# Signomial and Polynomial Optimization via Relative Entropy and Partial Dualization

Riley Murray[†], Venkat Chandrasekaran[†,‡], and Adam Wierman[†] *

[†] Department of Computing and Mathematical Sciences
[‡] Department of Electrical Engineering
California Institute of Technology
Pasadena, CA 91125

July 23, 2019

**Abstract**

We describe a generalization of the Sums-of-AM/GM Exponential (SAGE) relaxation methodology for obtaining bounds on constrained signomial and polynomial optimization problems. Our approach leverages the fact that relative entropy based SAGE certificates conveniently and transparently blend with convex duality, in a manner that Sums-of-Squares certificates do not. This more general approach not only retains key properties of ordinary SAGE relaxations (e.g. sparsity preservation), but also inspires a novel perspective-based method of solution recovery. 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.**

*Email: rmurray@caltech.edu, venkatc@caltech.edu, adamw@caltech.edu

Acknowledgements: R.M. was supported in part by an NSF Graduate Research Fellowship and by NSF grant CCF-1350590, NSF grant 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.

# 1 Introduction

A signomial is a function of the form $\boldsymbol{x} \mapsto \sum_{i=1}^{m} c_i \exp(\boldsymbol{\alpha}_i \cdot \boldsymbol{x})$ for real scalars $c_i$ and row vectors $\boldsymbol{\alpha}_i$ in $\mathbb{R}^{1\times n}$. Signomial optimization (often called signomial programming) concerns the minimization of a signomial, subject to signomial inequality and equality constraints. Signomial programming is a computationally challenging problem with applications in chemical engineering [3], aeronautics [29], circuit design [15], and communications network optimization [20]. Signomials are sometimes thought of as generalizations of polynomials over the positive orthant; 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 between polynomials and geometric-form signomials, we must bear in mind that signomials and polynomials have many significant differences. Where polynomials can be 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 the positive orthant.

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 [11, 12, 14, 16, 18, 23, 24] – ideas which precede the advent of modern convex optimization. By contrast, the field of polynomial optimization has been substantially influenced by semidefinite programming, specifically through Sums-of-Squares (SOS) certificates of polynomial nonnegativity [5, 8, 10]. In recent work, Chandrasekaran and Shah proposed the Sums-of-AM/GM Exponential or SAGE certificates of signomial nonnegativity, which provided a new convex relaxation framework for signomial programs akin to SOS methods for polynomial optimization. [26]. Where SOS certificates make use of semidefinite programming, SAGE certificates use the convex *relative entropy function*. The authors of the present article further demonstrated that a natural modification to SAGE certificates leads to a tractable relative entropy representable sufficient condition for global polynomial nonnegativity [30].

This article is concerned with how proof systems for function nonnegativity can be used in the service of constrained optimization. The basic idea here is simple: for a function $f$, a set $X$, and a real number $\gamma$, we have $\inf\{f(\boldsymbol{x}) : \boldsymbol{x} \in X\} \geq \gamma$ 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 $X \subsetneq \mathbb{R}^n$. For the polynomial case one usually performs this extension by appealing to representation theorems from real algebraic geometry. In the absence of such representation theorems, one typically relies on a dual problem obtained from the minimax inequality.

The primary contribution of this article is to show how SAGE certificates – by virtue of their roots in convex duality – provide a simple and powerful alternative method for describing functions which are nonnegative over proper subsets of $\mathbb{R}^n$.

Our method can be used both independently from and in conjunction with the minimax inequality. The space of possibilities with our method is large, and it is far from obvious as to which variations of this methodology are most useful for given problem structures. To facilitate research in this regard, we provide a user-friendly software package which implements all functionality described in this article. We provide detailed worked examples in several places alongside conceptual development. A dedicated section on computational experiments is provided, and several avenues of possible future research are outlined in a discussion section.

### 1.1 Article outline and our contributions

This article makes both mathematical and methodological contributions to signomial and polynomial optimization. Section 2 speaks to key questions which help place our work in a broader context. These questions include (1) What are the sources of error in nonnegativity-based relaxations of constrained optimization problems, and how are they usually mitigated? (2) How exactly are the original SAGE cones formulated? (3) How can we understand partial dualization in the context of existing nonnegativity and moment relaxations?

Once these questions are answered, we introduce the concept of conditional SAGE certificates for signomial nonnegativity (Section 3). We prove a representation result for the cone of these nonnegativity certificates (Theorem 6), and develop a solution recovery algorithm by investigating the dual cone (Algorithm 1). Section 3.4 describes two "hierarchies" of SAGE-based convex relaxations for signomial programs: 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 the idea of conditional SAGE certificates to polynomials. We discuss basic properties of the conditional SAGE polynomial cones before proving representation results (Theorems 9 and 10) which provide the basis for tractable relaxations of constrained polynomial optimization problems. Section 4.2 provides simple descriptions for dual conditional SAGE polynomial cones, and develops an efficient solution recovery algorithm based on these descriptions (Algorithm 2). Section 4.4 proposes reference hierarchies for polynomial optimization with SAGE certificates. Our minimax-free hierarchy has an interesting structure which reflects a link between SAGE signomials and SAGE polynomials, by way of the "signomial representatives" from [30].

Section 5 reports the effectiveness of our methodology on fifty-one problems appearing in the literature (sourced from [1, 2, 17, 23, 24, 28, 31, 34]), as well as randomly generated problems. A central component of our experiments is a desire to facilitate research both into 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.[1] `Sageopt` is a documented,

[1] `https://rileyjmurray.github.io/sageopt/`

tested, and convenient platform for constructing and solving SAGE relaxations, as well as analyzing the results thereof. We used `sageopt` for all experiments in this article.

### 1.2 Notation and preliminary definitions

Vectors and matrices always appear in boldface. The $i^{\text{th}}$ entry of a vector $\boldsymbol{v}$ is $v_i$, and the vector formed by deleting the $i^{\text{th}}$ 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 $i^{\text{th}}$ row of $\boldsymbol{A}$. All logarithms in this article are base-$e$. 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 $K \subset \mathbb{R}^r$, the dual cone is $K^\dagger \doteq \{\boldsymbol{y} : \boldsymbol{y}^\intercal \boldsymbol{x} \geq 0 \text{ for all } \boldsymbol{x} \text{ in } \mathbb{R}^r\}$. 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 $f = \text{Sig}(\boldsymbol{\alpha}, \boldsymbol{c})$ 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 $f = \text{Poly}(\boldsymbol{\alpha}, \boldsymbol{c})$ 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

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

and

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

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 $\mathsf{C}_{\mathsf{NNP}}(\boldsymbol{\alpha})$ [4]. The inner-approximations of nonnegativity cones developed in this article make use of the relative entropy function; this is the convex function "$D$" with domain $\mathbb{R}^m_+ \times \mathbb{R}^m_+$ taking values

$$D(\boldsymbol{u}, \boldsymbol{v}) = \sum_{i=1}^m u_i \log(u_i/v_i).$$

This article includes computational experiments with SAGE certificates and states solver runtimes for many of these examples. All of these examples rely on the MOSEK solver [32]. We use two different machines to provide a sense of when it may be practical to solve a SAGE relaxation with given computational resources. Machine **W** is an HP Z820 workstation, with two 8-core 2.6GHz Intel Xeon E5-2670 processors and 256GB 1600MHz DDR3 RAM. Machine **L** is a 2013 MacBook Pro, with a dual-core 2.4GHz Intel Core i5 processor and 8GB 1600MHz DDR3 RAM.

## 2 Background

In this article 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 $\phi$ 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 Section 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 the minimum background on SAGE and SOS nonnegativity certificates needed develop the contributions of this article. In Section 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 concludes with discussion on partial dualization. Until Section 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 $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) - \boldsymbol{\lambda}^\intercal g(\boldsymbol{x})$. The *Lagrange dual problem* is to compute

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

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

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 $(f, g)^{\text{d}}_{\mathbb{R}^n} \leq (f, g)^{\text{L}}_{\mathbb{R}^n}$. 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 problem as

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

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 $\gamma$ 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 SAGE and SOS nonnegativity certificates

In the development of the SAGE inner approximation for $\mathsf{C}_{\mathsf{NNS}}(\boldsymbol{\alpha})$, Chandrasekaran and Shah considered the structure where the coefficient vector $\boldsymbol{c}$ contained at most one negative entry $c_k$; if such a function was globally nonnegative, they called it an *AM/GM Exponential*, or an *AGE function* [26]. One thus defines the $k^{th}$ *AGE cone*

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

A key contribution of [26] was the use of convex duality to derive an efficient description of the AGE cones. The outcome of this derivation is that a vector $\boldsymbol{c}$ belongs to $\mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, k)$ iff $\boldsymbol{c}_{\setminus k} \geq \mathbf{0}$ and

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

The system of constraints given by (2) is crucially jointly convex in $\boldsymbol{c}$ and the auxiliary variable $\boldsymbol{\nu}$. The set defined by the *sum* of all AGE cones

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

is therefore efficiently representable.

The SAGE cone as defined above applies to signomials, but a similar construction exists for certifying global nonnegativity of polynomials [30]. Formally, we say that $f = \text{Poly}(\boldsymbol{\alpha}, \boldsymbol{c})$ is an *AGE polynomial* if it is nonnegative over $\mathbb{R}^n$, and if $f(\boldsymbol{x})$ contains at most one term $c_i \boldsymbol{x}^{\boldsymbol{\alpha}_i}$ that is not a monomial square. In conic form this writes as

$$\begin{aligned} \mathsf{C}_{\mathsf{AGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, k) = \{\boldsymbol{c} : \ & \text{Poly}(\boldsymbol{\alpha}, \boldsymbol{c})(\boldsymbol{x}) \geq 0 \text{ for all } \boldsymbol{x} \text{ in } \mathbb{R}^n, \text{ and} \\ & \boldsymbol{c}_{\setminus k} \geq \mathbf{0}, \ c_i = 0 \text{ for all } i \neq k \text{ with } \boldsymbol{\alpha}_i \notin 2\mathbb{N}^{1\times n}\}, \end{aligned} \tag{4}$$

and such AGE cones naturally give rise to

$$\mathsf{C}_{\mathsf{SAGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}) \doteq \sum_{k=1}^{m} \mathsf{C}_{\mathsf{AGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, k) \subset \mathsf{C}_{\mathsf{NNP}}(\boldsymbol{\alpha}). \tag{5}$$

The SAGE polynomial cone can also be described by an appropriate reduction to the SAGE signomial cone. For a nonnegative $m \times n$ integer matrix $\boldsymbol{\alpha}$ and a vector $\boldsymbol{c}$ in $\mathbb{R}^m$, we define the set of *signomial representative coefficient vectors* as

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

The name "signomial representative" derives from the fact that if $\hat{\boldsymbol{c}}$ belongs to $\text{SR}(\boldsymbol{\alpha}, \boldsymbol{c})$, then nonnegativity of the signomial $\text{Sig}(\boldsymbol{\alpha}, \hat{\boldsymbol{c}})$ would evidently imply nonnegativity of the polynomial $\text{Poly}(\boldsymbol{\alpha}, \boldsymbol{c})$ (see Section 5.1 of [30]). The set $\text{SR}(\boldsymbol{\alpha}, \boldsymbol{c})$

is useful because a constraint of the form "$\hat{\boldsymbol{c}}$ belongs to $\mathrm{SR}(\boldsymbol{\alpha}, \boldsymbol{c})$" is jointly convex in $\hat{\boldsymbol{c}}$ and $\boldsymbol{c}$. Lemma 19 of [30] proves that the SAGE polynomial cone defined by Equation 5 is equivalently given by

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

The generalization of SAGE polynomials considered in this article benefits from both the Sum-of-AGE-function and signomial-representative viewpoints of ordinary SAGE polynomials.

Lastly we consider *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 [5], and was subsequently developed by Parrilo [8] and Lasserre [10].

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

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 would be just as valid to have $\lambda_i$ be a *function*, provided that it was 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 $\lambda_i$ of the form $\lambda_i(x) = \eta_i \exp(x)$ with $\eta_i \geq 0$, the resulting dual bound is $-4 \leq (f, g)_{\mathbb{R}}^{\star}$.

Our 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 $w = \mathrm{Sig}(\boldsymbol{\alpha}, \mathbf{1})$ we can compute a non-decreasing sequence of lower bounds

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

Under appropriate conditions on $\boldsymbol{\alpha}$ (c.f. [26]), these lower bounds converge to $f^{\star}_{\mathbb{R}^n}$ as $\ell$ goes 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 $\gamma$.

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 could require that "$\psi(\gamma, \boldsymbol{\lambda}) \doteq w^{\ell}(\mathcal{L}(\boldsymbol{\lambda}) - \gamma)$ is SAGE." This would increase the size of the feasible set for variables $\gamma$ and $\boldsymbol{\lambda}$, and remain tractable due to the affine dependence of $\psi(\gamma, \boldsymbol{\lambda})$ on $\gamma$ 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

A *partial dual problem* is what results when the set "$X$" in Problem 1.1 is a proper subset of $\mathbb{R}^n$. In this case the natural generalization of the Lagrange dual is

$$(f, g)^{\mathrm{d}}_{X} \doteq \sup\{\gamma\ :\ \boldsymbol{\lambda} \geq \mathbf{0},\ \gamma \text{ in } \mathbb{R},\ \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) - \gamma \geq 0 \text{ for all } \boldsymbol{x} \text{ in } X\}. \tag{6}$$

The technique of *partial dualization* refers to the deliberate choice to restrict the Lagrangian to $X = \{\boldsymbol{x} : g_i(\boldsymbol{x}) \geq 0 \text{ for all } i \text{ in } \mathcal{I}\}$ for some $\mathcal{I} \subset [k]$, even when the constraints $\{g_i\}_{i \in \mathcal{I}}$ are of a functional form that is permitted in the Lagrangian. Note that in the extreme case with $X = \{\boldsymbol{x} : g(\boldsymbol{x}) \geq \mathbf{0}\}$, we are certain to 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.

Before getting into how SAGE certificates integrate with partial dualization, it is worth considering 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 single polynomial equality constraint $g(x) = 0$. In this case we could form a Lagrangian $\mathcal{L}(x, \mu) = f(x) - \mu g(x)$ with $\mu \in \mathbb{R}$, and find the largest constant $\gamma$ so that $\mathcal{L}(x, \mu) - \gamma$ was nonnegative over $x \in [a, b]$. A well-known result in real algebraic geometry is that a degree-$d$ polynomial "$p$" is nonnegative over an interval $[a, b]$ if and only if $p$ 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 [9]. Therefore the partial dual problem

$$(f,g)^{\mathrm{d}}_{[a,b]} = \sup\{\gamma \,:\, \gamma,\mu \in \mathbb{R},\ f(x) - \mu g(x) - \gamma \geq 0 \text{ for all } x \text{ in } [a,b]\}$$

can be framed as an SOS relaxation

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

Our last key concept is how partial dualization manifests in the dual of the dual. To develop this idea, consider $f = \mathrm{Poly}(\boldsymbol{\alpha}, \boldsymbol{c})$ with $\boldsymbol{\alpha}_1 = \mathbf{0}$, along with a set $X \subset \mathbb{R}^n$. The problem of computing $f^\star_X \doteq \inf_{\boldsymbol{x}\in X} f(\boldsymbol{x})$ has the following convex formulation

$$f^\star_X = \sup\{\,\gamma \,:\, \boldsymbol{c} - \gamma(1,0,\ldots,0) \ \in\ \mathsf{C}_{\mathsf{NNP}}(\boldsymbol{\alpha}, X)\}.$$

Of course, the above problem is intractable unless $\boldsymbol{\alpha}$ and $X$ satisfy very special conditions. In spite of the possible intractability, we can still compute the dual problem by applying standard rules of conic duality. The result of this process is

$$f^\star_X = \inf\{\boldsymbol{c}^\mathsf{T}\boldsymbol{v} \ :\ \boldsymbol{v} \in \mathsf{C}_{\mathsf{NNP}}(\boldsymbol{\alpha}, X)^\dagger,\ v_1 = 1\}$$

where $\mathsf{C}_{\mathsf{NNP}}(\boldsymbol{\alpha}, X)^\dagger$ is the dual cone to $\mathsf{C}_{\mathsf{NNP}}(\boldsymbol{\alpha}, X)$. This second problem is what we mean by "the dual of the dual." It appears prominently in the literature on polynomial optimization, where it is usually referred to as a *moment relaxation* [10]. The term "moment relaxation" derives from the fact that $\mathsf{C}_{\mathsf{NNP}}(\boldsymbol{\alpha}, \mathbb{R}^n)^\dagger$ is the smallest closed convex cone containing the vectors

$$(\mathbb{R}^n)^{\boldsymbol{\alpha}} \doteq \{(\boldsymbol{x}^{\boldsymbol{\alpha}_1}, \ldots, \boldsymbol{x}^{\boldsymbol{\alpha}_m}) \,:\, \boldsymbol{x} \in \mathbb{R}^n\},$$

and by thinking of a convex hull as computing expectations $\mathbb{E}_{\boldsymbol{x}\sim F}[(\boldsymbol{x}^{\boldsymbol{\alpha}_1}, \ldots, \boldsymbol{x}^{\boldsymbol{\alpha}_m})]$, where $F$ is a probability measure over $\mathbb{R}^n$. One can similarly understand the dual of a nonnegativty-based partial-dual problem in terms of probability and moment relaxations. When $X$ as a proper subset of $\mathbb{R}^n$, the convex hull of $X^{\boldsymbol{\alpha}}$ can be framed as the set of 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 over $X$. In this way, the "dual" of partial dualization can be understood in terms of *conditional moments*.

# 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 efficiently representable convex set $X$ gives rise to a parameterized and efficiently representable "$X$-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 $\{\boldsymbol{x} \,:\, g(\boldsymbol{x}) \leq \mathbf{1}\}$ 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

**Definition 1** (Conditional AGE signomial cones)**.** *For a matrix $\boldsymbol{\alpha}$ in $\mathbb{R}^{m\times n}$, a subset $X$ of $\mathbb{R}^n$, and an index $k$ in $[m]$, the $k^{th}$ AGE cone with respect to $\boldsymbol{\alpha}, X$ is*

$$\mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, k, X) = \{\boldsymbol{c} \in \mathbb{R}^m \,:\, \boldsymbol{c}_{\setminus k} \geq \mathbf{0} \text{ and } \mathrm{Sig}(\boldsymbol{\alpha}, \boldsymbol{c})(\boldsymbol{x}) \geq 0 \text{ for all } \boldsymbol{x} \text{ in } X\}.$$

**Definition 2** ($X$-SAGE signomials)**.** *If the vector $\boldsymbol{c}$ belongs to*

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

*then $f = \mathrm{Sig}(\boldsymbol{\alpha}, \boldsymbol{c})$ is an $X$-SAGE signomial.*

Conditional SAGE cones are order-reversing with respect to the second argument. That is, if $X_2 \subset X_1 \subset \mathbb{R}^n$, then $\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X_1) \subset \mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X_2)$ for all $\boldsymbol{\alpha}$ in $\mathbb{R}^{m\times n}$. Note that $\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)$ is defined for arbitrary $X \subset \mathbb{R}^n$, including nonconvex sets, and convex sets which admit no efficient description. Moreover, as a mathematical object, $\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)$ does not depend on the representation of $X$.

In practice we need conditions on $X$ in order to optimize over $\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)$. But before we get to those, it is worth mentioning some abstract results concerning optimization. For $f = \mathrm{Sig}(\boldsymbol{\alpha}, \boldsymbol{c})$ with $\boldsymbol{\alpha}_1 = \mathbf{0}$, define

$$f_X^{\mathrm{SAGE}} \doteq \sup\{\, \gamma \,:\, \gamma \text{ in } \mathbb{R},\ \boldsymbol{c} - \gamma(1, 0, \ldots, 0) \text{ in } \mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)\}$$

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

**Theorem 3.** *If $\boldsymbol{c} \geq \mathbf{0}$, then $f = \mathrm{Sig}(\boldsymbol{\alpha}, \boldsymbol{c})$ has $f_X^{\mathrm{SAGE}} = f_X^\star$ for all $X \subset \mathbb{R}^n$.*

*Proof.* Let $\boldsymbol{e}_1 = (1, 0, \ldots, 0)$. The signomial $\tilde{f} = \mathrm{Sig}(\boldsymbol{\alpha}, \boldsymbol{c} - f_X^\star \boldsymbol{e}_1)$ 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. □

**Theorem 4.** *If $X$ is bounded, then $f_X^{\mathrm{SAGE}} > -\infty$ 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 $\tilde{f}(\boldsymbol{x}) = c\exp(\boldsymbol{a}^\intercal \boldsymbol{x}) + L$ 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 $f_X^{\mathrm{SAGE}} \geq -L$. □

**Corollary 5** (See [30])**.** *Let $X \subset \mathbb{R}^n$ be arbitrary. If $\boldsymbol{c}$ is a vector in $\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)$ with nonempty $\mathcal{N} = \{i : c_i < 0\}$, then there exist vectors $\{\boldsymbol{c}^{(i)}\}_{i\in\mathcal{N}}$ satisfying*

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

*Proof.* This is simply the statement of Theorem 2 from [30], which was proven for ordinary SAGE cones, i.e. with $X = \mathbb{R}^n$. The entire proof of that theorem (including Lemmas 6 and 7 of [30]) extends to conditional SAGE cones simply by replacing references to "$\mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, i)$" and "$\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha})$" with "$\mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, i, X)$" and "$\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)$" respectively. □

In order to reliably optimize over $\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)$, we need $X$ to be a tractable convex set. This is essentially the only requirement on $X$, as is shown by the following theorem.

**Theorem 6.** *For a matrix $\boldsymbol{\alpha}$ in $\mathbb{R}^{m\times n}$, an index $i$ 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*

$$\begin{aligned}\mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, i, X) = \{\boldsymbol{c} : \boldsymbol{\nu} \text{ in } \mathbb{R}^{m-1}, \boldsymbol{c} \text{ in } \mathbb{R}^m, \boldsymbol{\lambda} \text{ in } \mathbb{R}^n \text{ satisfy} \\ \sigma_X(\boldsymbol{\lambda}) + D(\boldsymbol{\nu}, \boldsymbol{c}_{\setminus i}) - \boldsymbol{\nu}^\mathsf{T}\mathbf{1} \le c_i, \\ [\boldsymbol{\alpha}_{\setminus i} - \mathbf{1}\boldsymbol{\alpha}_i]^\mathsf{T}\boldsymbol{\nu} + \boldsymbol{\lambda} = \mathbf{0}, \text{ and } \boldsymbol{c}_{\setminus i} \ge \mathbf{0}\}.\end{aligned}$$

*Proof.* Let $\delta_X$ denote the indicator function of $X$, taking values

$$\delta_X(\boldsymbol{x}) = \begin{cases} 0 & \text{if } \boldsymbol{x} \text{ belongs to } X \\ +\infty & \text{if otherwise} \end{cases}.$$

A vector $\boldsymbol{c}$ with $\boldsymbol{c}_{\setminus i} \ge \mathbf{0}$ belongs to $\mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, i, X)$ if and only if

$$p^\star = \inf\{\delta_X(\boldsymbol{x}) + \textstyle\sum_{i=1}^{\ell} \tilde{c}_i \exp t_i \,:\, \boldsymbol{x} \in \mathbb{R}^n,\ \boldsymbol{t} \in \mathbb{R}^\ell,\ \boldsymbol{t} = \boldsymbol{W}\boldsymbol{x}\} \ge -L \tag{7}$$

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

The dual to the above optimization problem is easily calculated by applying Fenchel duality (c.f. [13]); the result of this process is

$$d^\star = \sup\{-\sigma_X(\boldsymbol{\lambda}) - D(\boldsymbol{\nu}, \tilde{\boldsymbol{c}}) + \boldsymbol{\nu}^\mathsf{T}\mathbf{1} \,:\, \boldsymbol{\lambda} \in \mathbb{R}^n, \boldsymbol{\nu} \in \mathbb{R}^{m-1},\ \boldsymbol{W}^\mathsf{T}\boldsymbol{\nu} + \boldsymbol{\lambda} = \mathbf{0}\}. \tag{8}$$

When $X$ is nonempty, one may verify that the hypothesis of Corollary 3.3.11 of [13] (concerning strong duality) hold for the primal-dual pair (7)-(8). In particular, $p^\star \ge -L$ holds if and only if $-d^\star \le L$, 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 $\boldsymbol{\nu}$ as zero vectors, we have $d^\star = +\infty$. The result follows. □

Theorem 6 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

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

An upper bound on the support function 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

$$\begin{aligned}\{\boldsymbol{c} : \boldsymbol{\nu} \text{ in } \mathbb{R}^{m-1}, \boldsymbol{c} \text{ in } \mathbb{R}^m, \text{ and } \boldsymbol{\eta} \text{ in } K^\dagger \\ D(\boldsymbol{\nu}, \boldsymbol{c}_{\setminus i}) - \boldsymbol{\nu}^\mathsf{T}\mathbf{1} + \boldsymbol{\eta}^\mathsf{T}\boldsymbol{b} \leq c_i, \\ [\boldsymbol{\alpha}_{\setminus i} - \mathbf{1}\boldsymbol{\alpha}_i]^\mathsf{T}\boldsymbol{\nu} = \boldsymbol{A}^\mathsf{T}\boldsymbol{\eta}, \text{ and } \boldsymbol{c}_{\setminus i} \geq \mathbf{0}\} \subset \mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, i, X).\end{aligned}$$

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

### 3.2 Dual perspectives and solution recovery

Here we discuss how 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 $g_i$, $\phi_i$ respectively, then

$$\inf\{\boldsymbol{c}^\mathsf{T}\boldsymbol{v} : \boldsymbol{v} \in \mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)^\dagger,\ v_1 = 1,\ \boldsymbol{G}\boldsymbol{v} \geq \mathbf{0},\ \boldsymbol{\Phi}\boldsymbol{v} = \mathbf{0}\} \tag{9}$$

is a convex relaxation of Problem 1. It is readily verified that if $\boldsymbol{v}^\star$ is an optimal solution to (9) and $\boldsymbol{v}^\star = \exp(\boldsymbol{\alpha}\boldsymbol{x})$ for some $\boldsymbol{x}$ in $X$, then $\boldsymbol{x}$ is optimal for (1).

The prospect of inverting the moment-type vector $\boldsymbol{v}^\star$ to obtain a feasible point $\boldsymbol{x} \in X$ drives our interest in understanding the dual cone $\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)^\dagger$. By standard rules in convex analysis, the dual SAGE cone is given by

$$\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)^\dagger = \cap_{i \in [m]} \mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, i, X)^\dagger.$$

An expression for the dual AGE cones can be recovered from Theorem 6. 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

$$\begin{aligned}\mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, i, X)^\dagger = \operatorname{cl}\{\boldsymbol{v} : v_i \log(\boldsymbol{v}/v_i) \geq [\boldsymbol{\alpha} - \mathbf{1}\boldsymbol{\alpha}_i]\boldsymbol{z} \\ (\boldsymbol{z}, v_i) \in \operatorname{co} X,\ \boldsymbol{v} \text{ in } \mathbb{R}^m_+, \text{ and } \boldsymbol{z} \text{ in } \mathbb{R}^n\}.\end{aligned} \tag{10}$$

The auxiliary variables "$\boldsymbol{z}$" appearing in the representation for the dual $X$-AGE cones are a powerful tool for solution recovery. If $\boldsymbol{z}, \boldsymbol{v}$ satisfy the constraints in (10) with $v_i > 0$, then $\boldsymbol{x} \doteq \boldsymbol{z}/v_i$ belongs to $X$. Additionally, if the inequality constraints involving the logarithm are binding *and* we meet the normalization condition $v_i = \exp(\boldsymbol{\alpha}_i \cdot \boldsymbol{x})$, then we will have $\exp(\boldsymbol{\alpha}\boldsymbol{x}) = \boldsymbol{v}$.

These observations form the core – but not the entirety – of our solution recovery algorithm. Depending on solver behavior and dimension-reduction techniques used to simplify a SAGE relaxation, it is possible that $\boldsymbol{v} = \exp(\boldsymbol{\alpha}\boldsymbol{x})$ for some $\boldsymbol{x}$ in $X$, and yet no dual AGE cone generated this value. Therefore it is also prudent to solve a constrained least-squares problem to find $\boldsymbol{x}$ in $X$ with $\boldsymbol{\alpha}\boldsymbol{x}$ near $\log \boldsymbol{v}$.

**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 $\boldsymbol{z}$ in $\mathbb{R}^n$ s.t. $v_j \log(\boldsymbol{v}/v_j) \geq [\boldsymbol{\alpha} - \mathbf{1}\boldsymbol{\alpha}_j]\boldsymbol{z}$ and $(\boldsymbol{z}, v_j) \in \operatorname{co} X$.
5: $\boldsymbol{x} \leftarrow \boldsymbol{z}/v_j$.
6: solutions.append($\boldsymbol{x}$).
7: **end for**
8: **if** $\boldsymbol{\alpha}\boldsymbol{x} \neq \log \boldsymbol{v}$ for all $\boldsymbol{x}$ in solutions **then**
9: Compute $\boldsymbol{x}_{\text{ls}}$ in argmin$\{\| \log \boldsymbol{v} - \boldsymbol{\alpha}\boldsymbol{x}\| : \boldsymbol{x} \text{ in } X\}$.
10: solutions.append($\boldsymbol{x}_{\text{ls}}$).
11: **end if**
12: solutions $\leftarrow$ [ $\boldsymbol{x}$ in solutions if $g(\boldsymbol{x}) \geq -\epsilon_{\text{ineq}}$ and $|\phi(\boldsymbol{x})| \leq \epsilon_{\text{eq}}$ ].
13: solutions.sort($f$, increasing).
14: **return** solutions.
15: **end procedure**

Assuming that Equation 10 is used to represent the dual AGE cones, the runtime of Algorithm 1 is dominated by Line 10. The runtime of Line 10, in turn, should be substantially smaller than the time required to compute $\boldsymbol{v} \in \mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha})^\dagger$ which is optimal for an appropriate SAGE relaxation.

Note how Algorithm 1 implicitly assumes that $\boldsymbol{v}$ is elementwise positive. For numerical reasons, this will be the case in practice. At a theoretical level, if $\boldsymbol{v}$ contains some component $v_i = 0$, then there is no $\boldsymbol{x}$ in $\mathbb{R}^n$ which could attain $v_i = \exp(\boldsymbol{\alpha}_i \cdot \boldsymbol{x})$, and so solution recovery is not a well-posed problem in such situations. Note also how in the important (and possibly nontrivial) case with $k_1 = k_2 = 0$, Algorithm 1 always returns at least one feasible solution.

In the authors experience it is useful to take solutions generated from Algorithm 1 and pass them to a local solver as initial conditions. This additional step is especially worthwhile when there is a gap between a SAGE bound and a problem's true optimal value, or when solution recovery to the desired precision of $\epsilon_{\text{ineq}}, \epsilon_{\text{eq}}$ fails. The term "Algorithm 1L" henceforth refers to the use of Algorithm 1, followed by local-solver solution refinement. For the examples in this article, the authors use Powell's COBYLA solver for the solution refinement [6].[2]

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

### 3.3 A first worked example

The following problem has appeared in many articles concerning algorithms for signomial programming [11, 14, 16, 18, 23].

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

This problem ("Example 1") is an excellent candidate for conditional SAGE relaxations, because each of the seven constraints defines an efficiently representable convex set. Constraint functions $g_{2:7}$ can be can be represented with six linear inequalities, and the constraint function $g_1$ can be represented with three exponential cones and one linear inequality. As a separate matter, 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 describe how SAGE relaxations fare for Example 1. We begin by setting $X = \{\boldsymbol{x} : g_{1:7}(\boldsymbol{x})\}$; since $X$ is bounded, Theorem 4 tells us that $f_X^{\text{SAGE}}$ is finite. The dual SAGE relaxation can be solved with MOSEK on Machine **L** in 0.01 seconds, and provides us with a lower bound $f_X^{\text{SAGE}} = -147.86 \leq f_X^\star$. By running Algorithm 1 on the dual solution, we recover

$$\boldsymbol{x}^\star = (5.01063529,\; 3.40119660, -0.48450710) \quad \text{satisfying} \quad f(\boldsymbol{x}^\star) = -147.66666.$$

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 $f_X^{\text{SAGE}} < f_X^\star$ can be attributed to how our solution recovery algorithm differs from traditional "moment" techniques. As it happens, the point $\boldsymbol{x}^\star$ recovered from Algorithm 1 is actually an optimal solution to Example 1. In order to certify this fact, we need stronger SAGE relaxations. Table 1 shows the results of these relaxations, using the minimax-free hierarchy described in Section 3.4.

| 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 |

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 $10^{-8}$.

### 3.4 Reference hierarchies for signomial programming

This section gives a particular set of choices regarding SAGE-based hierarchies for signomial programming. Because SAGE certificates are sparsity-preserving, one must be careful when describing relaxations which use nonconstant Lagrange multipliers, or positive-definite modulators. In particular, 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 that is identically equal to one.

First we describe a SAGE-based hierarchy that does not make use of the minimax inequality. This could be understood as a hierarchy for unconstrained optimization, but really applies whenever minimizing a signomial over a tractable convex set $X \subset \mathbb{R}^n$. In the event that we cannot certify nonnegativity $f - \gamma$ with $\gamma = f_X^\star$, we can use modulators as described in Section 2.3 to improve the largest SAGE-certifiable bound on $f$. Formally, for a signomial $f$ over exponents $\boldsymbol{\alpha}$, a nonnegative integer $\ell$, and a tractable convex set $X$, the *level-$\ell$ SAGE relaxation* for $f_X^\star$ is

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

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

Now we consider the case with functional constraints; let $f$, $g_i$, and $\phi_i$ 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 $\ell$. Starting from $p \geq 0$ and $q \geq 1$, define $\boldsymbol{\alpha}[p]$ as the matrix of exponent vectors for $\mathrm{Sig}(\boldsymbol{\alpha}, \mathbf{1})^p$, 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 $(p, q, \ell)$ is then

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

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 $s_h$ are required to be nonnegative only over $X$, rather than over the whole of $\mathbb{R}^n$.

Once an appropriate SAGE relaxation has been solved, there is the matter of attempting to recover a solution from the dual problem. Oftentimes a SAGE relaxation produces a tight bound on $(f, g, \phi)_X^\star$, and yet no feasible solution can be recovered from Algorithm 1 with reasonable values of $\epsilon_{\mathrm{eq}}$. Thus we also suggest that one eliminate equality constraints through substitution of variables, when possible. When it is not possible to eliminate all equality constraints, we recommend

allowing large violations of equality constraints in Algorithm 1 (e.g. $\epsilon_{\text{eq}} = 1.0$), and passing the returned values as near-feasible points to a local solver in the manner of Algorithm 1L.[3] This principle also extends to allowing large values of $\epsilon_{\text{ineq}}$ prior to solution-refinement, however the authors find that this is usually not necessary.

### 3.5 A second worked example

This section's example can be found in the 1976 PhD thesis of James Yan [1], where it illustrates signomial programming in the service of structural engineering design. This problem is notable because it is nonconvex even when written in exponential form. Such signomial programs 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]

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

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 seconds of solver time. This bound is very close to the optimal value claimed by Yan [1]. However, Algorithm 1 only returns candidate solutions "$\boldsymbol{x}$" with equality constraint violations $\phi(\boldsymbol{x}) \approx 70$.

In an effort 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 geometric-form) 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

$$\boldsymbol{A} = (7.0711000\mathsf{e}{-}04,\ 7.0711000\mathsf{e}{-}04,\ 1.00000000\mathsf{e}{-}08),\ P = \frac{70.7107A_3}{A_1 + A_3}.$$

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 seconds on Machine **W**.

[3] COBYLA is an excellent example of a solver which suppports infeasible starts.

[4] The objective and inequality constraint functions are multiplied by $10^4$ for numerical reasons; see equation environment (6.15) on page 106 of [1] 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 hitherto preferred the exponential form, primarily because it allows us to build upon the substantial theories of convex analysis and convex optimization. However it is important to acknowledge that 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 for signomial nonnegativity – both in effort to appeal to those who are accustomed to working with geometric-form signomial programs, and as a prelude to our discussion on conditional SAGE 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

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

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 $\mathsf{C}^{\mathsf{GEOM}}_{\mathsf{NNS}}(\boldsymbol{\alpha}, X) = \mathsf{C}_{\mathsf{NNS}}(\boldsymbol{\alpha}, \log X)$. Thus for $X \subset \mathbb{R}^n_{++}$, one arrives naturally at the definition

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

From here it should be easy to deduce various corollaries for $\mathsf{C}^{\mathsf{GEOM}}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)$, by appealing to results from Section 3.1. The most important such result is that one can efficiently optimize over $\mathsf{C}^{\mathsf{GEOM}}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)$ whenever $\log X$ is a convex set for which the epigraph of the support function is efficiently representable.

## 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 Section 3. Definitions, representations, and other basic theorems for the conditional SAGE polynomial cones are given in Section 4.1. Section 4.2 covers solution recovery from dual SAGE relaxations, and Section 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 "minmax free." Section 4.5 applies various manifestations of the minimax hierarchy to an example problem.

### 4.1 The conditional SAGE polynomial cones

We call $f = \mathrm{Poly}(\boldsymbol{\alpha}, \boldsymbol{c})$ an *$X$-AGE polynomial* if it is nonnegative over $X$, and $f(\boldsymbol{x})$ contains at most one term $c_i \boldsymbol{x}^{\boldsymbol{\alpha}_i}$ which is negative for some $\boldsymbol{x}$ in $X$.

**Definition 7** (Conditional AGE polynomial cones)**.** *For $\boldsymbol{\alpha}$ in $\mathbb{N}^{m \times n}$, a subset $X$ of $\mathbb{R}^n$, and an index $i$ in $[m]$, the $i^{th}$ AGE polynomial cone with respect to $\boldsymbol{\alpha}, X$ is*

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

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 $\boldsymbol{c} \in \mathsf{C}_{\mathsf{AGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, i, X)$ and $i \neq j$. 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 Definition 7 agrees with the definition of ordinary AGE polynomial cones, as proposed in [30] and as restated in Equation 4. 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 $\mathsf{C}_{\mathsf{AGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, i, X) = \mathsf{C}_{\mathsf{AGE}}(\boldsymbol{\alpha}, i, \log X)$. With these facts in mind, we define the conditional SAGE polynomial cone in the natural way.

**Definition 8** ($X$-SAGE polynomials)**.** *If the vector $\boldsymbol{c}$ belongs to*

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

*then we say that $f = \mathrm{Poly}(\boldsymbol{\alpha}, \boldsymbol{c})$ is an $X$-SAGE polynomial.*

Many of our earlier theorems for signomials apply to $X$-SAGE polynomials without any special assumptions on $X$. For example, it is easy to show that Theorem 4 applies to conditional SAGE polynomials: if $X$ is bounded, then $f = \mathrm{Poly}(\boldsymbol{\alpha}, \boldsymbol{c})$ with $\boldsymbol{\alpha}_1 = \mathbf{0}$ has

$$f_X^{\mathrm{SAGE}} \doteq \sup\{\, \gamma \, : \, \gamma \text{ in } \mathbb{R}, \ \boldsymbol{c} - \gamma(1, 0, \ldots, 0) \text{ in } \mathsf{C}_{\mathsf{SAGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, X)\} > -\infty.$$

Corollary 5 likewise extends to polynomials. Other than substituting AGE signomial cones with AGE polynomial cones, the only difference is that $\mathcal{N}$ becomes $\mathcal{N} = \{i : c_i \boldsymbol{x}^{\boldsymbol{\alpha}_i} < 0 \text{ for some } \boldsymbol{x} \text{ in } X\}$.

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 $X^\circ$ 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 9.** *Suppose $X \subset \mathbb{R}^n_+$ is representable as $X = \mathrm{cl}\{\boldsymbol{x} : \boldsymbol{0} < \boldsymbol{x}, H(\boldsymbol{x}) \leq \boldsymbol{1}\}$ for a continuous map $H : \mathbb{R}^n \to \mathbb{R}^r$. Then for $Y = \{\boldsymbol{y} : H(\exp \boldsymbol{y}) \leq \boldsymbol{1}\}$, we have $\mathsf{C}^{\mathsf{POLY}}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X) = \mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, Y)$.*

The proof of Theorem 9 is straightforward, and hence omitted. A more sophisticated result concerns when $X$ is not contained in any particular orthant, but nevertheless possesses a certain sign-symmetry.

**Theorem 10.** *Suppose $X \subset \mathbb{R}^n$ is representable as $X = \mathrm{cl}\{\boldsymbol{x} : \boldsymbol{0} < |\boldsymbol{x}|, H(|\boldsymbol{x}|) \leq \boldsymbol{1}\}$ for a continuous map $H : \mathbb{R}^n \to \mathbb{R}^r$. Then for $Y = \{\boldsymbol{y} : H(\exp \boldsymbol{y}) \leq \boldsymbol{1}\}$, we have*

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

By combining Theorem 6 with Theorems 9 and 10, 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 $\mathsf{C}^{\mathsf{POLY}}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)$; 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 10.* Suppose that $\boldsymbol{c}$ in $\mathsf{C}^{\mathsf{POLY}}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)$ admits the decomposition $\boldsymbol{c} = \sum_{i=1}^m \boldsymbol{c}^{(i)}$, where $\boldsymbol{c}^{(i)}$ belongs to the $i^{\text{th}}$ AGE polynomial cone with respect to $\boldsymbol{\alpha}, X$. Define $\{\tilde{\boldsymbol{c}}^{(i)}\}_{i=1}^m$ as follows

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

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

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

The signomials $\mathrm{Sig}(\boldsymbol{\alpha}, \tilde{\boldsymbol{c}}^{(i)})$ 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 Equation (12) contains $\mathsf{C}^{\mathsf{POLY}}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)$.

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 5. Setting $\mathcal{N} = \{i : \tilde{c}_i \leq 0\}$, there exist vectors $\tilde{\boldsymbol{c}}^{(i)}$ satisfying

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

Note how the definition of $\mathrm{SR}(\boldsymbol{\alpha}, \boldsymbol{c})$ ensures that $\mathcal{N} = \{i : \boldsymbol{\alpha}_i \text{ is not even, or } c_i \leq 0\}$. 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 $\boldsymbol{c} = \sum_{i \in \mathcal{N}} \boldsymbol{c}^{(i)}$, and each $\boldsymbol{c}^{(i)}$ has the necessary sign pattern for membership in the $i^{\text{th}}$ AGE cone with respect to $\boldsymbol{\alpha}, X$. Finally, note that

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

to complete the proof. □

### 4.2 Solution recovery for sparse moment problems

This section concerns using dual SAGE relaxations to recover solutions to optimization problems of the form (1.1), where $f$ and $g_i$ are polynomials over exponents $\boldsymbol{\alpha}$. If $\boldsymbol{G}$ a $k \times m$ matrix whose $i^{\text{th}}$ row is the coefficient vector of $g_i$, $\boldsymbol{\alpha}_1$ is the zero vector, and $\boldsymbol{c}$ is the coefficient vector of $f$, then the simplest such relaxation is

$$\inf\{\boldsymbol{c}^{\mathsf{T}}\boldsymbol{v} \,:\, \boldsymbol{v} \in \mathsf{C}_{\mathsf{SAGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, X)^{\dagger},\ v_1 = 1,\ \boldsymbol{G}\boldsymbol{v} \geq \boldsymbol{0}\}. \tag{13}$$

Overall, our goal is to recover vectors $\boldsymbol{x}$ in $X$ satisfying $\boldsymbol{v} = (\boldsymbol{x}^{\boldsymbol{\alpha}_1}, \ldots, \boldsymbol{x}^{\boldsymbol{\alpha}_m})$, where $\boldsymbol{v}$ is an optimal solution to a relaxation such as the one above. We assume that $X$ is of a form where one of Theorems 9 or 10 provide a tractable representation of $\mathsf{C}_{\mathsf{SAGE}}(\boldsymbol{\alpha}, X)$; this assumption allows us to leverage the following corollary.

**Corollary 11.** *Fix $Y = \{\boldsymbol{y} : H(\exp \boldsymbol{y}) \leq \boldsymbol{1}\}$ for a continuous $H : \mathbb{R}^n \to \mathbb{R}^r$.*

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

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

We make a running assumption that "$Y$" is convex.

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 Sections 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($f, g, \phi, \boldsymbol{\alpha}, \boldsymbol{v}, \hat{\boldsymbol{v}}, \epsilon_{\text{ineq}}, \epsilon_{\text{eq}}, \epsilon_0$)
    $M \leftarrow$ VariableMagnitudes($\boldsymbol{\alpha},\ \boldsymbol{v},\ \hat{\boldsymbol{v}},\ \epsilon_0$).  # Algorithm 3
    $S \leftarrow \{\mathbf{1}\}$
    **if** $X$ is not a subset of $\mathbb{R}_+^n$ **then**
        $S$.union( VariableSigns($\boldsymbol{\alpha},\ \boldsymbol{v}$) )  # Algorithm 4
    **end if**
    solutions $\leftarrow$ [].
    **for** $\boldsymbol{x}_{\text{mag}}$ in $M$ and $\boldsymbol{s}$ in $S$ **do**
        $\boldsymbol{x} \leftarrow \boldsymbol{x}_{\text{mag}} \odot \boldsymbol{s}$  # denotes elementwise multiplication
        **if** $g(\boldsymbol{x}) \geq -\epsilon_{\text{ineq}}$ and $|\phi(\boldsymbol{x})| \leq \epsilon_{\text{eq}}$ **then**
            solutions.append($\boldsymbol{x}$)
        **end if**
    **end for**
    solutions.sort($f,$ increasing).
    **return** solutions.
**end procedure**

If $\boldsymbol{v}$ is optimal for Problem (13) 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 1.1. As with Algorithm 1 in the signomial case, the authors find it useful to apply a simple local solver to the output of Algorithm 2 as a sort of solution refinement. We say "Algorithm 2L" in reference to such a method, with COBYLA as the local solver.

### 4.2.1 Recovering variable magnitudes

Given $\boldsymbol{v}$ in $\mathsf{C}_{\mathsf{SAGE}}^{\mathsf{POLY}}(\boldsymbol{\alpha}, X)^\dagger$, we want to find $\boldsymbol{x} \in X$ 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 Section 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

$$(\boldsymbol{x}_i^{\boldsymbol{\alpha}_1}, \ldots, \boldsymbol{x}_i^{\boldsymbol{\alpha}_m}) = \hat{\boldsymbol{v}} \tag{14}$$

under the binding-constraint and normalization conditions discussed in Section 3.2. Of course, if $\hat{\boldsymbol{v}} = |\boldsymbol{v}|$, then Equation (14) is precisely what we desire from our variable magnitudes. 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 $\hat{\boldsymbol{v}}$ does not equal $|\boldsymbol{v}|$. 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

$$\boldsymbol{y} \in \operatorname{argmin}\{\textstyle\sum_{i:v_i\neq 0}(\boldsymbol{\alpha}_i \cdot \boldsymbol{y} - \log|v_i|)^2 \ : \boldsymbol{y} \text{ in } Y, \quad \boldsymbol{\alpha}_i \cdot \boldsymbol{y} \leq \log(\epsilon_0) \text{ for all } v_i = 0\} \tag{15}$$

and exponentiates $\boldsymbol{x} = \exp \boldsymbol{y}$. The role of $\epsilon_0$ is to ensure that $\boldsymbol{x} = \exp \boldsymbol{y}$ satisfies $|\boldsymbol{x}|^{\boldsymbol{\alpha}_i} \leq \epsilon_0$ whenever $v_i = 0$. Extremely small values of $\epsilon_0$ (such as $10^{-100}$) would be 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

For $\boldsymbol{v}$ in $\mathbb{R}^m$, 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 that $s_i = +1$ 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 the equality $\boldsymbol{x}^{\boldsymbol{\alpha}_i} = v_i \neq 0$ 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

$$\boldsymbol{\alpha}[U, :]\boldsymbol{z} \equiv (\boldsymbol{v} < 0)[U] \mod 2 \quad \text{and} \quad z_j = 0 \text{ for all } j \text{ in } [n] \setminus W \tag{16}$$

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 (16) can be computed efficiently by Gaussian elimination over the finite field $\mathbb{F}_2$. Standard techniques for finite-field linear algebra also allow us to compute a basis for a null space of a matrix in mod 2 arithmetic (c.f. [19]), and so in principle one can readily recover all possible solutions to (16). In practice we must be careful, since the number of solutions to the linear system can easily be exponentially large in $n$ (for example, when $\boldsymbol{\alpha} \equiv \boldsymbol{0}_{n\times m} \mod 2$). Our formal algorithm for sign recovery accounts for this fact, and employs an additional hueristic to handle the case when (16) is inconsistent. 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 [7]. 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 [17]. 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 [21] to compute $f_X^\star = -7$ results in a semidefinite program that takes MOSEK 90 seconds to solve with Machine **W**.

SAGE relaxations automatically exploit the structure in this problem. Since the seven functions $f_i(\boldsymbol{x}) = 1 - 64 \prod_{j \neq i} x_i$ are $X$-AGE and sum to $f + 7$, we have that $-7 \leq f_X^{\text{SAGE}} \leq f_X^\star$. To address the dual SAGE relaxation and solution recovery, we introduce the $8 \times 7$ 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

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

MOSEK solves this problem in 0.01 seconds 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 Section 3.4, where "Sig" is replaced by "Poly" 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 $(p, q, \ell)$ is then

$$
\begin{aligned}
(f, g, \phi)_X^{(p,q,\ell)} = \sup \ \ \gamma \ \ \text{s.t.} \ \ & s_h, z_h \text{ are polynomials over exponents } \hat{\boldsymbol{\alpha}}[p] \\
& \mathcal{L} \doteq f - \gamma - \textstyle\sum_{h \in g[q]} s_h \cdot h - \sum_{h \in \phi[q]} z_h \cdot h \\
& \mathrm{Poly}(2\boldsymbol{\alpha}, \mathbf{1})^\ell \mathcal{L} \text{ is an } X\text{-SAGE polynomial} \\
& s_h \text{ are } X\text{-SAGE polynomials.}
\end{aligned}
\tag{17}
$$

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 (17) and it's signomial equivalent (11), is that the Lagrange multipliers are slightly more complex in (17). 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 $f = \mathrm{Poly}(\mathcal{A}(f), \mathcal{C}(f))$ 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

$$
\begin{aligned}
f_X^{(p,q)} = \sup \ \ & \gamma \\
\text{s.t.} \ \ & \psi \doteq \mathrm{Poly}(\boldsymbol{\alpha}, \boldsymbol{s})^p (f - \gamma) \\
& \boldsymbol{c} \in \mathrm{SR}(\mathcal{A}(\psi), \mathcal{C}(\psi)) \\
& [\mathrm{Sig}(\mathcal{A}(\psi), \mathbf{1})]^q \, \mathrm{Sig}(\mathcal{A}(\psi), \ \boldsymbol{c}) \text{ is } Y\text{-SAGE}
\end{aligned}
\tag{18}
$$

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

Formulation (18) uses two parameters out of desire to mitigate *both* sources of error in the SAGE polynomial cone: the error from replacing a polynomial by its signomial representative, and the error from replacing the signomial nonnegativity cone by the SAGE cone. As we show in Section 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

The following problem appears in work on "Bounded Degree Sums-of-Squares" (BSOS) and "Sparse Bounded Degree Sums-of-Squares" (Sparse-BSOS) methods for polynomial optimization [28, 31]. The latter paper reports that BSOS and Sparse-BSOS compute $(f, g)^\star_{\mathbb{R}^6} = -0.41288$ in 44.5 and 82.1 seconds respectively, when using SDPT3-4.0 on a machine with a quad-core 2.6GHz Core i7 processor and 16GB RAM.

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

This problem (Example 4) 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 (17) 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 seconds of solver time on Machine **W**.

A strictly preferable alternative is to apply partial dualization with $X = \mathbb{R}^6_+$. With this choice of $X$ it is natural to drop the first two and last six constraints from $g$ (all of which will be trivially satisfied), 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 seconds of solver time on Machine **W**, and 4.4 seconds of solver time on Machine **L**. It is significant that the SAGE relaxation can be solved in this amount of time on Machine **L**, since it is an order of magnitude faster than BSOS solve time reported in [31].

The most aggressive choice for partial dualization is $X = \{\boldsymbol{x} : \boldsymbol{x} \geq \mathbf{0}, g_{6:7} \geq \mathbf{0}\}$. With this choice of $X$ one has the option of using $\hat{g} = (g_{3:5}, g_{8:10})$, 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 seconds, and in the second case Machine **W** computes $(f,\hat{g})^{(1,1,0)}_X = -0.41288$ in 5.67 seconds. It is worth emphasizing that even 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 Section 5.1). Examples for polynomial optimization include twenty-two problems from the literature (Section 5.2), as well as randomly generated problems (Section 5.3).

All experiments described here were run with the provided `sageopt` python package. `Sageopt` is an extension and refinement of the "`sigpy`" package introduced by the authors in the appendix of [30]. In the spirit of Gloptipoly3 [21],

`sageopt` includes its own basic rewriting system to cast a SAGE relaxation into conic forms acceptable by appropriate solvers. The rewriting system also provides mechanisms for computing constraint violations, analyzing low-level problem data, and constructing a set $X$ (in an appropriate representation) from lists of constraint functions $g$ and $\phi$. Once problem data $f$, $g$, and $\phi$ are defined, a SAGE relaxation can be constructed and solved in just two lines of code. Solution recovery similarly requires no more than two lines of code. `Sageopt` currently supports the conic solvers ECOS [22, 25] and MOSEK [32].

Unless otherwise stated, experiments were conducted on Machine **W**. All experiments were conducted with the MOSEK solver, using the default solver tolerances. We note that although Sections 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 [1], a popular benchmarking paper by Rijckaert and Martens [2], and the more contemporary works [23, 24]. The organization of this section is chronological with respect to these three sources. Many of the problems considered here can be found elsewhere in the literature, including work by Shen et al. [11, 14, 18], Wang and Liang [12] and Qu et al. [16].

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-$(p, q, \ell)$ 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 constraint 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* [1]. 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.

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

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 seconds, but no solution can be recovered from Algorithm 1 unless $\epsilon$ is set to an unacceptably large value of 0.1. Instead we set $X = \{\boldsymbol{x} : g(\boldsymbol{x}) \geq \boldsymbol{0}\}$, compute $(f, g, \phi)^{(1,1,0)}_X = 1.92592593$ in 0.18 seconds, and by running Algorithm 1 recover $\boldsymbol{x}^\star$ satisfying $g(\boldsymbol{x}^\star) > $ 1E-11, $|\phi(\boldsymbol{x}^\star)| <$ 1E-8, and $f(\boldsymbol{x}^\star) = 1.92592593$.

$$\begin{aligned}
\inf_{\boldsymbol{y}\in\mathbb{R}^3_{++}}\ & y_1^{0.6}y_2 + y_2y_3^{-0.5} + 15.98y_1 + 9.0824y_2^2 - 60.72625y_3 \\
\text{s.t. } & y_2^{-2}y_3 - y_1y_2^{-2} - 0.48 \geq 0 \\
& y_1^{0.5}y_3^2 - y_1^{0.25}y_3 - y_2^2 - 5.75 \geq 0 \\
& (1000, 1000, 1000) \geq \boldsymbol{y} \geq (0.1, 0.1, 0.1) \\
& y_1^2 + 4y_2^2 + 2y_3^2 - 58 = 0 \\
& y_1y_2^{-1}y_3^{2.5} + y_2y_3 - y_2^2 - 16.55 = 0
\end{aligned} \tag{Ex6}$$

Setting $X = \{\boldsymbol{x} \in \mathbb{R}^3 : g(\boldsymbol{x}) \geq \boldsymbol{0}\}$, we can compute $(f, g_{1:2}, \phi)^{(0,1,0)}_X = -320.722913$ in 0.04 seconds. By running Algorithm 1 with $\epsilon_{\text{ineq}} =$1E-8 and $\epsilon_{\text{eq}} =$1E-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 parameter `RHOEND`=1E-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 Section 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-$(p, q, \ell)$ 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 attempted to solve problems 9 through 18 of the popular signomial-geometric programming benchmark paper by Rijckaert and Martens [2]. 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 qualitative 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.

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

The bound constraints appearing in Example 7 are not included in [2], however $f$ is unbounded below if we omit them. The solution proposed in [2] 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 seconds on Machine **W**.

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

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

| Num. in [2] | $n$ | $k$ | 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 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 [2]. 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.

| | SAGE relaxation | | Algorithm 1 | | Algorithm 1L | |
|---|---|---|---|---|---|---|
| Num. in [2] | $(p,q,\ell)$ | 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 |

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.

#### 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 [23], as well as four problems from the 2014 article by Xu [24]. 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 [23]-8 as our Example 9.

| source | num. | $n$ | $k_1$ | $k_2$ | objective | infeasibility | optimal? |
|---|---|---|---|---|---|---|---|
| [23] | 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 | 10122.493 | 4.00E-13 | unknown |
| - | 7 | 3 | 6 | 0 | -10.363636 | 2.00E-15 | yes |
| - | 8 | 15 | 37 | 6 | 156.21963 | 4.00E-14 | yes |
| [24] | 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 | 10252.790 | 8.00E-14 | unknown |

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.

$$\begin{aligned}
\inf_{\boldsymbol{y}\in\mathbb{R}^{15}_{++}} \ & \textstyle\sum_{i=1}^{4} y_{i+11}(12.62626 - 1.231059y_i) \qquad \text{(Ex9)}\\
\text{s.t. } & y_{12} - y_{11} \leq 0, \quad y_{11} - y_{12} \leq 50, \quad y_{10} - y_4 \leq 0\\
& y_9 - y_{10} \leq 0, \quad y_8 - y_9 \leq 0, \quad 2y_7 - y_1 \leq 1\\
& y_3 - y_4 \leq 0, \quad y_2 - y_3 \leq 0, \quad y_1 - y_2 \leq 0\\
& 50y_4 + y_{10}y_{15} - 50y_{10} - y_4y_{15} \leq 0\\
& 50y_{10} + y_4y_5 + y_9y_{14} - 50y_9 - y_3y_{14} - y_8y_{15} \leq 0\\
& 50y_7 + y_2y_{13} + y_7y_{12} - 50y_8 - y_1y_{12} - y_8y_{13} \leq 0\\
& 50y_8 + y_1y_{12} + y_8y_{13} - 50y_7 - y_2y_{13} - y_7y_{12} \leq 0\\
& 50y_8 + 50y_9 + y_3y_{14} + y_8y_{13} - y_2y_{13} - y_9y_{14} \leq 500\\
& y_6y_{11} + y_1y_{12} + y_7y_{11} - y_6y_{12} \leq 0\\
& 100y_{i+5} + 0.0975y_i^2 - 3.475y_i - 9.75y_iy_{i+5} \leq 0 \text{ for all } i \text{ in } [5]\\
& \boldsymbol{y} \geq (1.000000, 1, 9, 9, 9, 1, 1.000000, 1, 1, 1, 50, 0.0, 1.0, 50, 50)\\
& \boldsymbol{y} \leq (8.037732, 9, 9, 9, 9, 1, 4.518866, 9, 9, 9, 100, 50, 50, 50, 50)
\end{aligned}$$

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 [23]-8, in that a constraint "$x_3x_2 - x_3 \leq 0$" in the original problem statement was replaced by "$y_2 - y_3 \leq 0$" 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 $(f,g)^\star_X \geq 156.2196$ in 0.05 seconds. By running Algorithm 1L with $\epsilon_{\text{ineq}} = 100$, we subsequently obtain the geometric-form solution

$$\boldsymbol{y}^\star_{1:8} = (8.037732, 9, 9, 9, 9, 1, 1, 1.15686275)$$
$$\boldsymbol{y}^\star_{9:15} = (1.21505203, 1.58987319, 50, 3\text{E-}50, 1, 50, 50).$$

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.

| source-num. | $(p,q,\ell)$ | bound | $\epsilon_{\text{ineq}}$ | $\epsilon_{\text{eq}}$ | objective | infeasibility |
|---|---|---|---|---|---|---|
| [23]-5 | (0,1,0) | 9171.00 | 1.00E-08 | 0 | 10122.493 | 4.00E-13 |
| [24]-5 | (2,2,0) | -0.390 | 1.00E-08 | 1 | -0.3888114 | 5.00E-17 |
| [24]-7 | (0,1,0) | 9397.8 | 1.00E-08 | 1 | 10252.790 | 8.00E-14 |

Table 5: Signomial programs for which we did not certify optimality, but nevertheless recovered best-known solutions by using SAGE relaxations. Columns $\epsilon_{\text{ineq}}$ and $\epsilon_{\text{eq}}$ 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. [24]-7 reports a solution with smaller objective value, however that solution violates an equality constraint with forward error in excess of 0.11.

### 5.2 Polynomial optimization problems from the literature

Here we review results of the reference hierarchies from Section 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 [34] and [17] 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 [28]. 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 gives problem data for the unconstrained and box-constrained problems; three such problems are reproduced here, as our Examples 10 through 12.

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

The polynomial $f$ in Example 10 is known as the six-hump camel function; its minimum $f^\star_{\mathbb{R}^2} \approx -1.0316$ is attained at two points, which differ only by sign. By using polynomial modulators, a level-(3,0) relaxation returns a bound $-1.03170$ in 0.63 seconds 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 $-1.031630 \leq f^\star_{\mathbb{R}^2}$ in 0.19 seconds. Example 10 shows how the two-parameter hierarchy in Section 4.4 can be of practical importance.

Our next two examples are box-constrained problems from the work of Ray and Nataraj [17]; 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 (17) which includes those constraints in the Lagrangian.

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

Letting $X = \{\boldsymbol{x} \in \mathbb{R}^4 : -0.5 \leq x_i \leq 0.5\}$, one can compute $(f,g)^{(1,2,0)}_X = (f,g)^\star_{\mathbb{R}^4} = -3.1176903$, 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.

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

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 seconds. 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 to describe a set "$X$" for use in a conditional SAGE relaxation:

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

The resulting level (0,3,0) relaxation can be solved in 0.64 seconds. We recover a feasible solution with Algorithm 2, and obtain a solution matching 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 a problem statement, even when those constraints are weaker than those found in the problem.

| source | name | $n$ | $d$ | minimum | SAGE solved |
|---|---|---|---|---|---|
| [34] | 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 |
| [17] | 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 |

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 [30] show SAGE is tight for any number of variables. The Beale, Colville, and Goldstein-Price polynomials proved very difficult for optimization via SAGE certificates.

Now we turn to problems from [28], featuring nonconvex inequality constraints. One of these problems was introduced in Section 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 produces tight bounds for all of these problems; results are summarized in Table 7.

There are a few subtle distinctions between geometric-form signomial programs (SPs), and nonnegative polynomial optimization problems (POPs) such as those 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 typically 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

[28] took over 1600 and 200 seconds to solve P6_8 with BSOS and SOS respectively, while SAGE can solve the same problem in under 4 seconds 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 hierarchy (such as $(1,1,0)$, or $(0,q,0)$ with small $q$).

| name | $k$ | minimum | $(p,q,\ell)$ | **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 7: Generic polynomial optimization problems, over the nonnegative orthant. Problems can be found in both [28] and [31]; names "P$n_d$" 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 (in seconds).

| | Algorithm 2 | | Algorithm 2L | |
|---|---|---|---|---|
| name | $f(\boldsymbol{x})$ | $\min g(\boldsymbol{x})$ | $f(\boldsymbol{x})$ | $\min g(\boldsymbol{x})$ |
| 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 |

Table 8: Comparison of Algorithm 2 and Algorithm 2L for solution recovery for eight nonconvex polynomial optimization problems in the literature (ref. [28, 31]). 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.

### 5.3 Minimizing random sparse quartics over the sphere

In this section we describe how SAGE relaxations perform for minimizing sparse quartic forms over the unit sphere; this particular 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 [27].

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 $g(\boldsymbol{x}) = 1 - \boldsymbol{x}^\intercal \boldsymbol{x}$.[5] 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.

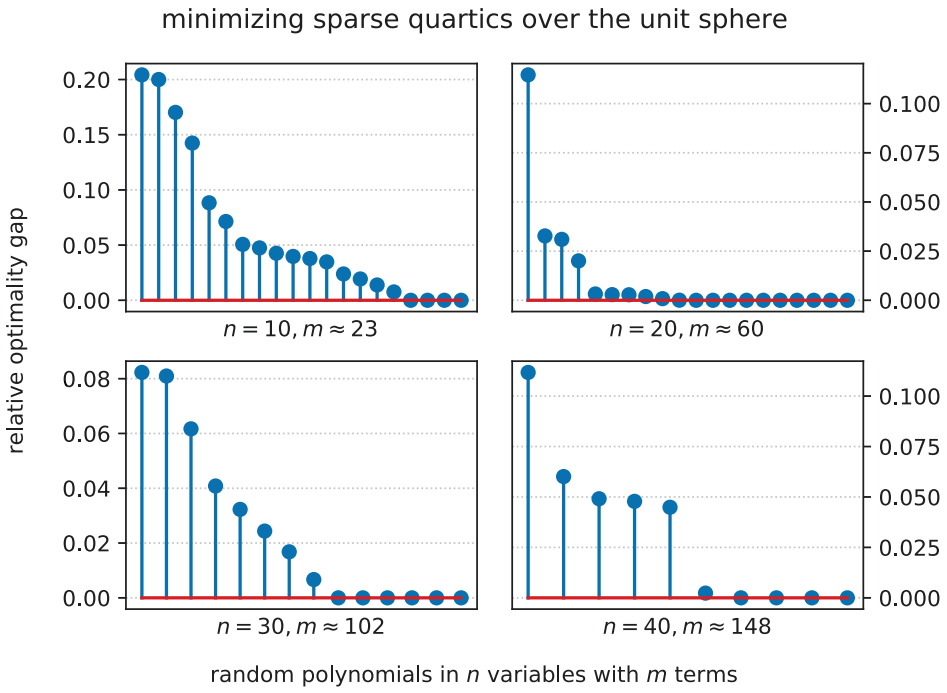


Figure 1: Upper-bounds on the optimality gap $|(f, g)^{(0,2,0)}_X - (f, g)^\star_{\mathbb{R}^n}| / |(f, g)^\star_{\mathbb{R}^n}|$. The value $(f, g)^\star_{\mathbb{R}^n}$ in these calculations was replaced by the objective value of a solution produced by Algorithm 2L. SAGE solved 4 problems in 10 variables,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

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

| solve time (s) | $n = 10$ | $n = 20$ | $n = 30$ | $n = 40$ |
|---|---|---|---|---|
| mean | 7.54E-01 | 6.50E-00 | 6.46E+01 | 4.59E+02 |
| std dev. | 8.74E-02 | 8.54E-01 | 1.38E+01 | 7.20E+01 |

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 a portion of the problem fits in RAM.

# 6 Discussion and Conclusion

In this article we introduced and developed notions of conditional SAGE certificates for both signomials and polynomials. In the signomial case, the underlying theory of the $X$-SAGE cones has deep roots in convex duality, while in the polynomial case, we derived efficient representations of important $X$-SAGE cones by employing "signomial representatives" introduced by the authors in earlier work. 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.

From an applications perspective, it would be interesting to see how conditional SAGE certificates can help with branch-and-bound algorithms. Whether considering signomials in geometric form, signomials in exponential form, or polynomials, bound constraints can easily be incorporated into $X$-SAGE cones. On the algorithmic front, important work remains to be done on solvers for relative entropy programs, primarily with regards to problems where the optimal solution contains a large number of variables along certain extreme rays of the exponential cone.

Conditional SAGE certificates raise many questions of potential interest to researchers in real algebraic geometry, and optimization-via-nonnegativity-certificates. Consider for example how the minimax-free hierarchy from Section 3.4 adopted a particular form for the modulating function: $\mathrm{Sig}(\boldsymbol{\alpha}, \mathbf{1})^\ell$. What benefit might there be to instead using a modulator $\mathrm{Sig}(\hat{\boldsymbol{\alpha}}, \mathbf{1})^\ell$, 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 $X$? And at the most fundamental level, one asks – for what exponents $\boldsymbol{\alpha}$ and what sets $X$ do $X$-SAGE cones coincide with $X$-nonnegativity cones? Prior work (c.f. [30], and more recently [33]) 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 $X$.

## References


[1] James Yan. "Signomial programs with equality constraints: numerical solution and applications". PhD thesis. University of British Columbia, 1976.

[2] M. J. Rijckaert and X. M. Martens. "Comparison of generalized geometric programming algorithms". In: *Journal of Optimization Theory and Applications* 26.2 (Oct. 1978), pp. 205–242. ISSN: 1573-2878.

[3] D. H. Rountree and A. K. Rigler. "A penalty treatment of equality constraints in generalized geometric programming". In: *Journal of Optimization Theory and Applications* 38.2 (Oct. 1982), pp. 169–178. ISSN: 1573-2878.

[4] Katta G. Murty and Santosh N. Kabadi. "Some NP-complete problems in quadratic and nonlinear programming". In: *Mathematical Programming* 39.2 (June 1987), pp. 117–129. ISSN: 1436-4646.

[5] N. Z. Shor. "Class of global minimum bounds of polynomial functions". In: *Cybernetics* 23.6 (Nov. 1987), pp. 731–734.

[6] M. J. D. Powell. "A Direct Search Optimization Method That Models the Objective and Constraint Functions by Linear Interpolation". In: *Advances in Optimization and Numerical Analysis*. Dordrecht: Springer Netherlands, 1994, pp. 51–67. ISBN: 978-94-015-8330-5.

[7] Jan Verschelde. "Algorithm 795: PHCpack: A General-purpose Solver for Polynomial Systems by Homotopy Continuation". In: *ACM Trans. Math. Softw.* 25.2 (June 1999), pp. 251–276. ISSN: 0098-3500.

[8] Pablo Parrilo. "Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization". PhD thesis. California Institute of Technology, May 2000.

[9] Victoria Powers and Bruce Reznick. "Polynomials that are positive on an interval". In: *Transactions of the American Mathematical Society* 352.10 (Oct. 2000), pp. 4677–4692.

[10] Jean B. Lasserre. "Global Optimization with Polynomials and the Problem of Moments". In: *SIAM Journal on Optimization* 11.3 (Jan. 2001), pp. 796–817.

[11] Peiping Shen and Kecun Zhang. "Global optimization of signomial geometric programming using linear relaxation". In: *Applied Mathematics and Computation* 150.1 (Feb. 2004), pp. 99–114.

[12] Yanjun Wang and Zhian Liang. "A Deterministic Global Optimization Algorithm for Generalized Geometric Programming". In: *Applied Mathematics and Computation* 168.1 (Sept. 2005), pp. 722–737. ISSN: 0096-3003.

[13] Jonathan Borwein and Adrian Lewis. *Convex Analysis and Nonlinear Optimization*. Springer New York, 2006.

[14] Peiping Shen and Hongwei Jiao. "A new rectangle branch-and-pruning approach for generalized geometric programming". In: *Applied Mathematics and Computation* 183.2 (Dec. 2006), pp. 1027–1038.

[15] R.A. Jabr. "Inductor design using signomial programming". In: *COMPEL - The international journal for computation and mathematics in electrical and electronic engineering* 26.2 (2007), pp. 461–475.

[16] Shao-Jian Qu, Ke-Cun Zhang, and Ying Ji. "A new global optimization algorithm for signomial geometric programming via Lagrangian relaxation". In: *Applied Mathematics and Computation* 184.2 (Jan. 2007), pp. 886–894.

[17] Shashwati Ray and P. S. V. Nataraj. "An efficient algorithm for range computation of polynomials using the Bernstein form". In: *Journal of Global Optimization* 45.3 (Dec. 2008), pp. 403–426.

[18] Peiping Shen, Yuan Ma, and Yongqiang Chen. "A robust algorithm for generalized geometric programming". In: *Journal of Global Optimization* 41.4 (Aug. 2008), pp. 593–612. ISSN: 1573-2916.

[19] Gregory V. Bard. "Some Basic Facts about Linear Algebra over GF(2)". In: *Algebraic Cryptanalysis*. Springer US, 2009, pp. 81–88.

[20] Mung Chiang. "Nonconvex Optimization for Communication Networks". In: *Advances in Applied Mathematics and Global Optimization: In Honor of Gilbert Strang*. Boston, MA: Springer US, 2009, pp. 137–196. ISBN: 978-0-387-75714-8.

[21] Didier Henrion, Jean-Bernard Lasserre, and Johan Löfberg. "GloptiPoly 3: moments, optimization and semidefinite programming". In: *Optimization Methods and Software* 24.4-5 (2009), pp. 761–779.

[22] A. Domahidi, E. Chu, and S. Boyd. "ECOS: An SOCP solver for embedded systems". In: *European Control Conference (ECC)*. 2013, pp. 3071–3076.

[23] Xueping Hou, Peiping Shen, and Yongqiang Chen. "A Global Optimization Algorithm for Signomial Geometric Programming Problem". In: *Abstract and Applied Analysis* 2014 (2014), pp. 1–12.

[24] Gongxian Xu. "Global optimization of signomial geometric programming problems". In: *Euro. Journal of Operational Research* 233.3 (2014), pp. 500–510.

[25] Santiago Akle Serrano. "Algorithms for unsymmetric cone optimization and an implementation for problems with the exponential cone". PhD thesis. Palo Alto, CA: Stanford University, 2015.

[26] Venkat Chandrasekaran and Parikshit Shah. "Relative Entropy Relaxations for Signomial Optimization". In: *SIAM Journal on Optimization* 26.2 (2016), pp. 1147–1173.

[27] Amir Ali Ahmadi and Anirudha Majumdar. *DSOS and SDSOS Optimization: More Tractable Alternatives to Sum of Squares and Semidefinite Optimization*. 2017. eprint: `arXiv:1706.02586`.

[28] Jean B. Lasserre, Kim-Chuan Toh, and Shouguang Yang. "A bounded degree SOS hierarchy for polynomial optimization". In: *EURO Journal on Computational Optimization* 5.1 (Mar. 2017), pp. 87–117. ISSN: 2192-4414.

[29] Max M. J. Opgenoord, Brian S Cohen, and Warren W. Hoburg. *Comparison of Algorithms for Including Equality Constraints in Signomial Programming*. Tech. rep. ACDL TR-2017-1. MIT, Aug. 2017.

[30] Riley Murray, Venkat Chandrasekaran, and Adam Wierman. *Newton Polytopes and Relative Entropy Optimization*. 2018. eprint: `arXiv:1810.01614`.

[31] Tillmann Weisser, Jean B. Lasserre, and Kim-Chuan Toh. "Sparse-BSOS: a bounded degree SOS hierarchy for large scale polynomial optimization with sparsity". In: *Mathematical Programming Computation* 10.1 (Mar. 2018), pp. 1–32. ISSN: 1867-2957.

[32] MOSEK ApS. *MOSEK 9.0.70(beta)*. 2019.

[33] Jens Forsgård and Timo de Wolff. *The algebraic boundary of the sonc cone*. 2019. eprint: `arXiv:1905.04776`.

[34] S. Surjanovic and D. Bingham. *Virtual Library of Simulation Experiments: Test Functions and Datasets*. Retrieved April 18, 2019, from `http://www.sfu.ca/~ssurjano`.

# 7 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 $\epsilon_0 > 0$.

**procedure** VARIABLEMAGNITUDES($\boldsymbol{\alpha}, \boldsymbol{v}, \hat{\boldsymbol{v}}, \epsilon_0$)
  $M \leftarrow []$
  **for** $j = 1, \ldots, m$ **do**
    **if** $\hat{v}_j = 0$ **then**
      **Continue**
    **end if**
    Recover $\boldsymbol{z}$ in $\mathbb{R}^n$ s.t. $\hat{v}_j \log(\hat{\boldsymbol{v}}/\hat{v}_j) \geq [\boldsymbol{\alpha} - \mathbf{1}\boldsymbol{\alpha}_j]\boldsymbol{z}$ and $(\boldsymbol{z}, \hat{v}_j) \in \operatorname{co} Y$.
    $\boldsymbol{y} \leftarrow \boldsymbol{z}/\hat{v}_j$
    $M$.append($\exp \boldsymbol{y}$)
  **end for**
  **if** $(\boldsymbol{x}^{\boldsymbol{\alpha}_1}, \ldots, \boldsymbol{x}^{\boldsymbol{\alpha}_m}) \neq |\boldsymbol{v}|$ for all $\boldsymbol{x}$ in $M$ **then**
    Compute $(\boldsymbol{y}, t)$ solving Problem 15, for given $\epsilon_0$.
    $M$.append($\exp \boldsymbol{y}$)
  **end if**
  **return** $M$.
**end procedure**

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

Let us describe the ways in which Algorithm 4 differs from the discussion in Section 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 $\alpha_{ij}$ 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 $\boldsymbol{s}$.