) by stacking columns. The inverse operator mat_{d_1,d_2} maps $\mathbb{R}^{d_1 d_2}$ to $\mathbb{R}^{d_1 \times d_2}$. For symmetric matrices, vec maps $\mathbb{S}^d$ to $\mathbb{R}^{\text{sd}(d)}$, where we define $\text{sd}(d) \coloneqq d(d+1)/2$, by rescaling off-diagonal elements by $\sqrt{2}$ and stacking columns of the upper triangle. For example, for $S \in \mathbb{S}^3$ we have $\text{sd}(3) = 6$ and $\text{vec}(S) = \big(S_{1,1}, \sqrt{2}S_{1,2}, S_{2,2}, \sqrt{2}S_{1,3}, \sqrt{2}S_{2,3}, S_{3,3}\big) \in \mathbb{R}^{\text{sd}(3)}$. The inverse mapping mat from $\mathbb{R}^{\text{sd}(d)}$ to $\mathbb{S}^d$ is well-defined. The linear operators vec and mat preserve inner products, so $\text{vec}(S)' \text{vec}(Z) = \text{tr}(S'Z)$ for $S, Z \in \mathbb{R}^{d_1 \times d_2}$ or $S, Z \in \mathbb{S}^d$.

3 Conic duality and standard form

Let $\mathcal{K}$ be a proper cone in $\mathbb{R}^q$, i.e. a conic subset of $\mathbb{R}^q$ that is closed, convex, pointed, and full-dimensional (see Skajaa and Ye [2015]). We call $\mathcal{K}$ a primitive (or irreducible) cone if it cannot be written as a Cartesian product of two or more lower-dimensional cones. $\mathcal{K}^* \subset \mathbb{R}^q$ is the dual cone of $\mathcal{K}$:

$$\mathcal{K}^* \coloneqq \{z \in \mathbb{R}^q : s'z \geq 0, \forall s \in \mathcal{K}\}. \tag{1}$$

$\mathcal{K}^*$ is a primitive proper cone if and only if $\mathcal{K}$ is a primitive proper cone.

In Section 4, we introduce a subset of Hypatia's predefined primitive proper cones and their dual cones. We use these cones to formulate NFs and EFs for our applied example problems in Section 5. In this paper, we omit the LHSCBs for the cones in Section 4, but these can be found in Coey et al. [2021a], Kapelevich et al. [2021], Coey et al. [2021d]. We note in Section 4 that for many of these cones, computing conjugate barrier oracles requires optimization, which is slow and numerically fraught in our experience. Fortunately, like the algorithm by Skajaa and Ye [2015] (and its implementation in Alfonso), Hypatia does not use conjugate barrier oracles.

Hypatia's generic cone interface allows defining any proper cone $\mathcal{K}$ by specifying a small list of oracles: an initial interior point $t \in \text{int}(\mathcal{K})$, a feasibility test for $\text{int}(\mathcal{K})$, and gradients and Hessians of an LHSCB f for $\mathcal{K}$. Recall that Nesterov and Nemirovskii [1994, Sections 2.3.1 and 2.3.3] defines an LHSCB for a proper cone. The cone interface also allows optional specification of other oracles that can improve performance. Once defined, the cone and its dual cone may be used in any combination with other cones recognized by Hypatia to construct the Cartesian product cone $\mathcal{K}$ in the primal conic form (2) below.

Hypatia's primal conic form over variable $x \in \mathbb{R}^n$ is:

$$\inf_x \quad c'x : \tag{2a}$$
$$b - Ax = 0, \tag{2b}$$
$$h - Gx \in \mathcal{K}, \tag{2c}$$

where $c \in \mathbb{R}^n$, $b \in \mathbb{R}^p$, and $h \in \mathbb{R}^q$ are vectors, $A : \mathbb{R}^n \to \mathbb{R}^p$ and $G : \mathbb{R}^n \to \mathbb{R}^q$ are linear maps, and $\mathcal{K} \subset \mathbb{R}^q$ is a Cartesian product $\mathcal{K} = \mathcal{K}_1 \times \cdots \times \mathcal{K}_K$ of primitive proper cones. Henceforth we use n, p, q to denote the variable, equality, and conic constraint dimensions of a conic problem in the form (2). This primal form matches CVXOPT's form, though CVXOPT only recognizes symmetric cones [Vandenberghe, 2010]. Unlike the conic form used by Skajaa and Ye [2015] (and Alfonso and MOSEK 9), which recognizes conic constraints of the form $x \in \mathcal{K}$, our form does not require the user to introduce slack variables to represent a more general constraint $h - Gx \in \mathcal{K}$.

The corresponding conic dual problem over variable $y \in \mathbb{R}^p$ associated with (2b), and $z \in \mathbb{R}^q$ associated with (2c), is:

$$\sup_{y,z} \quad -b'y - h'z : \tag{3a}$$
$$c + A'y + G'z = 0, \tag{3b}$$
$$z \in \mathcal{K}^*, \tag{3c}$$

where (3b) is associated with primal variable $x \in \mathbb{R}^n$. Note $\mathcal{K}^* = \mathcal{K}_1^* \times \cdots \times \mathcal{K}_K^*$.

If neither the primal nor the dual is *ill-posed*,[3] there exists a simple conic certificate providing an easily verifiable proof of infeasibility of the primal (2) or dual (3) or optimality of a given primal-dual solution. A primal improving ray x is a feasible direction for the primal along which the primal objective improves (i.e. $c'x < 0$, $-Ax = 0$, $-Gx \in \mathcal{K}$), and hence it certifies dual infeasibility via the conic generalization of Farkas' lemma. Similarly, a dual improving ray (y, z) certifies primal infeasibility (i.e. $-b'y - h'z > 0$, $A'y + G'z = 0$, $z \in \mathcal{K}^*$). Finally, a complementary solution (x, y, z) satisfies the primal-dual feasibility conditions (2b) to (2c) and (3b) to (3c) and has equal and attained objective values $c'x = -b'y - h'z$, and hence certifies optimality of (x, y, z) via conic weak duality.

4 Cones and extended formulations

Recall that we define the standard cones as those recognized by MOSEK 9, listed below.

Nonnegative cone. The self-dual nonnegative cone is $\mathcal{K}_{\geq} = \mathcal{K}_{\geq}^* := \mathbb{R}_{\geq}$.

[3] Intuitively, according to MOSEK ApS [2020b, Section 7.2], a conic problem is ill-posed if a small perturbation of the problem data can change the feasibility status of the problem or cause arbitrarily large perturbations to the optimal solution. See Permenter et al. [2017] for more details.

Euclidean norm cone. The self-dual Euclidean norm cone (or second order cone) is the epigraph of the ℓ_2 norm:

$$\mathcal{K}_{\ell_2} = \mathcal{K}_{\ell_2}^* := \left\{(u, w) \in \mathbb{R}_{\geq} \times \mathbb{R}^d : u \geq \|w\|\right\}. \tag{4}$$

Euclidean norm-squared cone. The self-dual Euclidean norm-squared cone (or rotated second order cone) is the epigraph of the perspective function of g for $g(w) = \frac{1}{2}\|w\|^2$:

$$\mathcal{K}_{\text{sqr}} = \mathcal{K}_{\text{sqr}}^* := \left\{(u, v, w) \in \mathbb{R}^2_{\geq} \times \mathbb{R}^d : 2uv \geq \|w\|^2\right\}. \tag{5}$$

Positive semidefinite cone. The self-dual (vectorized) positive semidefinite (PSD) cone is:

$$\mathcal{K}_{\succeq} = \mathcal{K}_{\succeq}^* := \left\{w \in \mathbb{R}^{\text{sd}(d)} : \text{mat}(w) \in \mathbb{S}^d_{\succeq}\right\}. \tag{6}$$

3-dimensional exponential cone. The exponential cone in $\mathbb{R}^3$ is a special case of our logarithm cone $\mathcal{K}_{\log}$ defined in Section 4.5 (let $d = 1$ in (17a)), so any 3-dimensional $\mathcal{K}_{\log}$ constraint is an exponential cone constraint.

3-dimensional power cone. The power cone in $\mathbb{R}^3$ (defined in MOSEK ApS [2020b, Section 4.1]) is a special case of Hypatia's generalized power cone (see Coey et al. [2021a]). However, none of our example NFs or EFs in Section 5 need power cones, so we omit these definitions here.

In Sections 4.1 to 4.7, we define a subset of Hypatia's predefined exotic cones. For simplicity, we refer to a particular exotic cone constraint as an NF, and an equivalent reformulation of such a constraint in terms of only standard cones as an EF. We describe general techniques for constructing EFs for the types of exotic conic constraints we use in our example NFs in Section 5. In general, an NF constraint has the form $h - Gx \in \mathcal{K}$, but in this section we write $s \in \mathcal{K}$ for simplicity, since $s = h - Gx$ can be substituted into the EF description. An EF may use auxiliary variables, linear equalities, and/or conic constraints, which affect the dimensions n, p, and q (respectively) of the primal conic form (2). In Table 1, we compare the dimensions and LHSCB parameters (ν) associated with equivalent NF and EF constraints; as we mention in Section 1.2, these properties may affect the performance of PDIPMs.

The EFs we describe below follow best practices from DCP modeling tools such as Convex.jl [Udell et al., 2014] and descriptions such as Ben-Tal and Nemirovski [2001, chapter 4]. We use JuMP [Dunning et al., 2017] to build the NFs and EFs in Section 5, so we contributed several exotic cones and the EFs described in Sections 4.1 to 4.3 to MathOptInterface's bridges [Legat et al., 2020] to permit automated EF construction. MathOptInterface does not currently recognize $\mathcal{K}_{\log}$ (for $d > 1$), $\mathcal{K}_{\text{SOS}}$, and $\mathcal{K}_{\text{matSOS}}$ (or their dual cones), so we construct the EFs in Sections 4.5 and 4.7 manually using JuMP. For some EFs with auxiliary variables and equalities, it is possible to perform eliminations to reduce dimensions slightly, but this can impact the sparsity of problem data (note that in our experiments in Section 5, both Hypatia and MOSEK perform preprocessing).

4.1 Infinity norm cone

The ℓ_∞ norm cone is the epigraph of ℓ_∞, and its dual cone is the ℓ_1 norm cone:

$$\mathcal{K}_{\ell_\infty} := \left\{(u, w) \in \mathbb{R}_{\geq} \times \mathbb{R}^d : u \geq \|w\|_\infty\right\}, \tag{7a}$$

$$\mathcal{K}_{\ell_\infty}^* := \left\{(u, w) \in \mathbb{R}_{\geq} \times \mathbb{R}^d : u \geq \|w\|_1\right\}. \tag{7b}$$

Table 1: Properties of NFs and EFs for the exotic conic constraints in Sections 4.1 to 4.7. q and ν are the dimension and LHSCB parameter for the NF cone, and $\bar{q}$ and $\bar{\nu}$ are the corresponding values for the EF Cartesian product cone. $\bar{n}$ and $\bar{p}$ are the EF auxiliary variable and equality dimensions. Note $\operatorname{sd}(k)$ is $\mathcal{O}(k^2)$.

NF	q	ν	EF	$\bar{q}$	$\bar{\nu}$	$\bar{n}$	$\bar{p}$
$\mathcal{K}_{\ell_\infty}$	$1+d$	$1+d$	$\mathcal{K}_\geq$	$2d$	$2d$	0	0
$\mathcal{K}^*_{\ell_\infty}$	$1+d$	$1+d$	$\mathcal{K}_\geq$	$1+2d$	$1+2d$	$2d$	d
$\mathcal{K}_{\ell_{\text{spec}}}$	$1+rs$	$1+r$	$\mathcal{K}_\geq$	$\operatorname{sd}(r+s)$	$r+s$	0	0
$\mathcal{K}^*_{\ell_{\text{spec}}}$	$1+rs$	$1+r$	$\mathcal{K}_\geq, \mathcal{K}_\geq$	$1+\operatorname{sd}(r+s)$	$1+r+s$	$\operatorname{sd}(r)+\operatorname{sd}(s)$	0
$\mathcal{K}_{\text{geo}}$	$1+d$	$1+d$	$\mathcal{K}_\geq, \mathcal{K}_{\log}$	$2+3d$	$2+3d$	$1+d$	0
$\mathcal{K}_{\text{rtdet}}$	$1+\operatorname{sd}(d)$	$1+d$	$\mathcal{K}_\geq, \mathcal{K}_{\log}, \mathcal{K}_\geq$	$2+3d+\operatorname{sd}(2d)$	$2+5d$	$1+d+\operatorname{sd}(d)$	0
$\mathcal{K}_{\log}$	$2+d$	$2+d$	$\mathcal{K}_\geq, \mathcal{K}_{\log}$	$1+3d$	$1+3d$	d	0
$\mathcal{K}_{\text{logdet}}$	$2+\operatorname{sd}(d)$	$2+d$	$\mathcal{K}_\geq, \mathcal{K}_{\log}, \mathcal{K}_\geq$	$1+3d+\operatorname{sd}(2d)$	$1+5d$	$1+d+\operatorname{sd}(d)$	0
$\mathcal{K}_{\text{SOS}}$	d	$\sum_l t_l$	$\mathcal{K}_\geq$	$\sum_l \operatorname{sd}(t_l)$	ν	$\bar{q}$	d
$\mathcal{K}^*_{\text{SOS}}$	d	$\sum_l s_l$	$\mathcal{K}_\geq$	$\sum_l \operatorname{sd}(s_l)$	ν	0	0
$\mathcal{K}_{\text{matSOS}}$	$\operatorname{sd}(t)d$	$t\sum_l s_l$	$\mathcal{K}_\geq$	$\sum_l \operatorname{sd}(ts_l)$	ν	$\bar{q}$	q

For $\mathcal{K}_{\ell_\infty}$, we use the LHSCB from Güler [1996, Section 7.5]. We are not aware of an LHSCB for $\mathcal{K}^*_{\ell_\infty}$ with similarly efficient oracles.

Our examples in Sections 5.1 and 5.4 use the following NF (left) and EF (right):

$$(u, w) \in \mathcal{K}_{\ell_\infty} \subset \mathbb{R}^{1+d} \quad \Leftrightarrow \quad (ue - w, ue + w) \in (\mathcal{K}_\geq)^{2d}, \tag{8}$$

and similarly, Section 5.1 uses:

$$(u, w) \in \mathcal{K}^*_{\ell_\infty} \subset \mathbb{R}^{1+d} \quad \Leftrightarrow \quad \exists \theta \in (\mathcal{K}_\geq)^d, \exists \lambda \in (\mathcal{K}_\geq)^d, w = \theta - \lambda, u - e'(\theta + \lambda) \in \mathcal{K}_\geq. \tag{9}$$

4.2 Spectral norm cone

The spectral norm cone is the epigraph of the matrix spectral norm, and its dual cone is the epigraph of the matrix nuclear norm:

$$\mathcal{K}_{\ell_{\text{spec}}(r,s)} \coloneqq \left\{(u, w) \in \mathbb{R}_\geq \times \mathbb{R}^{rs} : u \geq \sigma_1(W)\right\}, \tag{10a}$$

$$\mathcal{K}^*_{\ell_{\text{spec}}(r,s)} \coloneqq \left\{(u, w) \in \mathbb{R}_\geq \times \mathbb{R}^{rs} : u \geq \textstyle\sum_{i \in [\![r]\!]} \sigma_i(W)\right\}, \tag{10b}$$

where $W \coloneqq \operatorname{mat}_{r,s}(w) \in \mathbb{R}^{r \times s}$ and $r \leq s$ (this is nonrestrictive since the singular values are the same for W and W'). For $\mathcal{K}_{\ell_{\text{spec}}}$ we use the LHSCB from Nesterov and Nemirovskii [1994]. We are not aware of an LHSCB for $\mathcal{K}^*_{\ell_{\text{spec}}}$ with similarly efficient oracles.

Section 5.2 uses the EF from Ben-Tal and Nemirovski [2001, Section 4.2]:

$$(u, w) \in \mathcal{K}_{\ell_{\text{spec}}(r,s)} \subset \mathbb{R}^{1+rs} \quad \Leftrightarrow \quad \begin{bmatrix} uI(r) & W \\ W' & uI(s) \end{bmatrix} \in \mathbb{S}^{r+s}_\succeq. \tag{11}$$

Section 5.3 uses the EF from Recht et al. [2010]:

$$\begin{aligned} (u, w) \in \mathcal{K}^*_{\ell_{\text{spec}}(r,s)} \subset \mathbb{R}^{1+rs} \quad \Leftrightarrow \quad & \exists \theta \in \mathbb{R}^{\operatorname{sd}(r)}, \exists \lambda \in \mathbb{R}^{\operatorname{sd}(s)}, \begin{bmatrix} \Theta & W \\ W' & \Lambda \end{bmatrix} \in \mathbb{S}^{r+s}_\succeq, \\ & u - (\operatorname{tr}(\Theta) + \operatorname{tr}(\Lambda))/2 \in \mathcal{K}_\geq, \end{aligned} \tag{12}$$

where $\Theta \coloneqq \mathrm{mat}(\theta) \in \mathbb{S}^r, \Lambda \coloneqq \mathrm{mat}(\lambda) \in \mathbb{S}^s$.

4.3 Geometric mean cone

The geometric mean cone is the hypograph of the geometric mean function:

$$\mathcal{K}_{\text{geo}} \coloneqq \left\{(u, w) \in \mathbb{R} \times \mathbb{R}^d_{\geq} : u \leq \textstyle\prod_{i \in [\![d]\!]} w_i^{1/d}\right\}, \tag{13a}$$

$$\mathcal{K}^*_{\text{geo}} \coloneqq \left\{(u, w) \in \mathbb{R}_{\leq} \times \mathbb{R}^d_{\geq} : u \geq -d \textstyle\prod_{i \in [\![d]\!]} w_i^{1/d}\right\}. \tag{13b}$$

For $\mathcal{K}_{\text{geo}}$ we use the LHSCB from Nesterov [2006].

The example in Section 5.2 uses an EF for $\mathcal{K}_{\text{geo}}$, and the root-determinant variant of the example in Section 5.4 uses an EF for $\mathcal{K}_{\text{geo}}$ indirectly through a $\mathcal{K}_{\text{rtdet}}$ EF (see Section 4.4). We are aware of three EFs for $\mathcal{K}_{\text{geo}}$: a rotated second order cone EF (*EF-sec*) from Ben-Tal and Nemirovski [2001, Section 3.3.1], a power cone EF (*EF-pow*) from MOSEK ApS [2020b], and an exponential cone EF (*EF-exp*). We contributed EF-exp to MathOptInterface as a combination of two bridges (geometric mean cone to relative entropy cone to exponential cones):

$$(u, w) \in \mathcal{K}_{\text{geo}} \subset \mathbb{R}^{1+d} \quad \Leftrightarrow \quad \begin{array}{l} \exists \theta \in \mathcal{K}_{\geq}, \exists \lambda \in \mathbb{R}^d, e'\lambda \in \mathcal{K}_{\geq}, \\ (\lambda_i, u + \theta, w_i) \in \mathcal{K}_{\log}, \forall i \in [\![d]\!]. \end{array} \tag{14}$$

EF-pow is not currently available through MathOptInterface bridges, and it has a very similar size and structure to EF-exp, so we do not describe or test it. EF-sec uses multiple levels of variables and 3-dimensional $\mathcal{K}_{\text{sqr}}$ constraints and is complex to describe, so we refer the reader to Ben-Tal and Nemirovski [2001, Section 3.3.1]. In our empirical comparisons in Sections 5.2 and 5.4, EF-sec typically has larger variable and conic constraint dimensions but smaller barrier parameter than EF-exp.

4.4 Root-determinant cone

The root-determinant cone is the hypograph of the root-determinant function:

$$\mathcal{K}_{\text{rtdet}} \coloneqq \left\{(u, w) \in \mathbb{R}^{1+\text{sd}(d)} : W \in \mathbb{S}^d_{\succeq}, u \leq (\det(W))^{1/d}\right\}, \tag{15a}$$

$$\mathcal{K}^*_{\text{rtdet}} \coloneqq \left\{(u, w) \in \mathbb{R}_{\leq} \times \mathbb{R}^{\text{sd}(d)} : W \in \mathbb{S}^d_{\succeq}, u \geq -d(\det(W))^{1/d}\right\}, \tag{15b}$$

where $W \coloneqq \mathrm{mat}(w)$. For $\mathcal{K}_{\text{rtdet}}$ we propose an LHSCB with efficient oracles in Coey et al. [2021d].

Section 5.4 uses the EF from Ben-Tal and Nemirovski [2001, Section 4.2]:

$$(u, w) \in \mathcal{K}_{\text{rtdet}} \subset \mathbb{R}^{1+\text{sd}(d)} \quad \Leftrightarrow \quad \begin{array}{l} \exists \theta \in \mathbb{R}^{\text{sd}(d)}, (u, \mathrm{diag}(\Theta)) \in \mathcal{K}_{\text{geo}}, \\ \begin{bmatrix} W & \Theta \\ \Theta' & \mathrm{Diag}(\mathrm{diag}(\Theta)) \end{bmatrix} \in \mathbb{S}^{2d}_{\succeq}, \end{array} \tag{16}$$

where $\Theta \coloneqq \mathrm{mat}(\theta) \in \mathbb{S}^d$, and the $\mathcal{K}_{\text{geo}}$ constraint is itself replaced with one of the geometric mean cone EFs described in Section 4.3.

4.5 Logarithm cone

The logarithm cone is the hypograph of the perspective function of a sum of natural log functions:

$$\mathcal{K}_{\log} \coloneqq \mathrm{cl}\left\{(u, v, w) \in \mathbb{R} \times \mathbb{R}^{1+d}_{>} : u \leq \textstyle\sum_{i \in [\![d]\!]} v \log(w_i/v)\right\}, \tag{17a}$$

$$\mathcal{K}^*_{\log} \coloneqq \mathrm{cl}\left\{(u, v, w) \in \mathbb{R}_{<} \times \mathbb{R} \times \mathbb{R}^d_{>} : v \geq \textstyle\sum_{i \in [\![d]\!]} u(\log(-w_i/u) + 1)\right\}. \tag{17b}$$

For $\mathcal{K}_{\log}$ we propose an LHSCB with efficient oracles in Coey et al. [2021d].

Section 5.6 uses the EF (when $d > 1$):

$$(u, v, w) \in \mathcal{K}_{\log} \subset \mathbb{R}^{2+d} \quad \Leftrightarrow \quad \exists \theta \in \mathbb{R}^d, e'\theta - u \in \mathcal{K}_{\geq}, (\theta_i, 1, w_i) \in \mathcal{K}_{\log}, \forall i \in [\![d]\!]. \tag{18}$$

4.6 Log-determinant cone

The log-determinant cone is the hypograph of the perspective function of the log-determinant function:

$$\mathcal{K}_{\text{logdet}} := \text{cl}\big\{(u, v, w) \in \mathbb{R} \times \mathbb{R}_> \times \mathbb{R}^{\text{sd}(d)} : W \in \mathbb{S}^d_\succ, u \leq v \operatorname{logdet}(W/v)\big\}, \tag{19a}$$
$$\mathcal{K}^*_{\text{logdet}} := \text{cl}\big\{(u, v, w) \in \mathbb{R}_< \times \mathbb{R}^{1+\text{sd}(d)} : W \in \mathbb{S}^d_\succ, v \geq u(\operatorname{logdet}(-W/u) + d)\big\}, \tag{19b}$$

where $W := \text{mat}(w)$. For $\mathcal{K}_{\text{logdet}}$ we propose an LHSCB with efficient oracles in Coey et al. [2021d].

Section 5.4 adapts the root-determinant cone EF (16):

$$(u, v, w) \in \mathcal{K}_{\text{logdet}} \subset \mathbb{R}^{2+\text{sd}(d)} \quad \Leftrightarrow \quad \begin{array}{c} \exists \theta \in \mathbb{R}^{\text{sd}(d)}, (u, v, \operatorname{diag}(\Theta)) \in \mathcal{K}_{\log}, \\ \begin{bmatrix} W & \Theta \\ \Theta' & \operatorname{Diag}(\operatorname{diag}(\Theta)) \end{bmatrix} \in \mathbb{S}^{2d}_\succeq, \end{array} \tag{20}$$

where $\Theta := \text{mat}(\theta) \in \mathbb{S}^d$, and the $\mathcal{K}_{\log}$ constraint is itself replaced with the logarithm cone EF described in Section 4.5.

4.7 Polynomial weighted SOS scalar and matrix cones

Given a collection of matrices $P_l \in \mathbb{R}^{d \times s_l}, \forall l \in [\![r]\!]$ derived from basis polynomials evaluated at d interpolation points as in Papp and Yildiz [2019], the interpolant basis polynomial weighted SOS cone is:

$$\mathcal{K}_{\text{SOS}(P)} := \big\{w \in \mathbb{R}^d : \exists \Theta_l \in \mathbb{S}^{s_l}_\succeq, \forall l \in [\![r]\!], w = \textstyle\sum_{l \in [\![r]\!]} \operatorname{diag}\big(P_l \Theta_l P_l'\big)\big\}, \tag{21a}$$
$$\mathcal{K}^*_{\text{SOS}(P)} := \big\{w \in \mathbb{R}^d : P_l' \operatorname{Diag}(w) P_l \in \mathbb{S}^{s_l}_\succeq, \forall l \in [\![r]\!]\big\}. \tag{21b}$$

These cones are useful for polynomial and moment modeling; for example, a point in $\mathcal{K}_{\text{SOS}(P)}$ corresponds to a polynomial that is pointwise nonnegative on a semialgebraic domain defined by P.

Given a side dimension t of a symmetric matrix of polynomials (for simplicity, all using the same interpolant basis), and $P_l \in \mathbb{R}^{d \times s_l}, \forall l \in [\![r]\!]$ defined as for $\mathcal{K}_{\text{SOS}(P)}$ in Section 4.7, the interpolant basis polynomial weighted SOS matrix cone is:

$$\mathcal{K}_{\text{matSOS}(P)} := \left\{ \begin{array}{l} w \in \mathbb{R}^{\text{sd}(t)d} : \exists \Theta_l \in \mathbb{S}^{s_l t}_\succeq, \forall l \in [\![r]\!], \\ W_{i,j,:} = \sum_{l \in [\![r]\!]} \operatorname{diag}\big(P_l (\Theta_l)_{i,j} P_l'\big), \forall i, j \in [\![t]\!] : i \geq j \end{array} \right\}, \tag{22a}$$
$$\mathcal{K}^*_{\text{matSOS}(P)} := \big\{w \in \mathbb{R}^{\text{sd}(t)d} : \big[P_l' \operatorname{Diag}(W_{i,j,:}) P_l\big]_{i,j \in [\![t]\!]} \in \mathbb{S}^{s_l t}_\succeq, \forall l \in [\![r]\!]\big\}, \tag{22b}$$

where $W_{i,j,:} \in \mathbb{R}^d$ is the contiguous slice of w (scaled to account for symmetry) corresponding to the interpolant basis values in the (i,j)th position of the symmetric matrix, $(S)_{i,j}$ is the (i,j)th block in a symmetric matrix S with square blocks of equal dimensions, and $[g(W_{i,j,:})]_{i,j \in [\![t]\!]}$ is the symmetric matrix with square matrix $g(W_{i,j,:})$ in the (i,j)th block. A point in $\mathcal{K}_{\text{matSOS}(P)}$ corresponds to a polynomial matrix that is pointwise PSD on a semialgebraic domain defined by P. See Kapelevich et al. [2021] for more details.

Papp and Yildiz [2019] describe an LHSCB with efficient oracles for $\mathcal{K}^*_{\mathrm{SOS}(P)}$, but they state that one is not known for $\mathcal{K}_{\mathrm{SOS}(P)}$. We are not aware of a useful LHSCB for $\mathcal{K}_{\mathrm{matSOS}(P)}$; indeed, for $t = 1$, $\mathcal{K}_{\mathrm{matSOS}(P)}$ reduces to $\mathcal{K}_{\mathrm{SOS}(P)}$. Noting that (22b) implicitly constrains a linear function of w to a Cartesian product of PSD cones, we can use Nesterov and Nemirovskii [1994, Propositions 5.1.1 and 5.1.3] (with the $-\log\det$ LHSCB for $\mathcal{K}_{\succeq}$) to derive an LHSCB with efficient oracles for $\mathcal{K}^*_{\mathrm{matSOS}(P)}$. This LHSCB reduces to the $\mathcal{K}_{\mathrm{SOS}(P)}$ LHSCB for $t = 1$.

Our examples in Sections 5.5 to 5.7 use the EFs implicit in the definitions of $\mathcal{K}^*_{\mathrm{SOS}}$, $\mathcal{K}_{\mathrm{SOS}}$, and $\mathcal{K}_{\mathrm{matSOS}}$ in (21a), (21b) and (22a). These EFs each use r $\mathcal{K}_{\succeq}$ cones.

5 Numerical examples

In Sections 5.1 to 5.7, we present example problems with NFs using some of Hypatia's predefined cones and EFs constructed using the techniques from Section 4. For each example problem, we generate random instances of a wide variety of sizes, and we observe larger dimensions and often larger barrier parameters for EFs compared to NFs. In Tables 2 to 10, ν and n, p, q refer to the NF barrier parameter and primal variable, linear equality, and cone inequality dimensions (in our general conic form (2)), and $\bar{\nu}$, $\bar{n}$, $\bar{p}$, $\bar{q}$ refer to the corresponding EF values. For three solver/formulation combinations - Hypatia with NF (*Hypatia-NF*), Hypatia with EF (*Hypatia-EF*), and MOSEK with EF (*MOSEK-EF*) - we compare termination statuses, iteration counts, and solve times in seconds (columns *st*, *it*, and *time*) in Tables 2 to 10 and Figure 1. In Sections 5.2 and 5.4 we depend on a geometric mean cone EF, so we compare the *EF-exp* and *EF-sec* formulations from Section 4.3. Note that all of our instances are primal-dual feasible, so we expect solvers to return optimality certificates. Compared to Hypatia-EF and MOSEK-EF, Hypatia-NF generally converges faster and more reliably, and solves larger instances within time and memory limits.

We perform all instance generation, computational experiments, and results analysis with Ubuntu 21.04, Julia 1.7, and Hypatia 0.5.0, on dedicated hardware with an AMD Ryzen 9 3950X 16-core processor (32 threads) and 128GB of RAM. We limit each solver to using 16 threads. We use JuMP 0.21.5 and MathOptInterface 0.9.18 to build all instances. We use MOSEK 9 through MosekTools.jl 0.9.4 (which is maintained in part by MOSEK).[4] MOSEK uses its conic interior point method for all solves. We note that MOSEK heuristically determines whether it is more efficient to solve the primal or dual of an instance [MOSEK ApS, 2020a, Section 13.1]; Hypatia does not do this. We do not disable any MOSEK features. Hypatia uses one particular default algorithmic implementation that we describe broadly in Coey et al. [2021c] (the *combined directions method*); this is not the most efficient method for all instances, but Hypatia does not currently have heuristics for choosing which stepper or linear system solver procedure to use, for example. Simple scripts and instructions for reproducing all results are available in Hypatia's benchmarks/natvsext folder. A CSV file containing raw results is available at the Hypatia wiki page.

Hypatia and MOSEK use similar convergence criteria (see MOSEK ApS [2020a, Section 13.3.2]), and we set their feasibility and optimality gap tolerances to 10^{-7}. In the solver statistics tables, asterisks indicate missing data, and we use the following codes for the termination status (st) columns:

[4] We note that MOSEK 9's primal conic form only recognizes conic constraints of the form $x \in \mathcal{K}$ [MOSEK ApS, 2020b, Section 8], whereas Hypatia accepts the more general affine form $h - Gx \in \mathcal{K}$ (see (2c)). MathOptInterface recognizes both *VectorOfVariables* form $x \in \mathcal{K}$ and *VectorAffineFunction* form $h - Gx \in \mathcal{K}$. Since JuMP and MathOptInterface (including bridges) use the $x \in \mathcal{K}$ form whenever possible, unnecessary high dimensional slack variables are not introduced when instances in Hypatia's general conic form (2) are converted into MOSEK 9's form.

co - the solver claims it has an approximate optimality certificate,

tl - the solver stops itself due to a solve time limit of 1800 seconds, or the solve run is killed because it takes at least 1.2×1800 seconds,

rl - the solve is terminated because insufficient RAM is available,

sp - the solver reports *slow progress* during iterations,

er - the solver encounters a numerical error,

m - the model cannot be constructed with JuMP due to insufficient RAM or a model generation time limit of 1.2×1800 seconds (EFs tend to be slower and more memory-intensive to construct than NFs, so EF columns often have missing data),

sk - we skip the solve run because a smaller instance has a tl or rl status, or we skip model generation because a smaller instance has an m status.

For each solve run that yields a primal-dual point (x, y, z, s) (see Section 3; $s \in \mathcal{K}$ and $z \in \mathcal{K}^*$ are the solver's primal and dual cone interior points at termination), we compute:

$$\epsilon := \max\left\{ \frac{\|A'y + G'z + c\|_\infty}{1 + \|c\|_\infty}, \frac{\|-Ax + b\|_\infty}{1 + \|b\|_\infty}, \frac{\|-Gx + h - s\|_\infty}{1 + \|h\|_\infty}, \frac{|c'x + b'y + h'z|}{1 + |b'y + h'z|} \right\}, \tag{23}$$

and if $\epsilon < 10^{-5}$, we underline the corresponding status code (e.g. co, tl) to indicate that the solution approximately satisfies the optimality certificate conditions from Section 3. In our solve time plots in Figure 1, we only plot solve runs with underlined status codes. Finally, for each instance and each pair of corresponding solve runs with co status codes, we compute the relative difference of the primal objective values g_1 and g_2 as $\tilde{\epsilon} := |g_1 - g_2|/(1 + \max(|g_1|, |g_2|))$. We note $\tilde{\epsilon} < 10^{-5}$ for most instances and pairs of solvers, and $\tilde{\epsilon} < 10^{-3}$ in all cases.

5.1 Portfolio rebalancing

Suppose there are k possible investments with expected returns $g \in \mathbb{R}^k_>$ and covariance matrix $\Sigma \in \mathbb{S}^k_\succ$. We let $\rho \in [-1, 1]^k$ be the investment variable, which must also satisfy side constraints $F\rho = 0$, where $F \in \mathbb{R}^{l \times k}$. We formulate a risk-constrained portfolio rebalancing optimization problem as:

$$\begin{aligned} \max_{\rho \in \mathbb{R}^k} \quad & g'\rho : && \text{(24a)} \\ & e'\rho = 0, && \text{(24b)} \\ & F\rho = 0, && \text{(24c)} \\ & (1, \rho) \in \mathcal{K}_{\ell_\infty}, && \text{(24d)} \\ & (\gamma, \Sigma^{1/2}\rho) \in \mathcal{K}^*_{\ell_\infty}. && \text{(24e)} \end{aligned}$$

Note (24d) expresses $\rho \in [-1, 1]^k$ and (24e) is a risk constraint. The EFs for (24d) and (24e) follow (8) and (9). Note the EF is a standard linear program.

To build random instances of (24), we generate g with independent uniform positive entries, and $\Sigma^{1/2}$ and F with independent Gaussian entries, for $l = k/2$ and various values of k. We use $\Sigma^{1/2}$ to compute reasonable values for the risk parameter $\gamma > 0$. Our results are summarized in Table 2 and Figure 1a. Note that $\nu = q = 2k + 2$, $\bar{\nu} = \bar{q} = 4k + 1$, $n = k$, $\bar{n} = 2k$, $p = \bar{p} = k/2 + 1$. The variable and conic constraint dimensions of the EFs are approximately double those of the NFs. Hypatia-NF exhibits the fastest solve

times and solves much larger instances than Hypatia-EF and MOSEK-EF. MOSEK requires notably fewer PDIPM iterations than Hypatia.

Table 2: Portfolio rebalancing solver statistics.

	Hypatia-NF			Hypatia-EF			MOSEK-EF		
k	st	it	time	st	it	time	st	it	time
1000	co	31	0.6	co	25	2.7	co	9	1.7
2000	co	36	2.9	co	28	16.	co	10	7.0
4000	co	45	20.	co	29	92.	co	10	34.
6000	co	49	60.	co	34	292.	co	10	83.
8000	co	51	131.	co	33	615.	co	10	160.
10000	co	55	244.	co	36	1192.	co	12	305.
12000	co	62	421.	tl	32	1805.	co	10	433.
14000	co	61	624.	sk	*	*	rl	*	*
16000	co	63	924.	sk	*	*	sk	*	*
18000	co	64	1327.	sk	*	*	sk	*	*
20000	co	66	1810.	sk	*	*	sk	*	*

5.2 Matrix completion

Suppose there exists a matrix $F \in \mathbb{R}^{k\times l}$ and we know the entries $(F_{i,j})_{(i,j)\in\mathcal{S}}$ in the sparsity pattern $\mathcal{S}$. In the matrix completion problem, we seek to estimate the missing components $(F_{i,j})_{(i,j)\notin\mathcal{S}}$. We modify the formulation in Agrawal et al. [2019, Section 4.3] by replacing the spectral radius in the objective function with the spectral norm (allowing rectangular matrices) and using a convex relaxation of the geometric mean equality constraint:

$$\min_{\rho\in\mathbb{R}, X\in\mathbb{R}^{k\times l}} \quad \rho : \tag{25a}$$
$$X_{i,j} = F_{i,j} \quad \forall (i,j) \in \mathcal{S}, \tag{25b}$$
$$(\rho, \operatorname{vec}(X)) \in \mathcal{K}_{\ell_{\text{spec}}(k,l)}, \tag{25c}$$
$$\left(1, (X_{i,j})_{(i,j)\notin\mathcal{S}}\right) \in \mathcal{K}_{\text{geo}}. \tag{25d}$$

The EF for (25c) follows (11), and for (25d) we compare EF-exp and EF-sec (see Section 4.3).

To build random instances of (25), we generate sparse matrices F with independent Gaussian nonzero entries, for various values of k, column-to-row ratios $m \in \{10, 20\}$, and $l = mk$. Our results are summarized in Tables 3 and 4 and Figure 1b. Note we only plot EF-sec results for Hypatia-EF and MOSEK-EF, as MOSEK performs better with EF-sec (which only uses symmetric cones) than with EF-exp, though Hypatia exhibits the opposite trend. Hypatia-NF is much faster and solves more instances than the Hypatia-EFs and MOSEK-EFs.

5.3 Multi-response regression

In the multi-response linear regression problem, we seek to estimate a coefficient matrix $F \in \mathbb{R}^{m\times l}$ from a design matrix $X \in \mathbb{R}^{l\times k}$ and response matrix $Y \in \mathbb{R}^{m\times k}$. We use a similar formulation to the one proposed

Table 3: Matrix completion formulation statistics. Note $p = \bar{p} = |S|$.

		NF				EF-exp			EF-sec		
m	k	ν	n	p	q	$\bar{\nu}$	$\bar{n}$	$\bar{q}$	$\bar{\nu}$	$\bar{n}$	$\bar{q}$
10	5	57	251	200	302	207	302	1692	182	314	1730
	10	218	1001	794	1208	730	1208	6725	621	1256	6871
	15	472	2251	1795	2707	1532	2707	15062	1188	2762	15229
	20	846	4001	3176	4826	2694	4826	26784	2267	5024	27380
	25	1299	6251	4978	7524	4093	7524	41768	4370	8298	44092
	30	1858	9001	7174	10828	5810	10828	60095	4425	11048	60757
	35	2477	12251	9810	14692	7707	14692	81627	8576	16346	86591
	40	3256	16001	12786	19216	10084	19216	106664	8631	20096	109306
	45	4142	20251	16155	24347	12782	24347	135047	8686	24346	135046
20	5	114	501	393	609	428	609	5888	360	628	5947
	10	418	2001	1594	2408	1430	2408	23375	1233	2512	23689
	15	933	4501	3584	5418	3065	5418	52520	2362	5524	52840
	20	1663	8001	6359	9643	5345	9643	93335	4515	10048	94552
	25	2513	12501	10014	14988	7985	14988	145535	8716	16596	150361
	30	3643	18001	14389	21613	11465	21613	209600	8821	22096	211051

Table 4: Matrix completion solver statistics.

		NF			EF-exp						EF-sec					
		Hypatia			Hypatia			MOSEK			Hypatia			MOSEK		
m	k	st	it	time	st	it	time	st	it	time	st	it	time	st	it	time
10	5	co	14	0.0	co	19	0.1	co	15	0.9	co	18	0.2	co	11	0.7
	10	co	19	0.4	co	34	2.2	co	20	20.	co	30	2.3	co	10	11.
	15	co	23	2.6	co	42	16.	co	21	120.	co	41	18.	co	9	58.
	20	co	26	14.	co	52	70.	co	24	524.	co	47	78.	co	11	251.
	25	co	30	52.	co	59	225.	co	26	1758.	co	69	387.	co	11	770.
	30	co	34	166.	co	61	556.	tl	11	1817.	co	55	587.	co	10	1712.
	35	co	39	402.	co	61	1228.	sk	*	*	tl	61	1817.	rl	*	*
	40	co	48	949.	tl	34	1820.	sk	*	*	sk	*	*	sk	*	*
	45	co	47	1806.	sk	*	*	sk	*	*	sk	*	*	sk	*	*
20	5	co	15	0.1	co	29	0.8	co	17	14.	co	26	0.9	co	8	7.6
	10	co	22	2.2	co	48	25.	co	25	448.	co	45	27.	co	10	203.
	15	co	30	24.	co	59	179.	tl	14	1871.	co	52	176.	co	10	1375.
	20	co	33	119.	co	71	786.	sk	*	*	co	70	924.	rl	*	*
	25	co	41	448.	tl	47	1822.	sk	*	*	tl	26	1804.	sk	*	*
	30	co	52	1305.	sk	*	*	sk	*	*	sk	*	*	sk	*	*

in Yang et al. [2016], with nuclear norm loss and ℓ_2 norm regularization:

$$\min_{\rho\in\mathbb{R},\mu\in\mathbb{R},F\in\mathbb{R}^{m\times l}} \quad \rho + \gamma\mu : \tag{26a}$$

$$(\rho, \mathrm{vec}(Y - FX)) \in \mathcal{K}^*_{\ell_{\mathrm{spec}}(m,k)}, \tag{26b}$$

$$(\mu, \mathrm{vec}(F)) \in \mathcal{K}_{\ell_2}. \tag{26c}$$

The EF for NF constraint (26b) follows (12).

To build random instances of (26), we generate random X and Y with independent Gaussian entries, for various values of k with $l = m \in \{15, 30\}$, and we use regularization parameter $\gamma = 0.1$. Our results are summarized in Table 5 and Figure 1c. Note that $\nu = 3 + m$, $\bar{\nu} = \nu + k$, $n = 2 + m^2$, $p = \bar{p} = 0$, $\bar{q} = \bar{n} + mk$. The variable dimensions for the NFs only depend on k and are much smaller than those of the EFs. The EFs also have much larger conic constraint dimensions. Hypatia-NF exhibits faster solve times than Hypatia-EF and MOSEK-EF. Hypatia-NF solves much larger instances and takes a fairly consistent number of iterations.

Table 5: Multi-response regression formulation and solver statistics.

		form. stats.		Hypatia-NF			Hypatia-EF			MOSEK-EF		
m	k	$\bar{n}$	q	st	it	time	st	it	time	st	it	time
15	50	1622	977	co	11	0.1	co	12	1.1	co	4	0.6
	100	5397	1727	co	10	0.5	co	12	17.	co	5	6.7
	150	11672	2477	co	10	1.2	co	13	98.	co	5	36.
	250	31722	3977	co	10	3.3	co	14	1331.	co	5	308.
	500	125597	7727	co	10	17.	m	*	*	tl	*	*
	1000	*	15227	co	13	129.	sk	*	*	sk	*	*
	1500	*	22727	co	10	209.	sk	*	*	sk	*	*
	2000	*	30227	co	9	395.	sk	*	*	sk	*	*
	2500	*	37727	co	11	949.	sk	*	*	sk	*	*
	3000	*	45227	co	11	1375.	sk	*	*	sk	*	*
30	50	2642	2402	co	13	1.4	co	11	3.7	co	5	1.6
	100	6417	3902	co	14	4.9	co	11	23.	co	5	12.
	150	12692	5402	co	12	7.0	co	12	123.	co	5	47.
	250	32742	8402	co	15	41.	co	13	1412.	co	5	409.
	500	126617	15902	co	11	107.	m	*	*	tl	*	*
	750	*	23402	co	11	232.	sk	*	*	sk	*	*
	1000	*	30902	co	13	768.	sk	*	*	sk	*	*
	1250	*	38402	co	12	1098.	sk	*	*	sk	*	*
	1500	*	45902	co	12	1637.	sk	*	*	sk	*	*

5.4 D-optimal experiment design

In a continuous relaxation of the D-optimal experiment design problem (see Boyd and Vandenberghe [2004, Section 7.5]), the variable $\mu \in \mathbb{R}^m$ is the number of trials to run for each of m experiments, and our goal is to minimize the determinant of the error covariance matrix $(F \operatorname{Diag}(\mu) F')^{-1}$, given a menu of experiments $F \in \mathbb{R}^{k \times m}$ useful for estimating a vector in $\mathbb{R}^k$. We require that a total of j experiments are performed and that each experiment can be performed between 0 and l times. We formulate this problem as:

$$\max_{\rho \in \mathbb{R}, \mu \in \mathbb{R}^m} \quad \rho : \tag{27a}$$

$$e'\mu = j, \tag{27b}$$

$$(l/2, \mu - (l/2)e) \in \mathcal{K}_{\ell_\infty}, \tag{27c}$$

$$(\rho, \operatorname{vec}(F \operatorname{Diag}(\mu) F')) \in \mathcal{K}_{\text{rtdet}}. \tag{27d}$$

In an alternative *logdet* variant of the *rtdet* variant (27), we replace (27d) with:

$$(\rho, 1, \mathrm{vec}(F\,\mathrm{Diag}(\mu)F')) \in \mathcal{K}_{\mathrm{logdet}}, \tag{28}$$

noting that both variants have the same optimal solution set for μ. The EFs for (27c), (27d) and (28) follow (8), (16) and (20). Since the EF for $\mathcal{K}_{\mathrm{rtdet}}$ depends on a $\mathcal{K}_{\mathrm{geo}}$ EF, for the rtdet variant we compare EF-exp and EF-sec (see Section 4.3).

To build random instances of (27), we generate F with independent Gaussian entries, for various values of k, $m = j = 2k$, and $l = 5$. Our results are summarized in Tables 6 and 7 and Figure 1d. For the logdet variant, $\nu = 3 + 3k$, $\bar{\nu} = 1 + 9k$, $n = 1 + 2k$, $p = \bar{p} = 1$. The sizes for the rootdet formulations are similar to those of the logdet formulations, so we exclude these. Note for the rootdet variant, we only plot EF-sec results for Hypatia-EF and MOSEK-EF, as MOSEK typically performs slightly better with EF-sec than with EF-exp, though Hypatia exhibits the opposite trend. For both variants, the NFs have much lower variable and conic constraint dimensions than the EFs. Although the EF solvers typically solve instances up to $k = 150$, Hypatia-NF solves instances with k at least 900. Hypatia-NF is also much faster than the EF solvers for all k.

Table 6: D-optimal experiment design logdet variant formulation and solver statistics.

	form. stats.			Hypatia-NF			Hypatia-EF			MOSEK-EF		
k	$\bar{n}$	q	$\bar{q}$	st	it	time	st	it	time	st	it	time
50	1426	1378	5401	co	25	0.3	co	21	4.5	co	15	12.
100	5351	5253	20801	co	26	0.9	co	25	91.	co	15	277.
150	11776	11628	46201	co	29	3.0	co	27	690.	tl	14	1825.
200	20701	20503	81601	co	28	7.2	tl	17	1849.	sk	*	*
300	46051	45753	182401	co	36	36.	sk	*	*	sk	*	*
400	81401	81003	323201	co	36	81.	m	*	*	sk	*	*
500	126751	126253	504001	co	36	169.	sk	*	*	sk	*	*
600	182101	181503	724801	co	36	298.	sk	*	*	sk	*	*
700	*	246753	*	co	39	624.	sk	*	*	m	*	*
800	*	322003	*	co	37	838.	sk	*	*	sk	*	*
900	*	407253	*	co	37	1282.	sk	*	*	sk	*	*
1000	*	502503	*	tl	37	1838.	sk	*	*	sk	*	*

5.5 Polynomial minimization

Following Papp and Yildiz [2019], we use an interpolant basis weighted SOS dual formulation to find a lower bound for a multivariate polynomial f of maximum degree $2k$ in m variables over the unit hypercube $\mathcal{D} = [-1, 1]^m$. We let $U = \binom{m+2k}{m}$, $L = \binom{m+k}{m}$, $\tilde{L} = \binom{m+k-1}{m}$. We select multivariate Chebyshev basis polynomials $g_j, \forall j \in [\![L]\!]$ of increasing degree up to k, and suitable interpolation points $o_u \in \mathcal{D}, \forall u \in [\![U]\!]$. To parametrize $\mathcal{K}^*_{\mathrm{SOS}(P)}$, we set up the collection of matrices P by evaluating functions of basis polynomials at the points:

$$(P_1)_{u,j} = g_j(o_u) \qquad \forall u \in [\![U]\!], j \in [\![L]\!], \tag{29a}$$

$$(P_{1+i})_{u,j} = g_j(o_u)(1 - o_{u,i}^2) \qquad \forall i \in [\![m]\!], u \in [\![U]\!], j \in [\![\tilde{L}]\!]. \tag{29b}$$

Table 7: D-optimal experiment design rtdet variant solver statistics.

	NF			EF-exp						EF-sec					
	Hypatia			Hypatia			MOSEK			Hypatia			MOSEK		
k	st	it	time	st	it	time	st	it	time	st	it	time	st	it	time
50	co	25	0.3	co	22	4.7	co	14	11.	co	22	5.1	co	11	10.
100	co	25	0.9	co	25	93.	co	13	247.	co	26	97.	co	11	220.
150	co	26	2.6	co	27	696.	co	12	1580.	sp	36	921.	co	10	1432.
200	co	23	5.8	tl	17	1821.	tl	0	1821.	tl	17	1848.	tl	0	1868.
300	co	31	31.	sk	*	*	sk	*	*	sk	*	*	sk	*	*
400	co	29	67.	m	*	*	sk	*	*	m	*	*	sk	*	*
500	co	32	152.	sk	*	*	sk	*	*	sk	*	*	sk	*	*
600	co	33	281.	sk	*	*	sk	*	*	sk	*	*	sk	*	*
700	co	32	530.	sk	*	*	m	*	*	sk	*	*	sk	*	*
800	co	32	728.	sk	*	*	sk	*	*	sk	*	*	sk	*	*
900	co	36	1253.	sk	*	*	sk	*	*	sk	*	*	sk	*	*
1000	co	33	1729.	sk	*	*	sk	*	*	sk	*	*	sk	*	*

Letting $\bar{f} = (f(o_u))_{u \in U}$ be evaluations of f at the points, the conic formulation is:

$$\min_{\rho \in \mathbb{R}^U} \quad \bar{f}'\rho : \tag{30a}$$
$$e'\rho = 1, \tag{30b}$$
$$\rho \in \mathcal{K}^*_{\mathrm{SOS}(P)}. \tag{30c}$$

The EF for NF constraint (30c) uses $\mathcal{K}_{\succeq}$ and is implicit in (21b).

To build random instances of (30), we generate $\bar{f}$ (which implicitly defines a polynomial f) with independent Gaussian entries, for various values of m and k. Our results are summarized in Table 8. Note that $\nu = \bar{\nu}$, $p = \bar{p} = 1$, $n = \bar{n} = q$. For fixed m, the conic constraint dimensions are larger for the EFs and grow much faster for the EFs as the degree k increases. Hypatia-NF is faster than the EF solvers on all instances with $k > 1$, and solves instances with much higher degrees.

5.6 Smooth density estimation

$\mathbb{R}_{m,2k}[x]$ is the ring of polynomials of maximum degree $2k$ in m variables [Papp and Yildiz, 2019]. We seek a polynomial density function $f \in \mathbb{R}_{m,2k}[x]$ over the domain $\mathcal{D} = [-1, 1]^m$ that maximizes the log-likelihood of N given observations $z_i \in \mathcal{D}, \forall i \in [\![N]\!]$ (compare to Papp and Alizadeh [2014, Section 4.3]). For f to be a valid density it must be nonnegative on $\mathcal{D}$ and integrate to one over $\mathcal{D}$, so we aim to solve:

$$\max_{f \in \mathbb{R}_{m,2k}[x]} \quad \sum_{i \in [\![N]\!]} \log(f(z_i)) : \tag{31a}$$
$$\int_{\mathcal{D}} f(x)\, dx = 1, \tag{31b}$$
$$f(x) \geq 0 \quad \forall x \in \mathcal{D}. \tag{31c}$$

To find a feasible solution for (31), we build an SOS formulation. We obtain interpolation points and matrices P parametrizing $\mathcal{K}_{\mathrm{SOS}(P)}$, using the techniques from Section 5.5. From the interpolation points and the domain $\mathcal{D}$, we compute a vector of quadrature weights $\mu \in \mathbb{R}^U$. We compute a matrix $B \in \mathbb{R}^{N \times U}$ by

Table 8: Polynomial minimization formulation and solver statistics.

		form. stats.			Hypatia-NF			Hypatia-EF			MOSEK-EF		
m	k	ν	n	$\bar{q}$	st	it	time	st	it	time	st	it	time
1	100	201	201	10201	co	12	0.1	co	34	1.2	co	15	27.
1	200	401	401	40401	co	14	0.3	co	39	13.	co	11	409.
1	500	1001	1001	251001	co	18	2.4	co	57	329.	rl	*	*
1	1000	2001	2001	*	co	19	11.	m	*	*	sk	*	*
1	2000	4001	4001	*	co	21	73.	sk	*	*	sk	*	*
1	3000	6001	6001	*	co	24	235.	sk	*	*	sk	*	*
1	4000	8001	8001	*	co	24	508.	sk	*	*	sk	*	*
1	5000	10001	10001	*	co	24	916.	sk	*	*	sk	*	*
2	15	376	496	23836	co	15	0.4	co	21	5.0	co	10	87.
2	30	1426	1891	339946	co	25	10.	co	49	751.	rl	*	*
2	45	3151	4186	*	co	22	58.	m	*	*	sk	*	*
2	60	5551	7381	*	co	28	300.	sk	*	*	sk	*	*
2	75	8626	11476	*	co	30	1019.	sk	*	*	sk	*	*
3	6	252	455	8358	co	17	0.3	co	17	1.6	co	9	9.1
3	9	715	1330	65395	co	20	3.1	co	24	104.	co	9	799.
3	12	1547	2925	303030	co	23	20.	co	33	1775.	rl	*	*
3	15	2856	5456	*	co	26	89.	m	*	*	sk	*	*
3	18	4750	9139	*	er	34	1340.	sk	*	*	sk	*	*
4	4	210	495	5005	co	18	0.4	co	16	1.7	co	8	3.9
4	6	714	1820	54159	co	15	4.8	co	18	222.	co	10	579.
4	8	1815	4845	*	co	20	63.	m	*	*	m	*	*
4	10	3861	10626	*	co	22	458.	sk	*	*	sk	*	*
8	2	117	495	1395	co	26	0.5	co	21	0.7	co	11	0.9
8	3	525	3003	21975	co	18	15.	co	16	148.	co	8	125.
8	4	1815	12870	*	co	27	633.	m	*	*	m	*	*
16	1	33	153	169	co	13	0.1	co	12	0.7	co	7	0.0
16	2	425	4845	14229	co	27	86.	co	22	174.	sp	10	192.
32	1	65	561	593	co	15	0.7	co	12	1.0	co	7	0.2
64	1	129	2145	2209	co	15	14.	co	12	3.1	co	9	3.0

evaluating the U Lagrange basis polynomials corresponding to the interpolation points (see Papp and Yildiz [2019]) at the N observations. Letting variable ρ represent the coefficients on the Lagrange basis, the conic formulation is:

$$\max_{\psi\in\mathbb{R},\rho\in\mathbb{R}^U} \quad \psi : \tag{32a}$$
$$\mu'\rho = 1, \tag{32b}$$
$$(\psi, 1, B\rho) \in \mathcal{K}_{\log}, \tag{32c}$$
$$\rho \in \mathcal{K}_{\mathrm{SOS}(P)}. \tag{32d}$$

The EFs for NF constraints (32c) and (32d) follow (18) and (21a).

To build random instances of (32) for various values of m and k, we generate $N = 500$ independent uniform samples in $[-1, 1]^m$ for $z_i \in \mathcal{D}, \forall i \in [\![N]\!]$. As our method for computing μ is numerically unstable for larger

m, we only use $m \leq 16$. Our results are summarized in Table 9. Note that $\bar{\nu} = 999 + \nu$, $p = 1$, $\bar{p} = n$, $q = 501 + n$, $\bar{q} = 1001 + \bar{n} - n$. All dimensions are larger for the EFs than for the NFs. Hypatia-NF is faster than the EF solvers and solves instances with much higher degrees.

Table 9: Smooth density estimation.

		dimensions			Hypatia-NF			Hypatia-EF			MOSEK-EF		
m	$2k$	ν	n	$\bar{n}$	st	it	time	st	it	time	st	it	time
1	250	753	252	16628	co	35	0.2	sp	43	1040.	co	25	112.
1	500	1003	502	64003	co	42	1.1	rl	*	*	tl	18	1814.
1	1000	1503	1002	*	co	41	5.1	m	*	*	m	*	*
1	2000	2503	2002	*	co	56	23.	sk	*	*	sk	*	*
1	4000	4503	4002	*	co	82	185.	sk	*	*	sk	*	*
1	6000	6503	6002	*	co	106	663.	sk	*	*	sk	*	*
2	20	678	232	6023	co	50	0.3	co	29	73.	co	19	7.4
2	40	1153	862	72468	co	34	2.9	rl	*	*	sp	22	1522.
2	60	1928	1892	*	co	36	11.	m	*	*	m	*	*
2	80	3003	3322	*	co	53	64.	sk	*	*	sk	*	*
2	100	4378	5152	*	co	64	247.	sk	*	*	sk	*	*
3	12	754	456	9314	co	57	0.9	co	25	293.	sp	19	20.
3	18	1217	1331	67226	co	55	6.6	rl	*	*	sp	17	1216.
3	24	2049	2926	*	co	46	35.	m	*	*	m	*	*
3	30	3358	5457	*	co	63	348.	sk	*	*	sk	*	*
4	8	712	496	6001	co	57	1.0	co	25	133.	sp	22	12.
4	12	1216	1821	56480	co	72	16.	tl	*	*	sp	20	934.
4	16	2317	4846	*	co	70	192.	m	*	*	m	*	*
8	4	619	496	2391	co	96	2.1	co	30	9.8	sp	17	2.1
8	6	1027	3004	25479	co	90	62.	tl	*	*	sp	17	379.

5.7 Shape constrained regression

A common type of shape constraint imposes monotonicity or convexity of a polynomial over a basic semialgebraic set [Hall, 2019, Section 6]. Given an m-dimensional feature variable z and a scalar response variable g, we aim to fit a polynomial $f \in \mathbb{R}_{m,2k}[x]$ that is convex over $\mathcal{D} = [-1, 1]^m$ to N given observations $(z_i, g_i)_{i \in [\![N]\!]}$ with $z_i \in \mathcal{D}, \forall i \in [\![N]\!]$:

$$\min_{f \in \mathbb{R}_{m,2k}[x]} \quad \sum_{i \in [\![N]\!]} (g_i - f(z_i))^2 : \tag{33a}$$

$$y'(\nabla^2 f(x))y \geq 0 \quad \forall x \in \mathcal{D}, y \in \mathbb{R}^m. \tag{33b}$$

Constraint (33b) ensures the Hessian matrix $\nabla^2 f(x)$ of polynomials is PSD at every point $x \in \mathcal{D}$, which is equivalent to convexity of f over $\mathcal{D}$. To find a feasible solution for (33), we build an SOS formulation. The polynomial variable, represented in an interpolant basis with the optimization variable $\rho \in \mathbb{R}^U$, has degree $2k$ and $U = \binom{m+2k}{m}$ coefficients. Each polynomial entry of $\nabla^2 f(x)$ has degree $2k - 2$ and $\bar{U} = \binom{m+2k-2}{m}$ coefficients. Following the descriptions in Sections 5.5 to 5.6, we obtain interpolation points and a Lagrange polynomial basis for these U-dimensional and $\bar{U}$-dimensional spaces, and we define the matrix $B \in \mathbb{R}^{N \times U}$ containing evaluations of the U-dimensional Lagrange basis at the N feature observations. Finally, we let $F \in \mathbb{R}^{\mathrm{sd}(m)\bar{U} \times U}$ be such that $F\rho$ is a vectorization of the tensor $H \in \mathbb{R}^{m \times m \times \bar{U}}$ (scaled to account for

symmetry) with $H_{a,b,u}$ equal to the uth coefficient of the (a,b)th polynomial in $\nabla^2 f(x)$ for $a,b \in [\![m]\!]$ and $u \in [\![\bar{U}]\!]$. This yields the formulation:

$$\min_{\psi\in\mathbb{R},\rho\in\mathbb{R}^U} \quad \psi : \tag{34a}$$
$$(\psi, g - B\rho) \in \mathcal{K}_{\ell_2}, \tag{34b}$$
$$F\rho \in \mathcal{K}_{\text{matSOS}(P)}. \tag{34c}$$

Note that for $N > U$, we use a QR factorization to reduce the dimension of $\mathcal{K}_{\ell_2}$ in (34b) from $1+N$ to $2+U$.[5] The EF for NF constraint (34c) follows (22a).

To build random instances of (34) for various values of m and k, we generate $N = \lceil 1.1U \rceil$ independent observations with z_i sampled uniformly from $\mathcal{D}$ and $g_i = \exp(\|z\|^2/m) - 1 + \varepsilon_i$, where ε_i is a Gaussian sample yielding a signal to noise ratio of 10, for all $i \in [\![N]\!]$. We exclude the case $m = 1$, since $\mathcal{K}_{\text{SOS}(P)}$ could be used in place of $\mathcal{K}_{\text{matSOS}(P)}$. Our results are summarized in Table 10. Note that $\nu = \bar{\nu}$, $p = 0$, $\bar{p} = q - n - 1$, $\bar{q} = \bar{n} + 1$. All dimensions are larger for the EFs. The instances are numerically challenging, and MOSEK-EF often encounters slow progress. Hypatia-NF is faster than the EF solvers and solves instances with much higher degrees.

6 Conclusions

Although many convex problems are representable with conic EFs using the small number of standard cones currently recognized by some advanced conic solvers, these formulations can be much larger and more complex than NFs with exotic cones. In Section 4, we describe some of Hypatia's predefined exotic cones and analyze general techniques for constructing EFs from NFs that use these cones. For several example problems, we propose NFs and generate instances of a wide range of sizes. Across these instances, we observe much higher empirical dimensions (variable, equality, and conic constraint dimensions in the conic general form (2)) for the EFs than for the NFs. We demonstrate significant computational advantages from solving the NFs with Hypatia compared to solving the EFs with either Hypatia or MOSEK 9, especially in terms of solve time and memory usage. We also observe that the NFs are typically faster and less memory-intensive to generate using JuMP.

Our results suggest that when there exists an NF that is significantly smaller than any EF, it is probably worth trying to solve the NF with Hypatia. In deciding whether to formulate an NF or an EF, it can be helpful to examine our summary in Table 1 of computational properties for NFs and EFs of exotic cone constraints. For spectral and nuclear norm constraints, when the matrix ($W \in \mathbb{R}^{d_1 \times d_2}$) has many more columns than rows ($d_2 \gg d_1$), the dimensions look relatively more favorable for the NF. For SOS and SOS matrix constraints, the dimensions grow much more slowly for the NF as the polynomial degree increases. Sometimes the modeler has to choose between different EFs. For our matrix completion problem and experiment design root-determinant variant, we compare two EFs for the geometric mean cone and find that Hypatia performs better with the exponential cone EF (EF-exp) and MOSEK performs better with the second order cone EF (EF-sec).

If the modeler has an NF that uses a proper cone not already defined in Hypatia, the user can add support for the cone through Hypatia's generic cone interface. It may require some effort to make the cone oracles

[5] Let $[-B\ g] = QR$, where $Q \in \mathbb{R}^{N\times(U+1)}$ has orthonormal columns and $R \in \mathbb{R}^{(U+1)\times(U+1)}$ is upper triangular. Then $(\psi, g - B\rho) \in \mathcal{K}_{\ell_2}$ if and only if $(\psi, R(\rho, 1)) \in \mathcal{K}_{\ell_2}$.

Table 10: Shape constrained regression formulation and solver statistics.

		form. stats.				Hypatia-NF			Hypatia-EF			MOSEK-EF		
m	$2k$	ν	n	$\bar{n}$	q	st	it	time	st	it	time	st	it	time
2	10	72	67	952	203	co	18	0.0	co	24	0.6	sp	19	0.4
2	20	292	232	14527	803	co	37	0.5	sp	68	1348.	sp	20	60.
2	30	662	497	73727	1803	co	58	5.6	rl	$*$	$*$	tl	12	1845.
2	40	1182	862	$*$	3203	co	84	34.	m	$*$	$*$	sk	$*$	$*$
2	50	1852	1327	$*$	5003	er	52	161.	sk	$*$	$*$	sk	$*$	$*$
2	60	2672	1892	$*$	7203	er	120	939.	sk	$*$	$*$	sk	$*$	$*$
3	8	152	166	3391	671	co	19	0.1	co	27	14.	sp	23	4.3
3	12	485	456	31347	2173	co	38	3.5	tl	$*$	$*$	sp	15	244.
3	16	1118	970	161584	5051	co	61	44.	m	$*$	$*$	rl	$*$	$*$
3	20	2147	1772	$*$	9753	co	87	325.	sk	$*$	$*$	sk	$*$	$*$
3	24	3668	2926	$*$	16727	co	111	1605.	sk	$*$	$*$	sk	$*$	$*$
4	6	142	211	2881	912	co	17	0.4	co	23	7.8	sp	18	3.7
4	8	382	496	17686	2597	co	24	3.1	co	38	1621.	sp	14	94.
4	10	842	1002	79822	5953	co	38	35.	rl	$*$	$*$	tl	12	2068.
4	12	1626	1821	$*$	11832	co	58	283.	m	$*$	$*$	sk	$*$	$*$
4	14	2858	3061	$*$	21262	co	72	1430.	sk	$*$	$*$	sk	$*$	$*$
6	4	80	211	1240	800	co	13	0.3	co	15	0.7	co	9	0.6
6	6	422	925	20539	5336	co	22	15.	co	32	1630.	sp	17	260.
6	8	1514	3004	215440	22409	co	29	638.	m	$*$	$*$	rl	$*$	$*$
8	4	138	496	3412	2117	co	19	2.1	co	20	8.0	co	11	4.1
8	6	938	3004	89008	20825	co	31	515.	rl	$*$	$*$	rl	$*$	$*$
10	4	212	1002	7657	4633	co	26	13.	co	24	87.	co	13	25.
12	4	302	1821	15003	8920	co	29	73.	co	28	504.	co	11	125.
14	4	408	3061	26686	15662	co	33	346.	tl	0	1884.	sp	21	767.

as efficient and numerically stable as possible. However, we have already predefined over two dozen exotic cone types with tractable oracles in Hypatia [Coey et al., 2021c,a], many of which have multiple variants (such as real or complex flavors). We use these cones to model hundreds of formulations in our over three dozen examples available in Hypatia's examples folder. Our experience with these examples suggests that NFs tend to be more convenient for modeling and interpreting conic certificates.

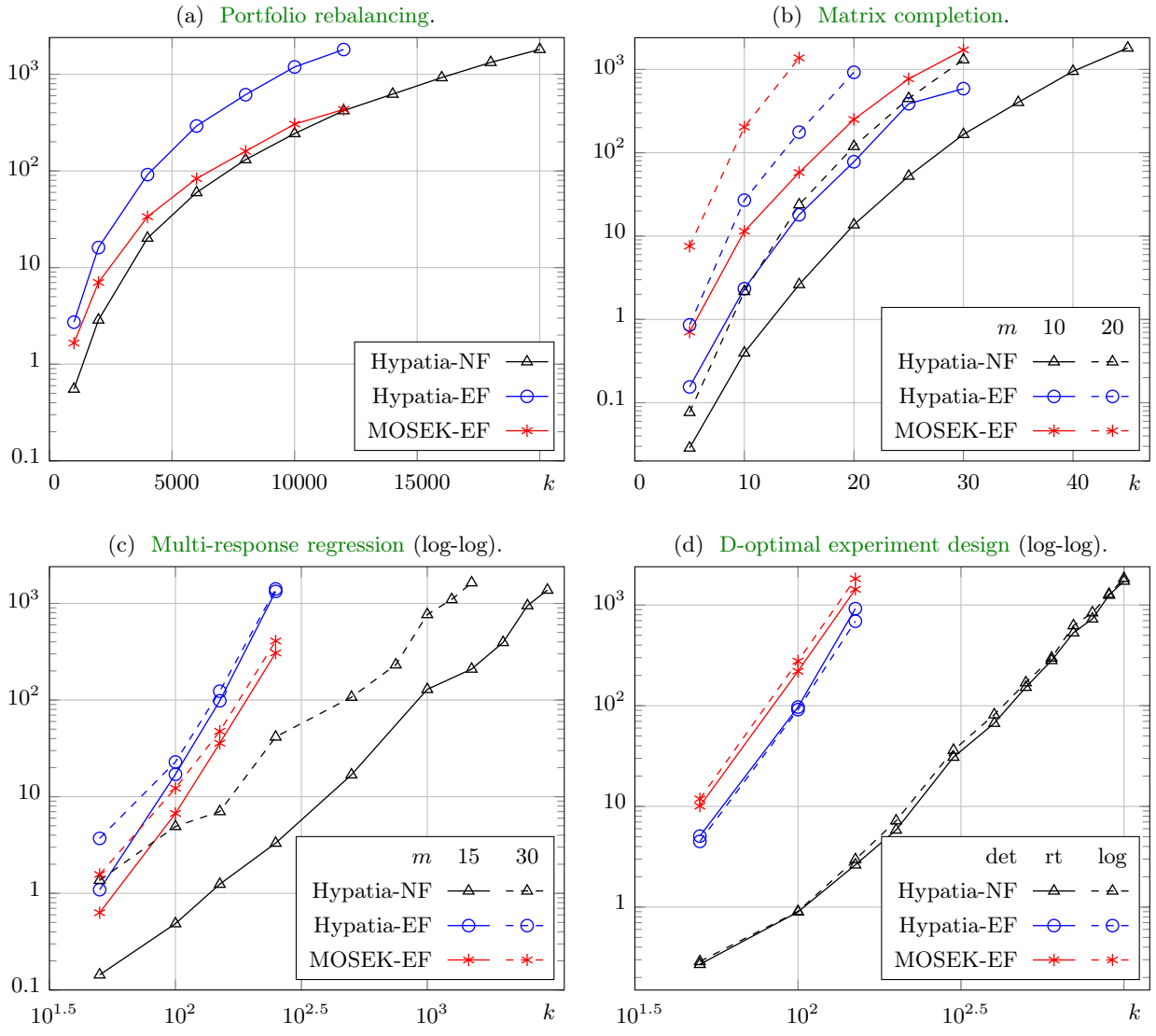
References

A. Agrawal, S. Diamond, and S. Boyd. Disciplined geometric programming. *Optimization Letters*, 13(5): 961–976, 2019.

M. Andersen, J. Dahl, Z. Liu, L. Vandenberghe, S. Sra, S. Nowozin, and S. Wright. Interior-point methods for large-scale cone programming. *Optimization for Machine Learning*, 5583, 2011.

A. Ben-Tal and A. Nemirovski. *Lectures on modern convex optimization: analysis, algorithms, and engineering applications*, volume 2. SIAM, 2001.

Figure 1: Solve times (in seconds) for solve runs satisfying the convergence check in (23).



J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah. Julia: A fresh approach to numerical computing. *SIAM Review*, 59(1):65–98, 2017.

B. Borchers. CSDP, a C library for semidefinite programming. *Optimization Methods and Software*, 11(1-4): 613–623, 1999.

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

C. Coey, M. Lubin, and J. P. Vielma. Outer approximation with conic certificates for mixed-integer convex problems. *Mathematical Programming Computation*, 12:249–293, 2020.

C. Coey, L. Kapelevich, and J. P. Vielma. Hypatia cones reference, 2021a. URL https://github.com/chriscoey/Hypatia.jl/wiki/. Online; accessed 1-June-2021.

C. Coey, L. Kapelevich, and J. P. Vielma. *Hypatia documentation*, 2021b. URL https://chriscoey.github.io/Hypatia.jl/dev/. Online; accessed 7-June-2021.

C. Coey, L. Kapelevich, and J. P. Vielma. Performance enhancements for a generic conic interior point algorithm. *arXiv preprint arXiv:?????*, 2021c.

C. Coey, L. Kapelevich, and J. P. Vielma. Conic optimization over epigraphs of spectral functions on cones of squares. *arXiv preprint arXiv:2103.04104*, 2021d.

J. Dahl and E. D. Andersen. A primal-dual interior-point algorithm for nonsymmetric exponential-cone optimization. *Mathematical Programming*, pages 1–30, 2021.

S. Diamond and S. Boyd. CVXPY: a Python-embedded modeling language for convex optimization. *The Journal of Machine Learning Research*, 17(1):2909–2913, 2016.

A. Domahidi, E. Chu, and S. Boyd. ECOS: an SOCP solver for embedded systems. In *2013 European Control Conference (ECC)*, pages 3071–3076. IEEE, 2013.

I. Dunning, J. Huchette, and M. Lubin. JuMP: a modeling language for mathematical optimization. *SIAM Review*, 59(2):295–320, 2017.

M. Grant and S. Boyd. CVX: MATLAB software for disciplined convex programming, version 2.1, 2014.

O. Güler. Barrier functions in interior point methods. *Mathematics of Operations Research*, 21(4):860–885, 1996.

G. Hall. Engineering and business applications of sum of squares polynomials. *arXiv preprint arXiv:1906.07961*, 2019.

L. Kapelevich, C. Coey, and J. P. Vielma. Sum of squares generalizations for conic sets. *arXiv preprint arXiv:2103.11499*, 2021.

M. Karimi and L. Tunçel. Domain-driven solver (DDS): a MATLAB-based software package for convex optimization problems in domain-driven form. *arXiv preprint arXiv:1908.03075*, 2019.

M. Karimi and L. Tunçel. Primal–dual interior-point methods for domain-driven formulations. *Mathematics of Operations Research*, 45(2):591–621, 2020.

B. Legat, O. Dowson, J. D. Garcia, and M. Lubin. MathOptInterface: a data structure for mathematical optimization problems. *arXiv preprint arXiv:2002.03447*, 2020.

M. Lubin, E. Yamangil, R. Bent, and J. P. Vielma. Extended formulations in mixed-integer convex programming. In Q. Louveaux and M. Skutella, editors, *Integer Programming and Combinatorial Optimization: 18th International Conference, IPCO 2016, Liège, Belgium, June 1-3, 2016, Proceedings*, pages 102–113. Springer International Publishing, 2016. ISBN 978-3-319-33461-5. doi: 10.1007/978-3-319-33461-5_9.

MOSEK ApS. MOSEK fusion API for Python, 2020a. URL https://docs.mosek.com/9.1/pythonfusion/index.html.

MOSEK ApS. Modeling Cookbook revision 3.2.1, 2020b. URL https://docs.mosek.com/modeling-cookbook/index.html.

Y. Nesterov. Constructing self-concordant barriers for convex cones. *CORE discussion paper*, 2006.

Y. Nesterov. Towards non-symmetric conic optimization. *Optimization Methods and Software*, 27(4-5): 893–917, 2012.

Y. Nesterov and A. Nemirovskii. *Interior-point polynomial algorithms in convex programming*. Studies in Applied Mathematics. Society for Industrial and Applied Mathematics, 1994.

Y. E. Nesterov and M. J. Todd. Self-scaled barriers and interior-point methods for convex programming. *Mathematics of Operations Research*, 22(1):1–42, 1997.

Y. E. Nesterov, M. J. Todd, and Y. Ye. Infeasible-start primal-dual methods and infeasibility detectors for nonlinear programming problems. Technical report, Cornell University Operations Research and Industrial Engineering, 1996.

B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd. Conic optimization via operator splitting and homogeneous self-dual embedding. *Journal of Optimization Theory and Applications*, 169(3):1042–1068, 2016.

D. Papp and F. Alizadeh. Shape-constrained estimation using nonnegative splines. *Journal of Computational and Graphical Statistics*, 23(1):211–231, 2014.

D. Papp and S. Yıldız. On "A homogeneous interior-point algorithm for non-symmetric convex conic optimization". *arXiv preprint arXiv:1712.00492*, 2017.

D. Papp and S. Yildiz. Sum-of-squares optimization without semidefinite programming. *SIAM Journal on Optimization*, 29(1):822–851, 2019.

D. Papp and S. Yıldız. alfonso: Matlab package for nonsymmetric conic optimization. *arXiv preprint arXiv:2101.04274*, 2021.

F. Permenter, H. A. Friberg, and E. D. Andersen. Solving conic optimization problems via self-dual embedding and facial reduction: a unified approach. *SIAM Journal on Optimization*, 27(3):1257–1282, 2017.

B. Recht, M. Fazel, and P. A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501, 2010.

S. A. Serrano. *Algorithms for unsymmetric cone optimization and an implementation for problems with the exponential cone*. PhD thesis, Stanford University, 2015.

A. Skajaa and Y. Ye. A homogeneous interior-point algorithm for nonsymmetric convex conic optimization. *Mathematical Programming*, 150(2):391–422, 2015.

M. Udell, K. Mohan, D. Zeng, J. Hong, S. Diamond, and S. Boyd. Convex optimization in Julia. In *Proceedings of the 1st First Workshop for High Performance Technical Computing in Dynamic Languages*, pages 18–28. IEEE Press, 2014.

L. Vandenberghe. The CVXOPT linear and quadratic cone program solvers, 2010. URL https://www.seas.ucla.edu/~vandenbe/publications/coneprog.pdf.

M. Yamashita, K. Fujisawa, and M. Kojima. Implementation and evaluation of SDPA 6.0 (semidefinite programming algorithm 6.0). *Optimization Methods and Software*, 18(4):491–505, 2003.

J. Yang, L. Luo, J. Qian, Y. Tai, F. Zhang, and Y. Xu. Nuclear norm based matrix regression with applications to face recognition with occlusion and illumination changes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(1):156–171, 2016.