Mathematical Programming Computation (2021) 13:257–295
https://doi.org/10.1007/s12532-020-00193-4

FULL LENGTH PAPER

Signomial and polynomial optimization via relative entropy and partial dualization

Riley Murray[1] · Venkat Chandrasekaran[1] · Adam Wierman[1]

Received: 21 July 2019 / Accepted: 11 August 2020 / Published online: 14 October 2020

Abstract
We describe a generalization of the Sums-of-AM/GM-Exponential (SAGE) methodology for relative entropy relaxations of constrained signomial and polynomial optimization problems. Our approach leverages the fact that SAGE certificates conveniently and transparently blend with convex duality, in a way which enables partial dualization of certain structured constraints. This more general approach retains key properties of ordinary SAGE relaxations (e.g. sparsity preservation), and inspires a projective method of solution recovery which respects partial dualization. We illustrate the utility of our methodology with a range of examples from the global optimization literature, along with a publicly available software package.

Keywords Global optimization · Exponential cone programs · SAGE certificates · SOS certificates · Signomial programming

1 Introduction

A signomial is a function $\boldsymbol{x} \mapsto \sum_{i=1}^{m} c_i \exp(\boldsymbol{\alpha}_i \cdot \boldsymbol{x})$ for real scalars c_i and row vectors $\boldsymbol{\alpha}_i$ in $\mathbb{R}^{1\times n}$. Signomial optimization and signomial programming are challenging problems with applications in chemical engineering [1], aeronautics [2], circuit design [3], and communications network optimization [4]. Signomials are sometimes thought of as generalized polynomials over the positive orthant $\mathbb{R}^n_{++}$; by a change of variables $y_i = \exp x_i$ one arrives at "geometric form" signomials $\boldsymbol{y} \mapsto \sum_{i=1}^{m} c_i \prod_{j=1}^{n} y_j^{\alpha_{ij}}$. Despite this aesthetic similarity, signomials and polynomials have many significant

✉ Riley Murray
rmurray@caltech.edu

Venkat Chandrasekaran
venkatc@caltech.edu

Adam Wierman
adamw@caltech.edu

[1] California Institute of Technology, Pasadena, CA 91125, USA

differences. Where polynomials are generated by a countably infinite basis, signomials require an uncountably infinite basis. Where polynomials are closed under composition, signomials are not. Where polynomials and exponential-form signomials are defined on all of $\mathbb{R}^n$, geometric-form signomials are only defined on $\mathbb{R}^n_{++}$.

For many years these abstract differences between signomials and polynomials have coincided with algorithmic disparities. Contemporary methods for signomial programming use some combination of local linearization, penalty functions, sequential geometric programming, and branch-and-bound [5–11]—ideas which precede the advent of modern convex optimization. By contrast, the study of polynomial optimization has been substantially influenced by semidefinite programming, specifically through Sums-of-Squares (SOS) certificates of polynomial nonnegativity [12–14]. In recent work, Chandrasekaran and Shah proposed Sums-of-AM/GM-Exponentials or "SAGE" decompositions, which use relative entropy programming to certify signomial nonnegativity [15]. The authors of the present article have further demonstrated that a certain extension of signomial SAGE certificates provides a tractable sufficient condition for polynomial nonnegativity [16]. This modification connects SAGE to Sums-of-Nonnegative-Circuit-Polynomials (SONC) [17], as we discuss in Sect. 2.2.

Here, we are concerned with how certain certificates of nonnegativity can be used for constrained optimization. The basic idea is simple: for a function f, a set X, and a scalar γ, we have $f^\star_X \doteq \inf\{f(\boldsymbol{x}) \,:\, \boldsymbol{x} \in X\} \geq \gamma$ if and only if $f - \gamma$ is nonnegative over X. The trouble is that to leverage this fact, we require ways to extend certificates for *global* nonnegativity (such as SOS or SAGE certificates) to prove nonnegativity over $X \subsetneq \mathbb{R}^n$. In the polynomial case, one usually performs this extension by appealing to representation theorems from real algebraic geometry. Absent such representation theorems, one typically relies on a dual problem obtained from the minimax inequality.

Our main contribution is to extend SAGE certificates to accommodate certain feasible sets "X" in a way which uses neither algebraic geometry nor the minimax inequality. When X is convex, this manifests in cones of "X-SAGE signomials" which are completely characterized by a relative entropy program involving the support function of X. Constraints which are not compatible with our new type of SAGE certificate are moved into a suitable Lagrangian, in a process known as partial dualization. We develop the core principles behind this approach for both signomials and polynomials. An abundance of examples are provided to illustrate the range of possibilities with this new framework.

1.1 Article outline and our contributions

Section 2 provides background material on the following points, all of which are central to the current article. (1) The sources of error in nonnegativity relaxations of constrained optimization problems, and how are these sources of error are usually mitigated. (2) The formulations for ordinary SAGE cones, and the relationships between SAGE and other nonnegativity certificates in the literature. (3) The definition of partial dualization, and how it can be understood in the context of existing nonnegativity and moment relaxations.

Once those topics are covered, we introduce *conditional SAGE certificates* for signomial nonnegativity over sets $X \subset \mathbb{R}^n$ (Sect. 3). We show that when X is a convex set, cones of "X-SAGE signomials" are completely characterized by a relative entropy program involving the support function of X (Theorem 1). This result is leveraged to obtain a representation for dual X-SAGE cones, which have a structure enabling a projective solution recovery method for convex relaxations to signomial programs (Algorithm 1). We go on to describe two SAGE-based "hierarchies" of convex relaxations for signomial optimization: one which uses the minimax inequality, and one which is minimax-free. The authors know of no analog to the minimax-free hierarchy in the polynomial optimization literature, and believe the underlying idea of the minimax-free hierarchy is of independent theoretical interest.

Section 4 extends conditional SAGE certificates to polynomials; the structure of this section is entirely analogous to that of Sect. 3. In this polynomial setting, it is not convexity of X which determines when an X-SAGE polynomial cone is tractable, but rather convexity of a certain logarithmic transform of X (Theorems 2 and 3). Solution recovery from SAGE relaxations of polynomial optimization problems is more complicated than in the signomial case; see Algorithm 2 and subroutines given by Algorithms 3 and 4. Our minimax-free hierarchy for polynomial optimization reflects a link between SAGE signomials and polynomials by way of the "signomial representatives" from [16].

Section 5 reports the effectiveness of our methodology on 51 problems from the literature, as well as randomly generated problems. A central goal of our experiments is to facilitate research into the theory underlying conditional SAGE relaxations, and the practice of using these relaxations in engineering design optimization. Towards this end, we provide the "`sageopt`" python package: a documented, tested, and convenient platform for constructing and solving SAGE relaxations. We use `sageopt` for all experiments in this article.

Concluding remarks and lines for future work are given in Sect. 6.

1.2 Notation and preliminary definitions

Vectors and matrices appear in boldface. The ith entry of a vector $\boldsymbol{v}$ is v_i, and the vector formed by deleting the ith entry of $\boldsymbol{v}$ is $\boldsymbol{v}_{\setminus i}$. A matrix $\boldsymbol{A}$ is built by stacking rows $\boldsymbol{a}_i \in \mathbb{R}^{1\times n}$, and $\boldsymbol{A}_{\setminus i}$ is the submatrix formed by deleting the ith row of $\boldsymbol{A}$. All logarithms are base-e. We use $\boldsymbol{e}_i$ to denote the ith standard basis vector in $\mathbb{R}^\ell$ where ℓ should be clear from context, and set $\mathbf{1} = \sum_{i=1}^{\ell} \boldsymbol{e}_i$. Elementary functions from $\mathbb{R}$ to $\mathbb{R}$ are extended first to vectors in an elementwise fashion, and subsequently to sets in a pointwise fashion. For a convex cone $K \subset \mathbb{R}^\ell$, the dual cone is $K^\dagger \doteq \{\boldsymbol{y} : \boldsymbol{y}^\mathsf{T}\boldsymbol{x} \geq 0 \text{ for all } \boldsymbol{x} \text{ in } K\}$. For $A, B \subset \mathbb{R}^n$, $A \subset B$ and $A \subsetneq B$ denote non-strict and strict inclusion respectively. The operator "cl" computes set-closure with respect to the standard topology.

For an $m \times n$ matrix $\boldsymbol{\alpha}$ and a vector $\boldsymbol{c}$ in $\mathbb{R}^m$, we write $f = \mathsf{Sig}(\boldsymbol{\alpha}, \boldsymbol{c})$ to mean that f takes values $f(\boldsymbol{x}) = \sum_{i=1}^{m} c_i \exp(\boldsymbol{\alpha}_i \cdot \boldsymbol{x})$. When $\boldsymbol{\alpha}$ is a matrix of nonnegative integers, we write $f = \mathsf{Pol}(\boldsymbol{\alpha}, \boldsymbol{c})$ to mean that $\boldsymbol{c}$ is the coefficient vector of f with respect to the monomial basis $\boldsymbol{x} \mapsto \boldsymbol{x}^{\boldsymbol{\alpha}_i} \doteq \prod_{j=1}^{n} x_j^{\alpha_{ij}}$. Given a matrix $\boldsymbol{\alpha}$ and a set $X \subset \mathbb{R}^n$, one

has the nonnegativity cones

$$\mathsf{C}_{\mathsf{NNS}}(\boldsymbol{\alpha}, X) \doteq \{\boldsymbol{c} \,:\, \mathsf{Sig}(\boldsymbol{\alpha}, \boldsymbol{c})(\boldsymbol{x}) \geq 0 \text{ for all } \boldsymbol{x} \text{ in } X\}$$

and

$$\mathsf{C}_{\mathsf{NNP}}(\boldsymbol{\alpha}, X) \doteq \{\boldsymbol{c} \,:\, \mathsf{Pol}(\boldsymbol{\alpha}, \boldsymbol{c})(\boldsymbol{x}) \geq 0 \text{ for all } \boldsymbol{x} \text{ in } X\}.$$

We write $\mathsf{C}_{\mathsf{NNS}}(\boldsymbol{\alpha})$ and $\mathsf{C}_{\mathsf{NNP}}(\boldsymbol{\alpha})$ in reference to the above cones when $X = \mathbb{R}^n$. Except in special cases on $\boldsymbol{\alpha}$, it is computationally intractable to check membership in either $\mathsf{C}_{\mathsf{NNS}}(\boldsymbol{\alpha})$ or $\mathsf{C}_{\mathsf{NNP}}(\boldsymbol{\alpha})$ [18]. Nonnegativity certificates developed in this article use the relative entropy function; this is the convex function "D" which continuously extends $D(\boldsymbol{u}, \boldsymbol{v}) = \sum_{i=1}^{m} u_i \log(u_i/v_i)$ to $\mathbb{R}^m_+ \times \mathbb{R}^m_+$. We use the standard convention that $D(\boldsymbol{u}, \boldsymbol{v}) = +\infty$ for $(\boldsymbol{u}, \boldsymbol{v}) \notin \mathbb{R}^m_+ \times \mathbb{R}^m_+$.

Our computational experiments use the MOSEK solver [19], with two different machines. Machine **W** is an HP Z820 workstation, with two 8-core 2.6 GHz Intel Xeon E5-2670 processors and 256 GB 1600 MHz DDR3 RAM. Machine **L** is a 2013 MacBook Pro, with a dual-core 2.4GHz Intel Core i5 processor and 8 GB 1600 MHz DDR3 RAM.

2 Problem statement and background material

We study constrained nonconvex optimization problems of the form

$$(f, g, \phi)^\star_X = \inf\{f(\boldsymbol{x}) : \boldsymbol{x} \text{ in } X \subset \mathbb{R}^n,\ g(\boldsymbol{x}) \geq \boldsymbol{0},\ \phi(\boldsymbol{x}) = \boldsymbol{0}\} \tag{1}$$

where f is a function from $\mathbb{R}^n$ to $\mathbb{R}$, g maps $\mathbb{R}^n$ to $\mathbb{R}^{k_1}$, and ϕ maps $\mathbb{R}^n$ to $\mathbb{R}^{k_2}$. Our primary goal is to produce lower bounds $(f, g, \phi)^{\text{lb}}_X \leq (f, g, \phi)^\star_X$. In the event that $(f, g, \phi)^{\text{lb}}_X = (f, g, \phi)^\star_X$, we are also interested in recovering optimal solutions to (1). For ease of exposition, this section focuses on problems of the form (1) with only inequality constraints– i.e. the problem of bounding

$$(f, g)^\star_X = \inf\{f(\boldsymbol{x}) : \boldsymbol{x} \text{ in } X \subset \mathbb{R}^n,\ g(\boldsymbol{x}) \geq \boldsymbol{0}\}. \tag{1.1}$$

In Sect. 2.1 we review the Lagrange dual relaxation of the above problem, both in minimax form and as a nonnegativity problem. Section 2.2 provides background on SAGE and related nonnegativity certificates, which is needed develop our contributions and frame them in the larger literature. In Sect. 2.3 we review standard techniques for strengthening nonnegativity-based relaxations of problems such as (1.1); this includes the use of redundant constraints, nonconstant Lagrange multipliers, and strengthening nonnegativity certificates via modulation. Section 2.4 reviews partial dualization, a concept which is central to this article. Until Sect. 2.4, the set X appearing in Problem 1.1 shall be the whole of $\mathbb{R}^n$.

2.1 Dual problems in nonconvex optimization

The simplest way to lower bound $(f, g)^{\star}_{\mathbb{R}^n}$ is via the Lagrange dual. For each coordinate function g_i of g, we introduce a dual variable $\lambda_i \geq 0$ and consider the Lagrangian $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) - \boldsymbol{\lambda}^{\mathsf{T}} g(\boldsymbol{x})$. The *Lagrange dual problem* is to compute

$$(f, g)^{\mathrm{L}}_{\mathbb{R}^n} = \sup_{\boldsymbol{\lambda} \geq \mathbf{0}} \inf_{\boldsymbol{x} \in \mathbb{R}^n} \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}).$$

By the minimax inequality, we can be certain that $(f, g)^{\mathrm{L}}_{\mathbb{R}^n} \leq (f, g)^{\star}_{\mathbb{R}^n}$.

There are many situations when the Lagrange dual problem is intractable. For signomial and polynomial optimization, one usually needs to compute yet another lower bound $(f, g)^{\mathrm{d}}_{\mathbb{R}^n} \leq (f, g)^{\mathrm{L}}_{\mathbb{R}^n}$. Contemporary approaches for computing such bounds begin by introducing a parameterized function $\psi(\gamma, \boldsymbol{\lambda})$ which takes values $\psi(\gamma, \boldsymbol{\lambda})(\boldsymbol{x}) = \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) - \gamma$. One reformulates the dual as

$$(f, g)^{\mathrm{L}}_{\mathbb{R}^n} = \sup\{\gamma \ : \ \boldsymbol{\lambda} \geq \mathbf{0}, \ \gamma \text{ in } \mathbb{R}, \ \psi(\gamma, \boldsymbol{\lambda})(\boldsymbol{x}) \geq 0 \text{ for all } \boldsymbol{x} \text{ in } \mathbb{R}^n\},$$

and the constraint that "$\psi(\gamma, \boldsymbol{\lambda})$ defines a nonnegative function" is then tightened to "$\psi(\gamma, \boldsymbol{\lambda})$ satisfies a particular sufficient condition for nonnegativity." The expectation is that the sufficient condition can be expressed by tractable convex constraints on variables γ and $\boldsymbol{\lambda}$. For example, SOS certificates for polynomial nonnegativity can be expressed via linear matrix inequalities, and SAGE certificates for signomial and polynomial nonnegativity can be expressed with the relative entropy function.

2.2 Global nonnegativity certificates

A signomial $f = \mathsf{Sig}(\boldsymbol{\alpha}, \boldsymbol{c})$ is called an *AM/GM-Exponential* or an *AGE function* if the coefficient vector $\boldsymbol{c}$ has at most one negative component, and f is nonnegative on $\mathbb{R}^n$ [15]. For an index $k \in [m]$, we define the kth AGE cone

$$\mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, k) = \{\boldsymbol{c} : \boldsymbol{c}_{\setminus k} \geq \mathbf{0} \text{ and } \boldsymbol{c} \text{ belongs to } \mathsf{C}_{\mathsf{NNS}}(\boldsymbol{\alpha})\}.$$

By applying duality to a suitable convex program, it may be shown that a vector $\boldsymbol{c}$ belongs to $\mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, k)$ if and only if

$$\text{some} \quad \boldsymbol{v} \text{ in } \mathbb{R}^{m-1} \quad \text{has} \quad [\boldsymbol{\alpha}_{\setminus k} - \mathbf{1}\boldsymbol{\alpha}_k]^{\mathsf{T}} \boldsymbol{v} = \mathbf{0} \quad \text{and} \quad D(\boldsymbol{v}, e\boldsymbol{c}_{\setminus k}) \leq c_k \tag{2}$$

where we have used the convention that $D(\boldsymbol{u}, \boldsymbol{v}) = +\infty$ for $(\boldsymbol{u}, \boldsymbol{v}) \notin \mathbb{R}^m_+ \times \mathbb{R}^m_+$. The system of constraints given by (2) is jointly convex in $\boldsymbol{c}$ and the auxiliary variable $\boldsymbol{v}$; therefore the set defined by the *sum* of all AGE cones

$$\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}) \doteq \left\{ \boldsymbol{c} : \ \text{there exist } \boldsymbol{c}^{(k)} \text{ in } \mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, k) \text{ where } \boldsymbol{c} = \sum_{k=1}^{m} \boldsymbol{c}^{(k)} \right\} \tag{3}$$

is tractable.

SAGE signomials can be used to certify global polynomial nonnegativity (see [16]). For a matrix $\boldsymbol{\alpha} \in \mathbb{N}^{m \times n}$ and a vector $\boldsymbol{c}$ in $\mathbb{R}^m$, we define the set of *signomial representative coefficient vectors* as

$$\begin{aligned} \mathrm{SR}(\boldsymbol{\alpha}, \boldsymbol{c}) = \{\hat{\boldsymbol{c}} : \ & \hat{c}_i = c_i \text{ whenever } \boldsymbol{\alpha}_i \text{ is in } 2\mathbb{N}^{1\times n}, \text{ and} \\ & \hat{c}_i \leq -|c_i| \text{ whenever } \boldsymbol{\alpha}_i \text{ is not in } 2\mathbb{N}^{1\times n}\}. \end{aligned}$$

Using a termwise argument, if $\hat{\boldsymbol{c}}$ belongs to $\mathrm{SR}(\boldsymbol{\alpha}, \boldsymbol{c})$ and $\mathsf{Sig}(\boldsymbol{\alpha}, \hat{\boldsymbol{c}})$ is nonnegative on $\mathbb{R}^n$, then $\mathsf{Pol}(\boldsymbol{\alpha}, \boldsymbol{c})$ must likewise be nonnegative on $\mathbb{R}^n$. We define the cone of coefficients for SAGE polynomials as

$$\mathsf{C}_{\mathsf{SAGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}) \doteq \{\boldsymbol{c} \ : \ \mathrm{SR}(\boldsymbol{\alpha}, \boldsymbol{c}) \cap \mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}) \text{ is nonempty } \}. \tag{4}$$

Alternatively, one may define an *AGE polynomial* as a nonnegative polynomial $\mathsf{Pol}(\boldsymbol{\alpha}, \boldsymbol{c})$ where at most one term $c_i \boldsymbol{x}^{\boldsymbol{\alpha}_i}$ attains a negative value as $\boldsymbol{x}$ varies over $\mathbb{R}^n$. Taking sums of such functions will also recover the cone in (4).

The theory of ordinary SAGE certificates has connections to a long-running history of similar nonnegativity certificates. The earliest developments here are the *agiforms* introduced by Reznick [20]. More recently, Pébay, Rojas, and Thompson studied maximization of circuit functions [21], Pantea, Koeppl, and Craciun introduced *monomial dominating posynomials* [22],[1] and Iliman and de Wolff proposed *Sums-of-Nonnegative-Circuit-Polynomials* (SONC) [17]. When polynomial SAGE certificates were introduced, it was shown that polynomial admits a SAGE decomposition if and only if it admits a SONC decomposition [16, Corollary 21]; this led to the first polynomial-time algorithm for optimizing over cones of SONC polynomials [16, Theorem 16].[2] Most recently, Katthän, Naumann, and Theobald proposed a class of SAGE-like functions which mix polynomials and geometric-form signomials [25]; the techniques presented in this article apply to such functions with straightforward changes. Some unconstrained optimization features of `sageopt` have counterparts in the POEM python package [26,27].

Lastly we mention *Sums-of-Squares* (SOS) polynomials. A polynomial f is said to be SOS if it can be written in the form $f = \sum_{i=1}^m f_i^2$ for appropriate polynomials f_i. In the context of polynomial optimization, one usually parameterizes the SOS cone by a number of variables n and a maximum degree $2d$; this cone can be represented as

$$\mathrm{SOS}(n, 2d) = \{p \ : \ p(\boldsymbol{x}) = L_d^n(\boldsymbol{x})^\mathsf{T} \boldsymbol{M} L_d^n(\boldsymbol{x}), \ \boldsymbol{M} \succeq \boldsymbol{0}\}$$

where $L_d^n : \mathbb{R}^n \to \mathbb{R}^{\binom{n+d}{d}}$ is the map from a vector $\boldsymbol{x}$ to the vector of all monomials of degree at-most-d evaluated at $\boldsymbol{x}$. The connection between SOS-representability and semidefinite programming was first observed by Shor [12], and was subsequently developed by Parrilo [13] and Lasserre [14]. GloptiPoly3 [28] and SOSTOOLS [29] are the SOS counterparts to `sageopt`.

[1] See also August, Koeppl, and Craciun [23].

[2] See [16, Section 5] for discussion on this topic and related results by Wang [24].

Although the bulk of the preceding discussion has been on polynomials, we must emphasize that SAGE is first and foremost about *signomials*.

2.3 Strengthening dual bounds in nonnegativity relaxations

A common method for strengthening dual problems is to introduce redundant constraints to the primal problem, particularly by taking products of existing constraint functions. As an example of this principle in action, consider the toy polynomial optimization problem

$$\inf\{-x^2 \,:\, -1 \leq x \leq 1\} = -1.$$

One may verify that $(f, g)^{\mathrm{L}}_{\mathbb{R}} = -\infty$, but by adding the single redundant constraint $(1-x)(1+x) \geq 0$, we can certify a dual bound $-1 \leq (f, g)^{\star}_{\mathbb{R}}$.

A more subtle method is to reconsider what is meant by "dual variables." For the Lagrange dual problem we use scalars $\lambda_i \geq 0$, however it is just as valid to have λ_i be a *function*, provided it is nonnegative over $\mathbb{R}^n$. Such a method is well-suited to our nonnegativity-based relaxations of the dual problem. The following toy signomial program illustrates the utility of this approach

$$\inf\{-\exp(2x) \,:\, 1 \leq \exp(x) \leq 2\} = -4.$$

Again the Lagrange dual problem returns a bound of $-\infty$, but by considering $\lambda_i(x) = \eta_i \exp(x)$ with $\eta_i \geq 0$, the resulting dual bound is $-4 \leq (f, g)^{\star}_{\mathbb{R}}$.

A third method for strengthening dual bounds only becomes relevant when working with strict inner-approximations of nonnegativity cones. For two functions w, f with w positive definite, it is clear that f is nonnegative if and only if the product $w \cdot f$ is nonnegative. The *method of modulation* is to choose a generic positive-definite function w so that if f fails a particular test for nonnegativity (say, being SOS, or being SAGE), there is still a chance that the product $w \cdot f$ passes a test for nonnegativity. Indeed, modulation is a crucial tool for computing successive bounds for unconstrained problems

$$f^{\star}_{\mathbb{R}^n} \doteq \inf\{f(\boldsymbol{x}) : \boldsymbol{x} \text{ in } \mathbb{R}^n\} = \sup\{\gamma \,:\, f(\boldsymbol{x}) - \gamma \geq 0 \text{ for all } \boldsymbol{x} \text{ in } \mathbb{R}^n\}.$$

Suppose for example that f is a signomial over exponents $\boldsymbol{\alpha}$; then for $w = \mathsf{Sig}(\boldsymbol{\alpha}, \mathbf{1})$ we can compute a non-decreasing sequence of lower bounds

$$f^{(\ell)}_{\mathbb{R}^n} = \sup\{\gamma : \gamma \text{ in } \mathbb{R},\ w^{\ell}(f-\gamma) \text{ is SAGE}\} \leq f^{\star}_{\mathbb{R}^n}.$$

For suitable conditions on $\boldsymbol{\alpha}$ (c.f. [15, Theorem 4.1]), we have $f^{(\ell)}_{\mathbb{R}^n} \to f^{\star}_{\mathbb{R}^n}$ as ℓ tends to infinity. From an implementation perspective, the constraint that "$\psi(\gamma) \doteq w^{\ell}(f-\gamma)$ is SAGE" is tractable because the coefficient vector of $\psi(\gamma)$ is an affine function of γ.

Modulation can similarly be applied to constrained optimization. Suppose that $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda})$ is the Lagrangian for Problem 1.1, and refer to the function $\boldsymbol{x} \mapsto \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda})$ as $\mathcal{L}(\boldsymbol{\lambda})$. Then rather than requiring that "$\mathcal{L}(\boldsymbol{\lambda}) - \gamma$ is SAGE", one can require that "$\psi(\gamma, \boldsymbol{\lambda}) \doteq w^{\ell}(\mathcal{L}(\boldsymbol{\lambda}) - \gamma)$ is SAGE." This increases the size of the feasible set for variables γ and $\boldsymbol{\lambda}$, and remains tractable due to the affine dependence of $\psi(\gamma, \boldsymbol{\lambda})$ on γ and $\boldsymbol{\lambda}$. Such modulation leads to a non-decreasing sequence of bounds which converge to $(f, g)^{\mathrm{L}}_{\mathbb{R}^n}$ under suitable conditions.

2.4 Partial dualization and conditional moment relaxations

Partial dualization is a technique for strengthening dual bounds, which is at least as strong as any choice of redundant constraints or nonconstant Lagrange multipliers. Considering (1.1) now with $X \subsetneq \mathbb{R}^n$, the natural generalization of the Lagrange dual is

$$(f, g)^{\mathrm{d}}_{X} \doteq \sup\{\gamma \,:\, \boldsymbol{\lambda} \geq \mathbf{0},\ \gamma \text{ in } \mathbb{R},\ \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) - \gamma \geq 0 \text{ for all } \boldsymbol{x} \text{ in } X\}. \tag{5}$$

In the important case when X is compact, we are guaranteed to have $(f, g)^{\mathrm{d}}_{X} > -\infty$, a property which is in stark contrast to the Lagrange dual. We call (5) a *partial dual* if $X = \{\boldsymbol{x} : g_i(\boldsymbol{x}) \geq 0 \text{ for all } i \text{ in } I\}$ was constructed from some subset $I \subset [k]$ of the constraint functions. Note that in the extreme case with $X = \{\boldsymbol{x} : g(\boldsymbol{x}) \geq \mathbf{0}\}$, we have $(f, 0)^{\mathrm{d}}_{X} = (f, g)^{\star}_{\mathbb{R}^n}$ – in this way, partial dualization provides a mechanism to completely eliminate duality gaps.

We now provide a simple example which combines partial dualization and nonnegativity certificates. Suppose we want to minimize a univariate polynomial f over an interval $[a, b]$, subject to a polynomial inequality constraint $g(x) \geq 0$. In this case we may form a Lagrangian $\mathcal{L}(x, \mu) = f(x) - \lambda g(x)$ with $\lambda \geq 0$, and find the largest constant γ so that $x \mapsto \mathcal{L}(x, \lambda) - \gamma$ is nonnegative over $x \in [a, b]$. A result by Powers and Reznick states that a degree-d polynomial "p" is nonnegative over an interval $[a, b]$ if and only if it can be written as $p(x) = s(x)^2 + h_{[a,b]}(x)t(x)^2$, where $h_{[a,b]}(x) = (b - x)(x - a)$, and s, t are polynomials of degree at most d and $d - 1$ respectively [30]. Therefore the partial dual $(f, g)^{\mathrm{d}}_{[a,b]}$ can be framed as an SOS relaxation

$$\begin{aligned}(f, g)^{\mathrm{d}}_{[a,b]} = \sup\{\gamma \,:\, & f - \lambda g - \gamma = s + t \cdot h_{[a,b]},\ \ \lambda \geq 0,\\ & s \in \mathrm{SOS}(1, 2d),\ t \in \mathrm{SOS}(1, 2(d-1))\}.\end{aligned}$$

Our last concept is how partial dualization manifests in the dual of the dual, also known as the *moment relaxation*. Consider $f = \mathsf{Pol}(\boldsymbol{\alpha}, \boldsymbol{c})$ with $\boldsymbol{\alpha}_1 = \mathbf{0}$, a set $X \subset \mathbb{R}^n$, and constraint functions $g_i = \mathsf{Pol}(\boldsymbol{\alpha}, \boldsymbol{g}_i)$ with coefficient vectors forming the rows of a matrix $\boldsymbol{G}$. The partial dual $(f, g)^{\mathrm{d}}_{X}$ can be written as the convex cone program

$$(f, g)^{\mathrm{d}}_{X} = \sup\{\gamma \,:\, \boldsymbol{\lambda} \geq \mathbf{0},\ \gamma \in \mathbb{R},\ \boldsymbol{c} - \gamma \boldsymbol{e}_1 - \boldsymbol{G}^{\mathsf{T}}\boldsymbol{\lambda} \in \mathsf{C}_{\mathsf{NNP}}(\boldsymbol{\alpha}, X)\},$$

and we can apply conic duality to obtain

$$(f, g)_X^{\mathrm{d}} = \inf\{\boldsymbol{c}^{\mathsf{T}}\boldsymbol{v} \,:\, \boldsymbol{v}^{\mathsf{T}}\boldsymbol{e}_1 = 1,\ \boldsymbol{G}\boldsymbol{v} \geq \boldsymbol{0},\ \boldsymbol{v} \in \mathsf{C}_{\mathsf{NNP}}(\boldsymbol{\alpha}, X)^{\dagger}\}.$$

The set $\mathsf{C}_{\mathsf{NNP}}(\boldsymbol{\alpha}, X)^{\dagger}$ is the closed cone generated by all vector-valued expectations $\mathbb{E}_{\boldsymbol{x}\sim F}[(\boldsymbol{x}^{\boldsymbol{\alpha}_1}, \ldots, \boldsymbol{x}^{\boldsymbol{\alpha}_m})]$, where F is a probability measure conditioned on $\boldsymbol{x} \in X$. Moment relaxations can be used in solution recovery schemes to certify $(f, g)_X^{\mathrm{d}} = (f, g)_X^{\star}$; see [14,31] for SOS-based moment relaxations and solution recovery methods.

3 Conditional SAGE certificates for signomials

In this section we show how SAGE certificates for signomial nonnegativity can fully leverage partial dualization, in the sense that any tractable convex set X gives rise to a parameterized and similarly tractable "X-SAGE" nonnegativity cone. The efficient representation of the X-SAGE cones (which we often call "conditional SAGE cones") leads to a practical, principled approach for solving and approximating a range of nonconvex signomial optimization problems. In this regard the most common sets X are of the form $\{\boldsymbol{x} : g(\boldsymbol{x}) \leq \boldsymbol{1}\}$ for signomials g_i with all nonnegative coefficients. An algorithm for solution recovery, and two worked examples are provided.

3.1 The conditional SAGE signomial cones

A signomial $\mathsf{Sig}(\boldsymbol{\alpha}, \boldsymbol{c})$ is called *X-AGE* if it is nonnegative on X, and at most one component of $\boldsymbol{c}$ is negative. The kth X-AGE cone for signomials over exponents $\boldsymbol{\alpha} \in \mathbb{R}^{m\times n}$ is

$$\mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, k, X) = \{\boldsymbol{c} \,:\, \boldsymbol{c}_{\setminus k} \geq \boldsymbol{0} \text{ and } \boldsymbol{c} \text{ belongs to } \mathsf{C}_{\mathsf{NNS}}(\boldsymbol{\alpha}, X)\}.$$

Note that X-AGE cones are defined for arbitrary $X \subset \mathbb{R}^n$, including nonconvex sets, and convex sets which admit no efficient description. A signomial $\mathsf{Sig}(\boldsymbol{\alpha}, \boldsymbol{c})$ is called *X-SAGE* if the coefficient vector $\boldsymbol{c}$ belongs to

$$\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X) \doteq \sum_{k=1}^{m} \mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, k, X).$$

It is possible to prove a range of results for X-SAGE signomials without knowing how to check membership in $\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)$. Here is one such result.

Corollary 1 *Let $X \subset \mathbb{R}^n$ be arbitrary. If $\boldsymbol{c}$ is a vector in $\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)$ with nonempty $N = \{i : c_i < 0\}$, then there exist vectors $\{\boldsymbol{c}^{(i)}\}_{i\in N}$ satisfying*

$$\boldsymbol{c}^{(i)} \in \mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, i, X) \qquad \boldsymbol{c} = \sum_{i\in N} \boldsymbol{c}^{(i)} \quad \textit{and} \quad c_j^{(i)} = 0 \textit{ for all } j \neq i \textit{ in } N.$$

Proof The cones $\mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, i, X)$ are of the form $C_i = \{\boldsymbol{c} \in K \,:\, \boldsymbol{c}_{\setminus i} \geq \boldsymbol{0}\}$ for a convex cone $K = \mathsf{C}_{\mathsf{NNS}}(\boldsymbol{\alpha}, X)$ which contains $\mathbb{R}^m_+$, and so they satisfy the hypothesis of [16, Lemma 6]. Therefore the proof of the known case with $X = \mathbb{R}^n$ ([16, Theorem 3]) generalizes immediately. □

Corollary 1 is important for theoretical and practical reasons. At a theoretical level, it says there is no loss of generality in restricting X-SAGE decompositions of a signomial $\mathsf{Sig}(\boldsymbol{\alpha}, \boldsymbol{c})$ to those with X-AGE functions also supported on $\boldsymbol{\alpha}$. Therefore it is equivalent to say "an X-SAGE function is a sum of X-AGE functions," without making reference to a fixed set of exponent vectors. At a practical level, Corollary 1 enables fast and substantial presolve procedures when using SAGE relaxations for signomial optimization.

To further illustrate that representations of $\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)$ are not required to prove results for X-SAGE signomials, we provide two elementary propositions concerning optimization with conditional SAGE certificates. For $f = \mathsf{Sig}(\boldsymbol{\alpha}, \boldsymbol{c})$ with $\boldsymbol{\alpha}_1 = \mathbf{0}$, define

$$f_X^{\mathsf{SAGE}} \doteq \sup\{\gamma \,:\, \gamma \text{ in } \mathbb{R},\ \boldsymbol{c} - \gamma \boldsymbol{e}_1 \text{ in } \mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)\}$$

so that $f_X^{\mathsf{SAGE}} \leq f_X^\star \doteq \inf\{f(\boldsymbol{x}) : \boldsymbol{x} \text{ in } X\}$.

Proposition 1 *If $\boldsymbol{c} \geq \mathbf{0}$, then $f = \mathsf{Sig}(\boldsymbol{\alpha}, \boldsymbol{c})$ has $f_X^{\mathsf{SAGE}} = f_X^\star$ for all $X \subset \mathbb{R}^n$.*

Proof The signomial $\tilde{f} = \mathsf{Sig}(\boldsymbol{\alpha}, \boldsymbol{c} - f_X^\star \boldsymbol{e}_1)$ is nonnegative over X, and its coefficient vector $\boldsymbol{c} - f_X^\star \boldsymbol{e}_1$ contains at most one negative entry. This implies that $\tilde{f}$ is X-AGE, and hence X-SAGE. □

Proposition 2 *If X is bounded, then $f_X^{\mathsf{SAGE}} > -\infty$ for every signomial f.*

Proof If X is empty then the result follows by verifying that $\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X) = \mathbb{R}^m$. Consider the case when X is nonempty. In this situation it suffices to prove the result for all f of the form $f(\boldsymbol{x}) = c\exp(\boldsymbol{a}\cdot\boldsymbol{x})$ where $c \neq 0$ and $\boldsymbol{a}$ belongs to $\mathbb{R}^{1\times n}$. Fixing such $c, \boldsymbol{a}$, the boundedness of X implies the existence of $L \neq 0$ with $\tilde{f}(\boldsymbol{x}) = c\exp(\boldsymbol{a}\cdot\boldsymbol{x}) + L$ nonnegative over $\boldsymbol{x}$ in X and $cL < 0$. Since $\tilde{f}$ is nonnegative over X and contains exactly one negative coefficient, we have that $f_X^{\mathsf{SAGE}} \geq -L$. □

Proposition 2 shows how convex relaxations based on conditional SAGE certificates respect compactness, as with the partial-duals from Sect. 2.4.

Now we turn to the computationally essential question of how to represent cones of X-SAGE signomials. The following theorem demonstrates that if X is a tractable convex set, then so is $\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)$.

Theorem 1 *For a matrix $\boldsymbol{\alpha}$ in $\mathbb{R}^{m\times n}$, an index k in $[m]$, and a convex set $X \subset \mathbb{R}^n$ with support function $\sigma_X(\boldsymbol{\lambda}) \doteq \sup_{\boldsymbol{x}\in X} \boldsymbol{\lambda}^\mathsf{T}\boldsymbol{x}$, we have*

$$\begin{aligned}\mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, k, X) = \{\boldsymbol{c} \in \mathbb{R}^m :\ & \text{there exist } \boldsymbol{v} \text{ in } \mathbb{R}^{m-1} \text{ and } \boldsymbol{\lambda} \text{ in } \mathbb{R}^n \\ & \text{satisfying } \ [\boldsymbol{\alpha}_{\setminus k} - \mathbf{1}\boldsymbol{\alpha}_k]^\mathsf{T}\boldsymbol{v} + \boldsymbol{\lambda} = \mathbf{0} \\ & \text{and } \ \sigma_X(\boldsymbol{\lambda}) + D(\boldsymbol{v}, e\boldsymbol{c}_{\setminus k}) \leq c_k\}.\end{aligned}$$

Proof Let δ_X denote the indicator function of X, taking values $\delta_X(\boldsymbol{x}) = 0$ when $\boldsymbol{x} \in X$, and $\delta_X(\boldsymbol{x}) = +\infty$ otherwise. A vector $\boldsymbol{c}$ with $\boldsymbol{c}_{\setminus k} \geq \mathbf{0}$ belongs to $\mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, k, X)$ if and

only if

$$p^\star = \inf \left\{ \delta_X(\boldsymbol{x}) + \sum_{i=1}^{\ell} \tilde{c}_i \exp t_i \ : \ \boldsymbol{x} \in \mathbb{R}^n, \ \boldsymbol{t} \in \mathbb{R}^\ell, \ \boldsymbol{t} = \boldsymbol{W}\boldsymbol{x} \right\} \geq -L \tag{6}$$

for $\ell = m - 1$, $\boldsymbol{W} = [\boldsymbol{\alpha}_{\setminus k} - \mathbf{1}\boldsymbol{\alpha}_k] \in \mathbb{R}^{\ell \times n}$, $\tilde{\boldsymbol{c}} = \boldsymbol{c}_{\setminus k} \in \mathbb{R}^\ell$, and $L = c_k$.

The dual to the above optimization problem is easily calculated by applying Fenchel duality (c.f. [32]); the result of this process is

$$d^\star = \sup\{-\sigma_X(\boldsymbol{\lambda}) - D(\boldsymbol{v}, e\tilde{\boldsymbol{c}}) \ : \ \boldsymbol{\lambda} \in \mathbb{R}^n, \boldsymbol{v} \in \mathbb{R}^{m-1}, \ \boldsymbol{W}^\mathsf{T}\boldsymbol{v} + \boldsymbol{\lambda} = \mathbf{0}\}. \tag{7}$$

When X is nonempty, one may verify that the hypothesis of [32, Corollary 3.3.11] (concerning strong duality) hold for the primal-dual pair (6)–(7). In particular, $p^\star \geq -L$ holds if and only if $-d^\star \leq L$, and the dual problem attains an optimal solution whenever finite. When X is empty, it is clear that $p^\star = +\infty$, and by taking both $\boldsymbol{\lambda}$ and $\boldsymbol{v}$ as zero vectors, we have $d^\star = +\infty$. The result follows. □

Theorem 1 is stated in terms of support functions for maximum generality. From an implementation perspective, it is useful to assume a representation of X. For example, if $X = \{\boldsymbol{x} \ : \ \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b} \in K\}$ for a matrix $\boldsymbol{A}$, a vector $\boldsymbol{b}$, and a convex cone K, then weak duality ensures

$$\sigma_X(\boldsymbol{\lambda}) \doteq \sup\{\boldsymbol{\lambda}^\mathsf{T}\boldsymbol{x} \ : \ \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b} \in K\} \leq \inf\{\boldsymbol{b}^\mathsf{T}\boldsymbol{\eta} \ : \ \boldsymbol{A}^\mathsf{T}\boldsymbol{\eta} + \boldsymbol{\lambda} = \mathbf{0}, \ \boldsymbol{\eta} \in K^\dagger\}.$$

The above is all we need to construct an inner-approximation of a given AGE cone. For all $X = \{\boldsymbol{x} \ : \ \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b} \in K\}$, we have

$$\begin{aligned} \{\boldsymbol{c} \in \mathbb{R}^m \ : \ & \text{there exist } \boldsymbol{v} \text{ in } \mathbb{R}^{m-1} \text{ and } \boldsymbol{\eta} \text{ in } K^\dagger \\ & \text{satisfying } \ [\boldsymbol{\alpha}_{\setminus k} - \mathbf{1}\boldsymbol{\alpha}_k]^\mathsf{T}\boldsymbol{v} = \boldsymbol{A}^\mathsf{T}\boldsymbol{\eta} \\ & \text{and } \ D(\boldsymbol{v}, e\boldsymbol{c}_{\setminus k}) + \boldsymbol{\eta}^\mathsf{T}\boldsymbol{b} \leq c_k, \} \subset \mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, k, X). \end{aligned}$$

If there exists an $\boldsymbol{x}_0$ where $\boldsymbol{A}\boldsymbol{x}_0 + \boldsymbol{b}$ is in the relative interior of K, then by Slater's condition the reverse inclusion in the preceding expression also holds.

The present article only considers X-SAGE signomials when X is a convex set, however there remains the possibility of using X-SAGE decompositions to certify nonnegativity when X is nonconvex. To give an example of when this is possible, suppose $X = X_1 \cup X_2$ where X_1 and X_2 are convex sets. In this case we trivially have that $\mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, k, X)$ is the intersection of $\mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, k, X_1)$ and $\mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, k, X_2)$, and so $\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)$ inherits a representation from Theorem 1.

3.2 Dual perspectives and solution recovery

Dual SAGE relaxations can be used to recover optimal and near-optimal solutions to signomial programs of the form (1). For concreteness, we state the simplest such relaxation here. Let f, $\{g_i\}_{i=1}^{k_1}$ and $\{\phi_i\}_{i=1}^{k_2}$ be signomials over exponents $\boldsymbol{\alpha}$, with

$\boldsymbol{\alpha}_1 = \mathbf{0}$. If $\boldsymbol{c}$ is the coefficient vector of f, and the rows of $\boldsymbol{G} \in \mathbb{R}^{k_1 \times m}$, $\boldsymbol{\Phi} \in \mathbb{R}^{k_2 \times m}$ specify coefficient vectors of g_i, ϕ_i respectively, then

$$\inf\{\boldsymbol{c}^{\mathsf{T}}\boldsymbol{v} \,:\, \boldsymbol{v} \in \mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)^{\dagger},\ v_1 = 1,\ \boldsymbol{G}\boldsymbol{v} \geq \mathbf{0},\ \boldsymbol{\Phi}\boldsymbol{v} = \mathbf{0}\} \tag{8}$$

is a convex relaxation of Problem 1. By standard rules in convex analysis, the dual SAGE cone is given by the intersection of the constituent dual AGE cones. An expression for the dual AGE cones can be recovered from Theorem 1 in the case when X is a convex set. Using $\operatorname{co} X = \{(\boldsymbol{x}, t) \,:\, t > 0,\ \boldsymbol{x}/t \in X\}$ to denote the cone over X, the usual conic-duality calculations yield

$$\begin{aligned}\mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, i, X)^{\dagger} = \operatorname{cl}\{\boldsymbol{v} :\ & v_i \log(\boldsymbol{v}/v_i) \geq [\boldsymbol{\alpha} - \mathbf{1}\boldsymbol{\alpha}_i]\boldsymbol{z}_i \\ & (\boldsymbol{z}_i, v_i) \in \operatorname{co} X,\ \boldsymbol{v} \text{ in } \mathbb{R}^m_{++},\ \text{and } \boldsymbol{z}_i \text{ in } \mathbb{R}^n\}.\end{aligned} \tag{9}$$

The auxiliary variables "$\boldsymbol{z}$" appearing in (9) are a powerful tool for solution recovery: if $\boldsymbol{z}_i, \boldsymbol{v}$ satisfy the constraints in (9), then $\boldsymbol{x} \doteq \boldsymbol{z}_i/v_i$ belongs to X. Beyond taking individual ratios, we note that for any index set $I \subset [m]$ of AGE cones, we have $\left(\sum_{i\in I} \boldsymbol{z}_i\right) / \left(\sum_{i\in I} v_i\right) \in X$. The ability to unconditionally recover X-feasible points by perspective transforms of a dual solution is an important aspect of conditional SAGE certificates.

In general there remain issues of recovering *optimal* points, and recovering solutions when some constraints cannot be pushed into X. Both of these issues can be resolved if some $\boldsymbol{x}^{\star} \in X$ satisfies $\exp(\boldsymbol{\alpha}\boldsymbol{x}^{\star}) = \boldsymbol{v}$ (as happens when all relative entropy constraints in (9) are binding and we meet one additional normalization condition). However it is possible that a SAGE relaxation produces a tight bound, and yet we cannot find a point $\boldsymbol{x}^{\star} \in X$ with $\exp(\boldsymbol{\alpha}\boldsymbol{x}^{\star}) = \boldsymbol{v}$. Therefore it is beneficial to include heuristics in the solution recovery process. Our basic solution recovery algorithm is given below.

Algorithm 1 signomial solution recovery from dual SAGE relaxations.

Input: Signomials f, $\{g_i\}_{i=1}^{k_1}$, and $\{\phi_i\}_{i=1}^{k_2}$ over exponents $\boldsymbol{\alpha}$ in $\mathbb{R}^{m\times n}$. A vector $\boldsymbol{v}$ in $\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)^{\dagger}$. Infeasibility tolerances $\epsilon_{\text{ineq}}, \epsilon_{\text{eq}} \geq 0$.

1: **procedure** SigSolutionRecovery($f, g, \phi, \boldsymbol{\alpha}, \boldsymbol{v}, \epsilon_{\text{ineq}}, \epsilon_{\text{eq}}$)
2: solutions $\leftarrow$ []
3: **for** $j = 1, \ldots,$ length($\boldsymbol{v}$) **do**
4: Recover $\boldsymbol{z}_i$ in $\mathbb{R}^n$ s.t. $v_i \log(\boldsymbol{v}/v_i) \geq [\boldsymbol{\alpha} - \mathbf{1}\boldsymbol{\alpha}_i]\boldsymbol{z}_i$ and $(\boldsymbol{z}_i, v_i) \in \operatorname{co} X$.
5: solutions.append($\boldsymbol{z}_i/v_i$).
6: **if** $\boldsymbol{\alpha}\boldsymbol{x} \neq \log \boldsymbol{v}$ for all $\boldsymbol{x}$ in solutions **then**
7: Compute $\boldsymbol{x}_{\text{ls}}$ in argmin$\{\|\log \boldsymbol{v} - \boldsymbol{\alpha}\boldsymbol{x}\| \,:\, \boldsymbol{x} \text{ in } X\}$.
8: solutions.append($\boldsymbol{x}_{\text{ls}}$).
9: solutions $\leftarrow$ [$\boldsymbol{x}$ in solutions if $g(\boldsymbol{x}) \geq -\epsilon_{\text{ineq}}$ and $|\phi(\boldsymbol{x})| \leq \epsilon_{\text{eq}}$].
10: solutions.sort(f, increasing).
11: **return** solutions.

Assuming (9) is used to represent $\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)^{\dagger}$, Algorithm 1's runtime is dominated by the constrained least-squares problem in Line 7. Note the only projective transformations used in Algorithm 1 are those with index sets $I = \{i\}$; this is due to

a present lack of theory for identifying which of the exponentially-many index sets $I \subset [m]$ might be useful for solution recovery. It is highly desirable to develop a systematic theory of solution recovery for dual X-SAGE relaxations, such as that found in the SOS-based *Lasserre relaxations* for polynomial optimization. In Lasserre relaxations, there are necessary and sufficient conditions for success of solution recovery based on a rank condition for dual variables to SOS multipliers (see [31] for a thorough treatment of this topic, and [33, Theorem 2.47] for a concise statement of such a result).

It is often useful to apply a local solver to the output of Algorithm 1. The term "Algorithm 1L" henceforth refers to the use of Algorithm 1, followed by solution refinement with Powell's COBYLA solver [34].[3] Our later experiments indicate that equality constraint functions ϕ_i can prevent finding a solution even when SAGE produces a tight bound on $(f, g, \phi)^\star_X$. We therefore suggest that one eliminate equality constraints through substitution of monomials $\exp(x_i)$, when possible. Alternatively, one can allow large violations of equality constraints in Algorithm 1, and pass the returned values as near-feasible points to a local solver as part of Algorithm 1L.

3.3 A first worked example

The following problem has appeared in many articles concerning algorithms for signomial programming [5,7–10].

$$\begin{aligned} \inf_{\boldsymbol{x}\in\mathbb{R}^3} \quad & f(\boldsymbol{x}) \doteq 0.5\exp(x_1 - x_2) - \exp x_1 - 5\exp(-x_2) \\ \text{s.t.} \quad & g_1(\boldsymbol{x}) \doteq 100 - \exp(x_2 - x_3) - \exp x_2 - 0.05\exp(x_1 + x_3) \geq 0 \\ & g_{2:4}(\boldsymbol{x}) \doteq \exp \boldsymbol{x} - (70,\ 1,\ 0.5) \geq \mathbf{0} \\ & g_{5:7}(\boldsymbol{x}) \doteq (150,\ 30,\ 21) - \exp \boldsymbol{x} \geq \mathbf{0} \end{aligned} \tag{Ex1}$$

This problem ("Example 1") is a good candidate for conditional SAGE relaxations, because each of the seven constraints defines a tractable convex set. The constraint functions $g_{2:7}$ can be represented with six linear inequalities, and the constraint function g_1 can be accommodated by three exponential cones and one linear inequality. Separately, Example 1 is interesting because the Lagrange dual problem performs poorly: regardless of how many products we take of existing constraint functions g_i, the "$-5\exp(-x_2)$" term in the objective will cause Lagrangians $f - \sum_I \lambda_I \prod_{j\in I} g_j$ to be unbounded below for all values of dual variables $\lambda_I \geq 0$.

Now we see how SAGE relaxations fare for Example 1. We begin by setting $X = \{\boldsymbol{x} : g_{1:7}(\boldsymbol{x}) \geq \mathbf{0}\}$; since X is bounded, Proposition 2 tells us f_X^{SAGE} is finite. The dual SAGE relaxation can be solved with MOSEK on Machine **L** in 0.01 s, and provides us with a lower bound $f_X^{\mathsf{SAGE}} = -147.86 \leq f_X^\star$. By running Algorithm 1 on the dual solution, we recover

$$\boldsymbol{x}^\star = (5.0106353,\ 3.4011966, -0.4845071) \quad \text{where} \quad f(\boldsymbol{x}^\star) = -147.66666.$$

[3] A FORTRAN implementation is accessible through SciPy's `optimize` submodule. The arguments we pass to that implementation are `RHOBEG`=1, `RHOEND` $= 10^{-7}$, and `MAXFUN`$= 10^5$.

Table 1 SAGE bounds for Example 1, with solver runtime for Machines **W** and **L**. A level-3 bound certifies the level-0 solution as optimal, within relative error 10^{-8}

Level	SAGE bound	**W** time (s)	**L** time (s)
0	−147.85713	0.03	0.01
1	−147.67225	0.05	0.02
2	−147.66680	0.08	0.08
3	−147.66666	0.19	0.26

From this solution, we know that the bound obtained from the SAGE relaxation is within 0.13% relative error of the true optimal value. The ability to recover near-optimal solutions even in the presence of a gap $f_X^{\mathsf{SAGE}} < f_X^\star$ can be attributed to how conditional SAGE certificates seamlessly integrate with convex duality and partial dualization. As it happens, the point $\boldsymbol{x}^\star$ returned by Algorithm 1 is actually *optimal* for Example 1; to certify this fact, we need stronger SAGE relaxations. Table 1 shows the results of such relaxations, using the minimax-free hierarchy described in Sect. 3.4.

3.4 Reference hierarchies for signomial programming

Here we give a particular set of choices regarding SAGE-based hierarchies for signomial programming. When we say "f and g_i are signomials over exponents $\boldsymbol{\alpha}$," we mean that $\{\boldsymbol{x} \mapsto \exp(\boldsymbol{\alpha}_j \cdot \boldsymbol{x})\}_{j=1}^m$ is the smallest monomial basis spanning all linear combinations of f, g_i, and the function $\boldsymbol{x} \mapsto 1$.

First we describe a SAGE-based hierarchy that does not make use of the minimax inequality, i.e. a hierarchy applicable when all constraints can be moved into X. If we cannot certify nonnegativity of $f - \gamma$ with $\gamma = f_X^\star$, we can use modulators as described in Sect. 2.3 to improve the largest SAGE-certifiable bound on f. Formally, for a signomial f over exponents $\boldsymbol{\alpha}$, a set $X \subset \mathbb{R}^n$, and an integer $\ell \geq 0$, the *level-ℓ SAGE relaxation* for $f_X^\star$ is

$$f_X^{(\ell)} \doteq \sup\{\gamma \,:\, \mathsf{Sig}(\boldsymbol{\alpha}, \mathbf{1})^\ell (f - \gamma) \text{ is } X\text{-SAGE}\}. \tag{10}$$

The special case with $\ell = 0$ was introduced earlier in this section as "f_X^{SAGE}."

Now we consider functional constraints; let f, g_i, and ϕ_i be signomials over exponents $\boldsymbol{\alpha}$. SAGE relaxations for the problem of computing $(f, g, \phi)_X^\star$ are indexed by three integer parameters: p, q, and ℓ. Starting from $p \geq 0$ and $q \geq 1$, define $\boldsymbol{\alpha}[p]$ as the matrix of exponent vectors for $\mathsf{Sig}(\boldsymbol{\alpha}, \mathbf{1})^p$, and define $g[q]$ as the set of all products of at-most-q elements of g (similarly define $\phi[q]$). The SAGE relaxation for $(f, g, \phi)_X^\star$ at level (p, q, ℓ) is then

$$\begin{aligned}(f, g, \phi)_X^{(p,q,\ell)} = \sup \;\; \gamma \;\; \text{s.t.} \;\; & s_h, z_h \text{ are signomials over exponents } \boldsymbol{\alpha}[p] \\ & \mathcal{L} \doteq f - \gamma - \textstyle\sum_{h \in g[q]} s_h \cdot h - \sum_{h \in \phi[q]} z_h \cdot h \\ & \mathsf{Sig}(\boldsymbol{\alpha}, \mathbf{1})^\ell \mathcal{L} \text{ is an } X\text{-SAGE signomial} \\ & s_h \text{ are } X\text{-SAGE signomials.}\end{aligned} \tag{11}$$

The decision variables in (11) are $\gamma \in \mathbb{R}$, the coefficient vectors of $\{s_h\}_{h\in g[q]}$, and the coefficient vectors of $\{z_h\}_{h\in\phi[q]}$. The most basic level of this hierarchy is $(p, q, \ell) = (0, 1, 0)$. This corresponds to using scalar Lagrange multipliers ($s_h \geq 0$ and $z_h \in \mathbb{R}$), the original constraints ($g[0] = g,\ \phi[0] = \phi$), and modulating the Lagrangian by the signomial that is identically equal to 1. Note that when $p > 0$, the Lagrange multipliers s_h need only be nonnegative on X, rather than over the whole of $\mathbb{R}^n$.

3.5 A second worked example

This section's example can be found in the 1976 PhD thesis of James Yan [35], where it illustrates signomial programming in the service of structural engineering design. This problem is nonconvex even when written in exponential form; such problems have received limited attention in the engineering design optimization community, largely due to a lack of reliable methods for solving them. We restate the problem here (as Example 2) in geometric form.[4]

$$\begin{aligned}
\inf_{\substack{\boldsymbol{A}\in\mathbb{R}^3_{++}\\ P\in\mathbb{R}_{++}}} \quad & 10^4(A_1 + A_2 + A_3) \\
\text{s.t.} \quad & 10^4 + 0.01A_1^{-1}A_3 - 7.0711A_1^{-1} \geq 0 \\
& 10^4 + 0.00854A_1^{-1}P - 0.60385(A_1^{-1} + A_2^{-1}) \geq 0 \\
& 70.7107A_1^{-1} - A_1^{-1}P - A_3^{-1}P = 0 \\
& 10^4 \geq 10^4 A_1 \geq 10^{-4} \qquad 10^4 \geq 10^4 A_2 \geq 7.0711 \\
& 10^4 \geq 10^4 A_3 \geq 10^{-4} \qquad 10^4 \geq 10^4 P \geq 10^{-4}
\end{aligned} \tag{Ex2}$$

Let $X \subset \mathbb{R}^4$ be the feasible set cut out by the eight bound constraints in Example 2. With an X-SAGE relaxation where all constraints appear in the Lagrangian, we obtain $(f, g, \phi)_X^{(0,1,0)} = 14.1423$ in 0.04 s of solver time. This bound is very close to the optimal value claimed by Yan [35]. However, Algorithm 1 only returns candidate solutions "$\boldsymbol{x}$" with equality constraint violations $\phi(\boldsymbol{x}) \approx 70$.

To improve our chances of solution recovery, we use the equality constraint to *define* the value $P \leftarrow 70.7107A_3/(A_3 + A_1)$. After clearing the denominator $(A_3 + A_1)$ for inequality constraints involving P, we obtain a signomial program in only the variables A_1, A_2, A_3. We solve a level-(0, 1, 0) dual conditional SAGE relaxation for this signomial program, and exponentiate the result of Algorithm 1 to recover

$$\boldsymbol{A} = (7.07110 \cdot 10^{-4}, 7.07110 \cdot 10^{-4}, 10^{-8}),\ P = \frac{70.7107A_3}{A_1 + A_3}.$$

This solution is feasible up to machine precision, and attains objective matching the 14.142300 SAGE bound. The entire process of solving the SAGE relaxation and recovering the optimal solution takes less than 0.05 s on Machine **W**.

[4] The objective and constraint functions are multiplied by 10^4 for numerical reasons; see equation environment (6.15) on page 106 of [35] for the original problem statement.

3.6 Remarks on "geometric-form" signomial programming

By now we have seen signomial programs in both exponential and geometric forms. The authors have so far preferred the exponential form, primarily because it allows us to build upon the substantial theories of convex analysis and convex optimization. However from an applications perspective, it is far more common to express signomial programs in geometric form. Here we briefly present a geometric-form parameterization of conditional SAGE certificates – both in effort to appeal to those who usually work with geometric-form signomial programs, and as a prelude to our discussion on polynomials.

Geometric form signomials $f(\boldsymbol{x}) = \sum_{i=1}^{m} c_i \boldsymbol{x}^{\boldsymbol{\alpha}_i}$ are defined at points $\boldsymbol{x}$ in $\mathbb{R}^n_{++}$, and so it only makes sense to discuss conditional nonnegativity cones for signomials over sets $X \subset \mathbb{R}^n_{++}$. Henceforth, define

$$\mathsf{C}^{\mathsf{GEO}}_{\mathsf{NNS}}(\boldsymbol{\alpha}, X) = \left\{ \boldsymbol{c} \,:\, \sum_{i=1}^{m} c_i \boldsymbol{x}^{\boldsymbol{\alpha}_i} \geq 0 \text{ for all } \boldsymbol{x} \text{ in } X \right\}$$

for matrices $\boldsymbol{\alpha}$ in $\mathbb{R}^{m \times n}$ and sets X contained in $\mathbb{R}^n_{++}$. By applying the change of variables $\boldsymbol{x} \mapsto \exp \boldsymbol{y}$ and considering the subsequent change of domain $X \mapsto \log X$, one may verify that $\mathsf{C}^{\mathsf{GEO}}_{\mathsf{NNS}}(\boldsymbol{\alpha}, X) = \mathsf{C}_{\mathsf{NNS}}(\boldsymbol{\alpha}, \log X)$. Thus for $X \subset \mathbb{R}^n_{++}$, one arrives naturally at the definition

$$\mathsf{C}^{\mathsf{GEO}}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X) \doteq \mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, \log X).$$

From here one may deduce various corollaries for $\mathsf{C}^{\mathsf{GEO}}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)$, by appealing to results from Sect. 3.1. The most important such result is that $\mathsf{C}^{\mathsf{GEO}}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)$ is tractable whenever $\log X$ is a tractable convex set. Recently, Agrawal, Diamond, and Boyd have provided a modeling framework to produce feasible sets X where $\log X$ is a tractable convex set [36].

4 Conditional SAGE certificates for polynomials

In the previous section we saw how conditional SAGE certificates for signomial nonnegativity are inextricably linked to convex duality. Here we show how the broader idea of conditional SAGE certificates can extend to polynomials. In this context it is not convexity of X that determines when an X-SAGE polynomial cone is tractable, but rather convexity of an appropriate logarithmic transform of X.

The organization of this section is similar to that of Sect. 3. Definitions, representations, and other basic theorems for the conditional SAGE polynomial cones are given in Sect. 4.1. Section 4.2 covers solution recovery from dual SAGE relaxations, and Sect. 4.3 provides a worked example with special focus on solution recovery. Section 4.4 describes reference hierarchies for optimization with conditional SAGE polynomial cones: one based on the minimax inequality, and one that is "minimax free". Section 4.5 applies various manifestations of the minimax hierarchy to an example problem.

4.1 The conditional SAGE polynomial cones

We call $f = \mathsf{Pol}(\boldsymbol{\alpha}, \boldsymbol{c})$ an *X-AGE polynomial* if it is nonnegative over X, and $f(\boldsymbol{x})$ contains at most one term $c_k \boldsymbol{x}^{\boldsymbol{\alpha}_k}$ which is negative for some $\boldsymbol{x}$ in X. The kth X-AGE polynomial cone is given by

$$\begin{aligned}\mathsf{C}_{\mathsf{AGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, k, X) = \{\boldsymbol{c} :\ & \mathsf{Pol}(\boldsymbol{\alpha}, \boldsymbol{c})(\boldsymbol{x}) \geq 0 \text{ for all } \boldsymbol{x} \text{ in } X,\\ & c_j \geq 0 \text{ if } j \neq k \text{ and } \boldsymbol{x}^{\boldsymbol{\alpha}_j} > 0 \text{ for some } \boldsymbol{x} \text{ in } X,\\ & c_j \leq 0 \text{ if } j \neq k \text{ and } \boldsymbol{x}^{\boldsymbol{\alpha}_j} < 0 \text{ for some } \boldsymbol{x} \text{ in } X\ \}.\end{aligned}$$

Naturally, $f = \mathsf{Pol}(\boldsymbol{\alpha}, \boldsymbol{c})$ is an *X-SAGE polynomial* if $\boldsymbol{c}$ belongs to

$$\mathsf{C}_{\mathsf{SAGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, X) \doteq \sum_{k=1}^{m} \mathsf{C}_{\mathsf{AGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, k, X).$$

Let us work through some consequences of the definition. For starters, if $\boldsymbol{x}^{\boldsymbol{\alpha}_j}$ takes on positive and negative values as $\boldsymbol{x}$ varies over X, then $c_j = 0$ whenever $\boldsymbol{c} \in \mathsf{C}_{\mathsf{AGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, k, X)$ and $k \neq j$. Note that $\boldsymbol{x}^{\boldsymbol{\alpha}_j}$ can only take on both positive and negative values when $\boldsymbol{\alpha}_j$ does not belong to the even integer lattice. If X contains an open ball around the origin, then $\boldsymbol{x}^{\boldsymbol{\alpha}_j}$ takes on both positive and negative values *if and only if* $\boldsymbol{\alpha}_j$ does not belong to the even integer lattice. Thus, the definition of X-AGE polynomials agrees with the definition of ordinary AGE polynomials, as proposed in [16]. Another important case is when X is a subset of the nonnegative orthant. This point is addressed in some detail later in this section; as a preliminary remark, we note that by considering the connection between polynomials and geometric-form signomials, one can easily see that if $X \subset \mathbb{R}^n_{++}$ then $\mathsf{C}_{\mathsf{AGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, k, X) = \mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, k, \log X)$.

Many of our earlier theorems for signomials directly apply to X-SAGE polynomials. For example, it is easy to show the polynomial analog to Proposition 2: if X is bounded, then $f = \mathsf{Pol}(\boldsymbol{\alpha}, \boldsymbol{c})$ with $\boldsymbol{\alpha}_1 = \mathbf{0}$ has

$$f_X^{\mathsf{SAGE}} = \sup\{\gamma\ :\ \gamma \text{ in } \mathbb{R},\ \boldsymbol{c} - \gamma \boldsymbol{e}_1 \text{ in } \mathsf{C}_{\mathsf{SAGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, X)\} > -\infty.$$

Corollary 1 likewise extends to polynomials. Other than substituting AGE signomial cones with AGE polynomial cones, the only difference is that N becomes $N = \{i : c_i \boldsymbol{x}^{\boldsymbol{\alpha}_i} < 0 \text{ for some } \boldsymbol{x} \text{ in } X\}$.

Now we turn to representation of SAGE polynomial cones. By applying a simple continuity argument one can show that if $X = \mathrm{cl}\, X^\circ \subset \mathbb{R}^n_+$ – where X° is the interior of X – then $\mathsf{C}_{\mathsf{SAGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, X) = \mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, \log X^\circ)$. This claim is strengthened slightly and made more explicit through the following theorem.

Theorem 2 *Suppose* $X = \mathrm{cl}\{\boldsymbol{x} : \mathbf{0} < \boldsymbol{x}, H(\boldsymbol{x}) \leq \mathbf{1}\}$ *for a continuous map* $H : \mathbb{R}^n \to \mathbb{R}^r$. *Then for* $Y = \{\boldsymbol{y} : H(\exp \boldsymbol{y}) \leq \mathbf{1}\}$, *we have* $\mathsf{C}_{\mathsf{SAGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, X) = \mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, Y)$.

The proof of Theorem 2 is straightforward, and hence omitted. A more sophisticated result concerns when X possesses a certain sign-symmetry.

Theorem 3 *Suppose* $X = \mathrm{cl}\{\boldsymbol{x} : \mathbf{0} < |\boldsymbol{x}|, H(|\boldsymbol{x}|) \leq \mathbf{1}\}$ *for a continuous map* $H : \mathbb{R}^n \to \mathbb{R}^r$. *Then for* $Y = \{\boldsymbol{y} : H(\exp \boldsymbol{y}) \leq \mathbf{1}\}$, *we have*

$$\mathsf{C}_{\mathsf{SAGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, X) = \{\boldsymbol{c} \ : \mathrm{SR}(\boldsymbol{\alpha}, \boldsymbol{c}) \cap \mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, Y) \text{ is nonempty } \}. \tag{12}$$

By combining Theorem 1 with Theorems 2 and 3, we know that there exist a range of sets X for which optimization over X-SAGE polynomials is tractable. There remains the potentially nontrivial task of formulating a problem so that one of these theorems provides an efficient representation of $\mathsf{C}_{\mathsf{SAGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, X)$; important examples of when this is possible include constraints such as

$$-a \leq x_j \leq a, \qquad \|\boldsymbol{x}\|_p \leq a, \qquad |\boldsymbol{x}^{\boldsymbol{\alpha}_i}| \geq a, \quad \text{and} \quad x_j^2 = a$$

where $a > 0$ is a fixed constant.

Proof of Theorem 3 Suppose that $\boldsymbol{c}$ in $\mathsf{C}_{\mathsf{SAGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, X)$ admits the decomposition $\boldsymbol{c} = \sum_{i=1}^m \boldsymbol{c}^{(i)}$, where $\boldsymbol{c}^{(i)}$ belongs to the ith AGE polynomial cone with respect to $\boldsymbol{\alpha}$, X. Define $\{\tilde{\boldsymbol{c}}^{(i)}\}_{i=1}^m$ as follows

$$\tilde{\boldsymbol{c}}_j^{(i)} = \begin{cases} -|c_j^{(i)}| & \text{if } \boldsymbol{\alpha}_i \text{ is not even, and } j = i \\ c_j^{(i)} & \text{if otherwise} \end{cases}.$$

By the invariance of X under reflection about hyperplanes $\{\boldsymbol{x} : x_j = 0\}$, and continuity of polynomials, we have that

$$\begin{aligned} 0 \leq \inf\{\mathsf{Pol}(\boldsymbol{\alpha}, \boldsymbol{c}^{(i)})(\boldsymbol{x}) \ : \ \boldsymbol{x} \text{ in } X\} &= \inf\{\mathsf{Pol}(\boldsymbol{\alpha}, \tilde{\boldsymbol{c}}^{(i)})(\boldsymbol{x}) \ : \ \boldsymbol{x} \text{ in } X \cap \mathbb{R}_+^n\} \\ &= \inf\{\mathsf{Sig}(\boldsymbol{\alpha}, \tilde{\boldsymbol{c}}^{(i)})(\boldsymbol{y}) \ : \ \boldsymbol{y} \text{ in } Y\}. \end{aligned}$$

The signomials $\mathsf{Sig}(\boldsymbol{\alpha}, \tilde{\boldsymbol{c}}^{(i)})$ are thus nonnegative over $Y = \{\boldsymbol{y} : H(\exp \boldsymbol{y}) \leq 1\}$, and posses at most one negative coefficient. This implies that $\tilde{\boldsymbol{c}} \doteq \sum_{i=1}^m \tilde{\boldsymbol{c}}^{(i)}$ belongs to $\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, Y)$. One may verify that $\tilde{\boldsymbol{c}}$ also satisfies $\tilde{\boldsymbol{c}} \in \mathrm{SR}(\boldsymbol{\alpha}, \boldsymbol{c})$, and so we conclude that the right-hand-side of Eq. 12 contains $\mathsf{C}_{\mathsf{SAGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, X)$.

Now we address the reverse inclusion. Let $\boldsymbol{c}$ be such that $\mathrm{SR}(\boldsymbol{\alpha}, \boldsymbol{c}) \cap \mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, Y)$ is nonempty. One may verify that basic properties of $\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, Y)$ and $\mathrm{SR}(\boldsymbol{\alpha}, \boldsymbol{c})$ ensure that if the intersection is nonempty, it contains an element $\tilde{\boldsymbol{c}}$ satisfying $|\boldsymbol{c}| = |\tilde{\boldsymbol{c}}|$. Henceforth fix $\tilde{\boldsymbol{c}}$ satisfying these conditions. Next we appeal to a relaxed form of Corollary 1. Setting $N = \{i : \tilde{c}_i \leq 0\}$, there exist vectors $\tilde{\boldsymbol{c}}^{(i)}$ satisfying

$$\tilde{\boldsymbol{c}} = \textstyle\sum_{i \in N} \tilde{\boldsymbol{c}}^{(i)}, \quad \tilde{\boldsymbol{c}}^{(i)} \in \mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, i, Y), \quad \text{and} \quad \tilde{c}_j^{(i)} = 0 \text{ for all } i \neq j \text{ in } N.$$

Note the definition of $\mathrm{SR}(\boldsymbol{\alpha}, \boldsymbol{c})$ ensures that $N = \{i : \boldsymbol{\alpha}_i \notin 2\mathbb{N}^n, \text{or } c_i \leq 0\}$. Thus we define $\boldsymbol{c}^{(i)}$ by

$$c_j^{(i)} = \begin{cases} (\operatorname{sgn} c_j)|\tilde{c}_j| & \text{if } \boldsymbol{\alpha}_i \text{ is not even, and } j = i \\ \tilde{c}_j^{(i)} & \text{if otherwise} \end{cases}$$

so that $\boldsymbol{c} = \sum_{i \in N} \boldsymbol{c}^{(i)}$, and each $\boldsymbol{c}^{(i)}$ has the necessary sign pattern for membership in the ith AGE cone with respect to $\boldsymbol{\alpha}$, X. Finally, note that

$$\inf\{\mathsf{Pol}(\boldsymbol{\alpha}, \boldsymbol{c}^{(i)})(\boldsymbol{x}) : \boldsymbol{x} \text{ in } X\} = \inf\{\mathsf{Sig}(\boldsymbol{\alpha}, \tilde{\boldsymbol{c}}^{(i)})(\boldsymbol{y}) : \boldsymbol{y} \text{ in } Y\} \geq 0.$$

to complete the proof. □

4.2 Solution recovery for sparse moment problems

Throughout this section we assume that X is of a form where one of Theorems 2 or 3 provide a representation of $\mathsf{C}_{\mathsf{SAGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, X)$; this assumption allows us to leverage the following corollary.

Corollary 2 *Fix $Y = \{\boldsymbol{y} : H(\exp \boldsymbol{y}) \leq \mathbf{1}\}$ for a continuous $H : \mathbb{R}^n \to \mathbb{R}^r$.*

- *If $X = \mathrm{cl}\{\boldsymbol{x} : \mathbf{0} < \boldsymbol{x},\ H(\boldsymbol{x}) \leq \mathbf{1}\}$, then $\mathsf{C}_{\mathsf{SAGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, X)^\dagger = \mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, Y)^\dagger$.*
- *If $X = \mathrm{cl}\{\boldsymbol{x} : \mathbf{0} < |\boldsymbol{x}|,\ H(|\boldsymbol{x}|) \leq \mathbf{1}\}$, then*

$$\mathsf{C}_{\mathsf{SAGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, X)^\dagger = \{\boldsymbol{v} : \text{ there exists } \hat{\boldsymbol{v}} \text{ in } \mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, Y)^\dagger \text{ with } |\boldsymbol{v}| \leq \hat{\boldsymbol{v}}, \text{ and } v_i = \hat{v}_i \text{ when } \boldsymbol{\alpha}_i \in 2\mathbb{N}^{1 \times n}\}.$$

We further assume Y is convex.[5]

Solution recovery for polynomial optimization is more difficult than for signomial optimization, because monomials possess both signs and magnitudes. We propose a two-phase approach for this problem, where different techniques are used to recover variable magnitudes and variable signs. The main ideas for each phase are described in Sects. 4.2.1 and 4.2.2, while the formal algorithms are given in the appendix. The recovered signs and magnitudes are then combined in an elementary way, as given by the following algorithm.

Algorithm 2 solution recovery for dual SAGE polynomial relaxations.

Input: Polynomials f, $\{g_i\}_{i=1}^{k_1}$, and $\{\phi_i\}_{i=1}^{k_2}$ over exponents $\boldsymbol{\alpha} \in \mathbb{N}^{m \times n}$. Vectors $\boldsymbol{v} \in \mathsf{C}_{\mathsf{SAGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, X)^\dagger$ and $\hat{\boldsymbol{v}} \in \mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, Y)^\dagger$. Tolerances $\epsilon_{\text{ineq}}, \epsilon_{\text{eq}}, \epsilon_0 > 0$.

```
procedure PolySolutionRecovery(f, g, φ, α, v, v̂, ε_ineq, ε_eq, ε_0)
    M ← VariableMagnitudes(α, v, v̂, ε_0).   # Algorithm 3
    S ← {1}
    if X is not a subset of ℝ^n_+ then
        S.union( VariableSigns(α, v) )   # Algorithm 4
    solutions ← [].
    for x_mag in M and s in S do
        x ← x_mag * s   # denotes elementwise multiplication
        if g(x) ≥ −ε_ineq and |φ(x)| ≤ ε_eq then
            solutions.append(x)
    solutions.sort(f, increasing).
    return solutions.
```

[5] Corollary 2 holds regardless of whether or not this is the case.

If $\boldsymbol{v}$ is optimal for an appropriate SAGE relaxation and $\boldsymbol{v} = (\boldsymbol{x}^{\boldsymbol{\alpha}_1}, \ldots, \boldsymbol{x}^{\boldsymbol{\alpha}_m})$ for an elementwise nonzero $\boldsymbol{x}$ in X, then Algorithm 2 will return an optimal solution to Problem 1. As with Algorithm 1 in the signomial case, it is useful to apply a simple local solver to the output of Algorithm 2 as a sort of solution refinement. We use the term "Algorithm 2L" in reference to such a method, with COBYLA as the local solver.

4.2.1 Recovering variable magnitudes

Given a vector $\boldsymbol{v}$ in $\mathsf{C}_{\mathsf{SAGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, X)^\dagger$, we want to find a point $\boldsymbol{x} \in X$ satisfying $(\boldsymbol{x}^{\boldsymbol{\alpha}_1}, \ldots, \boldsymbol{x}^{\boldsymbol{\alpha}_m}) = |\boldsymbol{v}|$.

Regardless of whether X is sign-symmetric or $X \subset \mathbb{R}^n_+$, the variable $\boldsymbol{v} \in \mathsf{C}_{\mathsf{SAGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, X)$ is associated with an auxiliary variable $\hat{\boldsymbol{v}}$ in $\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, Y)$, and the variable $\hat{\boldsymbol{v}}$ is associated with additional auxiliary variables $\boldsymbol{z}_i$ as part of the dual Y-AGE signomial cones. As we discussed in Sect. 3.2, the vectors $\boldsymbol{y}_i = \boldsymbol{z}_i / \hat{v}_i$ belong to Y, and so the vectors $\boldsymbol{x}_i = \exp \boldsymbol{y}_i$ must belong to X. These vectors $\boldsymbol{x}_i$ are not only feasible with respect to X, but also satisfy $(\boldsymbol{x}_i^{\boldsymbol{\alpha}_1}, \ldots, \boldsymbol{x}_i^{\boldsymbol{\alpha}_m}) = \hat{\boldsymbol{v}}$ under the binding-constraint and normalization conditions alluded to in Sect. 3.2. Since $\hat{\boldsymbol{v}} = |\boldsymbol{v}|$ always holds at least for $X \subset \mathbb{R}^n_+$, the vectors $\boldsymbol{x}_i = \exp(\boldsymbol{z}_i / \hat{v}_i)$ are reasonable candidates for variable magnitudes.

When X is sign-symmetric, it is possible that $|\boldsymbol{v}|$ does not equal $\hat{\boldsymbol{v}}$. This is particularly likely when $\boldsymbol{v}$ is subject to additional linear constraints, such as $\boldsymbol{G}\boldsymbol{v} \geq \boldsymbol{0}$. Therefore when X is sign-symmetric it is worth considering variable magnitudes which supplement the ones described above. We propose that one picks a threshold $\epsilon_0 > 0$, computes

$$\begin{aligned} \boldsymbol{y} \in \operatorname{argmin} \Big\{ \textstyle\sum_{i:v_i \neq 0} (\boldsymbol{\alpha}_i \cdot \boldsymbol{y} - \log |v_i|)^2 \ : \ \boldsymbol{y} \text{ in } Y, \\ \boldsymbol{\alpha}_i \cdot \boldsymbol{y} \leq \log(\epsilon_0) \text{ for all } v_i = 0 \Big\} \end{aligned} \tag{13}$$

and exponentiates $\boldsymbol{x} = \exp \boldsymbol{y}$. The role of ϵ_0 is to ensure $\boldsymbol{x}$ satisfies $|\boldsymbol{x}|^{\boldsymbol{\alpha}_i} \leq \epsilon_0$ whenever $v_i = 0$. Values of ϵ_0 below machine precision are reasonable here.

A formal statement of our method for magnitude recovery (Algorithm 3) can be found in the appendix.

4.2.2 Recovering variable signs

Let $\boldsymbol{\alpha}^{-1}(\boldsymbol{v})$ denote the set of $\boldsymbol{x} \in \mathbb{R}^n$ satisfying $\boldsymbol{v} = (\boldsymbol{x}^{\boldsymbol{\alpha}_1}, \ldots, \boldsymbol{x}^{\boldsymbol{\alpha}_m})$. Henceforth, fix $\boldsymbol{v}$ and assume $\boldsymbol{\alpha}^{-1}(\boldsymbol{v})$ is nonempty. Here we describe how to find vectors $\boldsymbol{s}$ in $\{+1, 0, -1\}^n$ so that at least one $\boldsymbol{x} \in \boldsymbol{\alpha}^{-1}(\boldsymbol{v})$ satisfies $x_i > 0$ when $s_i = +1$, $x_i = 0$ when $s_i = 0$, and $x_i < 0$ when $s_i = -1$. Once we describe this process, we relax the problem slightly so $s_i = +1$ allows $x_i = 0$.

First we address when s_i should equal zero. Let $U = \{i \in [m] \ : \ v_i \neq 0\}$. Consider how if some $\boldsymbol{x} \in \boldsymbol{\alpha}^{-1}(\boldsymbol{v})$ has $x_j = 0$, then we must have $\alpha_{ij} = 0$ for all i in U (else $\boldsymbol{x}^{\boldsymbol{\alpha}_i} = v_i \neq 0$ would fail). Thus when $\alpha_{ij} = 0$ for all i in U, we set $s_j = 0$ without loss of generality. Now let $W = \{j \in [n] \ : \ \alpha_{ij} > 0 \text{ for some } i \text{ in } U\}$; these are indices for which s_j is not yet decided. Consider the vector $(\boldsymbol{v} < 0) \in \{0, 1\}^n$ with values

$(\boldsymbol{v} < 0)_i = 1$ if $v_i < 0$, and zero if otherwise. Let $\boldsymbol{\alpha}[U, :]$ be the submatrix of $\boldsymbol{\alpha}$ formed by rows $\{\boldsymbol{\alpha}_i\}_{i \in U}$, and similarly index $(\boldsymbol{v} < 0)$. Finally, solve

$$\boldsymbol{\alpha}[U, :]\boldsymbol{z} \equiv (\boldsymbol{v} < 0)[U] \mod 2 \quad \text{and} \quad z_j = 0 \text{ for all } j \text{ in } [n] \setminus W \tag{14}$$

for $\boldsymbol{z}$ in $\{0, 1\}^n$. The remaining $(s_j)_{j \in W}$ are $s_j = -1$ if $z_j = 1$ and $s_j = 1$ otherwise.

An individual solution to (14) can be computed efficiently by Gaussian elimination over the finite field $\mathbb{F}_2$. Our formal algorithm for solution recovery provides the option to recover all solutions to (14), using additional techniques from finite-field linear algebra (c.f. [37]). See the appendix for details.

4.3 A first worked example

This section's example is to minimize a function appearing in the formulation of the cyclic n-roots problem. The general cyclic n-roots problem is a challenging benchmark problem in computer algebra [38]. Our problem is to minimize

$$f(\boldsymbol{x}) = -64 \sum_{i=1}^{7} \prod_{j \in [7] \setminus \{i\}} x_j \tag{Ex3}$$

over the box $X = [-1/2, 1/2]^7$. To the authors' knowledge, this problem was first used as an optimization benchmark in the work by Ray and Nataraj, on computing the extrema of polynomials over boxes [39]. One may verify that $f_X^\star = -7$, and that this objective value is attained at $\boldsymbol{x}^{(1)} = \mathbf{1}/2$ and $\boldsymbol{x}^{(2)} = -\mathbf{1}/2$. Despite this problem's simplicity, it requires nontrivial computational effort with SOS methods. The lowest relaxation order that allows Gloptipoly3 [28] to compute $f_X^\star = -7$ results in a semidefinite program that takes MOSEK 90 s to solve with Machine **W**.

SAGE relaxations automatically exploit the structure in this problem. Since the seven functions $f_i(\boldsymbol{x}) = 1 - 64 \prod_{j \neq i} x_j$ are X-AGE and sum to $f + 7$, we have that $-7 \leq f_X^{\mathsf{SAGE}} \leq f_X^\star$. To address the dual SAGE relaxation and solution recovery, we introduce the 8×7 matrix $\boldsymbol{\alpha}$, with final row $\boldsymbol{\alpha}_8 = \mathbf{0}$, $\alpha_{ii} = 0$ for $i \leq 7$, and $\alpha_{ij} = 1$ for the remaining entries. Next we write $X = \{\boldsymbol{x} : \boldsymbol{x}^2 \leq \mathbf{1}/4\}$, and for $Y = \{\boldsymbol{y} : \exp(2\boldsymbol{y}) \leq \mathbf{1}/4\}$ numerically solve

$$\begin{aligned} f_X^{\mathsf{SAGE}} = \inf\{-64 \cdot \mathbf{1}^\mathsf{T} \boldsymbol{v}_{1:7} \,:\, -\hat{\boldsymbol{v}} \leq \boldsymbol{v} \leq \hat{\boldsymbol{v}}, \\ \hat{\boldsymbol{v}} \text{ in } \mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, Y)^\dagger,\ v_8 = \hat{v}_8 = 1\} = -7. \end{aligned}$$

MOSEK solves this problem in 0.01 s with Machine **W**.

We recover candidate magnitudes by using the eight Y-AGE cones associated with the auxiliary variable $\hat{\boldsymbol{v}} \in \mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, Y)^\dagger$. To machine precision, each of these AGE cones yields the same candidate magnitude $|\boldsymbol{x}| = \mathbf{1}/2$. The optimal moment vector $\boldsymbol{v} = \mathbf{1}/64$ is elementwise positive, and so sign-pattern recovery is a matter of finding all solutions to the system $\boldsymbol{\alpha}\boldsymbol{z} \equiv \mathbf{0} \mod 2$. There are exactly two solutions to this system: $\boldsymbol{z}^{(1)} = \mathbf{0}$, and $\boldsymbol{z}^{(2)} = \mathbf{1}$. The first of these gives rise to signs $\boldsymbol{s}^{(1)} = \mathbf{1}$,

and the second of these results in $\boldsymbol{s}^{(2)} = -\mathbf{1}$. By combining these candidate signs with candidate magnitudes, we obtain candidate solutions $\{\mathbf{1}/2, -\mathbf{1}/2\}$; since these solutions are feasible and obtain objective values matching the SAGE bound, we conclude that both candidate solutions are minimizers of f over X.

4.4 Reference hierarchies for POPs

If $X \subset \mathbb{R}^n_+$, then one should use the same hierarchies described in Sect. 3.4, where "Sig" is replaced by "Pol" and constraints that a function is "an X-SAGE signomial" are replaced by constraints that the function is "an X-SAGE polynomial." This section focuses on the more complicated case when X is sign-symmetric.

Our reference hierarchy for functionally constrained polynomial optimization is similar to that used for signomial programming. Let f, $\{g_i\}_{i=1}^{k_1}$, and $\{\phi_i\}_{i=1}^{k_2}$ be polynomials over common exponents $\boldsymbol{\alpha} \in \mathbb{N}^{m\times n}$, and fix sign-symmetric $X \subset \mathbb{R}^n$. Define $\hat{\boldsymbol{\alpha}}$ as the matrix formed by stacking $\boldsymbol{\alpha}$ on top of $2\boldsymbol{\alpha}$, and then removing any duplicate rows. The SAGE relaxation for $(f, g, \phi)^\star_X$ at level (p, q, ℓ) is then

$$\begin{aligned}(f, g, \phi)_X^{(p,q,\ell)} = \sup\ \ \gamma\ \ \text{s.t.}\ \ & s_h, z_h \text{ are polynomials over exponents } \hat{\boldsymbol{\alpha}}[p] \\ & \mathcal{L} \doteq f - \gamma - \textstyle\sum_{h\in g[q]} s_h \cdot h - \sum_{h\in\phi[q]} z_h \cdot h \\ & \mathsf{Pol}(2\boldsymbol{\alpha}, \mathbf{1})^\ell \mathcal{L} \text{ is an } X\text{-SAGE polynomial} \\ & s_h \text{ are } X\text{-SAGE polynomials.} \end{aligned} \tag{15}$$

As before, the decision variables are $\gamma \in \mathbb{R}$, and the coefficient vectors of $\{s_h\}_{h\in g[q]}$, $\{z_h\}_{h\in\phi[q]}$. The main difference between (15) and it's signomial equivalent (11), is that the Lagrange multipliers are slightly more complex in (15). This change was made to improve performance for problems where only a few rows of $\boldsymbol{\alpha}$ belong to the nonnegative even integer lattice.

Our minimax-free reference hierarchy for polynomial optimization is meaningfully different from the signomial case. We begin by assuming a representation $X = \mathrm{cl}\{\boldsymbol{x} : \mathbf{0} < |\boldsymbol{x}|,\ H(|\boldsymbol{x}|) \leq \mathbf{1}\}$, and subsequently defining $Y = \{\boldsymbol{y} : H(\exp \boldsymbol{y}) \leq \mathbf{1}\}$. Let $\mathcal{A}$ and $\mathcal{C}$ be operators on polynomials so that $f = \mathsf{Pol}(\mathcal{A}(f), \mathcal{C}(f))$ always holds, and let $\boldsymbol{s}$ be the vector in $\mathbb{R}^m$ with $s_i = 1$ when $\boldsymbol{\alpha}_i$ is even, and $s_i = 0$ otherwise. The SAGE relaxation for $f^\star_X$ at level (p, q) is

$$\begin{aligned} f_X^{(p,q)} = \sup\ \ \gamma\ \ \text{s.t.}\ \ & \psi \doteq \mathsf{Pol}(\boldsymbol{\alpha}, \boldsymbol{s})^p (f - \gamma) \\ & \boldsymbol{c} \in \mathrm{SR}(\mathcal{A}(\psi),\ \mathcal{C}(\psi)) \\ & [\mathsf{Sig}(\mathcal{A}(\psi),\ \mathbf{1})]^q\, \mathsf{Sig}(\mathcal{A}(\psi),\ \boldsymbol{c}) \text{ is } Y\text{-SAGE} \end{aligned} \tag{16}$$

over optimization variables $\boldsymbol{c}$ and γ.

Formulation 16 uses two parameters out of desire to mitigate *both* sources of error in the SAGE polynomial cone: that attributable to the use of signomial representatives, and that attributable to the gap between Y-SAGE and Y-nonnegativity. As we show

in Sect. 5.2, the signomial representative complexity parameter "q" can make the difference in our ability to solve problems when $X = \mathbb{R}^n$.

4.5 A second worked example

Our next problem appears in work on "Bounded Degree Sums-of-Squares" (BSOS) and "Sparse Bounded Degree Sums-of-Squares" (Sparse-BSOS) methods for polynomial optimization [40,41]. The latter paper reports BSOS and Sparse-BSOS compute $(f, g)^\star_{\mathbb{R}^6} = -0.41288$ in 44.5 and 82.1 s respectively, when using SDPT3-4.0 on a machine with a 4-core 2.6 GHz Core i7 processor and 16 GB RAM.

$$\begin{aligned}
\inf_{\boldsymbol{x}\in\mathbb{R}^6} \quad & f(\boldsymbol{x}) \doteq x_1^6 - x_2^6 + x_3^6 - x_4^6 + x_5^6 - x_6^6 + x_1 - x_2 \quad \text{subject to} \qquad \text{(Ex4)}\\
& g_1(\boldsymbol{x}) \doteq 2x_1^6 + 3x_2^2 + 2x_1x_2 + 2x_3^6 + 3x_4^2 + 2x_3x_4 + 2x_5^6 + 3x_6^2 + 2x_5x_6 \geq 0\\
& g_2(\boldsymbol{x}) \doteq 2x_1^2 + 5x_2^2 + 3x_1x_2 + 2x_3^2 + 5x_4^2 + 3x_3x_4 + 2x_5^2 + 5x_6^2 + 3x_5x_6 \geq 0\\
& g_3(\boldsymbol{x}) \doteq 3x_1^2 + 2x_2^2 - 4x_1x_2 + 3x_3^2 + 2x_4^2 - 4x_3x_4 + 3x_5^2 + 2x_6^2 - 4x_5x_6 \geq 0\\
& g_4(\boldsymbol{x}) \doteq x_1^2 + 6x_2^2 - 4x_1x_2 + x_3^2 + 6x_4^2 - 4x_3x_4 + x_5^2 + 6x_6^2 - 4x_5x_6 \geq 0\\
& g_5(\boldsymbol{x}) \doteq x_1^2 + 4x_2^6 - 3x_1x_2 + x_3^2 + 4x_4^6 - 3x_3x_4 + x_5^2 + 4x_6^6 - 3x_5x_6 \geq 0\\
& g_{6:10}(\boldsymbol{x}) \doteq 1 - g_{1:5}(\boldsymbol{x}) \geq \mathbf{0}\\
& g_{11:16}(\boldsymbol{x}) \doteq \boldsymbol{x} \geq \mathbf{0}
\end{aligned}$$

Example 4 is very sparse: it includes only 22 of the $\binom{12}{6} = 924$ distinct monomials that could appear in a degree 6 polynomial optimization problem in 6 variables.[6] This problem is a good example for conditional SAGE polynomial relaxations, because it allows for several choices in partial dualization.

The simplest choice is to use no partial dualization at all—simply solve relaxations of the form (15) with $X = \mathbb{R}^n$. Indeed, it is possible to solve Example 3 with only these ordinary SAGE certificates, however the necessary level of the hierarchy $(f, g)^{(1,1,0)}_{\mathbb{R}^6} = -0.41288$ requires 101 s of solver time on Machine **W**.

A preferable alternative is to use partial dualization with $X = \mathbb{R}^6_+$. With this choice of X it is natural to drop now trivially-satisfied constraints from g, and work with $\hat{g} = g_{3:10}$. This allows us to compute $(f, \hat{g})^{(1,1,0)}_X = -0.41288$ in 3.04 s of solver time on Machine **W**, and 4.4 s of solver time on Machine **L**. Significantly, the SAGE relaxation solve time on Machine **L** is an order of magnitude smaller than the BSOS solve time reported in [41].

The most aggressive choice for partial dualization is $X = \{\boldsymbol{x} : \boldsymbol{x} \geq \mathbf{0}, g_{6:7}(\boldsymbol{x}) \geq \mathbf{0}\}$. With this choice of X one may use $\hat{g} = (g_{3:5}, g_{8:10})$, or $\hat{g} = g_{3:10}$; in the first case Machine **W** computes $(f, \hat{g})^{(1,1,0)}_X = -0.47121$ in 3.3 s, and in the second case Machine **W** computes $(f, \hat{g})^{(1,1,0)}_X = -0.41288$ in 5.67 s. We emphasize that even

[6] The problem is referred to as "dense" in the Sparse-BSOS article because it does not satisfy the running-intersection property that Sparse-BSOS is built upon.

though $g_{6:7}$ were incorporated into X, the SAGE bound with Lagrange multiplier complexity $p = 1$ improved by including $g_{6:7}$ in the Lagrangian.

5 Computational experiments

This section presents the results of some computational experiments with SAGE relaxations. Experiments with signomial programs consist of twenty-nine problems drawn from the literature, of which seventeen are solved to optimality (see Sect. 5.1). Examples for polynomial optimization include twenty-two problems from the literature (Sect. 5.2), as well as randomly generated problems (Sect. 5.3).

All experiments described here were run with the provided `sageopt` python package [42]. `Sageopt` includes its own basic rewriting system to cast SAGE relaxations into conic forms acceptable by ECOS [43,44] and MOSEK [19]. The rewriting system also provides mechanisms for computing constraint violations, analyzing low-level problem data, and constructing a set X from lists of constraint functions g and ϕ. Problem data (f, g, ϕ) is usually constructed in an algebraic way, with `sageopt`'s provided operator-overloaded signomial and polynomial objects. `Sageopt` can also automatically construct signomial data (f, g, ϕ) by reading a GPKit model [45]. Once problem data is defined, a SAGE relaxation can be constructed and solved in two lines of code; solution recovery similarly requires no more than two lines of code.

All experiments were conducted on Machine **W** using the MOSEK solver with default tolerances. We note that although Sects. 3.4 and 4.4 only stated the SAGE relaxations in primal form, these experiments were conducted by symbolically constructing primal and dual problems, and solving them separately from one another. In order to communicate the quality of these numeric solutions, we generally report "SAGE bounds" to the farthest decimal point where the primal and dual objectives agree.

5.1 Signomial programs from the literature

The examples in this section were drawn from the PhD thesis of James Yan [35], a popular benchmarking paper by Rijckaert and Martens [46], and the more contemporary works [10,11]. This section is organized chronologically with respect to these sources. Many of the problems considered here can be found elsewhere in the literature; see Shen et al. [5,7,9], Wang and Liang [6] and Qu et al. [8].

SAGE recovers best-known solutions for all but six of the twenty nine problems considered here. For every one of these six problematic examples, numerical issues resulted in solver failures for level-(p, q, ℓ) relaxations whenever $p > 0$; the results for these six problems should not be taken as definitive. For the twenty-three problems where SAGE recovered best-known solutions, there are two important trends we can observe. First, our solution recovery algorithms are more likely to succeed with a conditional SAGE relaxation than with an ordinary SAGE relaxation, even when the ordinary SAGE relaxation is tight. Second—the local solver refinement in Algorithm 1L can help tremendously not only in the presence of suboptimal strictly-feasible initial solutions (Example 8), but also in the presence of both large and small con-

straint violations (Examples 9 and 6 respectively). The initial condition from a SAGE relaxation in Algorithm 1L is important; the underlying COBYLA solver can and will return suboptimal solutions if initialized poorly.

5.1.1 Problems from the PhD thesis of James Yan

We attempted to solve nine example problems appearing James Yan's 1976 PhD thesis *Signomial programs with equality constraints : numerical solution and applications* [35]. This section reproduces two of the six problems which we solved to global optimality via SAGE certificates. Yan's "Problem B" (page 88) and "Problem C" (page 89) serve as our Examples 5 and 6 respectively.

$$\begin{aligned} \inf_{\boldsymbol{x}\in\mathbb{R}^4} \quad & f(\boldsymbol{x}) \doteq 2 - \exp(x_1 + x_2 + x_3) \\ \text{s.t.} \quad & g_1(\boldsymbol{x}) \doteq 4 - \exp x_3 - 15\exp(x_2 + x_3) - 15\exp(x_3 + x_4) \geq 0 \\ & g_{2:5}(\boldsymbol{x}) \doteq (1,1,1,2) - \exp \boldsymbol{x} \geq \mathbf{0} \\ & g_{6:9}(\boldsymbol{x}) \doteq \exp \boldsymbol{x} - (1,1,1,1)/10 \geq \mathbf{0} \\ & \phi_1(\boldsymbol{x}) \doteq \exp x_1 + 2\exp x_2 + 2\exp x_3 - \exp x_4 = 0 \end{aligned} \tag{Ex5}$$

It is possible to quickly compute $(f, g, \phi)^{\star}_{\mathbb{R}^4} = 1.\overline{925}$ with both conditional and ordinary SAGE certificates, although conditional SAGE certificates exhibit better performance for solution recovery. Specifically, $(f, g, \phi)^{(1,1,0)}_{\mathbb{R}^4} = 1.92592592$ can be computed in 0.12 s, but no solution can be recovered from Algorithm 1 unless ϵ is set to an unacceptably large value of 0.1. Instead we set $X = \{\boldsymbol{x} : g(\boldsymbol{x}) \geq \mathbf{0}\}$, compute $(f, g, \phi)^{(1,1,0)}_{X} = 1.92592593$ in 0.18 s, and by running Algorithm 1 recover $\boldsymbol{x}^\star$ satisfying $g(\boldsymbol{x}^\star) > 1\text{E-}11$, $|\phi(\boldsymbol{x}^\star)| < 1\text{E-}8$, and $f(\boldsymbol{x}^\star) = 1.92592593$.

$$\begin{aligned} \inf_{\boldsymbol{y}\in\mathbb{R}^3_{++}} \quad & y_1^{0.6} y_2 + y_2 y_3^{-0.5} + 15.98 y_1 + 9.0824 y_2^2 - 60.72625 y_3 \\ \text{s.t.} \quad & y_2^{-2} y_3 - y_1 y_2^{-2} - 0.48 \geq 0 \\ & y_1^{0.5} y_3^2 - y_1^{0.25} y_3 - y_2^2 - 5.75 \geq 0 \\ & (1000, 1000, 1000) \geq \boldsymbol{y} \geq (0.1, 0.1, 0.1) \\ & y_1^2 + 4y_2^2 + 2y_3^2 - 58 = 0 \\ & y_1 y_2^{-1} y_3^{2.5} + y_2 y_3 - y_2^2 - 16.55 = 0 \end{aligned} \tag{Ex6}$$

With $X = \{\boldsymbol{x} \in \mathbb{R}^3 : g(\boldsymbol{x}) \geq \mathbf{0}\}$, we can compute $(f, g_{1:2}, \phi)^{(0,1,0)}_{X} = -320.722913$ in 0.04 s. By running Algorithm 1 with $\epsilon_{\text{ineq}} = 1\text{E-}8$ and $\epsilon_{\text{eq}} = 1\text{E-}6$, we recover $\boldsymbol{x}$ with objective $f(\boldsymbol{x}) = -320.722913$ and that is feasible up to tolerance 8E-7. We then pass this solution to COBYLA with `RHOEND` $= 1\text{E-}10$, and subsequently recover recover $\boldsymbol{x}^\star$ with the same objective, but constraint violation of only 5E-13.

The remaining problems which we solved to optimality were "Problem A" on page 60, "Problem A" on page 88, "Problem D" on page 89, and the problem in equation environment "(6.15)" on page 106. The last of these was introduced in Sect. 3.5 as

"Example 2." The problems which we did not solve to optimality were "Problem B" on page 61, the problem in equation environment "(6.29)" on page 113, and the problem in equation environment "(6.36)" on page 120. In each of these unsolved cases, we encountered solver-failures for level-(p, q, ℓ) relaxations whenever $p > 0$. Therefore the bounds computed for each of these problems were essentially limited to those of Lagrange dual problems, with modest partial dualization.

5.1.2 Problems from the benchmarking paper of Rijckaert and Martens

We consider problems 9 through 18 of the popular signomial-geometric programming benchmark paper by Rijckaert and Martens [46]. Of these ten problems, seven met with at least moderate success, in that SAGE relaxations produced meaningful lower bounds on a problem's optimal value, and also facilitated recovery of best-known solutions to these problems. SAGE certificates allow us to certify global optimality for four of these seven problems. Problem statistics and a summary of SAGE performance is given in Table 2.

We reproduce Rijckaert and Martens' problems 10 and 15 as our Examples 7 and 8 respectively; both problems are written in exponential-form.

$$\begin{aligned} \inf_{\boldsymbol{x}\in\mathbb{R}^3} \quad & f(\boldsymbol{x}) \doteq 0.5\exp(x_1 - x_2) - \exp x_1 - 5\exp(-x_2) \\ \text{s.t.} \quad & g_1(\boldsymbol{x}) \doteq 100 - \exp(x_2 - x_3) - \exp x_1 - 0.05\exp(x_1 + x_3) \geq 0 \\ & g_{2:4}(\boldsymbol{x}) \doteq (100, 100, 100) - \exp \boldsymbol{x} \geq \mathbf{0} \\ & g_{5:7}(\boldsymbol{x}) \doteq \exp \boldsymbol{x} - (1, 1, 1) \geq \mathbf{0} \end{aligned} \tag{Ex7}$$

The bound constraints appearing in Example 7 are not included in [46], however f is unbounded below if we omit them. The solution proposed in [46] has $\exp \boldsymbol{x} = (88.310, 7.454, 1.311)$, and objective value $f(\boldsymbol{x}) = -83.06$. The actual optimal solution has value -83.25, and this can be certified by running Algorithm 1 on a dual solution for $f_X^{(3)} = -83.2510$, where $X = \{\boldsymbol{x} : g(\boldsymbol{x}) \geq \mathbf{0}\}$. Solving the necessary SAGE relaxation takes 0.1 s on Machine **W**.

$$\begin{aligned} \inf_{\boldsymbol{x}\in\mathbb{R}^{10}} \quad & f(\boldsymbol{x}) \doteq 0.05\exp x_1 + 0.05\exp x_2 + 0.05\exp x_3 + \exp x_9 \\ \text{s.t.} \quad & g_1(\boldsymbol{x}) \doteq 1 + 0.5\exp(x_1 + x_4 - x_7) - \exp(x_{10} - x_7) \geq 0 \\ & g_2(\boldsymbol{x}) \doteq 1 + 0.5\exp(x_2 + x_5 - x_8) - \exp(x_7 - x_8) \geq 0 \\ & g_3(\boldsymbol{x}) \doteq 1 + 0.5\exp(x_3 + x_6 - x_9) - \exp(x_8 - x_9) \geq 0 \\ & g_4(\boldsymbol{x}) \doteq 1 - 0.25\exp(-x_{10}) - 0.5\exp(x_9 - x_{10}) \geq 0 \\ & g_5(\boldsymbol{x}) \doteq 1 - 0.79681\exp(x_4 - x_7) \geq 0 \\ & g_6(\boldsymbol{x}) \doteq 1 - 0.79681\exp(x_5 - x_8) \geq 0 \\ & g_7(\boldsymbol{x}) \doteq 1 - 0.79681\exp(x_6 - x_9) \geq 0 \end{aligned} \tag{Ex8}$$

A level (1,1,0) ordinary SAGE relaxation for Example 8 can be solved in 2.8 s on Machine **W**; this returns the bound $(f, g)_{\mathbb{R}^{10}}^{(1,1,0)} = 0.2056534$. When Algorithm 1 is

Table 2 Columns n and k give number of variables and number of inequality constraints for the indicated problem. "Solution quality" is "same" (resp. "improved") if Algorithm 1L returned a feasible solution with objective equal to (resp. less than) that proposed in [46]. Problems 9, 12, and 14 are discussed in Table 3. We encountered solver failures for level-(1, 1, 0) relaxations of problems 13, 17, and 18

Num. in [46]	n	k	Solution quality	Optimal?
9	2	1	Same	Unknown
10	3	1	Improved	Yes
11	4	2	Same	Yes
12	8	4	Same	Unknown
13	8	6	No solution	No
14	10	7	Same	Unknown
15	10	7	Same	Yes
16	10	7	Same	Yes
17	11	8	No solution	No
18	13	9	No solution	No

Table 3 Problems for which we did not certify optimality, but nevertheless recovered best-known solutions by using SAGE relaxations. Note that Algorithm 1 returned strictly-feasible solutions in each of these cases. In the next section we present examples where Algorithm 1 does not return feasible solutions, and so solution refinement (i.e. Algorithm 1L) becomes more important

Num. in [46]	SAGE relaxation		Algorithm 1		Algorithm 1L	
	(p, q, ℓ)	Bound	$f(\boldsymbol{x})$	$\min g(\boldsymbol{x})$	$f(\boldsymbol{x})$	$\min g(\boldsymbol{x})$
9	(3,3,1)	11.7	12.500	0.00438	11.9600	2.00E-10
12	(0,2,1)	− 6.4	− 5.7677	0.00034	− 6.0482	− 5.00E-09
14	(0,4,0)	0.7	2.5798	0.01541	1.14396	− 8.00E-09

run on the dual solution, it returns a point $\boldsymbol{x}$ satisfying $f(\boldsymbol{x}) \approx 0.38$ and $g(\boldsymbol{x}) \geq 0.053$. However by subsequently running Algorithm 1L, we obtain $\boldsymbol{x}^\star$ satisfying $f(\boldsymbol{x}^\star) = 0.20565341$ and $g_i(\boldsymbol{x}^\star) \geq$ 1E-8 for all i in $[k]$. We thus conclude that the level-(1, 1, 0) SAGE relaxation was tight.

5.1.3 Problems from contemporary sources

Here we describe our attempts at solving six problems from the 2014 article by Hou, Shen, and Chen [10], as well as four problems from the 2014 article by Xu [11]. SAGE relaxations are quite successful in this regard: seven of the ten problems are solved to global optimality (verified SAGE bounds), while best-known (but possibly suboptimal) solutions are obtained for the remaining three problems. Summary results can be found in Tables 4 and 5. We explicitly reproduce problem [10]-8 as our Example 9.

$$\begin{aligned} \inf_{\boldsymbol{y}\in\mathbb{R}^{15}_{++}} \quad & \textstyle\sum_{i=1}^{4} y_{i+11}(12.62626 - 1.231059 y_i) \\ \text{s.t.} \quad & y_{12} - y_{11} \leq 0, \quad y_{11} - y_{12} \leq 50, \quad y_{10} - y_4 \leq 0 \end{aligned} \tag{Ex9}$$

Table 4 Columns n, k_1, and k_2 specify the number of variables, inequality constraints, and equality constraints in the indicated problem. The last three columns specify the objective value and constraint violation of a solution obtained by running Algorithm 1L on the output of a dual SAGE relaxation, as well as a note on whether the objective matched a SAGE bound. Problems with "unknown" optimality status are described in Table 5

Source	Num.	n	k_1	k_2	Objective	Infeasibility	Optimal?
[10]	1	4	10	0	0.7650822	0	Yes
–	2	2	5	0	11.964337	0	Yes
–	3	3	7	0	− 147.66666	0	Yes
–	5	5	16	0	10,122.493	4.00E-13	Unknown
–	7	3	6	0	− 10.363636	2.00E-15	Yes
–	8	15	37	6	156.21963	4.00E-14	Yes
[11]	4	2	1	1	1.3934649	2.00E-10	Yes
–	5	6	9	4	− 0.3888114	5.00E-17	Unknown
–	6	2	4	2	1.1771243	4.00E-12	Yes
–	7	6	20	3	10,252.790	8.00E-14	Unknown

$$
\begin{aligned}
&y_9 - y_{10} \le 0, \quad y_8 - y_9 \le 0, \quad 2y_7 - y_1 \le 1\\
&y_3 - y_4 \le 0, \quad y_2 - y_3 \le 0, \quad y_1 - y_2 \le 0\\
&50y_4 + y_{10}y_{15} - 50y_{10} - y_4y_{15} \le 0\\
&50y_{10} + y_4y_5 + y_9y_{14} - 50y_9 - y_3y_{14} - y_8y_{15} \le 0\\
&50y_7 + y_2y_{13} + y_7y_{12} - 50y_8 - y_1y_{12} - y_8y_{13} \le 0\\
&50y_8 + y_1y_{12} + y_8y_{13} - 50y_7 - y_2y_{13} - y_7y_{12} \le 0\\
&50y_8 + 50y_9 + y_3y_{14} + y_8y_{13} - y_2y_{13} - y_9y_{14} \le 500\\
&y_6y_{11} + y_1y_{12} + y_7y_{11} - y_6y_{12} \le 0\\
&100y_{i+5} + 0.0975y_i^2 - 3.475y_i - 9.75y_iy_{i+5} \le 0 \text{ for all } i \text{ in } [5]\\
&\boldsymbol{y} \ge (1.000000, 1, 9, 9, 9, 1, 1.000000, 1, 1, 1, 50, 0.0, 1.0, 50, 50)\\
&\boldsymbol{y} \le (8.037732, 9, 9, 9, 9, 1, 4.518866, 9, 9, 9, 100, 50, 50, 50, 50)
\end{aligned}
$$

Six of the fifteen variables in Example 9 have matching upper and lower bounds—these are the six equality constraints alluded to in Table 4. Our formulation differs from [10]-8, in that a constraint "$x_3x_2 - x_3 \le 0$" in the original problem statement was replaced by "$y_2 - y_3 \le 0$" in our problem statement. This change is necessary because the original problem is actually infeasible.

We approach Example 9 by maximizing our use of partial dualization: the set $X \subset \mathbb{R}^{15}$ includes all bound constraints, all but two of the first nine inequality constraints, as well as the constraint fourth from the end of the problem statement. The equality constraints implied for variables y_3, y_4, y_5, y_6, y_{14}, y_{15} are not included in the Lagrangian. A level-(0,1,0) conditional SAGE relaxation then produces a bound $(f, g)^\star_X \ge 156.2196$ in 0.05 s. By running Algorithm 1L with $\epsilon_{\text{ineq}} = 100$, we subse-

Table 5 Signomial programs for which we did not certify optimality, but nevertheless recovered best-known solutions by using SAGE relaxations. Columns ϵ_{ineq} and ϵ_{eq} indicate the value of infeasibility tolerances when running Algorithm 1, prior to feeding the output of Algorithm 1 to COBYLA as part of Algorithm 1L. The last two columns list the objective function value and constraint violations for the output of Algorithm 1L. [11]-7 reports a solution with smaller objective value, however that solution violates an equality constraint with forward error in excess of 0.11

Source-num.	(p, q, ℓ)	Bound	ϵ_{ineq}	ϵ_{eq}	Objective	Infeasibility
[10]-5	(0,1,0)	9171.00	1.00E-08	0	10,122.493	4.00E-13
[11]-5	(2,2,0)	−0.390	1.00E-08	1	−0.3888114	5.00E-17
[11]-7	(0,1,0)	9397.8	1.00E-08	1	10,252.790	8.00E-14

quently obtain the geometric-form solution

$$\begin{aligned} \boldsymbol{y}^{\star}_{1:8} &= (8.037732, 9, 9, 9, 9, 1, 1, 1.15686275) \\ \boldsymbol{y}^{\star}_{9:15} &= (1.21505203, 1.58987319, 50, 3\text{E}-50, 1, 50, 50). \end{aligned}$$

The solution $\boldsymbol{y}^{\star}$ is feasible up to forward-error 3.6E-14, and attains an objective value of 156.219629. Because this objective matches the SAGE bound, we conclude that $\boldsymbol{y}^{\star}$ is optimal up to relative error 2E-7.

5.2 Polynomial optimization problems from the literature

Here we review results of the reference hierarchies from Sect. 4.4, as applied to twenty-two polynomial optimization problems from the literature. We begin with six unconstrained and eight box-constrained problems (drawn from [47] and [39] respectively). There are two important lessons which we highlight with the box-constrained problems. First, bound constraints should still be included in the Lagrangian, even if they can be completely absorbed into the set "X" in a conditional SAGE relaxation. Second, even if the original problem does not feature many sign-symmetric constraints, it is often easy to infer valid sign-symmetric constraints which can improve performance of a conditional SAGE relaxation. The remaining eight problems discussed in this section have nonconvex objectives, nonconvex inequality constraints, and constraints that the optimization variables are nonnegative [40]. Our experience with such problems is that partial dualization plays a crucial role in solving them efficiently, primarily with the simpler constraints $\boldsymbol{x} \geq 0$.

Table 6 describes the unconstrained and box-constrained problems; three such problems are reproduced here, as our Examples 10 through 12.

$$\inf\{f(\boldsymbol{x}) \doteq 4x_1^2 - 2.1x_1^4 + x_1^6/3 + x_1x_2 - 4x_2^2 + 4x_2^4 \,:\, \boldsymbol{x} \text{ in } \mathbb{R}^2\} \qquad \text{(Ex10)}$$

The polynomial f in Example 10 is known as the six-hump camel function; its minimum is $f^{\star}_{\mathbb{R}^2} \approx -1.0316$. By using polynomial modulators, a level-(3,0) relaxation returns a bound -1.03170 in 0.63 s of solver time on Machine **W**. By instead solving a level-(0,2) relaxation (i.e. modulating the signomial representative of $f - \gamma$) we obtain

$-1.031630 \leq f_{\mathbb{R}^2}^\star$ in 0.19 s. Example 10 shows how the two-parameter hierarchy (16) can be of practical importance.

Our next two examples are box-constrained problems from the work of Ray and Nataraj [39]; their problems "Capresse 4" and "Butcher 6" serve as our Examples 11 and 12. A consistent trend for these problems is that even when a feasible set X can be incorporated entirely into an X-SAGE cone, it is still useful to take products of constraints, and solve a relaxation such as (15) which includes those constraints in the Lagrangian.

$$\begin{aligned} \inf_{\boldsymbol{x}\in\mathbb{R}^4} \quad & f(\boldsymbol{x}) \doteq -x_1x_3^3 + 4x_2x_3^2x_4 + 4x_1x_3x_4^2 + 2x_2x_4^3 + 4x_1x_3 \\ & \quad + 4x_3^2 - 10x_2x_4 - 10x_4^2 + 2 \\ \text{s.t.} \quad & g_{1:4}(\boldsymbol{x}) \doteq \boldsymbol{x} + (1,1,1,1)/2 \geq \mathbf{0} \\ & g_{5:8}(\boldsymbol{x}) \doteq (1,1,1,1)/2 - \boldsymbol{x} \geq \mathbf{0} \end{aligned} \tag{Ex11}$$

Letting $X = \{\boldsymbol{x} \in \mathbb{R}^4 : -0.5 \leq x_i \leq 0.5\}$, one can compute $(f,g)_X^{(1,2,0)} = -3.180096$, where the equality is verified by recovering a solution with Algorithm 2. Example 11 is noteworthy because the recovered solution required no local-solver refinement that occurs in Algorithm 2L.

$$\begin{aligned} \inf_{\boldsymbol{x}\in\mathbb{R}^6} \quad & f(\boldsymbol{x}) \doteq x_6x_2^2 + x_5x_3^2 - x_1x_4^2 + x_4^3 + x_4^2 - 1/3x_1 + 4/3x_4 \\ \text{s.t.} \quad & g_{1:6}(\boldsymbol{x}) \doteq \boldsymbol{x} + (1,\ 0.1,\ 0.1,\ 1,\ 0.1,\ 0.1) \geq \mathbf{0} \\ & g_{7:12}(\boldsymbol{x}) \doteq (0,\ 0.9,\ 0.5, -0.1, -0.05, -0.03) - \boldsymbol{x} \geq \mathbf{0} \end{aligned} \tag{Ex12}$$

We can produce a tight bound for Example 12 with ordinary SAGE certificates: a level-(0,3,0) relaxation returns $-1.4392999 \leq (f,g)^\star$ in 0.67 s. Solution recovery is not so easy. Unless we move to a computationally expensive level-(0,3,1) ordinary SAGE relaxation, Algorithm 2 fails to return a feasible point. Instead, we infer valid inequalities for use in a conditional SAGE relaxation:

$$\begin{aligned} &|x_1| \leq 1, \quad |x_2| \leq 0.9, \quad |x_3| \leq 0.5, \quad \text{and} \\ &0.1 \leq |x_4| \leq 1, \quad 0.05 \leq |x_5| \leq 0.1, \quad 0.03 \leq |x_6| \leq 0.1. \end{aligned}$$

The resulting level (0,3,0) relaxation can be solved in 0.64 s. We recover a feasible solution with Algorithm 2, which matches the SAGE bound after refinement by COBYLA. Example 12 reinforces a message from signomial optimization: even if an ordinary SAGE relaxation can produce a tight bound, a conditional SAGE relaxation is likely to fare better with solution recovery. Example 12 also shows how useful sign-symmetric constraints can be inferred from bound constraints which are not sign-symmetric.

Now we turn to problems featuring nonconvex inequality constraints [40]. One of these problems was introduced in Sect. 4.5 as "Example 4," and all of these problems have a similar structure to that of Example 4. Most importantly, problems featured here include nonnegativity constraints $\boldsymbol{x} \geq \mathbf{0}$. The natural SAGE hierarchy solves all of these problems; see Table 7.

Table 6 Results for SAGE on unconstrained and box-constrained polynomial minimization problems. Column "d" indicates the degree of the polynomial to be minimized. The Rosenbrock example allows for different numbers of variables, though results from [16] show SAGE is tight for any number of variables. The Beale, Colville, and Goldstein-Price polynomials proved very difficult for optimization via SAGE certificates

Source	Name	n	d	Minimum	SAGE solved
[47]	Rosenbrock	Variable	4	0	Yes
–	6-hump camel	2	6	−1.0316	Yes
–	3-hump camel	2	6	0	Yes
–	Beale	2	8	0	No
–	Colville	4	4	0	No
–	Goldstein-Price	2	8	3	No
[39]	L.V. 4	4	4	− 20.8	Yes
–	Cap 4	4	4	− 3.117690	Yes
–	Hun 5	5	7	− 1436.515	No
–	Cyc 5	5	4	− 3	Yes
–	C.D. 6	6	2	− 270397.4	No
–	But 6	6	3	− 1.4393	Yes
–	Heart 8	8	4	− 1.367754	Yes
–	Viras 8	8	2	− 29	Yes

There are a few subtle distinctions between geometric-form signomial programs (SPs), and the nonnegative polynomial optimization problems (POPs) considered here. While a polynomial optimization problem over $\boldsymbol{x} \geq \mathbf{0}$ may include $x_i = 0$ in the feasible set, geometric-form SPs cannot allow this (since there is the possibility of dividing by zero, or encountering indeterminate forms). Thus solution recovery from SAGE relaxations is nominally more challenging for a true nonnegative POP, relative to a geometric-form SP. Despite this challenge, Algorithm 2L successfully recovers optimal solutions for all of these problems. See Table 8 for details.

The other important distinction between geometric-form SPs and nonnegative POPs, is that there exist established Sums-of-Squares based methods for dealing with nonnegative POPs. Thus it is useful to understand the performance of SAGE-based methods in the context of SOS-based methods for polynomial optimization. Although SAGE relaxations took a very long time to solve problems `P4_6` and `P4_8`, the runtimes for problems such as `P6_8` are remarkable. The unspecified machine in [40] took over 1600 and 200 s to solve `P6_8` with BSOS and SOS respectively, while SAGE can solve the same problem in under 4 s on a mid-tier laptop from 2013. It seems to the authors that SAGE provides a compelling option for nonnegative polynomial optimization problems, at least for low levels of the reference hierarchy (such as $(1, 1, 0)$, or $(0, q, 0)$ with small q).

Table 7 Generic polynomial optimization problems, over the nonnegative orthant [40,41]. Names "Pn_d" indicate the number of variables n and degree d of the given problem. Column k gives the number of inequality constraints, excluding constraints $\boldsymbol{x} \geq \mathbf{0}$, as well as those which trivially follow from $\boldsymbol{x} \geq \mathbf{0}$. SAGE solved all problems listed here, at the indicated level of the hierarchy, and with the indicated solver runtimes for the primal-form relaxations

Name	k	Minimum	(p, q, ℓ)	**W** time (s)	**L** time (s)
P4_4	8	−0.033538	(1,1,0)	0.47	0.7
P4_6	7	−0.060693	(1,1,1)	289	292
P4_8	7	−0.085813	(2,1,0)	396	460
P6_4	8	−0.576959	(1,1,0)	3.45	4.1
P6_6	8	−0.412878	(1,1,0)	3.04	4.37
P6_8	8	−0.409020	(1,1,0)	3.25	3.83
P8_4	8	−0.436026	(1,1,0)	7.18	7.25
P8_6	8	−0.412878	(1,1,0)	8.67	8.21

Table 8 Comparison of Algorithms 2 and 2L for solution recovery for eight nonconvex polynomial optimization problems in the literature (Ref. [40,41]). Both algorithms were initialized with solutions to a level-(1,1,0) conditional SAGE relaxation, and Algorithm 2L always recovers an optimal solution. It is especially notable that Algorithm 2L recovers optimal solutions for problems P4_6 and P4_8, since level-(1,1,0) relaxations do not produce tight bounds for these problems

Name	Algorithm 2		Algorithm 2L	
	$f(\boldsymbol{x})$	$\min g(\boldsymbol{x})$	$f(\boldsymbol{x})$	$\min g(\boldsymbol{x})$
P4_4	−0.033386	0.00E-00	−0.033538	0.00E-00
P4_6	−0.057164	4.06E-02	−0.060693	−2.44E-14
P4_8	−0.066671	1.42E-01	−0.085813	−3.46E-14
P6_4	−0.570848	4.04E-08	−0.576959	−1.03E-13
P6_6	−0.412878	5.46E-09	−0.412878	−1.68E-13
P6_8	−0.409018	1.07E-07	−0.409020	−5.82E-14
P8_4	−0.436024	3.27E-08	−0.436026	−2.58E-13
P8_6	−0.412878	2.78E-43	−0.412878	−2.55E-12

5.3 Minimizing random sparse quartics over the sphere

Here we describe how SAGE relaxations fare for minimizing sparse quartic forms over the unit sphere. This class of test problems is inspired from similar computational experiments by Ahmadi and Majumdar in their work on LP and SOCP-based inner-approximations of the SOS cone [48].

Our method for generating these problems is as follows: initialize $f = 0$ as a polynomial in n variables, and proceed to iterate over all tuples "t" in $[n]^4$. With probability $n \log n / n^4$, sample a coefficient c_t from the standard normal distribution, and add the term $c_t \boldsymbol{x}^{\boldsymbol{\alpha}_t}$ to f, where $\boldsymbol{\alpha}_t \in [4]^n$ has $\alpha_{tj} = |\{i : t_i = j\}|$. The expected number of terms in f after this procedure is roughly $n \log n$. Once a polynomial is generated, we solve a level-(0,2,0) conditional SAGE relaxation for $(f, g)^\star_{\mathbb{R}^n}$, where

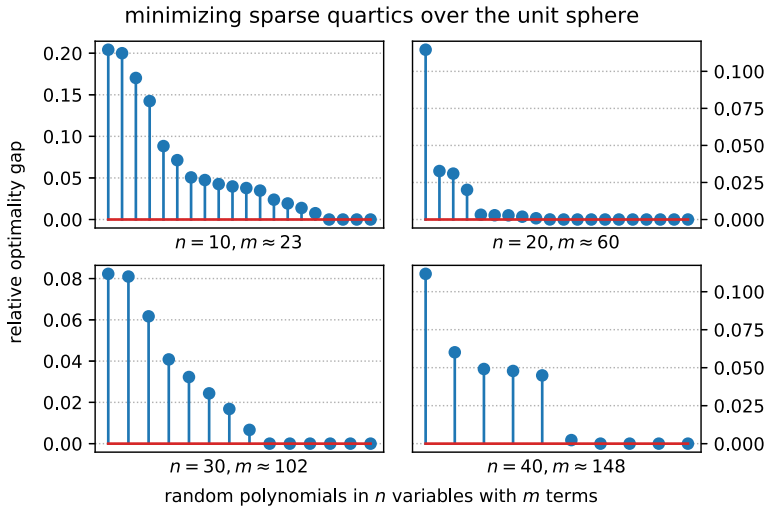

Fig. 1 Upper-bounds on the optimality gap $|(f,g)_X^{(0,2,0)} - (f,g)_{\mathbb{R}^n}^\star|/|(f,g)_{\mathbb{R}^n}^\star|$. The value $(f,g)_{\mathbb{R}^n}^\star$ in these calculations was replaced by the objective value of a solution produced by Algorithm 2L. SAGE solved 4 problems in 10 variables, 10 problems in 20 variables, 6 problems in 30 variables, and 4 problems in 40 variables

Table 9 Solver runtimes for level-(0,2,0) conditional SAGE relaxations, on Machine **W**. Similar runtimes can be expected for Machine **L** with $n \in \{10, 20, 30\}$. Solve times with Machine **L** can take much longer for $n \geq 40$, since only part of the problem fits in RAM

Solve time (s)	$n = 10$	$n = 20$	$n = 30$	$n = 40$
Mean	7.54E-01	6.50E-00	6.46E+01	4.59E+02
SD	8.74E-02	8.54E-01	1.38E+01	7.20E+01

$g(\boldsymbol{x}) = 1 - \boldsymbol{x}^\mathsf{T}\boldsymbol{x}$.[7] The set "$X$" in the conditional SAGE relaxation is $X = \{\boldsymbol{x} : g(\boldsymbol{x}) \geq 0\}$. Figure 1 and Table 9 report results for 20 problems in 10 variables, 20 problems in 20 variables, 14 problems in 30 variables, and 10 problems in 40 variables.

5.4 Broader observations from numerical experiments

Here we provide expanded remarks on three aspects of our numerical experiments. First, we address which SAGE relaxations appear to be numerically difficult, and provide a reason for why this might happen. Then we report some of the sizes of the SAGE relaxations as represented by cone programs suitable for low-level solvers. Finally, we demonstrate that these low-level cone programs actually have an extremely useful substructure which is not exploited by MOSEK or ECOS.

For both signomial and polynomial optimization problems, there is significant benefit to solving level-(p, q, ℓ) relaxations with Lagrange multiplier complexity $p > 0$. However we encountered several problems where any choice of $p > 0$ resulted in a

[7] Because f is homogeneous, $\boldsymbol{x}^\mathsf{T}\boldsymbol{x} = 1$ may be relaxed to $\boldsymbol{x}^\mathsf{T}\boldsymbol{x} \leq 1$ without loss of generality

Table 10 Dimensions of $\boldsymbol{A} \in \mathbb{R}^{N \times d}$ and sparsity $s = \text{nnz}(\boldsymbol{A})/(Nd)$ in `sageopt`'s low-level representation of feasible sets for dual relaxations of [40, Problem P4_6]. "Slacks" is the default behavior for `sageopt` version 0.2, which was used for experiments in this article. "Direct" (no slacks) is the default for `sageopt` version 0.5.2. See also Tables 7 and 8

Level (p, q, ℓ)	(0,1,0)	(1,1,0)	(1,1,1)
(N, d, s)-slacks	(953, 342, 4E-3)	(80,542, 21,896, 8E-5)	(2,275,860, 574,775, 3E-6)
(N, d, s)-direct	(840, 229, 6E-3)	(62,653, 4007, 5E-4)	(1,714,437, 13,352, 1E-4)

solver failure due to numerical issues. One explanation for the difficulty of such SAGE relaxations could be how (11) and (15) set the complexity of a Lagrange multiplier without consideration to its associated constraint function. Compare to the usual SOS-based Lasserre hierarchy, where a degree k constraint polynomial $g_i(\boldsymbol{x}) \geq 0$ appearing in a degree $2d$ relaxation gets an SOS multiplier of degree $2\lfloor (2d-k)/2 \rfloor$. For SAGE relaxations, one could set the support of a Lagrange multiplier with consideration to how the product of the constraint function and Lagrange multiplier affect the sparsity pattern of the final Lagrangian. Suitably chosen supports for nonconstant Lagrange multipliers could also result in bounds which are stronger than those produced by reference hierarchies (11) and (15).

One of the main functions of `sageopt` is to cast abstract SAGE relaxations into low-level standard forms, with feasible sets $\{\boldsymbol{x} \in \mathbb{R}^d : \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b} \in K\}$ for some $\boldsymbol{A} \in \mathbb{R}^{N \times d}$ and $K \subset \mathbb{R}^N$ which is a product of elementary convex cones. There are several settings within `sageopt` which affect how this compilation process is performed. The impact of different settings for the use of slack variables becomes very apparent as one solves SAGE relaxations farther up a hierarchy (Table 10). Regardless of compilation settings, the resulting cone programs end up being large and sparse as reference hierarchy parameters increase. It is possible to construct smaller cone programs by inferring signs on certain coefficients of a modulated Lagrangian, and then appealing to Corollary 1. `Sageopt` already performs a simple version of such dimension-reduction, which is particularly helpful for the minimax-free hierarchy defined in Equation 10.

Solving linearized KKT equations is the largest computational expense in each iteration of an interior-point algorithm for convex cone programming. For ease of exposition, suppose equality constraints in a low-level cone program are represented by two-sided inequality constraints. Under this assumption, both MOSEK and ECOS solve linearized KKT equations by performing a sparse LDL factorization of a symmetric indefinite matrix of order $N + d$. Another approach to solving the linearized KKT equations applies a block-elimination to the indefinite system, to obtain a symmetric positive definite system of dimension $d \ll N$ [49]. In the case of cone programs generated by `sageopt` for dual SAGE relaxations, the only coupling across dual AGE cones occurs through the moment vector $\boldsymbol{v}$, therefore reduction to positive definite KKT systems would be extremely efficient (see Sect. 2). It is of interest to see how SAGE relaxations scale with a solver which could take advantage of this structure.

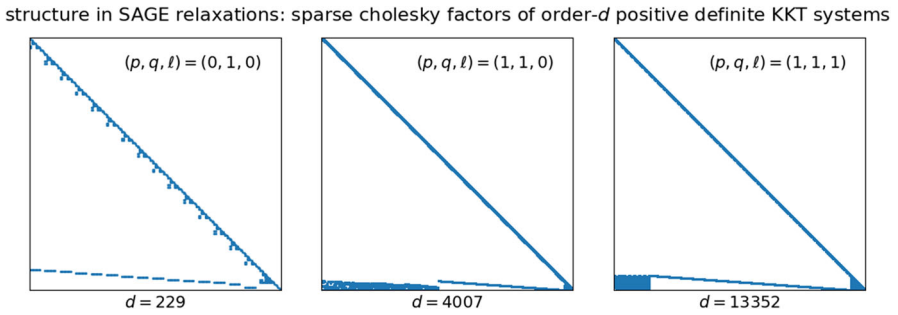

Fig. 2 Sparsity patterns of Cholesky factors, which can be used to solve linearized KKT systems in interior-point methods for SAGE relaxations to [40, Problem P4_6]. All Cholesky factors used a simple reversed elimination order $d, d-1, \ldots, 1$ relative to the original positive definite KKT systems which follow from `sageopt` 0.5.2's low-level problem data

6 Outlook

In this article we introduced and developed notions of conditional SAGE certificates for both signomials and polynomials. Through worked examples and computational experiments, we have demonstrated that subsequent convex relaxations can be used to solve many signomial and polynomial optimization problems from the literature. The authors believe that conditional SAGE certificates are a fertile area for research in both the theory and practice of constrained optimization; we briefly describe some possible directions here.

Branch-and-bound. Bound constraints can be completely incorporated into conditional SAGE cones for signomials (in geometric or exponential form). It would be valuable to determine how conditional SAGE certificates can be used to aid the "bounding" step in existing branch-and-bound algorithms for signomial programming. Branch-and-bound also blends with SAGE polynomials. Of course, sign-symmetric bounds can be accommodated directly. If X encodes bound constraints where each variable has a fixed sign, we can still obtain representations for $\mathsf{C}_{\mathsf{SAGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, X)$ in a way which is similar to when $X \subset \mathbb{R}^n_+$.

Solution recovery. Solution recovery from conditional SAGE relaxations to signomial programs is a problem of great importance. This article identified a projective structure in dual X-SAGE cones which leads to a practical method of solution recovery. However, we did not explore the theoretical properties of this method, and in fact we introduced additional heuristics (such as local-refinement) to obtain good performance. It is highly desirable to develop a coherent theory around such projective methods of solution recovery.

The minimax-free hierarchy. The minimax-free hierarchy from Sect. 3.4 adopted a particular form for the modulating function: $\mathsf{Sig}(\boldsymbol{\alpha}, \mathbf{1})^\ell$. What benefit might there be to instead using a modulator $\mathsf{Sig}(\hat{\boldsymbol{\alpha}}, \mathbf{1})^\ell$, where $\hat{\boldsymbol{\alpha}}$ was chosen with consideration to X? Equally important, how could one efficiently identify good candidates for such $\hat{\boldsymbol{\alpha}}$, given only $\boldsymbol{\alpha}$ and a description of X? Finally, under what conditions on $\boldsymbol{\alpha}$, X are the bounds $f_X^{(\ell)}$ certain to converge to $f_X^\star$?

SAGE versus nonnegativity. For what exponents $\boldsymbol{\alpha}$ and what sets X do X-SAGE cones coincide with X-nonnegativity cones? Prior work (c.f. [16], and more recently [50]) has uncovered meaningful sufficient conditions for this problem when $X = \mathbb{R}^n$. These sufficient conditions have hitherto been stated in terms of the combinatorial geometry of the exponent vectors $\{\boldsymbol{\alpha}_i\}_{i=1}^m$. It will be very interesting to see how such results do (or do not) generalize to X-SAGE cones for arbitrary X.

Acknowledgements The authors thank Fangzhou Xiao and two anonymous referees for helpful feedback. R.M. was supported in part by an NSF Graduate Research Fellowship, NSF grants CCF-1350590 and CCF-1637598, and AFOSR grant FA9550-16-1-0210. V.C. was supported in part by NSF grants CCF-1350590 and CCF-1637598, AFOSR grant FA9550-16-1-0210, and a Sloan Research Fellowship. A.W. was supported in part by NSF grant CCF-1637598.

Appendix

Algorithm 3 magnitude recovery for dual SAGE polynomial relaxations.

Input: A matrix $\boldsymbol{\alpha} \in \mathbb{N}^{m\times n}$. Vectors $\boldsymbol{v} \in \mathsf{C}_{\mathsf{SAGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, X)^\dagger$ and $\hat{\boldsymbol{v}} \in \mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, Y)$. Zero threshold parameter $\epsilon_0 > 0$.

1: **procedure** VARIABLEMAGNITUDES($\boldsymbol{\alpha}$, $\boldsymbol{v}$, $\hat{\boldsymbol{v}}$, ϵ_0)
2: $M \leftarrow []$
3: **for** $j = 1, \ldots, m$ **do**
4: **if** $\hat{v}_j = 0$ **then**
5: **Continue**
6: Recover $\boldsymbol{z}_j$ in $\mathbb{R}^n$ s.t. $\hat{v}_j \log(\hat{\boldsymbol{v}}/\hat{v}_j) \geq [\boldsymbol{\alpha} - \mathbf{1}\boldsymbol{\alpha}_j]\boldsymbol{z}_j$ and $(\boldsymbol{z}_j, \hat{v}_j) \in \operatorname{co} Y$.
7: M.append($\exp(\boldsymbol{z}_j/\hat{v}_j)$)
8: **if** $(\boldsymbol{x}^{\boldsymbol{\alpha}_1}, \ldots, \boldsymbol{x}^{\boldsymbol{\alpha}_m}) \neq |\boldsymbol{v}|$ for all $\boldsymbol{x}$ in M **then**
9: Compute $(\boldsymbol{y}, t)$ solving Problem 13, for given ϵ_0.
10: M.append($\exp \boldsymbol{y}$)
11: **return** M.

As in the signomial case, Algorithm 3 always returns a vector $\boldsymbol{x} \in X$. Assuming that $\boldsymbol{z}$ from Line 7 are already computed as part of representing $\hat{\boldsymbol{v}}$, the complexity of this algorithm is dominated by Line 12. The runtime of Line 12 is in turn negligible relative to solving a SAGE relaxation to obtain vectors $\boldsymbol{v}$ and $\hat{\boldsymbol{v}}$. Infeasibility errors encountered in Line 12 should be handled by jumping to Line 15.

Algorithm 4 sign recovery for dual SAGE polynomial relaxations.

Input: A matrix $\boldsymbol{\alpha} \in \mathbb{N}^{m\times n}$. A vector $\boldsymbol{v}$ in $\mathbb{R}^m$. A Boolean `heuristic`.

1: **procedure** VARIABLESIGNS($\boldsymbol{\alpha}$, $\boldsymbol{v}$, `heuristic`)
2: $U \leftarrow \{i : v_i \neq 0$ and $\boldsymbol{\alpha}_i$ is not even $\}$
3: $W \leftarrow \{j : \alpha_{ij} \equiv 1 \mod 2$ for some i in $U\}$
4: $Z \leftarrow \{\boldsymbol{z} \in \{0,1\}^n : \boldsymbol{\alpha}[U, :]\boldsymbol{z} \equiv (\boldsymbol{v} < 0)[U] \mod 2,\ z_i = 0$ for i in $[n] \setminus W\}$
5: $S \leftarrow \{\}$
6: **for** $\boldsymbol{z}$ in Z **do**
7: $\boldsymbol{s} \leftarrow \mathbf{1}$
8: **for** j in $\{j : \alpha_{ij} > 0$ for some i in $U\}$ **do**
9: $s_j \leftarrow -1$ if $z_j = 1$, 1 if $z_j = 0$
10: $S \leftarrow S \cup \{\boldsymbol{s}\}$
11: If $S = \emptyset$ and `heuristic`, update $S \leftarrow \{$HueristicSigns($\boldsymbol{\alpha}$, $\boldsymbol{v}$)$\}$.
12: **return** S.

Let us describe the ways in which Algorithm 4 differs from the discussion in Sect. 4.2.2. First- there are changes to the sets U and W. The set U now drops any rows $\boldsymbol{\alpha}_i$ from $\boldsymbol{\alpha}$ where $\boldsymbol{\alpha}_i$ is even; it is easy to verify that this does not affect the set of solutions to the appropriate linear system. The set W changes by only considering j where at least one $\alpha_{ij} \equiv 1 \mod 2$. This change is valid because if α_{ij} is even for all i, then the sign of variable x_j is irrelevant to the underlying optimization problem, and we make take $x_j \geq 0$ without loss of generality.

Next we speak to the "hueristic" sign recovery. We partly mean to leave this as open-ended, however for completeness we describe the algorithm used in `sageopt`. The goal is to find a vector $\boldsymbol{s}$ in $\{+1, -1\}$ so that the signs of $\boldsymbol{s}^{\boldsymbol{\alpha}} \doteq (\boldsymbol{s}^{\boldsymbol{\alpha}_1}, \ldots, \boldsymbol{s}^{\boldsymbol{\alpha}_m})$ match the signs of $\boldsymbol{v}$ to the greatest extent possible. However, we consider how having $\boldsymbol{s}^{\boldsymbol{\alpha}_i}$ match the sign of v_i may not be very important if v_i is very small. Therefore we use a merit function $M(\boldsymbol{s}) = \boldsymbol{v}^\intercal \boldsymbol{s}^{\boldsymbol{\alpha}}$ to evaluate the quality of candidate signs $\boldsymbol{s}$. We apply a greedy algorithm to maximize the merit function $M(\boldsymbol{s})$ as follows: initialize $\boldsymbol{s} = \mathbf{1}$, and a set of undecided coordinates $C = \{1, \ldots, n\}$. As long as the set C is nonempty, find an index $i^\star \in C$ so that changing $s_{i^\star} = 1$ to $s_{i^\star} = -1$ maximizes improvement in the merit function. If the improvement is positive, then perform the update $s_{i^\star} \leftarrow -1$. Regardless of whether or not the improvement is positive, remove $i^\star$ from C. Once C is empty, return $\boldsymbol{s}$.

References

1. Rountree, D.H., Rigler, A.K.: A penalty treatment of equality constraints in generalized geometric programming. J. Optim. Theory Appl. **38**(2), 169–178 (1982). issn: 1573-2878
2. Kirschen, P.G., et al.: Application of signomial programming to aircraft design. J. Aircr. **55**(3), 965–987 (2018)
3. Jabr, R.A.: Inductor design using signomial programming. COM-PEL Int. J. Comput. Math. Electr. Electron. Eng. **26**(2), 461–475 (2007)
4. Chiang, M.: Nonconvex optimization for communication networks. In: Honor of Gilbert Strang, Advances in Applied Mathematics and Global Optimization, Springer US, Boston, pp. 137–196. ISBN: 978-0-387-75714-8 (2009)
5. Shen, P., Zhang, K.: Global optimization of signomial geometric programming using linear relaxation. Appl. Math. Comput. **150**(1), 99–114 (2004)
6. Wang, Y., Liang, Z.: A deterministic global optimization algorithm for generalized geometric programming. Appl. Math. Comput. **168**(1), 722–737 (2005). issn: 0096-3003
7. Shen, P., Jiao, H.: A new rectangle branch-and-pruning approach for generalized geometric programming. Appl. Math. Comput. **183**(2), 1027–1038 (2006)
8. Shao-Jian, Q., Zhang, K.-C., Ji, Y.: A new global optimization algorithm for signomial geometric programming via Lagrangian relaxation. Appl. Math. Comput. **184**(2), 886–894 (2007)
9. Shen, P., Ma, Y., Chen, Y.: A robust algorithm for generalized geometric programming. J. Global Optim. **41**(4), 593–612 (2008). issn: 1573-2916
10. Hou, X., Shen, P., Chen, Y.: A global optimization algorithm for signomial geometric programming problem. Abstract Appl. Anal. **2014**, 1–12 (2014)
11. Gongxian, X.: Global optimization of signomial geometric programming problems. Eur. J. Oper. Res. **233**(3), 500–510 (2014)
12. Shor, N.Z.: Class of global minimum bounds of polynomial functions. Cybernetics **23**(6), 731–734 (1987)
13. Parrilo, P.: Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization. PhD thesis, California Institute of Technology (2000)
14. Lasserre, J.B.: Global optimization with polynomials and the problem of moments. SIAM J. Optim. **11**(3), 796–817 (2001)

15. Chandrasekaran, V., Shah, P.: Relative entropy relaxations for signomial optimization. SIAM J. Optim. **26**(2), 1147–1173 (2016)
16. Murray, R., Chandrasekaran, V., Wierman, A.: Newton polytopes and relative entropy optimization (2018). arXiv:1810.01614
17. Iliman, S., de Wolff, T.: Amoebas, nonnegative polynomials and sums of squares supported on circuits. Res. Math. Sci. **3**, 9 (2016)
18. Murty, K.G., Kabadi, S.N.: Some NP-complete problems in quadratic and nonlinear programming. Math. Program. **39**(2), 117–129 (1987). issn: 1436-4646
19. MOSEK ApS. MOSEK 9.0.70(beta) (2019)
20. Reznick, B.: Forms derived from the arithmetic-geometric inequality. Math. Ann. **283**(3), 431–464 (1989)
21. Pébay, P.P., Rojas, J.M., Thompson, D.C.: Optimization and NP R-completeness of certain fewnomials. In: Proceedings of the 2009 Conference on Symbolic Numeric Computation, ACM Press (2009)
22. Pantea, C., Koeppl, H., Craciun, G.: Global injectivity and multiple equilibria in uni- and bi-molecular reaction networks. Discrete Contin. Dyn. Syst. Ser. B **17**(6), 2153–2170 (2012)
23. August, E., Craciun, G., Koeppl, H.: Finding invariant sets for biological systems using monomial domination. In: 2012 IEEE 51st IEEE Conference on Decision and Control (CDC), IEEE (2012)
24. Wang, J.: Nonnegative polynomials and circuit polynomials (2018). arXiv:1804.09455
25. Katthän, L., Naumann, H., Theobald, T.: A unified framework of SAGE and SONC polynomials and its duality theory (2019). arXiv:1903.08966
26. Seidler, H., de Wolff, T.: An experimental comparison of SONC and SOS certificates for unconstrained optimization (2018). arXiv:1808.08431
27. Seidler, H., de Wolff, T.: POEM: effective methods in polynomial optimization, version 0.2.1.0(a) (2019). http://www.iaa.tu-bs.de/AppliedAlgebra/POEM/index.html
28. Henrion, D., Lasserre, J.-B., Löfberg, J.: GloptiPoly 3: moments, optimization and semidefinite programming. Optim. Methods Softw. **24**(4–5), 761–779 (2009)
29. Papachristodoulou, A., et al.: SOSTOOLS version 3.00 sum of squares optimization toolbox for MATLAB (2013). arXiv:1310.4716
30. Powers, V., Reznick, B.: Polynomials that are positive on an interval. Trans. Am. Math. Soc. **352**(10), 4677–4692 (2000)
31. Laurent, M.: Sums of squares, moment matrices and optimization over polynomials. In: Putinar, M., Sullivant, S. (eds.) Emerging Applications of Algebraic Geometry, pp. 157–270. Springer, New York (2009). isbn: 978-0-387-09686-5
32. Borwein, J., Lewis, A.: Convex Analysis and Nonlinear Optimization. Springer, New York (2006)
33. Lasserre, J.B.: An Introduction to Polynomial and Semi-algebraic OptimizationCambridge Texts in Applied Mathematics. Cambridge University Press, Cambridge (2015)
34. Powell, M.J.D.: A direct search optimization method that models the objective and constraint functions by linear interpolation. In: Advances in Optimization and Numerical Analysis, Springer, Dordrecht, pp. 51–67. ISBN: 978-94-015-8330-5 (1994)
35. Yan, J.: Signomial programs with equality constraints: numerical solution and applications. PhD thesis, University of British Columbia (1976)
36. Agrawal, A., Diamond, S., Boyd, S.: Disciplined geometric programming. Optim. Lett. **13**(5), 961–976 (2019)
37. Bard, G.V.: Some basic facts about linear algebra over GF(2). In: Algebraic Cryptanalysis, Springer, Berlin, pp. 81–88 (2009)
38. Verschelde, J.: Algorithm 795: PHCpack—a general-purpose solver for polynomial systems by homotopy continuation. ACM Trans. Math. Softw. **25**(2), 251–276 (1999). issn: 0098-3500
39. Ray, S., Nataraj, P.S.V.: An efficient algorithm for range computation of polynomials using the Bernstein form. J. Global Optim. **45**(3), 403–426 (2008)
40. Lasserre, J.B., Toh, K.-C., Yang, S.: A bounded degree SOS hierarchy for polynomial optimization. EURO J. Comput. Optim. **5**(1), 87–117 (2017). issn: 2192- 4414
41. Weisser, T., Lasserre, J.B., Toh, K.-C.: Sparse-BSOS: a bounded degree SOS hierarchy for large scale polynomial optimization with sparsity. Math. Program. Comput. **10**(1), 1–32 (2018). issn: 1867-2957
42. Murray, R.: Sageopt 0.5.3 (2020). https://doi.org/10.5281/ZENODO.4017991
43. Domahidi, A., Chu, E., Boyd, S.: ECOS: an SOCP solver for embedded systems. In: European Control Conference (ECC), pp. 3071–3076 (2013)

44. Serrano, S.A.: Algorithms for unsymmetric cone optimization and an implementation for problems with the exponential cone. PhD Thesis, Stanford University, Palo Alto, CA (2015)
45. Burnell, E., Damen, N.B., Hoburg, W.: GPkit: a human-centered approach to convex optimization in engineering design. In: Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (2020)
46. Rijckaert, M.J., Martens, X.M.: Comparison of generalized geometric programming algorithms. J. Optim. Theory Appl. **26**(2), 205–242 (1978). issn: 1573-2878
47. Surjanovic, S., Bingham, D.: Virtual library of simulation experiments: test functions and datasets. Retrieved April 18 from http://www.sfu.ca/~ssurjano (2019)
48. Ahmadi, A.A., Majumdar, A.: DSOS and SDSOS optimization: more tractable alternatives to sum of squares and semidefinite optimization. SIAM J. Appl. Algebra Geom. **3**(2), 193–230 (2019)
49. Vandenberghe, L.: The CVXOPT linear and quadratic cone program solvers (2010). http://www.seas.ucla.edu/~vandenbe/publications/coneprog.pdf
50. Forsgård, J., de Wolff, T.: The algebraic boundary of the sonc cone (2019). arXiv:1905.04776