Computation of sum of squares polynomials from data points

Bruno Després, Maxime Herda

COMPUTATION OF SUM OF SQUARES POLYNOMIALS FROM DATA POINTS

BRUNO DESPRÉS* AND MAXIME HERDA†

Abstract. We propose an iterative algorithm for the numerical computation of sums of squares of polynomials approximating given data at prescribed interpolation points. The method is based on the definition of a convex functional G arising from the dualization of a quadratic regression over the Cholesky factors of the sum of squares decomposition. In order to justify the construction, the domain of G, the boundary of the domain and the behavior at infinity are analyzed in details. When the data interpolate a positive univariate polynomial, we show that in the context of the Lukacs sum of squares representation, G is coercive and strictly convex which yields a unique critical point and a corresponding decomposition in sum of squares. For multivariate polynomials which admit a decomposition in sum of squares and up to a small perturbation of size ε, G^ε is always coercive and so it minimum yields an approximate decomposition in sum of squares. Various unconstrained descent algorithms are proposed to minimize G. Numerical examples are provided, for univariate and bivariate polynomials.

Key words. Positive polynomials, sum of squares, convex analysis, positive interpolation, iterative methods.

AMS subject classifications. 90C30, 65K05, 90C25

1. Introduction.

The numerical and algorithmic motivation of the present paper comes from a recent work [5] where an iterative algorithm for positive interpolation (meaning that a sign condition on a given closed interval $\mathbb{I}$ must be respected) was proposed for of univariate polynomials. A practical scenario which illustrates the interest of iterative positive interpolation is the following. Take a polynomial without knowing its sign on $\mathbb{I}$. If the iterative method converges and recover p at the limit (it can be checked at a finite number of points), then p is non negative on $\mathbb{I}$ (that is for an infinite number of points). In this case the algorithm provides an iterative *certificate of positivity* [14, 15]. But if the iterations do not recover p at the limit (or if one stops the algorithm after a finite number of iterations), then p is (or might be) non positive on $\mathbb{I}$. In this case of non convergence, the iterations provide nevertheless a non negative surrogate to p. We refer to the quoted work for an illustration of the interest of non negative polynomial surrogates in the context of Scientific Computing (SC). However two important restrictions in the previous algorithm [5] are that the polynomials are univariate and the interpolation points, where the data of the polynomials are given, are sliding points (it allowed for strong convergence properties). It brings severe constraints for applications in SC. In the present work, we relax these restrictions by constructing a new iterative algorithm for positive interpolation. The algorithm aims at computing a sum of squares (SOS) decomposition from the sole knowledge of prescribed interpolation data at prescribed interpolation points. Also the method is much more general so it is formulated for multivariate polynomials as well and does not need tensorization, something that was impossible with the previous method.

A modern reference in SC for control of the sign of polynomials at a *finite number of prescribed interpolation points* is in the works of C.-W. Shu [27], with application to the discretization of hyperbolic equations with high order methods. The point of view developed in this article is to control the sign of polynomials on *all points in a given compact (semi-algebraic) set* $\mathbb{K} \subset \mathbb{R}^d$ which is much more demanding. Preliminary tests for the construction of such algorithms are in [6], but the methods were inefficient in terms of the time of restitution. In a fully different direction, one must mention the theory of numerical approximation with splines, see [16, 2]: splines are widely used in scientific computing and computer aided design (CAD) but often needs tensorization in multi-dimension; this limitation is not encountered by our new methods because they can be implemented on any semi-algebraic set $\mathbb{K}$ in any

*Sorbonne-Université, CNRS, Université de Paris, Laboratoire Jacques-Louis Lions (LJLL), F-75005 Paris, France, and Institut Universitaire de France

†Inria, Univ. Lille, CNRS, UMR 8524 Laboratoire Paul Painlevé, F-59000 Lille, France

dimension.

In the community of numerical optimization [14] from which we borrow most of our notations, SOS algorithms based on SemiDefinite Programming (SDP) are extensively used. It had been noticed by Powers and Wörmann [24] that finding an SOS decomposition is equivalent to SDP, that is optimization in the cone of non-negative quadratic forms. Then algorithms based on interior-point methods were developed to solve these problems [21, 20, 28]. However, these methods seem to be hardly directly applicable in SC because they are based more on algebraic properties and not on interpolation data which are of major importance in numerical analysis and SC. This leads us to the development of the algorithm of the present paper, which is not based on SDP but rather on the iterative resolution of a non-convex quadratic problem over Cholesky factors of the SOS decomposition. We solve the quadratic program using a dualization of the problem, which leads us to a nonlinear convex program. Let us mention that a similar reformulation of general SDP was proposed by Burer and Monteiro [4]. In our case however, we use the particular structure of the interpolation data of the SOS to obtain some useful coercivity properties on the dual function. Also, similar dualization ideas can be found in [18, 10], but unlike here they are formulated on the Gram matrix rather than on the Cholesky factors. Our construction will generate a functional with strong convexity properties for which standard descent algorithms are efficient, as shown in the numerical section.

Let $\mathrm{P}[\mathbf{X}] := \mathrm{P}[X_1, \dots X_d]$ be the set of real polynomials with d variables. The subset of polynomials of total degree less than or equal to $n \geqslant 1$ is denoted by $\mathrm{P}^n[\mathbf{X}]$, with $r_* = \dim \mathrm{P}^n[\mathbf{X}]$. Let $\mathbb{K} \subset \mathbb{R}$ be a closed semi-algebraic set defined through a finite number j_* of polynomial inequalities

$$\mathbb{K} = \left\{ \mathbf{x} \in \mathbb{R}^d \text{ such that } g_j(\mathbf{x}) \geqslant 0 \text{ for } g_j \in \mathrm{P}[\mathbf{X}], 1 \leqslant j \leqslant j_* \right\}. \tag{1.1}$$

Most standard cells (intervals in 1D, squares and triangles in 2D, ...) in SC can be implemented as semi-algebraic sets, so it is not a restriction for further applications. The convex set of non-negative polynomials of maximal degree n on $\mathbb{K}$ is

$$\mathrm{P}^n_{\mathbb{K},+}[\mathbf{X}] = \{ p \in \mathrm{P}^n[\mathbf{X}] \text{ such that } p(\mathbf{x}) \geqslant 0 \text{ for any } \mathbf{x} \in K \}. \tag{1.2}$$

Famous examples of characterizations as SOS are the Lukacs theorem [29] or Putinar's Positvstellensatz [25]: a recent state of the art can be found in the books of Lasserre [14, 15]; some recent algorithmic issues in the context of optimal control can be found in [12] and therein. In order to be constructive, we focus in this work on the following version

$$p = \sum_{j=1}^{j_*} g_j \left(\sum_{i=1}^{i_*} q_{ij}^2 \right) = \sum_{i=1}^{i_*} \left(\sum_{j=1}^{j_*} g_j q_{ij}^2 \right) = \sum_{j=1}^{j_*} \sum_{i=1}^{i_*} g_j q_{ij}^2, \tag{1.3}$$

where the maximal number of squares is equal to a predefined value $i_* \geqslant 1$ independent of j. In this work, the number of squares i_* and the degree of the polynomials q_{ij} are prescribed in function of n, see below (1.4) for the prescription on i_* and (1.5) for the prescription the degree of the polynomials q_{ij}. It can be compared with the Schmügden's or Putinar's Positvstellensatz where the degree of the polynomials q_{ij} can be exponentially large [22, 14]. With our notations, it is sufficient to embed p in a set of polynomials of larger degree, that is to say to take $n \gg \deg(p)$, to recover this case.

Next, the notion of unisolvence which comes from the Finite Element Method (FEM) is convenient to formalize properties of interpolation points. A unisolvent set of points $(\mathbf{x}_r)_{1 \leqslant r \leqslant r_*}$ is such that any polynomial $p \in \mathrm{P}^n[\mathbf{X}]$ is uniquely determined by its values $y_r = p(\mathbf{x}_r)$ for $1 \leqslant r \leqslant r_*$. The number i_* of polynomials in the SOS (1.3) is *a priori* independent from the number of interpolation points. However in our context the function G below is more naturally constructed assuming that

$$i_* = r_*. \tag{1.4}$$

That is why we will assume (1.4) throughout this work, except at early stages of the construction. With these notations, one formulates the notion of positive interpolation: it is a recent adaptation [5] to SC of the notion of a *certificate of positivity* for which the reader can find information in [14, 15]. A practical way to understand the model problem below is the following: from the knowledge of the values of p at only a *finite number of given interpolation points*, get a control of the sign of p at *infinite number of points* (the whole set $\mathbb{K}$).

PROBLEM 1.1 (Iterative positive interpolation on $\mathbb{K}$). *Let $p \in \mathrm{P}^n_{\mathbb{K},+}[\mathbf{X}]$. Take a unisolvent set $(\mathbf{x}_r)_{1\leqslant r\leqslant r_*}$, and consider the interpolated values $y_r = p(\mathbf{x}_r)$. From $(\mathbf{x}_r, y_r)_{1\leqslant r\leqslant r_*}$, compute iteratively polynomials $(q_{ij})_{ij}$ such that the SOS representation (1.3) holds at the limit.*

The methods and results studied in this work can be summarized as follows. Consider the parametrization

$$q_{ij} \in \mathrm{P}^{n_j}[\mathbf{X}] \text{ with } n_j = \lfloor (n - \deg(g_j))/2 \rfloor \tag{1.5}$$

where $\lfloor\cdot\rfloor$ denotes the integer part of a real number. Consider the canonical basis made of monomials (but other basis can be taken as well, see Remark 2.1), with the standard multi-index notation $\alpha = (\alpha_1, \ldots, \alpha_d) \in \mathbb{N}^d$, $|\alpha| = \alpha_1 + \cdots + \alpha_d$ and $\mathbf{X}^\alpha = X_1^{\alpha_1} \ldots X_d^{\alpha_d}$. The polynomials q_{ij} write $q_{ij}(\mathbf{X}) = \sum_{|\alpha|\leqslant n_j} c_\alpha^{ij} \mathbf{X}^\alpha$ and we store the coefficients in a vector of coefficients $c^{ij} = (c_\alpha^{ij})_\alpha \in \mathbb{R}^{r_j}$ where $r_j = \dim(\mathrm{P}^{n_j}[\mathbf{X}]) = \binom{d+n_j}{d}$. Gather the coefficients c^{i1}, c^{i2}, ..., c^{ij_*} in a single column vector (called a Cholesky factor) $\mathbf{U}_i = \left(c^{i1}, c^{i2}, \ldots, c^{ij_*}\right)^t \in \mathbb{R}^{r_*}$ where $r_* = \sum_{j=1}^{j_*} r_j$. Define the Hankel matrices $D^{n_j}_{\alpha,\beta}(\mathbf{X}) = \mathbf{X}^\alpha \mathbf{X}^\beta$ for $|\alpha|, |\beta| \leqslant n_j$. Define the polynomial valued block matrix $B(\mathbf{X}) = B(\mathbf{X})^t \in \mathbb{R}^{r_* \times r_*}$

$$B(\mathbf{X}) = \operatorname{diag}\left(g_1(\mathbf{X}) D^{n_1}(\mathbf{X}), , \ldots, g_{j_*}(\mathbf{X}) D^{n_{j_*}}(\mathbf{X})\right). \tag{1.6}$$

This matrix is a block diagonal localizing matrix [14]. The first diagonal block is square $r_1 \times r_1$, ... until the last block which is square $r_{j_*} \times r_{j_*}$: all other terms are zero. By construction, one has the identity

$$\sum_{j=1}^{j_*} g_j(\mathbf{X}) \sum_{i=1}^{i_*} q_{ij}^2(\mathbf{X}) = \sum_{i=1}^{i_*} \left(\sum_{j=1}^{j_*} g_j(\mathbf{X}) q_{ij}^2(\mathbf{X}) \right) = \sum_{i=1}^{i_*} \langle B(\mathbf{X}) \mathbf{U}_i, \mathbf{U}_i \rangle . \tag{1.7}$$

Denote the evaluation of $B(\mathbf{X})$ at interpolation points as $B_r = B(\mathbf{x}_r) \in \mathbb{R}^{r_* \times r_*}$. Define the function $G : \mathbb{R}^{r_*} \to \overline{\mathbb{R}} = \mathbb{R} \cup \{+\infty\}$ with domain $\mathcal{D} = \{\lambda \in \mathbb{R}^{r_*}$ such that $I + \sum_{r=1}^{r_*} \lambda_r B_r \succ 0\}$ as follows. For $\lambda \in \mathcal{D}$ then

$$G(\lambda) = \operatorname{tr}\left[\left(I + \sum_{r=1}^{r_*} \lambda_r B_r \right)^{-1} \right] + \sum_{r=1}^{r_*} y_r \lambda_r, \tag{1.8}$$

otherwise $G(\lambda) = +\infty$. In the previous formula, $\operatorname{tr}(\cdot)$ denotes the trace. Our main results are the following.

THEOREM 1.2. *The function G has the following properties:*

1. *It is a proper closed convex function. It is C^∞ on its non-empty open convex domain $\mathcal{D}$, tends to infinity at $\partial\mathcal{D}$ and is infinite everywhere else by definition.*
2. *Each $\lambda \in \mathcal{D}$ defines computable polynomials $(q_{ij}[\lambda])_{1\leqslant i\leqslant r_*, 1\leqslant j\leqslant j_*}$ such that*

$$\frac{\partial G}{\partial \lambda_r}(\lambda) = y_r - \sum_{j=1}^{j_*} g_j(\mathbf{x}_r) \sum_{i=1}^{r_*} q_{ij}^2[\lambda](\mathbf{x}_r), \quad 1 \leqslant r \leqslant r_*. \tag{1.9}$$

If $\lambda_* \in \mathcal{D}$ is a critical point of G, that is $\nabla G(\lambda_*) = 0$, then the family $(q_{ij}[\lambda_*])_{ij}$ is solution to (1.3), that is a SOS.

THEOREM 1.3 (Existence of critical points in $\mathcal{D}$). *It is proved in two cases.*

1. *Take $d > 1$, $\mathbb{K}$ a semi-algebraic set and $p \in \mathrm{P}^n_{\mathbb{K},+}[\mathbf{X}]$. Assume that a technical condition on the linear independence of the matrices B_r is satisfied. Then, up to an infinitesimally small perturbation (the perturbed polynomial p^ε has the interpolation data $(y^\varepsilon_r)_{1\leqslant r\leqslant r_*}$), the function G^ε is strictly convex, coercive and admits a unique critical point in $\mathcal{D}$.*
2. *Take $d = 1$, $\mathbb{K}$ a segment and $p > 0$ on $\mathbb{K}$. Then the technical condition the linear independence of the matrices B_r is satisfied. Moreover G is strictly convex, coercive and admits a unique critical point in $\mathcal{D}$.*

COROLLARY 1.4 (Solution to Problem 1.1). *Under the hypothesis of Theorem 1.3, the minimum of G (or G^ε) in $\mathcal{D}$ yields a SOS decomposition of p (or p^ε). It can be computed by standard descent algorithms.*

As stated in the introduction, a practical scenario which in our mind has interest for SC is the following. Take a polynomial without knowing its sign on $\mathbb{K}$. If the descent method converges and recover p at the limit, then p is non negative on $\mathbb{K}$. If the descent does not recover p at the limit, then for monovariate polynomials, p is non positive on $\mathbb{K}$. It shows that the descent method provides an iterative *certificate of positivity*. In case of non convergence, the iterations provide nevertheless a non negative surrogate to p. We refer to [5] for an illustration of the interest of non negative polynomial surrogates.

The outline of this paper is as follows. In Section 2, we propose a dual interpretation of Problem 1.1. This leads us to the introduction of the function G and its domain. Then, in Section 3, we discuss necessary and sufficient conditions characterizing asymptotic properties and strict convexity of G. In Section 4, we show that for univariate positive polynomials on a segment, the former conditions are satisfied yielding strict convexity and coercivity of the associated function G. Besides, we provide a more precise description the structure of the domain. In Section 5, we present the specific descent and Newton type methods we use to compute the critical points of G. In Section 6 we provide numerical illustrations of the efficiency of our new approach for computing SOS decomposition of polynomials in one variable on segment and two variables on triangle. Finally, we provide in Appendix A some additional theoretical results concerning the links between the asymptotic cone of the set $\mathcal{D}$ and the Lagrange polynomials in the case of univariate polynomials.

Acknowledgements. Both authors are greatly indebted to Jean-Bernard Lasserre and Didier Henrion for their kind explanations on the theory and state of the art of semidefinite programming and sum of squares and would like to thank them for their invitation at LAAS and for their hospitality. The authors would also like to thank the anonymous referees for their suggestions and comments which helped to improve the quality of this paper.

2. Construction of G (Proof of Theorem 1.2). The construction of G, leading to (1.8), is done by recasting the model problem 1.1 as the convex dual of a Quadratically Constrained Quadratic Program (QCQP) (see[3]). In order to have a more general discussion, we relax the condition (1.4) in this part and in the next Section 2.1. It means that

$$i_* \neq r_*$$

is possible as well. The condition (1.4) is reintroduced end of Section 2.1. We begin with some remarks on the objects introduced in the first section.

Remark 2.1. In the numerical experiments of Section 6, we use other polynomials than the monomials in order to optimize the robustness and accuracy of the algorithms. It only changes the definition of the matrix $B(\mathbf{X})$ in (1.6) and thus of $B_r = B(\mathbf{x}_r)$ but every result of this paper still hold. More generally, one could even generalize Problem 1.1 and replace the constraint of equality at interpolated

values by constraints of the type $y_r = L_r(p)$ where the family $\{L_r : P^n[\mathbf{X}] \to \mathbb{R},\ r = 1, \dots, r_*\}$ is any basis of the dual space of $P^n[\mathbf{X}]$. In the context of SC more precisely for the numerical resolution of partial differential equations, one deals with data points in finite difference discretizations. However, if one is considering a finite volume discretization, one would rather work with mean values on some mesh cells. This variant is easily manageable with our method by choosing the adequate linear forms L_r and modifying the matrices B_r accordingly.

Remark 2.2. An interesting consequence of the Caratheodory Theorem ([11, Theorem III.1.3.6 page 98]) is that if a formula like (1.3) holds for $i_* > r_*$, then a similar one holds also for $i_* = r_*$ (but for different polynomials q_{ij}). Indeed the set $\mathcal{W} = \sum_{j=1}^{j_*} g_j(\mathbf{X})\mathrm{P}^{n_j}[\mathbf{X}]^2$ is a closed convex cone embedded in $p \in \mathrm{P}^n[\mathbf{X}]$. Therefore any convex combination of $i_* > r_*$ elements of $\mathcal{W}$ can be expressed as a convex combination of only $r_* = \dim \mathrm{P}^n[\mathbf{X}]$ elements of $\mathcal{W}$ (the coefficients of the convex combination can be set to 1 after proper rescaling of the new q_{ij}).

2.1. Lagrangian duality (Theorem 1.2 item 1). In any dimension, the notation $\langle \cdot, \cdot \rangle$ will denote the Euclidean dot product and $\| \cdot \|$ will denote the associated norm. We define the algebraic manifold

$$\mathcal{U} = \{\mathbf{U} = (\mathbf{U}_1, \dots, \mathbf{U}_{i_*}) \in (\mathbb{R}^{r*})^{i*} \text{ such that } \sum_{i=1}^{i_*} \langle B_r \mathbf{U}_i, \mathbf{U}_i \rangle = y_r \text{ for all } 1 \leqslant r \leqslant r_*\}. \tag{2.1}$$

The vectors $\mathbf{U}_i$ are called the Cholesky factors [20, 28] of the decomposition. With the unisolvence assumption, finding a SOS (1.3) amounts to finding one element $\mathbf{U} \in \mathcal{U}$. In order to find a $\mathbf{U} \in \mathcal{U}$ in a constructive manner, our strategy is to start at a given $\mathbf{V}$ (probably outside $\mathcal{U}$) and to project on $\mathcal{U}$ in the quadratic norm. It writes as follows.

Problem 2.3. *Take* $\mathbf{V} = (\mathbf{V}_1, \dots, \mathbf{V}_{i_*}) \in (\mathbb{R}^{r*})^{i*}$. *Calculate* $\mathbf{U} = \underset{\mathbf{U} \in \mathcal{U}}{\mathit{argmin}}\ \frac{1}{2} \sum_{i=1}^{i_*} \|\mathbf{U}_i - \mathbf{V}_i\|^2$.

The vectors $\mathbf{V} = (\mathbf{V}_i)_i$ may be thought of as a good initial guesses for the $\mathbf{U} = (\mathbf{U}_i)_i$. The optimal value of the cost does not matter. However Problem 2.3 seems even harder to solve than the original problem we were concerned with. The finding is that the Lagrangian dual problem is endowed with good properties provided $\mathbf{V}$ is conveniently chosen. In this case, the new Problem 2.3 provides a way to determine an admissible $\mathbf{U} \in \mathcal{U}$.

Still for any $\mathbf{V}$, introduce the Lagrangian which is the sum of the functional and of the dualization of the constraint (2.1) with a Lagrange multiplier $\lambda \in \mathbb{R}^{r*}$

$$\mathcal{L}(\mathbf{U}, \lambda) = \frac{1}{2} \sum_{i=1}^{i_*} \left(\|\mathbf{U}_i - \mathbf{V}_i\|^2 + \sum_{r=1}^{r_*} \lambda_r \langle B_r \mathbf{U}_i, \mathbf{U}_i \rangle \right) - \frac{1}{2} \langle \lambda, \mathbf{y} \rangle \qquad \text{where } \mathbf{y} = (y_r)_{1 \leqslant r \leqslant r_*}.$$

The first-order optimality constraints are $\nabla_{\mathbf{U}} \mathcal{L} = 0$ and $\nabla_{\lambda} \mathcal{L} = 0$. The first-order optimality constraint $\nabla_{\mathbf{U}} \mathcal{L} = 0$ is linear with respect to $\mathbf{U}$. Define the symmetric matrix $M(\lambda) = M(\lambda)^t \in \mathbb{R}^{r_* \times r_*}$

$$M(\lambda) = I + \sum_{r=1}^{r_*} \lambda_r B_r \tag{2.2}$$

where I is the identity matrix in $\mathbb{R}^{r_* \times r_*}$. The condition $\nabla_{\mathbf{U}} \mathcal{L} = 0$ writes

$$M(\lambda)\mathbf{U}_i = \mathbf{V}_i \text{ for } 1 \leqslant i \leqslant i_* \iff M(\lambda)\mathbf{U} = \mathbf{V}. \tag{2.3}$$

If the multiplier $\lambda \in \mathbb{R}^{r*}$ is such that the matrix $M(\lambda)$ is invertible, then the candidate solution $\mathbf{U}$ can be computed explicitly in terms of λ and $\mathbf{V}$ as the solution of the linear system (2.3).

It is therefore natural to concentrate on a condition on λ such that $M(\lambda)$ is invertible. In order to obtain convexity properties in the following we even restrict λ to the set of positive definiteness of $M(\lambda)$. To our knowledge, this at this stage that our analysis differs from the standard exposition of dual QCQP [3, 11] and from other dualizations in the context of SOS [4, 18, 10].

DEFINITION 2.4. *The domain of positive definiteness of M is $\mathcal{D} = \{\lambda \in \mathbb{R}^{r_*} \mid M(\lambda) \succ 0\} \subset \mathbb{R}^{r_*}$. It is an open set and it is non empty since $0 \in \mathcal{D}$.*

For a Lagrange multiplier $\lambda \in \mathcal{D}$, the inverse transformation of (2.3) is $\mathbf{U}(\lambda) = M(\lambda)^{-1}\mathbf{V}$. Then, one can evaluate the Lagrangian at $\mathbf{U}(\lambda)$. An elementary computation yields $\mathcal{L}(\mathbf{U}(\lambda), \lambda) = \frac{1}{2}\sum_{i=1}^{i_*} \left(\|\mathbf{V}_i\|^2 - \left\langle \mathbf{V}_i, M(\lambda)^{-1}\mathbf{V}_i \right\rangle \right) - \frac{1}{2}\langle \lambda, \mathbf{y} \rangle$. This motivates the introduction of the dual objective function $G_{\mathbf{V}} : \mathcal{D} \longrightarrow \mathbb{R}$ defined by

$$G_{\mathbf{V}}(\lambda) = \sum_{i=1}^{i_*} \left\langle \mathbf{V}_i, M(\lambda)^{-1}\mathbf{V}_i \right\rangle + \langle \lambda, \mathbf{y} \rangle \, , \tag{2.4}$$

and which one should think of as a function to be minimized.

LEMMA 2.5. *The function $G_{\mathbf{V}}$ is smooth on $\mathcal{D}$. The first and second derivatives are*

$$\frac{\partial G_{\mathbf{V}}}{\partial \lambda_r}(\lambda) = y_r - \sum_{i=1}^{i_*} \langle \mathbf{U}_i(\lambda), B_r \mathbf{U}_i(\lambda) \rangle \quad \text{and} \quad \frac{\partial^2 G_{\mathbf{V}}}{\partial \lambda_r \partial \lambda_s}(\lambda) = 2 \sum_{i=1}^{i_*} \left\langle B_r \mathbf{U}_i(\lambda), M(\lambda)^{-1} B_s \mathbf{U}_i(\lambda) \right\rangle . \tag{2.5}$$

In particular $G_{\mathbf{V}}$ is convex on $\mathcal{D}$.

Proof. The proof stems from the identity $\partial_{\lambda_r} M(\lambda)^{-1} = -M(\lambda)^{-1} B_r M(\lambda)^{-1}$ and the symmetry of the various matrices involved. Convexity follows from the positivity of $M(\lambda)$ and the expression of second derivatives yielding $\left\langle \nabla^2 G_{\mathbf{V}}(\lambda)\mu, \mu \right\rangle = 2\sum_{i=1}^{i_*} \left\langle A_i(\mu, \lambda), M(\lambda)^{-1} A_i(\mu, \lambda) \right\rangle \geqslant 0$ where $A_i(\mu, \lambda) = \sum_{r=1}^{r_*} \mu_r B_r \mathbf{U}_i(\lambda)$ for $1 \leqslant i \leqslant i_*$. ∎

In order to address the behavior of $G_{\mathbf{V}}$ near the boundary, we will make us of the following notion.

DEFINITION 2.6. *A convex function $f : \mathbb{R}^{r_*} \mapsto \mathbb{R} \cup \{+\infty\}$ is said to be closed over its domain $\mathcal{D}_f = \{\mathbf{x} \mid f(\mathbf{x}) < \infty\}$ if and only if the level sets $\{\mathbf{x} \mid f(\mathbf{x}) \leqslant t\}$ are closed for $t < +\infty$: see [11] or [3, Appendix A.3.3.].*

This property is extremely important in our approach because it yields a strong control of the objective function at finite distance.

LEMMA 2.7. *Assume the equality of dimensions (1.4))*[1]*, that is $i_* = r_*$, and that $\mathbf{V} \in \mathbb{R}^{r_* \times r_*}$ is an orthogonal matrix. Then one has the simpler expression where $\mathrm{tr}(\cdot)$ denotes the trace of a square matrix*

$$G(\lambda) := G_{\mathbf{V}}(\lambda) = \mathrm{tr}(M^{-1}(\lambda)) + \langle \lambda, \mathbf{y} \rangle . \tag{2.6}$$

Moreover the extension of $G := G_{\mathbf{V}}$ with value $+\infty$ outside of $\mathcal{D}$ is a closed convex function with open domain $\mathcal{D}$.

Proof. The formula is a direct consequence of (2.4), because the number i_* of orthogonal vectors $\mathbf{V}_i$ is equal to the dimension r_* of the space. Thanks to the continuity on $\mathcal{D}$, the closedness of $G_{\mathbf{V}}$ on $\mathbb{R}^{r_*}$ amounts to showing that for any sequence $(\mu_k)_k$ in $\mathcal{D}$ converging to a point of the boundary of the domain $\partial\mathcal{D} = \left\{ \lambda \in \overline{\mathcal{D}} \mid \det\left(I + \sum_{r=1}^{r_*} \lambda_r B_r\right) = 0 \right\}$, then one has $G_{\mathbf{V}}(\mu_k) \to +\infty$ as $k \to +\infty$. In the light of the representation formula (2.6) involving the trace of $M(\lambda)^{-1}$ it is the case since the minimal eigenvalue of $M(\mu_k)$ goes to 0 as $k \to +\infty$. Clearly the function is independent of $\mathbf{V}$. ∎

[1] As a consequence, the notation i_* will not be used anymore in the rest of the presentation, only r_*.

In order to have a better intuition of the structure of G, an illustration of its graph is given on Figure 1. Near the boundary of its domain, the function G behaves by construction like a rational barrier function [21, 3]. This barrier does not introduce any kind of approximation, and it is an exact one. This property is a strong algorithmic asset of G with respect to more standard logarithmic barrier methods. The figure provides three different illustrations of a closed convex function which is infinite outside of its domain. Actually we will show in the sections below that G is linear at infinity (in the direction of the asymptotic cone). The whole point will be to understand under which conditions G is coercive, which corresponds to the rightmost plot on Figure 1. It will prove the existence of a multiplier in $\mathcal{D}$, without explicitly requiring to use the methods of Lagrangian duality.

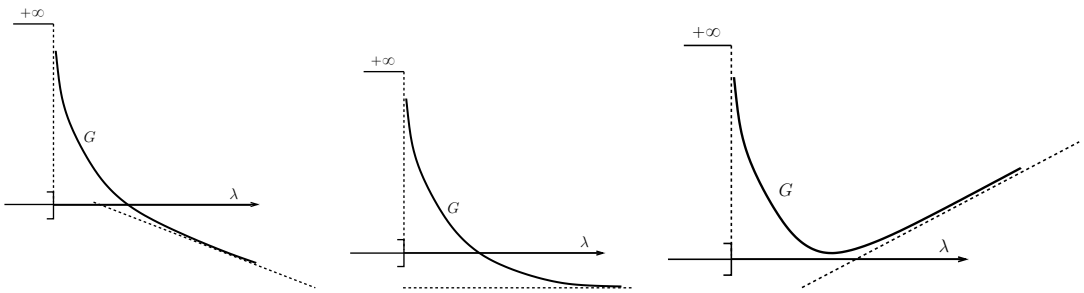


FIG. 1. *Three cases of the graph of a closed convex function f which is convex over its open domain and asymptotically linear at infinity. On the left, the function is not lower bounded and not coercive. In the center the function is lower bounded but not coercive. On the right, the function is lower bounded and coercive.*

Even if the Lagrange multiplier λ is constrained in the domain $\mathcal{D}$, the minimization of the dual function G can be done by essentially unconstrained descent algorithms thanks to its coercivity properties. We will make this clearer in Section 5. This behavior is another asset of the function G.

2.2. Critical points of G (Theorem 1.2 item 2). In this section, we formalize natural consequences of the formulas (2.5) for the derivatives of G, which are preparatory to establish that the G is naturally coercive in the domain $\mathcal{D}$. These first properties are essentially a reformulation of the previous material.

These first properties are essentially a reformulation of the previous material. For each Lagrange multiplier $\lambda \in \mathcal{D}$ one defines the vectors $(c_\alpha^{ij}[\lambda])_{\alpha,j} \in \mathbb{R}^{r_j}$ which are the components of $\mathbf{U}_i(\lambda)$, the latter being the ith column of $M(\lambda)^{-1}$. It defines the polynomials $q_{ij}[\lambda] \in \mathrm{P}^{n_j}[\mathbf{X}]$ by $q_{ij}[\lambda](\mathbf{X}) = \sum_{|\alpha|\leqslant n_j} c_\alpha^{ij}[\lambda]\mathbf{X}^\alpha$. With (1.7), these polynomials define a sum of square $p[\lambda] \in \mathrm{P}^n_{\mathbb{K},+}[\mathbf{X}]$

$$p[\lambda](\mathbf{X}) = \sum_{j=1}^{j_*} g_j(\mathbf{X}) \left(\sum_{i=1}^{r_*} q_{ij}^2[\lambda](\mathbf{X}) \right). \tag{2.7}$$

Using (1.7), $p[\lambda](\mathbf{x}_r) = \sum_{i=1}^{r_*} \langle B_r \mathbf{U}_i, \mathbf{U}_i \rangle$. So (2.5) is rewritten as

$$\frac{\partial G}{\partial \lambda_r}(\lambda) = y_r - p[\lambda](\mathbf{x}_r). \tag{2.8}$$

The Proposition below characterizes that in order to solve Problem 1.1 it is sufficient to find critical points of G. It is part of the Lagrangian duality between the primal formulation of Problem 2.3 and the dual formulations (2.4) or (2.6).

PROPOSITION 2.8. *Take $p \in \mathrm{P}^n_{\mathbb{K},+}[\mathbf{X}]$ and an unisolvent set of interpolation points $(\mathbf{x}_r)_{1\leqslant r\leqslant r_*}$ in $\mathbb{K}$. Consider $y_r = p(x_r)$ for $1 \leqslant r \leqslant r_*$. The following properties are equivalents*

- $\lambda^* \in \mathcal{D}$ *is a critical point of* G*, namely* $\nabla G(\lambda^*) = 0$.
- $\lambda^* \in \mathcal{D}$ *minimizes* G.
- $p(\mathbf{X}) = p[\lambda^*](\mathbf{X})$.

Proof. Since G is closed convex, local minima coincide exactly with critical points, so the first two points are equivalent. The equivalence between the first and third assertions follows from (2.8) and the unisolvence assumption. ∎

2.3. Number of squares. Let us precise the number of squares in the SOS formula (2.7). This information is additional with respect to Theorem 1.2. It brings the possibility to have a cheaper implementation.

LEMMA 2.9. *The number of non zero polynomials in* $\sum_{i=1}^{r*} q_{ij}^2[\lambda](\mathbf{X})$ *is less or equal to* r_j.

Proof. By construction $\big(\mathbf{U}_1(\lambda), \dots, \mathbf{U}_{r_*}(\lambda)\big) = \mathbf{U}(\lambda) = M(\lambda)^{-1}$ is a block diagonal matrix. The blocks have size $r_1 \times r_1$ until $r_{j_*} \times r_{j_*}$. So, for a given j, the polynomials $q_{ij}[\mathbf{X}]$ vanish for $1 \leqslant i \leqslant r_1 + \cdots + r_{j-1}$ and for $r_1 + \cdots + r_{j-1} + r_j + 1 \leqslant i \leqslant r_*$. ∎

Remark 2.10. The result of Lemma 2.9 is nevertheless non optimal in dimension $d = 1$. Indeed consider the Lukács Theorem (see Proposition 4.1) in the odd case $n = 2k+1$ and take $g_1(\mathbf{X}) = \mathbf{X}$ and $g_2(\mathbf{X}) = (1-\mathbf{X})$ as in (4.3). So $r_* = n+1$ and $r_1 = r_2 = k+1$. Assume that there exists a critical point λ_* to G. Then (2.7) yields a representation $p(\mathbf{X}) = \mathbf{X}\sum_{i=1}^{k} p_{i1}^2[\lambda_*](\mathbf{X}) + (1-\mathbf{X})\sum_{i=k+1}^{2k} p_{i2}^2[\lambda_*](\mathbf{X})$. In terms of the number of squares, here $2k$, it is clearly non optimal with respect to the result of the Lukács Theorem which involves only two polynomials whatever n.

3. Coercivity of G (Proof of Theorem 1.3 Item 1). A sufficient condition for the existence of a critical point is that G is infinite at infinity, this is called coercivity,

$$\lim_{\|\lambda\| \to +\infty} G(\lambda) = +\infty. \tag{3.1}$$

A sufficient condition for the uniqueness of the critical points is strict convexity.

In the following, we start in Section 3.1 by investigating the asymptotic behavior of G along rays starting at 0. From this knowledge we derive conditions characterizing coercivity in Section 3.2. We characterize strict convexity in Section 3.3.

3.1. The asymptotic cone. There are two types of directions in $\mathcal{D}$. For $\mathbf{d} \in \mathbb{R}^{r}*$ with $\|\mathbf{d}\| = 1$, one defines the rays $R_{\mathbf{d}} := \{\lambda = t\mathbf{d} \mid t \geqslant 0\}$ issued from the starting point $0 \in R_{\mathbf{d}}$. Two possibilities occur: either $R_{\mathbf{d}}$ intersects the boundary ∂D either it does not. In the first case if one notes $t_{\mathbf{d}} > 0$ the unique real number such that $t_{\mathbf{d}}\mathbf{d} \in \partial\mathcal{D}$, then $\lim_{t \to t_{\mathbf{d}}^-} G(t\mathbf{d}) = +\infty$. So the function G is bounded from below and coercive in the direction $\mathbf{d}$.

In this section one is interested in the rest of the directions. They generate the so-called asymptotic cone or recession cone of $\mathcal{D}$. The asymptotic cone is closed, independent of the starting point and is classically defined [11] by $C_\infty = \{\lambda \in \mathbb{R}^{r}* \text{ such that } \forall \mu \in \mathcal{D},\ t \geqslant 0,\ \mu + t\lambda \in \mathcal{D}\}$.

LEMMA 3.1. *The asymptotic cone of* $\mathcal{D}$ *is* $C_\infty = \{\lambda \in \mathbb{R}^{r}* \mid \sum_{r=1}^{r*} \lambda_r B_r \succeq 0\}$.

Proof. Let λ, μ such that $\sum_{r=1}^{r*} \lambda_r B_r \succeq 0$ and $I + \sum_{r=1}^{r*} \mu_r B_r \succ 0$. Then, $I + \sum_{r=1}^{r*} (\mu_r + t\lambda_r) B_r \succ 0$ for all $t \geqslant 0$, so λ belongs to the asymptotic cone. Conversely let λ such that for all μ and $t \geqslant 0$, $\mu + t\lambda \in \mathcal{D}$. If $\sum_{r=1}^{r*} \lambda_r B_r$ had a negative eigenvalue then for t large enough $I + t\sum_{r=1}^{r*} \lambda_r B_r$ would also have a negative eigenvalue which would contradict the fact that $t\lambda \in \mathcal{D}$. ∎

The main question is the asymptotic behavior of G in directions in C_∞.

Some preparatory material is provided. One introduces the polynomial valued vector $L(\mathbf{X})$ with components being the Lagrange polynomials associated with the set of points $(\mathbf{x}_r)_{1 \leqslant r \leqslant r_*}$ evaluated at

$\mathbf{x}$, namely

$$L(\mathbf{X}) = (l_r(\mathbf{X}))_{1\leqslant r\leqslant r_*} \in \mathbb{R}^{r*}, \tag{3.2}$$

where the Lagrange interpolation polynomials $l_r \in \mathrm{P}^n[\mathbf{X}]$ are defined by $l_r(\mathbf{x}_s) = \delta_{rs}$ for $1 \leqslant r, s \leqslant r_*$, where δ_{rs} denotes the Kronecker symbol. The vector $L(\mathbf{X})$ will be called a Lagrange vector. The polynomial p which takes the value y_r at $\mathbf{x}_r$ satisfies the Lagrange interpolation formula

$$p(\mathbf{X}) = \sum_{r=1}^{r_*} y_r l_r(\mathbf{X}) = \langle \mathbf{y}, L(\mathbf{X}) \rangle. \tag{3.3}$$

One can show another interpolation property characteristics of our problem.

LEMMA 3.2. *One has* $B(\mathbf{X}) = \sum_{r=1}^{r*} l_r(\mathbf{X}) B_r$. *For* $\mathbf{x} \in \mathbb{K}$, $B(\mathbf{x})$ *is positive semidefinite and* $L(\mathbf{x}) \in C_\infty$.

Proof. Let $\mathbf{W}, \mathbf{Z} \in \mathbb{R}^{r*}$ be the coefficients of some polynomials $(p_j)_{1\leqslant j\leqslant j_*}$ and $(q_j)_{1\leqslant j\leqslant j_*}$. By definition (1.6-1.7) of $B(\mathbf{x})$ which is symmetric one knows that

$$\left\langle \mathbf{W}, \left(B(\mathbf{x}) - \sum_{r=1}^{r_*} l_r(\mathbf{x}) B_r \right) \mathbf{Z} \right\rangle = \sum_{j=1}^{j_*} \left(g_j(\mathbf{x}) p_j(\mathbf{x}) q_j(\mathbf{x}) - \sum_{r=1}^{r_*} l_r(\mathbf{x}) g_j(\mathbf{x}_r) p_j(\mathbf{x}_r) q_j(\mathbf{x}_r) \right) = 0.$$

Since $\mathbf{W}, \mathbf{Z}$ are arbitrary, it yields the first part of the claim. Also for $\mathbf{x} \in \mathbb{K}$, one has that $g_j(\mathbf{x}) \geqslant 0$. Therefore $\langle \mathbf{W}_1, B(\mathbf{x})\mathbf{W}_1 \rangle = \sum_{j=1}^{j*} g_j(\mathbf{x}) p_j(\mathbf{x})^2 \geqslant 0$ which yields that $B(\mathbf{x}) \succeq 0$. One gets that $L(\mathbf{x}) \in C_\infty$. ∎

In the following there are three different results concerning the behavior of G in the asymptotic cone: either, Lemma 3.3, $\inf_{t>0, \lambda\in C_\infty} G(t\lambda) = -\infty$; or, Proposition 3.4, $\inf_{t>0,\lambda\in C_\infty} G(t\lambda) > -\infty$; or even better, Proposition 3.9, the function G is coercive.

LEMMA 3.3. *Assume that there exists* $\mathbf{z} \in \mathbb{K}$ *such that* $p(\mathbf{z}) < 0$. *Then* $\lim_{t\to+\infty} G(tL(\mathbf{z})) = -\infty$ *and thus the corresponding function* G *is not bounded from below in* C_∞.

Proof. The half line generated by $L(\mathbf{z})$ is included in $\mathcal{D}$ by Lemma 3.2 and so all for $t \geqslant 0$, one has $G(tL(\mathbf{z})) = \mathrm{tr}\left(M(tL(\mathbf{z}))^{-1}\right) + tp(\mathbf{z})$. Since $\lambda = tL(\mathbf{z}) \in C_\infty$, one has $M(\lambda) \succeq I$ so $G(t\lambda) \leqslant r_* + tp(\mathbf{z}) \underset{t\to\infty}{\longrightarrow} -\infty$. ∎

PROPOSITION 3.4. *Consider* $p \in \mathrm{P}^n_{\mathbb{K},+}[\mathbf{X}]$, *a unisolvent set of interpolation points* $(\mathbf{x}_r)_{1\leqslant r\leqslant r_*}$ *in* $\mathbb{K}$ *and define* $y_r = p(\mathbf{x}_r)$ *for* $1 \leqslant r \leqslant r_*$. *The following properties are equivalent.*

- *For any* $\lambda \in C_\infty$, *one has* $\langle \lambda, \mathbf{y} \rangle \geqslant 0$.
- *There exists polynomials* q_{ij} *for* $1 \leqslant j \leqslant j_*$ *and* $1 \leqslant i \leqslant r_* = r_*$ *such that*

$$p(\mathbf{X}) = \sum_{j=1}^{j_*} g_j(\mathbf{X}) \sum_{i=1}^{r_*} q_{ij}^2(\mathbf{X}).$$

Proof. For $\mathbf{W} \in \mathbb{R}^{r*}$, define the vector $s_{\mathbf{W}} = (\langle B_r \mathbf{W}, \mathbf{W} \rangle)_{1\leqslant r\leqslant r_*} \in \mathbb{R}^{r*}$. A equivalent definition of C_∞ is $C_\infty = \{\lambda \in \mathbb{R}^{r*} \text{ such that } \langle s_{\mathbf{W}}, \lambda \rangle \geqslant 0 \text{ for all } \mathbf{W} \in \mathbb{R}^{r*}\}$. In order to prove the result, one can invoke the Generalized Farkas Theorem ([11, Theorem III.4.3.4 page 131] with the correspondence $\mathbf{y} = \mathbf{b}$). It already states that our first assertion is equivalent to $\mathbf{y}$ being in the closed convex conical hull of the linear forms $s_{\mathbf{W}}$, that is $\mathbf{y} = \sum_{i=1}^{r*} \alpha_i s_{\mathbf{W}_i}$ where $\alpha_i \geqslant 0$ for all i, and r_* is sufficiently large. It is rewritten as $\mathbf{y} = \sum_{i=1}^{r*} s_{\mathbf{Z}_i}$ for $\mathbf{Z}_i = (\alpha_i)^{\frac{1}{2}} \mathbf{W}_i$. Using (1.7), the latter rewrites as our second assertion. ∎

3.2. Coercivity. Now we investigate the conditions such that G is infinite at infinity (coercivity). A first negative result about coercivity is the following. The proof easily adapted from the one of Lemma 3.3.

LEMMA 3.5. *Assume there exists* $\mathbf{z} \in \mathbb{K}$ *such that* $p(\mathbf{z}) = 0$. *Then* $G(tL(\mathbf{z}))$ *remains bounded as* $t \to +\infty$ *and* G *is not coercive.*

Thus we can only hope for coercivity starting from strictly positive polynomials. Let us know define a specific useful polynomial denoted as p_B.

DEFINITION 3.6. *Define the polynomial* $p_B(\mathbf{X}) = \mathrm{tr}\,(B(\mathbf{X})) \in \mathrm{P}^n_{\mathbb{K},+}[\mathbf{X}]$, *where* $B(\mathbf{X})$ *is the matrix defined in* (1.6).

A key property of this polynomial is the following.

LEMMA 3.7. *Assume that the matrices* $\{B_r\}_{1\leqslant r\leqslant r_*}$ *are linearly independent. Then there exists a constant* $c_* > 0$ *such that*

$$c_*\|\lambda\| \leqslant \sum_{r=1}^{r_*} \lambda_r p_B(\mathbf{x}_r), \qquad \forall \lambda \in C_\infty. \tag{3.4}$$

Proof. Let $\lambda \in C_\infty$. The matrix $\sum_r \lambda_r B_r$ is symmetric and positive semidefinite. So its matrix norm can be controlled by its largest eigenvalue and thus by its trace, namely $\|\sum_{r=1}^{r_*} \lambda_r B_r\| \leqslant \mathrm{tr}\,(\sum_{r=1}^{r_*} \lambda_r B_r) = \sum_{r=1}^{r_*} \lambda_r p_B(\mathbf{x}_r)$. Second we also know that $\lambda \to \sum_{r=1}^{r_*} \lambda_r B_r$ is injective thanks to the linear independence assumption. Thus there a constant $c_* > 0$ such that $c_*\|\lambda\| \leqslant \|\sum_{r=1}^{r_*} \lambda_r B_r\|$. Combining both inequalities ends the proof. ∎

Remark 3.8. The assumption of linear independence of $\{B_r\}_{1\leqslant r\leqslant r_*}$ is close but different than the condition of Linear Independence Constraint Qualification (LICQ), see [23, Section 12.2], which in our setting says that the matrices $\{B_r\mathbf{U}\}_{1\leqslant r\leqslant r_*}$ are linearly independent for any matrix $\mathbf{U}$ such that $\sum_{i=1}^{r_*} \langle B_r\mathbf{U}_i, \mathbf{U}_i\rangle = y_r$ for $1 \leqslant r \leqslant r_*$. One may prove by contradiction that LICQ implies our assumption.

PROPOSITION 3.9. *Let* $p \in \mathrm{P}^n_{\mathbb{K},+}[\mathbf{X}]$ *which admits a SOS (1.3). Take a unisolvent set of interpolation points* $(\mathbf{x}_r)_{1\leqslant r\leqslant r_*}$ *in* $\mathbb{K}$ *and assume that the corresponding matrices* $\{B_r\}_{1\leqslant r\leqslant r_*}$ *are linearly independent. Take* $\varepsilon > 0$ *and set* $p^\varepsilon = p + \varepsilon p_B$. *Then the function* G^ε *built from* $\mathbf{x}_r$ *and* $y_r^\varepsilon = p^\varepsilon(\mathbf{x}_r) = y_r + \varepsilon p_B(\mathbf{x}_r)$ *for* $1 \leqslant r \leqslant r_*$ *is coercive.*

Proof. The asymptotic cone C_∞ does not depend on $\mathbf{y}$ or $\mathbf{y}^\varepsilon$ and we desire to show firstly that G^ε grows linearly to infinity for directions in C_∞. One has the identity $\sum_{r=1}^{r_*} \lambda_r y_r^\varepsilon = \sum_{r=1}^{r_*} \lambda_r y_r + \varepsilon \sum_{r=1}^{r_*} \lambda_r p_B(\mathbf{x}_r)$. Take $\lambda \in C_\infty$: proposition 3.4 yields $\sum_{r=1}^{r_*} \lambda_r y_r \geqslant 0$ because p is a SOS by assumption; then Lemma 3.7 shows that for any $\lambda \in C_\infty$ $\sum_{r=1}^{r_*} \lambda_r y_r \geqslant 0 + \varepsilon c_*\|\lambda\|$ which yields uniform coercivity in the directions in the asymptotic cone.

In order to show coercivity (3.1) which is a stronger statement, the proof is by contradiction. Assume it does not hold. Then there exists a constant $K \in \mathbb{R}$ as well as a sequence $(t_m, \mathbf{d}_m)_{m\in\mathbb{N}}$ such that $t_m \to +\infty$, $\|\mathbf{d}_m\| = 1$ and $G(t_m\mathbf{d}_m) \leqslant K$. By convexity, and since $G(0) = r_*$, one has $G(t\mathbf{d}_m) \leqslant \max(r_*, K)$ for $t \in [0, t_m]$. Up to the extraction of a sub-sequence there exists $\mathbf{d}_*$ with $\|\mathbf{d}_*\| = 1$, such that $G(t\mathbf{d}_*) \leqslant \max(r_*, K)$ for $t \in \mathbb{R}^+$. In particular the ray with direction $\mathbf{d}_*$ cannot intersect the boundary $\partial\mathcal{D}$ so it belongs to the asymptotic cone C_∞. By the first estimate $G(t\mathbf{d}_*) \geqslant \varepsilon c_* t$, so it cannot be bounded which yields the contradiction. ∎

3.3. Strict convexity. Strict convexity, if it holds, yields uniqueness of a critical point. This information is additional to Item 1 of Theorem 1.3.

PROPOSITION 3.10. *Let $p \in \mathrm{P}^n_{\mathbb{K},+}[\mathbf{X}]$ be strictly positive on $\mathbb{K}$. Take a unisolvent set of interpolation points $(\mathbf{x}_r)_{1\leqslant r\leqslant r_*}$ in $\mathbb{K}$ and assume that the corresponding matrices $\{B_r\}_{1\leqslant r\leqslant r_*}$ are linearly independent. Then G is strictly convex over its domain $\mathcal{D}$.*

Proof. From (2.5) one has that $\langle \nabla^2 G(\lambda)\mu, \mu\rangle = 2\sum_{i=1}^{r_*} \langle A_i(\mu,\lambda), M(\lambda)^{-1}A_i(\mu,\lambda)\rangle \geqslant 0$ for all $\mu \in \mathbb{R}^{r_*}$, where $A_i(\mu,\lambda) = (\sum_{r=1}^{r_*} \mu_r B_r)\mathbf{U}_i(\lambda)$ for $1 \leqslant i \leqslant r_*$. Since $M(\lambda)^{-1}$ is positive definite, its columns $\mathbf{U}_i(\lambda)$ form a basis.

By contradiction, assume now G is not strictly convex. There exists $\mu \neq 0$ such that $\langle \nabla^2 G(\lambda)\mu, \mu\rangle = 0$. So the vectors $A_i(\mu,\lambda)$ vanish for all i. So $\sum_{r=1}^{r_*} \mu_r B_r = 0$, and $\mu = 0$ by linear independence of the matrices $(B_r)_{r=1,\ldots,r_*}$. This is a contradiction so $\nabla^2 G(\lambda) > 0$ and G is strictly convex. ∎

The strict convexity of G can be measured with the minimal eigenvalue of its Hessian $\alpha(\lambda) = \inf_{\mu\neq 0} \frac{\langle \nabla^2 G_{\mathbf{V}}(\lambda)\mu,\mu\rangle}{\|\mu\|^2} > 0$, for any $\lambda \in \mathcal{D}$. An important property which motivates the design of one of our numerical methods is the following.

LEMMA 3.11. *Under the assumptions of Proposition 3.10, then α has a cubic degeneracy at infinity in the interior of the asymptotic cone of $\mathcal{D}$. For all $\mathbf{d} \in \mathbb{R}^{r_*}$ such that $\|\mathbf{d}\| = 1$ and $\sum_{r=1}^{r_*} d_r B_r \succ 0$, there is $C_{\mathbf{d}} > 0$ such that $\alpha(t\mathbf{d}) \leqslant C_{\mathbf{d}}(1+t)^{-3}$ for all $t \geqslant 0$.*

Proof. Let $\lambda = t\mathbf{d}$. For a constant C depending only on the data, one has $\langle \nabla^2 G(\lambda)\mu,\mu\rangle \leqslant C\|M(\lambda)^{-1}\|^3\|\mu\|^2$. Under the assumptions the minimal eigenvalue of $M(\lambda)$ is given by $1 + e_{\mathbf{d}}t$ with $e_{\mathbf{d}}$ the minimal eigenvalue of $\sum_{r=1}^{r_*} d_r B_r$. Hence $\|M(\lambda)^{-1}\| = O((1+t)^{-1})$.

4. Univariate polynomials on a segment (Theorem 1.3 Item 2). In this section, we focus on univariate polynomials, namely when $d = 1$, over the segment $\mathbb{K} = [0,1]$. This case is interesting because it is central for for numerical computation of functions of one variable and also one can easily prove the coercivity and the strict convexity. The notation is simplified by using the real variable $x \in \mathbb{R}$, more adapted to analytical methods.

We check that the various assumptions granting coercivity and strict convexity are satisfied. In view of Proposition 3.4, Proposition 3.9 and Proposition 3.10 of the previous section, it suffices to exhibit an appropriate choice of functions $(g_j)_j$ and of interpolation points such that: any non-negative polynomial admits a (possibly non-explicit) SOS decomposition; and the matrices $\{B_r\}_r$ are linearly independent. The first point follows from the *Markov-Lukács* Theorem, see [29, 6, 7, 13] for a proof.

PROPOSITION 4.1 (*Markov-Lukács*). *Let us consider $p \in \mathrm{P}^n[x]$ and $\mathbb{K} = [0,1]$.*

- ***Even case:*** *If $n = 2k$, then p is non-negative on $\mathbb{K}$ if and only if there are polynomials a and b with degree less or equal to k and $k-1$ respectively such that*

$$p(x) = a^2(x) + x(1-x)b^2(x). \tag{4.1}$$

- ***Odd case:*** *If $n = 2k+1$, then p is non-negative on $\mathbb{K}$ if and only if there are polynomials a and b with degree less or equal to k such that*

$$p(x) = xa^2(x) + (1-x)b^2(x). \tag{4.2}$$

Now let us precise the setting. One takes $j_* = 2$ and

$$\begin{cases} \text{for } n \text{ is even}: & g_1(x) = 1 \quad \text{and} \quad g_2(x) = x(1-x), \\ \text{for } n \text{ is odd}: & g_1(x) = x \quad \text{and} \quad g_2(x) = 1-x. \end{cases} \tag{4.3}$$

Concerning the interpolation points, we choose any $r_* = n+1$ distinct points $(x_r)_{r=1,\ldots,n+1}$ on the segment $[0,1]$. The polynomials are represented along monomials so that the matrices B_r have the

block structure

$$B_r = \begin{pmatrix} g_1(x_r)\mathbf{w}_1^r \otimes \mathbf{w}_1^r & 0 \\ 0 & g_2(x_r)\mathbf{w}_2^r \otimes \mathbf{w}_2^r \end{pmatrix} \in \mathbb{R}^{(n+1)\times(n+1)} \tag{4.4}$$

where

$$\begin{cases} \text{for } n = 2k: & \mathbf{w}_1^r = \left(1, x_r, \dots, x_r^k\right)^t \text{ and } \mathbf{w}_2^r = \left(1, x_r, \dots, x_r^{k-1}\right)^t, \\ \text{for } n = 2k+1: & \mathbf{w}_1^r = \mathbf{w}_2^r = \left(1, x_r, \dots, x_r^k\right)^t. \end{cases}$$

With these notations, the equalities (4.1) and (4.2) are equivalent to $y_r = \langle B_r \mathbf{U}, \mathbf{U}\rangle$ for $1 \leqslant r \leqslant n+1$. In the odd case $n = 2k+1$ one has $\mathbf{U} = (a_0, \dots, a_k, b_0, \dots b_k)^t \in \mathbb{R}^{n+1}$ with $a(x) = \sum_{l=0}^k a_l x^l$ and $b(x) = \sum_{l=0}^k b_l x^l$. In the even case $n = 2k$, $\mathbf{U} = (a_0, \dots, a_k, b_0, \dots b_{k-1})^t \in \mathbb{R}^{n+1}$.

COROLLARY 4.2 (of Proposition 3.4). *Take $p \in P^n_{[0,1],+}$ and set $y_r = p(x_r)$. Then, for all $\lambda \in C_\infty$, one has that $\langle \lambda, \mathbf{y}\rangle \geqslant 0$.*

Proof. Indeed the second statement of Proposition 3.4 holds with $i_* = 1$ by taking $p_{11} = a$ and $p_{12} = b$ with a, b provided by Proposition 4.1. ∎

Let $\lambda \in \mathbb{R}^{n+1}$. Using the structure (4.4) of the matrices B_r, one has the Hankel matrices

$$\sum_{r=1}^{n+1} \lambda_r B_r = \begin{pmatrix} H_1 & 0 \\ 0 & H_2 \end{pmatrix} \tag{4.5}$$

where

$$\begin{cases} \text{for } n = 2k: & \langle H_1\mathbf{v}, \mathbf{w}\rangle = \displaystyle\sum_{i,j=0}^{k} s_{i+j+1} v_i w_j, \quad \langle H_2\mathbf{v}, \mathbf{w}\rangle = \sum_{i,j=0}^{k} (s_{i+j} - s_{i+j+1}) v_i w_j, \\ \text{for } n = 2k+1: & \langle H_1\mathbf{v}, \mathbf{w}\rangle = \displaystyle\sum_{i,j=0}^{k} s_{i+j+1} v_i w_j, \quad \langle H_2\mathbf{v}, \mathbf{w}\rangle = \sum_{i,j=0}^{k-1} (s_{i+j+1} - s_{i+j+2}) v_i w_j. \end{cases}$$

The s_i's are given by $s_i = \sum_{r=1}^{n+1} \lambda_r x_r^i$. The linear map $\lambda \mapsto (s_0, \dots, s_n)$ is one to one, since $(s_0, \dots, s_n)$ is obtained by multiplying λ by a Vandermonde matrix, which is invertible. A direct consequence is the following.

LEMMA 4.3. *The matrices $\{B_r\}_{1\leqslant r\leqslant r_*}$ are linearly independent.*

Proof. Assume $\sum_{r=0}^n \lambda_r B_r = 0$. Then (4.5) and the definition of H_1 and H_2 yields that $s_0 = \cdots = s_n = 0$. It yields $\lambda = 0$. So the $\{B_r\}_{1\leqslant r\leqslant r_*}$ are linearly independent. ∎

PROPOSITION 4.4. *For any univariate polynomial p that is strictly positive on $\mathbb{K} = [0,1]$, the associated function G is strictly convex and coercive. As a consequence, it has a unique critical point $\lambda^* \in \mathcal{D}$ which defines a sum of squares decomposition $p[\lambda^*] = p$.*

Proof. Thanks to Corollary 4.2 and Lemma 4.3, the assumptions of Proposition 3.9 and Proposition 3.10 are satisfied which yields the result. ∎

5. Numerical algorithms. The numerical methods are based on the minimization of the dual function G either by a descent type algorithm, either by the direct search of a critical point with a Newton type methods. Let us emphasize that as G is a proper strictly convex and coercive function on its domain, its minimization is equivalent to the search of a critical point. All the methods enter the generic iterative framework

$$\lambda^{m+1} = \lambda^m - \tau_m H_m^{-1} \nabla G(\lambda^m), \qquad \lambda^0 = 0, \tag{5.1}$$

with H_m and τ_m to be defined. In terms of complexity, the cost of one iteration is essentially the cost of computation $\nabla G(\lambda^m)$ with formula (2.5). Indeed, one needs to compute the inverse of $M^{-1}(\lambda)$ and then do $O(r*)$ matrix multiplications with the matrices B_r. It yields a cost in $O(r_*^3)+O(r_*\times r_*^3)$. Observe that the Hessian of G is not more expensive to compute since all the vectors $B_r\mathbf{U}_i$ and $M(\lambda)^{-1}$ have already been computed. Therefore, one needs to do $O(r*)$ matrix multiplications to get the $M(\lambda)^{-1}B_r\mathbf{U}_i$ and from these matrices the assembling of the Hessian is no more than $O(r_*^4)$. In the end, the cost of the inversion of H_m, whatever how it is defined, is less than the evaluation of the gradient.

5.1. Choices for H_m. Let us first explain the various choice for the matrix H_m.

5.1.1. Forward descent method. The first method we use is the classical descent method which consists in taking $H_m = I$ the identity matrix.

5.1.2. Backward descent method. Given a sequence of positive time steps τ_m, the following iterative scheme $\tilde{\lambda}^{m+1} = \arg\min_{\lambda\in\mathcal{D}} G(\lambda) + \frac{1}{2\tau_m}\|\lambda - \tilde{\lambda}^m\|^2$ with initial guess $\tilde{\lambda}^0 = 0$ is well defined since G is convex. It corresponds exactly to the implicit Euler discretization of the gradient flow with variable time steps. At step m we look for the critical point of the strictly convex objective function by making one step of a Newton method starting at λ^m, yielding the scheme (5.1) with $H_m = I + \tau_m\nabla^2 G(\lambda_m)$.

5.1.3. Newton-Raphson method. A straightforward method for a direct search of the critical point of G is the classical Newton method $H_m = \nabla^2 G(\lambda_m)$, with $\nabla^2 G$ the Hessian of G.

5.1.4. Modified Newton-Raphson method. The Hessian of G degenerates at infinity as showed in Lemma 3.11. In practice, a classical Newton-Raphson method for solving $\nabla G(\lambda) = 0$ can be inaccurate at the first iterations in some cases. Instead one may notice that λ_* is a critical point of $G(\lambda)$ if and only if it is a critical point of $(G(\lambda) - C)^2$ where C is a constant which is smaller than the infimum of G. One expects the latter function to grow quadratically at infinity thus improving the conditioning of the Hessian. This suggests the modified Newton method (5.1) with $H_m = \alpha_m\nabla G(\lambda_m)\otimes\nabla G(\lambda_m) + \nabla^2 G(\lambda_m)$. Several choices are possible for α_m. Following the heuristic one could impose $\alpha_m = (G(\lambda_m) - C)^{-1}$ but C is not known *a priori*. In practice, we found out that the empirical choice $\alpha_m = \|\nabla G(\lambda_m)\|/(\|\nabla G(\lambda_m)\| + \|\nabla G(0)\|)$ yields good results. This choice is motivated by the fact that close to the critical point, the method degenerates back to the classical Newton-Raphson method.

5.2. Choice of the time step τ_m. Now we detail the choice of adaptive time step. A preliminary concern is whether one can ensure that every iterate stays in the domain of G.

5.2.1. Maximal time step. It is possible to guarantee the condition $\lambda_m \in \mathcal{D}$ for any m by imposing a simple threshold on the time step. Indeed start from $\lambda_m \in \mathcal{D}$. Since $\lambda^{m+1} = \lambda^m - \tau_m\mathbf{d}^m$ for a given direction $\mathbf{d}^m = (d_r^m)$, the condition $\lambda^{m+1} \in \mathcal{D}$ is satisfied provided $M(\lambda^m) - \tau_m\sum_r d_r^m B_r \succeq 0$. A sufficient condition is that $\tau_m \leqslant \tau_{\max}$ with $\tau_{\max}$ such that $\mu_{\max}\left(\sum_r d_r^n B_r\right)\tau_{\max} \leqslant \mu_{\min}(M(\lambda^m))$, where $\mu_{\max}(A)$ and $\mu_{\min}(A)$ denote respectively the maximum and minimum of the absolute values of the eigenvalue of a real symmetric square matrix A (*i.e.* the spectral radius and, if A is positive definite, the spectral gap). This condition is very much like a CFL stability condition. In various test cases we observed that $\tau_{\max}$ is of the order of 1 initially and tends to increase as iterates get closer to the solution.

5.2.2. Empirical adaptive time step. The first choice of time step relies on a criteria of decay of $\|\nabla G(\lambda_m)\|$. In the case of descent methods, it differs from the more usual Wolfe condition [23] which enforces a decay of $G(\lambda_m)$ and it is fairly close to the so-called strong Wolfe condition. We make this choice because it is well adapted to our particular setting. Indeed we recall that $\|\nabla G(\lambda_m)\|$ actually measures the Euclidean norm between the current sum of squares $(p[\lambda^m](x_r))_r$ and $\mathbf{y}$. By equivalence of norms and unisolvance one has $\|p - p[\lambda^m]\|_{P^n[\mathbf{X}]} \leqslant c_n\|\nabla G(\lambda^m)\|$, for some constant $c_n > 0$ depending

only on n, whatever the choice of norm of the space of polynomials. It is thus the natural measure of the error which has to be decreased by the iterative algorithm.

The adaptive time step τ_m is defined as follows. We choose *a priori* $0 < \tau_{\min} \leqslant \tau_0 \leqslant \tau_{\max}$. Then we define $\lambda_m^{(k)} = \lambda_n - 2^{-k}\tau_m^{(k)} H_m^{-1}\nabla G(\lambda_m)$ and denote by k_m the smallest integer such that $\|\nabla G(\lambda_m^{(k)})\| < \|\nabla G(\lambda_{m+1})\|$. From there we define $\tau_{m+1} = \max(2^{-k_m}\tau_m, \tau_{\min})$ for $k_m > 0$ and $\tau_{m+1} = \min(2\tau_m, \tau_{\max})$ for $k_m = 0$.

In the case of Newton methods, we take $\tau_0 = \tau_{\max} = 1$ in order to achieve the expected quadratic rate of convergence. As we shall see in the numerical results, the decrease of the time step in Newton methods coincides with a bad conditioning of the Hessian matrix.

5.2.3. Barzilai and Borwein time step. In the case of the forward descent method of Section 5.1.1, there is a particular choice of time step relying on the two previous iterates due to Barzilai and Borwein in their seminal paper [1] (see [26] for an improvement of the original method which ensures global convergence, two other recent works are [9, 30]). It can be seen as an intermediate between the classical gradient descent method of Cauchy and the Newton method as it generalizes the secant method in higher dimensions. The corresponding time step is given, for $m \geqslant 1$, by either

$$\tau_m = \frac{\langle \nabla G(\lambda_m) - \nabla G(\lambda_{m-1}), \lambda_m - \lambda_{m-1}\rangle}{\|\nabla G(\lambda_m) - \nabla G(\lambda_{m-1})\|^2}, \tag{5.2}$$

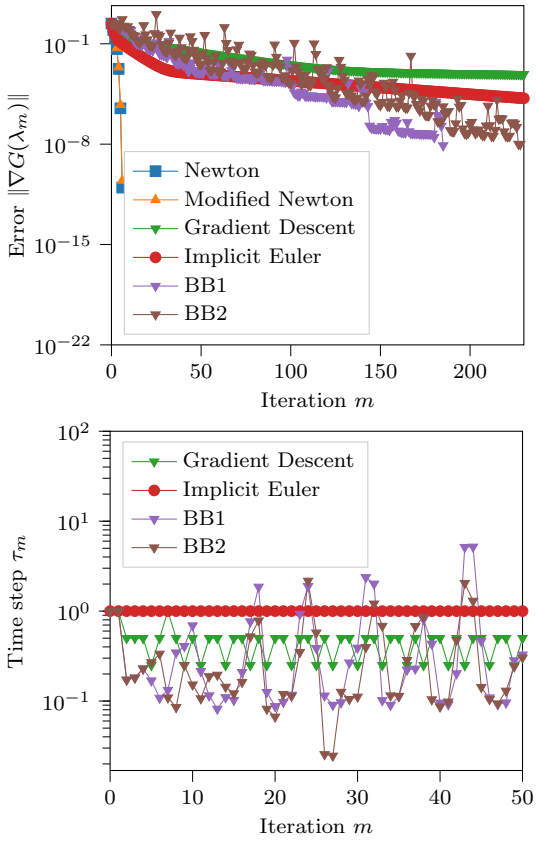
or

$$\tau_m = \frac{\|\lambda_m - \lambda_{m-1}\|^2}{\langle \nabla G(\lambda_m) - \nabla G(\lambda_{m-1}), \lambda_m - \lambda_{m-1}\rangle}. \tag{5.3}$$

It is known that this method does not yield a monotone decay of either G or $\|\nabla G\|$ in general. In our case it may be that $\lambda_{m+1} \notin \mathcal{D}$ with the choices (5.2) and (5.3). Thus the methods are stabilized by replacing τ_m with $\tau_{\max}$ given in Section 5.2.1 whenever $\tau_m > \tau_{\max}$.

6. Numerical experiments. In this section, we perform various numerical experiments in order to illustrate the theoretical results and to explore the behavior of the numerical algorithms. The implementation has been performed with Matlab and Python, with no noticeable difficulties. The maximal time step of Section 5.2.1 is calculated with built-in subroutines, the extra-cost is negligible. In the following, we denote by "Gradient descent", "BB1"and "BB2" the methods where $H_m = I$ and the time step is taken as in Section 5.2.2 and Section 5.2.3 with (5.2) and (5.3) respectively. The methods "Implicit Euler", "Newton" and "Modified Newton" correspond respectively to the choices of Section 5.1.2, Section 5.1.3 and Section 5.1.4 and the time step is taken as in Section 5.2.2

6.1. Univariate polynomials on a segment. Here we consider univariate SOS polynomials. We proceed as explained in Section 4, except that the monomial basis is replaced here by the orthogonal basis of shifted Chebychev polynomials $(T_i(x))_{i=1,\dots,k}$ satisfying $T_i(\cos(\theta) + 1)/2) = \cos(i\theta)$, for all $\theta \in \mathbb{R}$. The only modification of the method presented earlier concerns the definition of the D_r matrices which become $D_r = \mathbf{w}_r^t \mathbf{w}_r \in \mathbb{R}^{r_k \times r_k}$ with $\mathbf{w}_r = (T_0(x_r), T_1(x_r), \dots, T_k(x_r))^t \in \mathbb{R}^{r_k}$. The reason is that shifted Chebychev polynomials have much better behavior in terms of numerical approximation, since they produce "uniformly distributed" polynomials in $[0, 1]$, see [8] for comprehensive mathematical treatment. On the opposite, monomials x^i which concentrate at $x = 1$ for $i \to +\infty$ are non optimal for numerical approximation in the segment $[0, 1]$. One can refer to [6] for a comparison between the use of Chebychev polynomials and monomials. In the following we propose different test cases to illustrate the properties of the various descent and Newton-Raphson type methods proposed in Section 5. For univariate polynomials, the tests 1-2-3 are performed with the odd order option (4.3) of the weights: similar results are observed with $g_1(x) = 1$ and $g_2(x) = x(1 - x)$, and so are not reported. Test 5 is performed with both the odd and even options.



Method	Computation time	Number of iterates
Newton	0.0051	6
Modif. Newton	0.0048	6
Grad. Descent	0.7755	2727
Implicit Euler	0.2445	573
BB1	0.0378	185
BB2	0.0415	228

Iterations stop when $\|\nabla G(\lambda_m)\| \leqslant 10^{-8}$

FIG. 2. *Test case 1. Sum of square interpolation of* $p(x) = x^5 + 1$. *(Top left) Error* $\|\nabla G(\lambda_m)\|$ *vs. iteration* m*; (Top right) Objective function* $G(\lambda_m)$ *vs. iteration* m*; (Bottom left) Step size* τ_m *vs. iteration* m*; (Bottom right) Number of iterates and computation time to reach* $\|\nabla G(\lambda)\| < 10^{-8}$ *for each method.*

6.1.1. Test case 1. We compare the convergence of the methods for an easy objective polynomial, that is a polynomial with low degree and far above 0: we take $n = 5$, $r_* = i_* = n+1 = 6$, $p(x) = x^5 + 1$ and the weights $g_1(x) = x$ with $g_2(x) = 1 - x$ (so $j_* = 2$).
We observe on Figure 2 that the Newton type methods both reach the threshold precision of 10^{-8} after only 6 iterations. The implicit Euler and gradient descent methods need respectively 573 and 2727 iterations to reach the same error: this low convergence has been observed for many other test cases. This is why we continue the tests with the Newton and Barzilai and Borwein methods only.

6.1.2. Test case 2. In this second test case, we illustrate the better performance of the modified Newton-Raphson method compared to the other methods. We choose a highly oscillating objective polynomial with lower bound equal to 0. It is given by $n = 21$, $r_* = i_* = n + 1 = 22$, $p(x) = T_{21}(x) + 1$ and the weights $g_1(x) = x$ with $g_2(x) = 1 - x$ (so $j_* = 2$).

We observe on Figure 3 that the modified Newton-Raphson method reaches a precision of around 10^{-8} in 40 iterations. In the case of the standard Newton-Raphson method, the adaptive time step quickly reduces to a very small value in order to keep decreasing the error at each iteration. A similar phenomena happens near convergence for the modified Newton-Raphson method. These behaviors

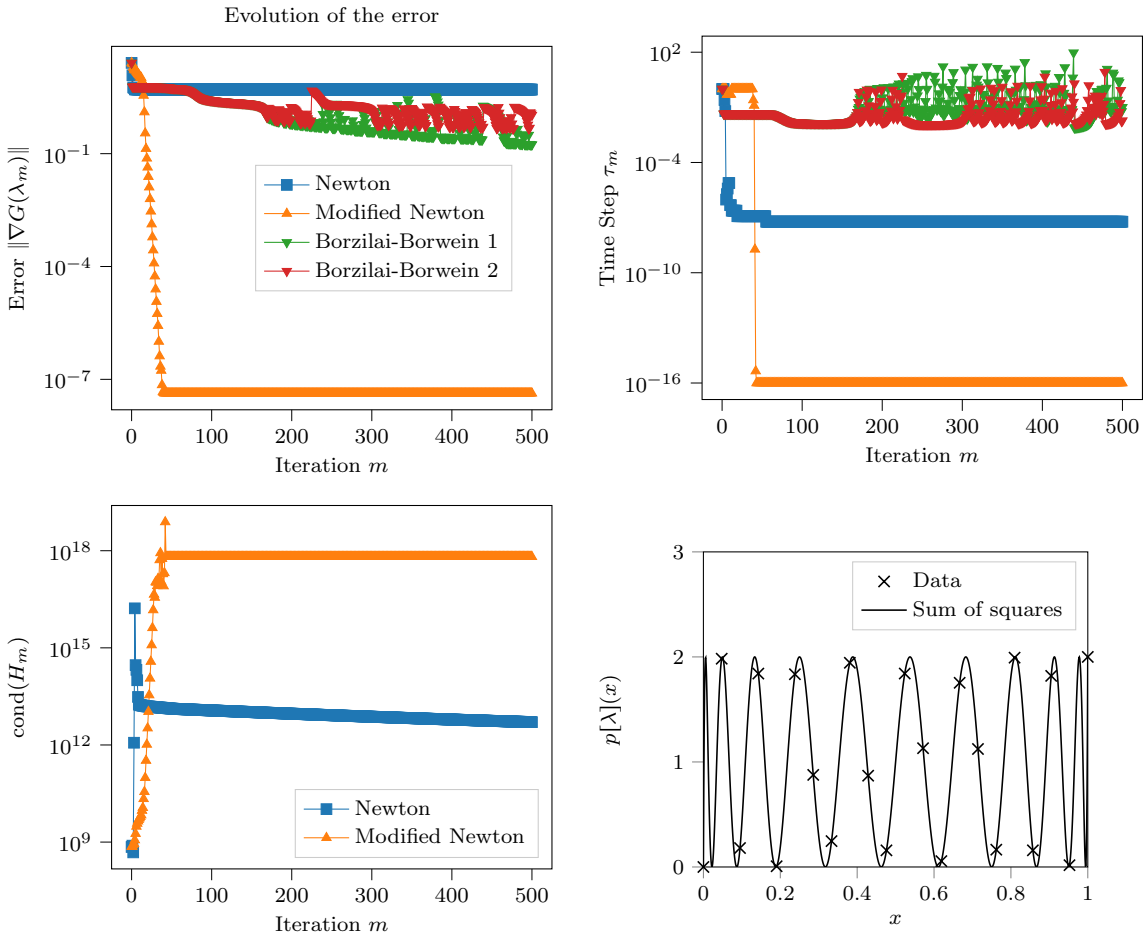


FIG. 3. *Test case 2. Sum of square interpolation of* $p(x) = T_{21}(x) + 1$. *(Top left) error* $\|\nabla G(\lambda_m)\|_2$ *vs. iteration* m*; (Top right) Step size* τ_m *vs. iteration* m*; (Bottom left) Condition number of* H_m *vs. iteration* m*; (Bottom right) Data* $(y_r = p(x_r))_r$ *and sum of squares* $p[\lambda](x)$ *satisfying* $\|\nabla G(\lambda)\| < 10^{-6}$.

can be interpreted thanks to the evolution of the condition number of the matrix H_m also showed on Figure 3. Let us recall that this matrix needs to be inverted at each iteration. On the first hand, for the Newton-Raphson method, H_m is the Hessian of G which degenerates when λ is far from the minimizer of G, as explained in Lemma 3.11. The modified Newton-Raphson method seems to prevent a bad condition number of the tweaked Hessian in the first few iterations. On the second hand, since the objective polynomial has a 0 lower bound, strict convexity and coercivity of G are not granted and it may explain the bad conditioning of H_m near convergence in the case of the modified Newton-Raphson method. Indeed recall that when $\nabla G(\lambda_m)$ is small H_m almost coincides with the Hessian in the modified Newton-Raphson method.

Concerning the Barzilai and Borwein methods, we found out that the convergence is very slow on this test case. A minimal error $\|\nabla G(\lambda_m)\|$ of around 10^{-4} is attained after 100000 iterations for the "BB1" method, and worse performances are obtained with the "BB2" method. While these methods are well-suited for many nonlinear programming problems, it seems that despite the cost of the Hessian

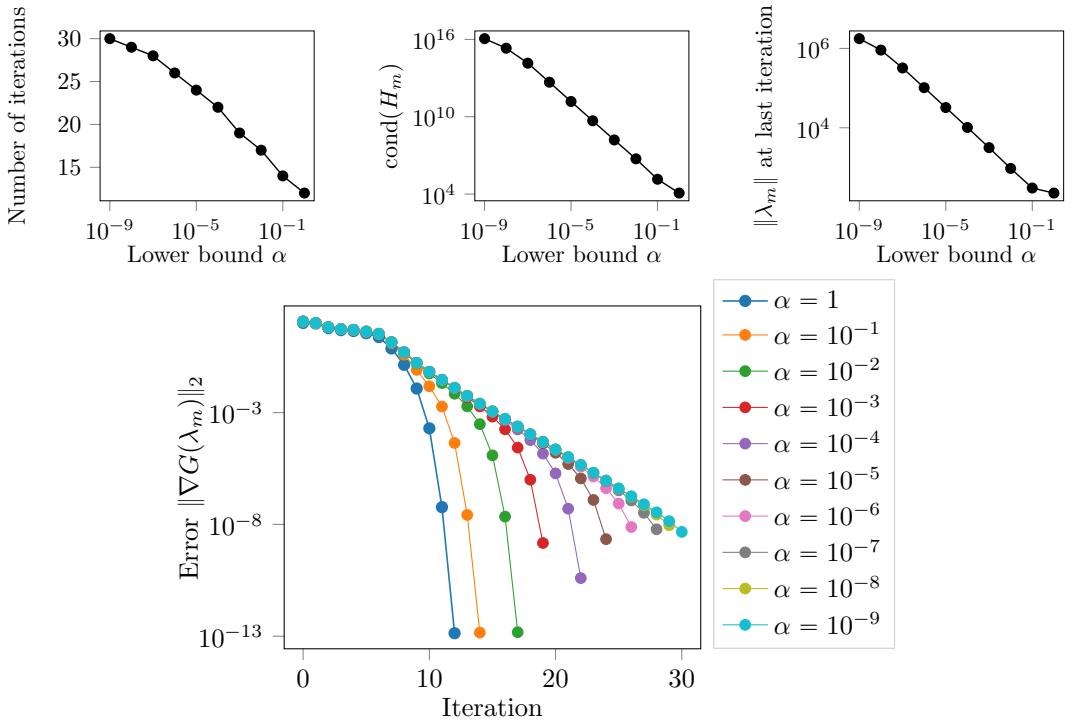


FIG. 4. *Test case 3. Influence of the lower bound α in the sum of square interpolation of $p(x) = (T_{11}(x) + 1) + \alpha$. (Top left) Number of iterations to converge vs. α; (Top right) Condition number of H_m at the last iteration vs. α; (Bottom) Error $\|\nabla G(\lambda_m)\|_2$ vs. iteration m for different lower bounds α.*

inversion, it is significantly cheaper in terms of computational effort to use Newton type iterations on our particular problem.

Eventually we found in many numerical experiments that, on the particular problem addressed in this paper, the modified Newton method is by far the most robust and efficient method among the ones we tested. This the reason why we only use the modified Newton-Raphson method in the following series of tests.

6.1.3. Test case 3. Now, we illustrate the influence of the lower bound of p on the convergence of the method. To proceed, we compute a sum of squares approximation of the polynomial $p(x) = T_{11}(x) + 1 + \alpha$ for various lower bounds α ($n = 5$, $r_* = i_* = n + 1 = 6$, $j_* = 2$).
The results are displayed on Figure 4. We observe that the number of iterations required to reach a precision of 10^{-8} seems to increase proportionally with $|\log(\alpha)|$. The condition number of H_m and the norm of λ_m at convergence decays like some negative power of α. Interestingly enough, one also sees that the quadratic convergence of the (modified) Newton method seems to degenerate to linear convergence when α goes to 0. All these behaviors can be interpreted thanks to the results of Lemma 3.5 and Lemma 3.11. We know from Lemma 3.5 that for $\alpha = 0$, p has a root x_0 in $[0, 1]$, and thus the coercivity of G is lost in some direction of the asymptotic cone of $\mathcal{D}$ (that of the Lagrange vector $L(x_0)$). Thus as $\alpha \to 0$, the minimizer λ_α^* may go to $+\infty$ in the asymptotic cone which would explain here the explosion of the norm of λ and of the condition number of H_m as predicted by Lemma 3.11 and shown on Figure 4.

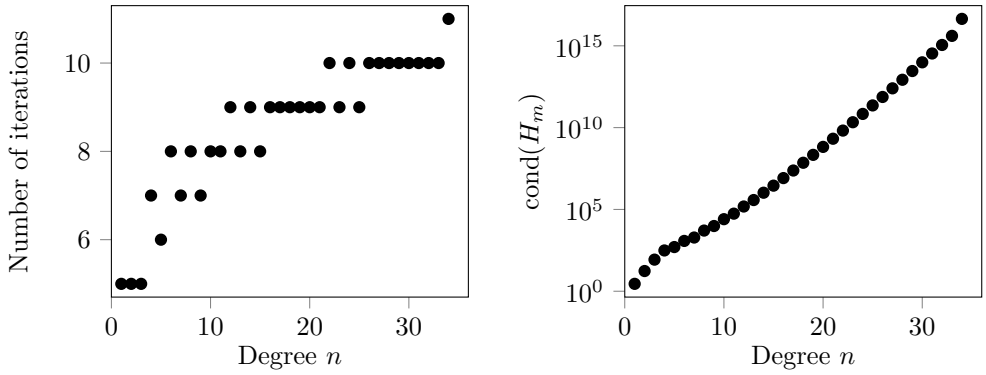


FIG. 5. ***Test case 4.*** *Influence of the degree n in the sum of square interpolation of $p(x) = x^n + 1$. (Left) Number of iterations to converge vs. n; (Right) Condition number of H_m at the last iteration vs. n.*

6.1.4. Test case 4. In this fourth test case we illustrate the influence of the degree n of the objective polynomial $p(x) = x^n + 1$ on the convergence of our method, with $g_1(x) = x$ and $g_2(x) = 1 - x$ for n odd and $g_1(x) = 1$ and $g_2(x) = x(1 - x)$ for n even.
The result are displayed on Figure 5. We observe that the number of iterations required to reach an error of 10^{-8} increases with the degree, but weakly. We also observe that the condition number $\mathrm{cond}(H_m) = \|H_m\|\|H_m^{-1}\|$ near convergence deteriorates with n, approximately quadratically.

6.2. Bivariate polynomials on a triangle. We use the minimization algorithm for the computation of a sum of squares representation of some positive polynomial $p \in P_n[X, Y]$ on the triangle.

6.2.1. Numerical setting. The barycentric coordinates corresponding to the vertices S_1, S_2 and S_3 of the triangle are denoted as μ_j for $j = 1, 2, 3$: $\mu_1(x, y) = 1 - x - y$, $\mu_2(x, y) = x$ and $\mu_3(x, y) = y$. The triangle is $\mathbb{K} = \{\mathbf{x} = (x, y) \in \mathbb{R}^2 \mid \ \mu_1(\mathbf{x}) \geqslant 0\,,\ \mu_2(\mathbf{x}) \geqslant 0\,,\ \mu_3(\mathbf{x}) \geqslant 0\}$. The interpolation points are $\mathbf{x}_r = (x_r, y_r)$ for $1 \leqslant r \leqslant r_* = (n+1)(n+2)/2$ are the distinct points of a cartesian grid intersected with the triangle. For a given polynomial $p \in \mathrm{P}^n[\mathbf{X}]$ of a given degree, the data is $\mathbf{z} \in \mathbb{R}^{r*}$ which is the vector with components $z_r = p(x_r, y_r)$. An illustration of the geometry is provided in Figure 6 where the degree is $n = 4$.

We consider the ansatz ($\mathbf{x} = (x, y)$)

$$p[\lambda](\mathbf{x}) \ = \ \sum_{i=1}^{r_j} g_1(\mathbf{x})\, p_{i1}[\lambda](\mathbf{x})^2 + g_2(\mathbf{x})\, p_{i2}[\lambda](\mathbf{x})^2 + g_3(\mathbf{x})\, p_{i3}[\lambda](\mathbf{x})^2 + g_4(\mathbf{x})\, p_{i4}[\lambda](\mathbf{x})^2, \tag{6.1}$$

where, arbitrarily with respect to the literature [14], the weights are

$$\begin{cases} \text{for } n = 2k+1, & g_i = \mu_i \text{ for } i = 1, 2, 3 \text{ and } g_4 = \mu_1\mu_2\mu_3, \\ \text{for } n = 2k, & g_1 = \mu_2\,\mu_3,\ g_2 = \mu_3\,\mu_1,\ g_3 = \mu_1\,\mu_2 \text{ and } g_4 = 1. \end{cases} \tag{6.2}$$

With this choice we recover in every cases $r_* = r_1 + r_2 + r_3 + r_4$. All polynomials are parametrized on the basis of bivariate monomials since Chebychev polynomials are not available on the triangle.

6.2.2. Test case 5. We approach the polynomial $p(x, y) = (T_4(x) + 1)(T_4(y) + 1)/4 + 10^{-3}$ on the 2D simplex with the modified Newton method. The parameters are $n = 8$, $r_* = i_* = 45$ and $j_* = 4$.

We observe on Figure 7 that our method converges in this multivariate setting and reaches a precision of less than 10^{-8} in 210 iterations. The error decays slowly during the first 200 iterations

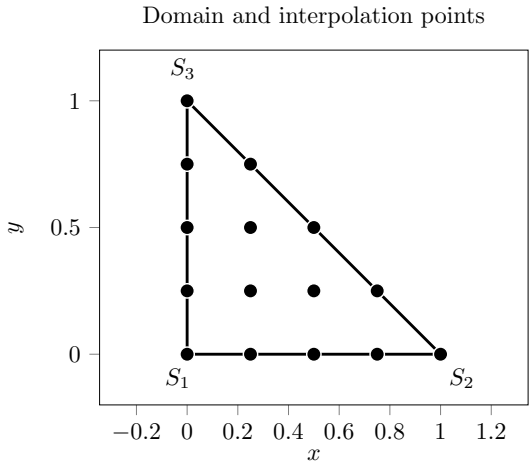


FIG. 6. *The simplex* $\mathbb{K}$ *and interpolation points for* $n = 4$.

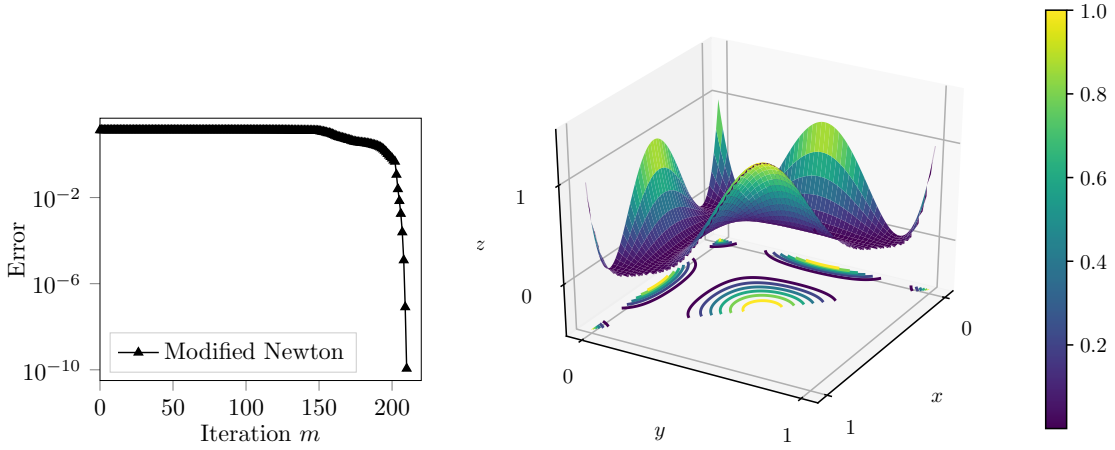


FIG. 7. *Test case 5. Bivariate sum of square interpolation of the degree* 8 *polynomial* $p(x, y) = (T_4(x) + 1)(T_4(y) + 1)/4 + 10^{-3}$ *on the 2D simplex. (Left) error* $\|\nabla G(\lambda_m)\|_2$ *vs. iteration* m*; (Right) surface plot of the converged sum of square.*

before reaching usual quadratic speed of convergence of the Newton method near the minimizer of G. This result illustrates the ability of our algorithms to provided a computational strategy for the computation of positive polynomials on bi-dimensional sets.

6.2.3. Test case 6. In this last test case we are interested in the SOS approximation of the Motzkin polynomial [19] $p(x, y) = x^2y^4 + y^2x^4 - 3x^2y^2 + 1$.

This polynomial is non-negative over $\mathbb{R}^2$ and famous for not being a sum of square in the sense that it admits no decomposition (1.3) with weigths $\widetilde{g}_1 = \cdots = \widetilde{g}_{j_*} = 1$ (whatever the choice of i_* or, equivalently in this particular case, j_*). The parameters are $n = 6$, $r_* = i_* = 28$ and $j_* = 4$. We use our method to approach this polynomial with the sum of square ansatz (6.1) but with two different weights: on the one hand we use the weights g_i (6.2) for which we expect some convergence of the algorithm; on the other other hand we use the weights $\widetilde{g}_i = 1$ for $i = 1, 2, 3, 4$.

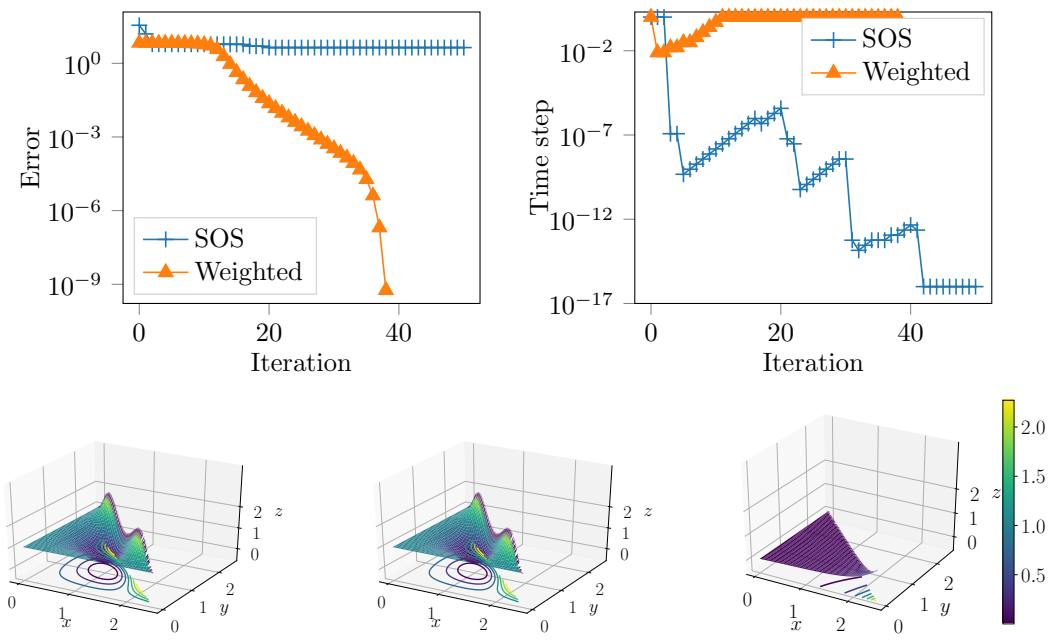


FIG. 8. ***Test case 6.*** *Bivariate sum of square approximations (of degree $n = 6$) of the Motzkin polynomial. (Top left) error $\|\nabla G(\lambda_m)\|_2$ vs. iteration m; (Top right) time step τ_m vs. iteration m; (Bottom left) The Motzkin polynomial; (Bottom center) Sum of square approximation with weights $g_1 = \mu_2\,\mu_3$, $g_2 = \mu_3\,\mu_1$, $g_3 = \mu_1\,\mu_2$ and $g_4 = 1$, the algorithm has converged; (Bottom right) Sum of square approximation without weights ($g_1 = g_2 = g_3 = g_4 = 1$), the algorithm has not converged;*

In the latter case our experiment on Figure 8 show the method does not converge (in coherence with the non-existence of a sum of square decomposition for the Motzkin polynomial). The algorithm with weights g_i converges while the algorithm with weights $\widetilde{g}_i$ does not converge (bottom right illustration in the Figure).

7. Concluding remarks. In this paper, we reformulated the problem of computing SOS decompositions into a new nonlinear convex program. On the theoretical side we analyzed this reformulation in detail, and particular the domain which guarantees convexity, and showed that up to a perturbation the problem is proper strictly convex and coercive, which ensures the existence of a solution which may be explicitly approached via iterative methods. As the literature in numerical optimization is immense, we did not give an exhaustive numerical comparison of all the methods that could be used to solve our problem. Preferably, we tried to design robust methods specifically adapted to the structure of our objective function and compare it with some classical algorithms in numerical optimization. Our numerical results show that the modified Newton-Raphson algorithm is robust to compute polynomials which respect a sign condition on a given simple semi-algebraic set. However more needs to be investigated to compare with different methods and evaluate the full potential of such methods. Here we detail possible domains of research which are consequences of the multiple connections of our methods with the ones of Scientific Computing.

• It is possible to look in more details in the case $i_* \neq r_*$. It allows greater generality of the construction, which can be convenient for optimization purposes. In such cases, the function G should be replaced by $G_{\mathbf{V}}$.

• The technical conditions on the linear independence of the matrices B_r in the multivariate case

needs further examinations. In this direction there may be links with algebraic properties such as the Archimedeanity of the quadratic module associated with the weights g_j (see [14, proof of Theorem 2.14]) or the condition of Linear Independence Constraint Qualification (LICQ) [23].

• A C++ implementation needs to be tested. On this basis it will be possible to couple with codes in scientific computing (such as the ones evoked in [27] and the references therein) to evaluate the gain in robustness with the new algorithms. Comparisons with other established softwares like the primal-dual interior-point SDP Mosek-Yalmip package [17] in Matlab will be a plus. Such benchmarks are left for future work.

Appendix A. The asymptotic cone for univariate polynomials.

One can obtain a much better understanding of the cone at infinity, which exemplifies the role of the Lagrange interpolating polynomials. Given a subset $S \subset \mathbb{R}^{n+1}$ one denotes by $\mathrm{coni}(S)$ the *conical hull* of S that is the set of linear combinations with non-negative coefficients of elements of S. The asymptotic cone can be constructed from the matrices (4.4) or (4.5) in the univariate case. The main result is the following, where the the Lagrange vectors are defined in (3.2).

THEOREM A.1. *The asymptotic cone of $\mathcal{D}$ is generated by the Lagrange vectors $L(x)$ for $0 \leqslant x \leqslant 1$, that is $C_\infty = \mathrm{coni}(\{L(x) \in \mathbb{R}^{n+1} \mid x \in [0,1]\})$.*

We need some intermediate results in order to prove Theorem A.1. First, let us define $C^1_\infty = \{\lambda \in C_\infty \mid \sum_{r=1}^{n+1} \lambda_r = 1\} \subset C_\infty$. Since $\sum_{r=1}^{r_*} l_r(x_r) = 1$ for all $1 \leqslant r \leqslant r_*$, one has $\sum_{r=1}^{r_*} l_r(X) = 1$. Therefore, with Lemma 3.2, we know that $\left\{L(x) \in \mathbb{R}^{n+1} \mid x \in [0,1]\right\} \subset C^1_\infty$. The main point of the proof is to show that $C^1_\infty \subset \left\{L(x) \in \mathbb{R}^{n+1}, \mid x \in [0,1]\right\}$. To do so we identify C^1_∞ with a subset of Borel probability measures on $[0,1]$ using the theory of the moment problem for which an comprehensive reference is [13]. The proof of the Theorem invoked below in the proof is strongly related to the Lukacs decomposition of Theorem 4.1.

PROPOSITION A.2. *Let $\lambda \in \mathbb{R}^{n+1}$. The following are equivalents: a) The vector λ belongs to C^1_∞; b) There is a Borel probability measure σ on $[0,1]$ such that*

$$\sum_{r=1}^{n+1} \lambda_r B_r = \int_{[0,1]} B(x) \mathrm{d}\sigma(x). \tag{A.1}$$

Proof. Using (4.5), one can say that $\lambda \in C^1_\infty \Longleftrightarrow (s_0, \ldots, s_n)$ are such that H_1 and H_2 are positive semidefinite matrices and $s_0 = 1$. By [13, Theorem 2.3, Theorem 2.4], this is equivalent to the existence of a Borel probability measure σ such that (A.1) holds. ∎

COROLLARY A.3. *The set C^1_∞ is compact.*

Proof. Since, by Proposition A.2, the s_i's are moments of a Borel probability measure on $[0,1]$, one has $(s_0, \ldots, s_n) \in [0,1]^{n+1}$. Therefore, since $\lambda \mapsto (s_0, \ldots, s_n)$ is linear and invertible (see Lemma 4.3), C^1_∞ is bounded. ∎

We recall that a point λ of a convex set C is said to be an extreme point (see [11, III, Definition 2.3.1]) of C if for any $\lambda_1, \lambda_2 \in C$ such that $\lambda = (\lambda_1 + \lambda_2)/2$, one has $\lambda = \lambda_1 = \lambda_2$. We denote by $\mathrm{ext}(C)$ the set of extreme points of C.

PROPOSITION A.4. *The set of extreme points of C^1_∞ is $\mathrm{ext}(C^1_\infty) = \{L(x) \mid x \in [0,1]\}$.*

Proof. Let $\lambda \in \mathrm{ext}(C^1_\infty)$. Since extreme points of a convex set are located on its boundary there is a vector $\mathbf{V} \neq 0$ such that $\left\langle \sum_{r=1}^{n+1} \lambda_r B_r \mathbf{V}, \mathbf{V} \right\rangle = 0$. Let σ be a Borel measure satisfying (A.1) and define $q(X) = \langle B(X)\mathbf{V}, \mathbf{V}\rangle \geqslant 0$. One has $\int_{[0,1]} q(x) \mathrm{d}\sigma(x) = 0$. Since q is not identically zero, the measure σ must be supported on a subset of the finite set of roots of q intersected with $[0.1]$. Since q has degree

n, σ has the form $\sigma = \sum_{k=1}^n \alpha_k \delta_{x_k}$ where $\sum_{k=1}^n \alpha_k = 1$, $0 \leqslant \alpha_k \leqslant 1$ and $x_k \in [0,1]$, for some distinct $x_1, \dots, x_n$ and where δ_{x_k} is the Dirac measure at x_k. Now assume that for some index k, $\alpha_k \in (0,1)$. Then there is $k' \neq k$ such that $\alpha_{k'} \in (0,1)$. Then let $0 \leqslant \varepsilon < \min(\alpha_k, \alpha_{k'}, 1-\alpha_k, 1-\alpha_{k'})$ and define $\sigma_1 = \sigma - \varepsilon\delta_{x_k} + \varepsilon\delta_{x_{k'}}$ and $\sigma_2 = \sigma + \varepsilon\delta_{x_k} - \varepsilon\delta_{x_{k'}}$. The measures σ_1 and σ_2 are two Borel probability measures generating different sets of moments for at least some ε in the range. Since $\lambda \mapsto (s_0, \dots, s_n)$ is linear and invertible there are distinct $\lambda_1, \lambda_2 \in C^1_\infty$ satisfying (A.1) for the respective measures σ_1 and σ_2 and one has $\lambda = (\lambda_1 + \lambda_2)/2$. There is a contradiction. Therefore either $\alpha_k = 0$ or $\alpha_k = 1$ so σ must be a dirac measure at some point $x_* \in [0,1]$. Hence $\sum_{r=1}^{n+1} \lambda_r B(x_r) = B(x_*)$ so in particular $\sum_{r=1}^{n+1} \lambda_r x_r^k = x_*^k$ for any $0 \leqslant k \leqslant n$ which yields $\lambda = L(x_*)$. Conversely if $\lambda = L(x_*)$ and $\lambda = (\lambda_1 + \lambda_2)/2$, then there are probability measures σ_1 and σ_2 such that $\delta_{x_*} = (\sigma_1 + \sigma_2)/2$. Therefore σ_1 and σ_2 are supported at x_* and since they have the same mass one has $\delta_{x_*} = \sigma_1 = \sigma_2$, so $\lambda \in \mathrm{ext}(C^1_\infty)$. ∎

Proof of Theorem A.1. Denote by $\mathrm{co}(S)$ the *convex hull* of S, the set of linear combinations of elements of S with non-negative coefficients whose sum equals 1. By the Minkowski (or Krein-Milman) theorem [11, III, Theorem 2.3.4], any compact convex set is the convex hull of its extreme points, therefore $C^1_\infty = \mathrm{co}(\mathrm{ext}(C^1_\infty))$. Remark that $C_\infty = \bigcup_{t \geqslant 0} tC^1_\infty$ and the result follows. ∎

REFERENCES

[1] J. Barzilai and J. M. Borwein, *Two-point step size gradient methods*, IMA J. Numer. Anal., 8 (1988), pp. 141–148.

[2] L. Beirão da Veiga, A. Buffa, G. Sangalli, and R. Vázquez, *Mathematical analysis of variational isogeometric methods*, Acta Numer., 23 (2014), pp. 157–287.

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

[4] S. Burer and R. D. C. Monteiro, *A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization*, Math. Program., 95 (2003), pp. 329–357. Computational semidefinite and second order cone programming: the state of the art.

[5] F. Charles, M. Campos-Pinto, and B. Després, *Algorithms for positive polynomial approximation*, to appear in Siam J. Numer. Analysis, (2017). Online at https://hal.sorbonne-universite.fr/hal-01527763.

[6] B. Després, *Polynomials with bounds and numerical approximation*, Numerical Algor., 76 (2017), pp. 829–859.

[7] B. Despres and M. Herda, *Correction to: Polynomials with bounds and numerical approximation*, Num. Algor., (2017).

[8] R. A. DeVore and G. G. Lorentz, *Constructive approximation*, vol. 303 of Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], Springer-Verlag, Berlin, 1993.

[9] D. di Serafino, V. Ruggiero, G. Toraldo, and L. Zanni, *On the steplength selection in gradient methods for unconstrained optimization*, Appl. Math. Comput., 318 (2018), pp. 176–195.

[10] D. Henrion and J. Malick, *Projection methods for conic feasibility problems: applications to polynomial sum-of-squares decompositions*, Optimization Methods & Software, 26 (2011), pp. 23–46.

[11] J.-B. Hiriart-Urruty and C. Lemaréchal, *Convex analysis and minimization algorithms. I*, vol. 305 of Fundamental Principles of Mathematical Sciences, Springer-Verlag, 1993.

[12] M. Korda, D. Henrion, and C. N. Jones, *Convergence rates of moment-sum-of-squares hierarchies for optimal control problems*, Systems Control Lett., 100 (2017), pp. 1–5.

[13] M. G. Kreĭn and A. A. Nudel'man, *The Markov moment problem and extremal problems*, American Mathematical Society, Providence, R.I., 1977.

[14] J. B. Lasserre, *Moments, positive polynomials and their applications*, vol. 1 of Imperial College Press Optimization Series, Imperial College Press, London, 2010.

[15] J. B. Lasserre, *An introduction to polynomial and semi-algebraic optimization*, Cambridge Texts in Applied Mathematics, Cambridge University Press, Cambridge, 2015.

[16] B.-G. Lee, T. Lyche, and K. Mørken, *Some examples of quasi-interpolants constructed from local spline projectors*, in Mathematical methods for curves and surfaces (Oslo, 2000), Innov. Appl. Math., Vanderbilt Univ. Press, Nashville, TN, 2001, pp. 243–252.

[17] J. Löfberg, *Yalmip : A toolbox for modeling and optimization in matlab*, in In Proceedings of the CACSD Conference, Taipei, Taiwan, 2004.

[18] J. Malick, *A dual approach to semidefinite least-squares problems*, SIAM J. Matrix Anal. Appl., 26 (2004), pp. 272–284.

[19] T. S. Motzkin, *The arithmetic-geometric inequality*, Inequalities (Proc. Sympos. Wright-Patterson Air Force Base, Ohio, 1965), (1967), pp. 205–224.

[20] Y. Nesterov, *Squared functional systems and optimization problems*, in High performance optimization, vol. 33 of Appl. Optim., Kluwer Acad. Publ., Dordrecht, 2000, pp. 405–440.
[21] Y. Nesterov and A. Nemirovskii, *Interior-point polynomial algorithms in convex programming*, vol. 13 of SIAM Studies in Applied Mathematics, SIAM, Philadelphia, PA, 1994.
[22] J. Nie and M. Schweighofer, *On the complexity of Putinar's Positivstellensatz*, J. Complexity, 23 (2007), pp. 135–150.
[23] J. Nocedal and S. J. Wright, *Numerical optimization*, Springer Series in Operations Research and Financial Engineering, Springer, New York, second ed., 2006.
[24] V. Powers and T. Wörmann, *An algorithm for sums of squares of real polynomials*, J. Pure Appl. Algebra, 127 (1998), pp. 99–104.
[25] M. Putinar, *Positive polynomials on compact semi-algebraic sets*, Ind. Univ. Math. J., 42 (1993), pp. 969–984.
[26] M. Raydan, *The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem*, SIAM J. Optim., 7 (1997), pp. 26–33.
[27] C.-W. Shu, *Bound-preserving high order finite volume schemes for conservation laws and convection-diffusion equations*, in Finite volumes for complex applications VIII, vol. 199 of Springer, 2017, pp. 3–14.
[28] J. F. Sturm, *Theory and algorithms of semidefinite programming*, in High performance optimization, vol. 33 of Appl. Optim., Kluwer Acad. Publ., Dordrecht, 2000, pp. 3–19.
[29] G. Szegő, *Orthogonal polynomials*, American Mathematical Society, Providence, R.I., fourth ed., 1975. American Mathematical Society, Colloquium Publications, Vol. XXIII.
[30] B. Zhou, L. Gao, and Y.-H. Dai, *Gradient methods with adaptive step-sizes*, Comput. Optim. Appl., 35 (2006), pp. 69–86.