Exploiting Sparsity for Semi-Algebraic Set Volume Computation *

Matteo Tacchi, Tillmann Weisser, Jean-Bernard Lasserre, Didier Henrion

Exploiting Sparsity for Semi-Algebraic Set Volume Computation*

M. Tacchi[1,2] T. Weisser[1] J. B. Lasserre[1,3] D. Henrion[1,4]

February 5, 2019

Abstract

We provide a systematic deterministic numerical scheme to approximate the volume (i.e. the Lebesgue measure) of a basic semi-algebraic set whose description follows a sparsity pattern. As in previous works (without sparsity), the underlying strategy is to consider an infinite-dimensional linear program on measures whose optimal value is the volume of the set. This is a particular instance of a generalized moment problem which in turn can be approximated as closely as desired by solving a hierarchy of semidefinite relaxations of increasing size. The novelty with respect to previous work is that by exploiting the sparsity pattern we can provide a sparse formulation for which the associated semidefinite relaxations are of much smaller size. In addition, we can decompose the sparse relaxations into completely decoupled subproblems of smaller size, and in some cases computations can be done in parallel. To the best of our knowledge, it is the first contribution that exploits sparsity for volume computation of semi-algebraic sets which are possibly high-dimensional and/or non-convex and/or non-connected.

1 Introduction

This paper is in the line of research concerned with computing approximations of the volume (i.e. the Lebesgue measure) of a given compact basic semi-algebraic set $\mathbf{K}$ of $\mathbb{R}^n$ neither necessarily convex nor connected. Computing or even approximating the volume of a convex body is hard theoretically and in practice as well. Even if $\mathbf{K}$ is a convex polytope, exact computation of its volume or integration over $\mathbf{K}$ is a computational challenge. Computational complexity of these problems is discussed in, e.g. [1, 2, 3]. In particular, any deterministic algorithm with polynomial-time complexity that would compute an upper bound and a lower bound on the volume cannot yield an estimate on the bound ratio better than polynomial in the dimension n. For more detail, the interested reader is referred to the discussion in [4] and to [5] for a comparison.

If one accepts randomized algorithms that fail with small probability, then the situation is more favorable. Indeed, the probabilistic approximation algorithm of [6] computes the volume to fixed arbitrary relative precision $\epsilon > 0$ in time polynomial in $1/\epsilon$. The algorithm uses approximation schemes based on rapidly mixing Markov chains and isoperimetric inequalities, see also hit-and-run algorithms described in e.g. [7, 8, 9]. So far, it seems that the recent work [10] has provided the best algorithm of this type.

[1]LAAS-CNRS, Université de Toulouse, France.
[2]Réseau de Transport d'Electricité (RTE), France.
[3]Institut de Mathématiques de Toulouse, Université de Toulouse, France.
[4]Faculty of Electrical Engineering, Czech Technical University in Prague, Czechia.
*This work was partly funded by the RTE company and by the ERC Advanced Grant Taming.

The moment approach for volume computation

In [4] a general deterministic methodology was proposed for approximating the volume of a compact basic semi-algebraic set $\mathbf{K}$, not necessarily convex or connected. It was another illustration of the versatility of the so-called *moment-SOS (sums of squares) hierarchy* developed in [11] for solving the Generalized Moment Problem (GMP) with polynomial data.

Briefly, the underlying idea is to view the volume as the optimal value of a GMP, i.e., an infinite-dimensional Linear Program (LP) on an appropriate space of finite Borel measures. Then one may approximate the value from above by solving a hierarchy of *semidefinite programming (SDP) relaxations* with associated sequence of optimal values indexed by an integer d. Monotone convergence of the bounds is guaranteed when d increases. Extensions to more general measures and possibly non-compact sets were then proposed in [12]. The order d in the hierarchy encodes the amount of information that is used, namely the number of moments of the Lebesgue measure up to degree d. It is a crucial factor for the size of the associated SDP problem. More precisely, the size grows in d^n as d increases, which so far limits its application to sets of small dimension n, typically up to 4 or 5.

In view of the growth of the size of the SDP problem with increasing order it is desirable to converge quickly towards the optimal value of the LP. However, this convergence is expected to be slow in general. One reason becomes clear when looking at the dual LP which attempts to compute a polynomial approximation (from above) of the (discontinuous) indicator function of the set $\mathbf{K}$. Hence one is then faced with a typical Gibbs effect[1], well-known in the theory of approximation. To overcome this drawback the authors in [4] have proposed to use an alternative criterion for the LP, which results in a significantly faster convergence. However in doing so, the monotonicity of the convergence (a highly desirable feature to obtain a sequence of upper bounds) is lost. In the related work [12] the author has proposed an alternative strategy which consists of strengthening the relaxations by incorporating additional linear moment constraints known to be satisfied at the optimal solution of the LP. These constraints come from a specific application of Stokes' theorem. Remarkably, adding these *Stokes constraints* results in a significantly faster convergence while keeping monotonicity.

Motivation

The measure approach for the volume computation problem is intimately linked to the use of occupation measures, in dynamical systems theory, for computing the region of attraction (ROA) of a given target set. Indeed, in [13], the problem of estimating the ROA is formulated as a GMP very similar to the volume computation problem. The idea is to maximize the volume of a set of initial conditions that yield trajectories ending in the target set after a given time.

This problem of estimating the ROA of a target set is crucial in power systems safety assessment, since the power grids must have good stability properties. Perturbations (such as short-circuits or unscheduled commutations) should be tackled before they get the system outside the ROA of the nominal operating domain. Currently the stability of power grids is estimated through numerous trajectory simulations that prove very expensive. In the wake of the energy transition, it is necessary to find new tools for estimating the stability of power systems. The conservative, geometric characterization of the region of attraction as formulated in [13] is a very promising approach for this domain of application, see [14].

As in volume computation, the main limitation of this method is that only problems of modest dimension can be handled by current solvers. Exploiting sparsity seems to be the best approach to allow scalability both in volume computation and ROA estimation. Since volume estimation is a simpler instance of the GMP than ROA estimation, we decided to address first the former problem.

[1]The Gibbs effect appears at a jump discontinuity when one approximates a piecewise continuously differentiable function with a continuous function, e.g. by its Fourier series.

In addition, volume computation with respect to a measure satisfying some conditions (e.g. compactly supported or Gaussian measure) also has applications in the fields of geometry and probability computation, which is the reason why many algorithms were already proposed for volume computation of convex polytopes and convex bodies in general.

Contribution

We design deterministic methods that provide approximations with strong asymptotic guarantees of convergence to the volume of $\mathbf{K}$. The methodology that we propose is similar in spirit to the one initially developed in [4] as described above and its extension to non-compact sets and Gaussian measures of [12]. However it is not a straightforward or direct extension of [4] or [12], and it has the following important distinguishing features:

(i) It can handle sets $\mathbf{K} \subset \mathbb{R}^n$ of potentially large dimension n provided that some sparsity pattern is present in the description of $\mathbf{K}$. This is in sharp contrast with [4].

(ii) The computation of upper and lower bounds can be decomposed into smaller independent problems of the same type, and depending on the sparsity pattern, some of the computations can even be done in parallel. This fact alone is remarkable and unexpected.

To the best of our knowledge, this is the first deterministic method for volume computation that takes benefit from a sparsity pattern in the description of $\mathbf{K}$ in the two directions of (a) decomposition into problems of smaller size and (b) parallel computation.

The key idea is to provide a new and very specific *sparse formulation* of the original problem in which one defines a set of marginal measures whose (small dimensional) support is in accordance with the sparsity pattern present in the description of the set $\mathbf{K}$. However, those marginal measures are not similar to the ones used in the sparse formulation [15] of polynomial optimization problems over the same set $\mathbf{K}$. Indeed they are not expected to satisfy the consistency condition of [15][2].

Finally, in principle, our floating point volume computation in large dimension n is faced with a crucial numerical issue. Indeed as in Monte-Carlo methods, up to rescaling, one has to include the set $\mathbf{K}$ into a box $\mathbf{B}$ of unit volume. Therefore the volume of $\mathbf{K}$ is of the order ϵ^n for some $\epsilon \in (0,1)$ and therefore far beyond machine precision as soon as n is large. To handle this critical issue we develop a *sparsity-adapted rescaling* which allows us to compute very small volumes in potentially very high dimension with good precision.

A motivating example

Consider the following set

$$\mathbf{K} := \{\mathbf{x} \in [0,1]^{100} \ : \ x_{i+1} \leq x_i^2,\ i = 1, \ldots, 99\}.$$

This is a *high-dimensional non-convex sparse semi-algebraic set*. The precise definition of a *sparse semi-algebraic set* will be given later on, but so far notice that in the description of $\mathbf{K}$ each constraint involves only 2 variables out of 100. The volume of $\mathbf{K}$ is hard to compute, but thanks to the structured description of the set we are able to prove numerically that its volume is smaller than $1.2624 \cdot 10^{-30}$ in less than 200 seconds on a standard computer.

For this we have implemented a specific version of the moment-SOS hierarchy of SDP relaxations to solve the GMP, in which we exploit the sparsity pattern of the set $\mathbf{K}$. The basic idea is to replace the original GMP that involves an unknown measure on $\mathbb{R}^{100}$ (whose SDP relaxations are hence untractable) with a GMP involving 99 measures on $\mathbb{R}^2$ (hence tractable). In addition, this new GMP can be solved either in one shot (with the 99 unknown measures) or by solving sequentially 99 GMPs involving (i) one measure on $\mathbb{R}^2$ and (ii) some data obtained from the GMP solved at previous step. Our approach can be sketched as follows.

[2] If two measures share some variables then the consistency condition requires that their respective marginals coincide.

First, we rescale the problem so that the set $\mathbf{K}$ is included in the unit box $\mathbf{B} := [0,1]^n$ on which the moments of the Lebesgue measure are easily computed.

Next, we describe the volume problem on $\mathbf{K}$ as a chain of volume subproblems on the projections on the subspaces generated by the coordinates (x_i, x_{i+1}), with a link between the i-th and $(i+1)$-th sub-problems.

Finally, in this example, as $n = 100$ and $\mathbf{K} \subset \mathbf{B}$, the volume of $\mathbf{K}$ is very small and far below standard floating point machine precision. To handle this numerical issue, we have implemented a sparsity-adapted strategy which consists of rescaling each subproblem defined on the projections of $\mathbf{K}$ to obtain intermediate values all with the same order of magnitude. Once all computations (involving quantities of the same order of magnitude) have been performed, the correct value of the volume is obtained by a reverse scaling.

The sparse formulation stems from considering some measure marginals appropriately defined according to the sparsity pattern present in the description of $\mathbf{K}$. It leads to a variety of algorithms to compute the volume of sparse semi-algebraic sets.

The outline of the paper is as follows. In Section 2.1 we describe briefly the moment-SOS hierarchy for semi-algebraic set volume approximation, as well as our notion of sparse semi-algebraic set. In Section 3, we introduce a first constructive theorem that allows to efficiently compute the volume of specific sparse sets with the hierarchy. Section 4 is dedicated to a method for accelerating the convergence of the sparse hierarchy, as well as some numerical examples. Eventually, Section 5 presents the general theorem for computing the volume of any sparse set using parallel computations, accompanied with some other examples.

2 Preliminaries

2.1 Notations and definitions

Given a closed set $\mathbf{K} \subset \mathbb{R}^n$, we denote by $\mathscr{C}(\mathbf{K})$ the space of continuous functions on $\mathbf{K}$, $\mathscr{M}(\mathbf{K})$ the space of finite signed Borel measures on $\mathbf{K}$, with $\mathscr{C}_+(\mathbf{K})$ and $\mathscr{M}_+(\mathbf{K})$ their respective cones of positive elements.

The Lebesgue measure on $\mathbb{R}^n$ is $\lambda^n(d\mathbf{x}) := dx_1 \otimes dx_2 \otimes \cdots dx_n = d\mathbf{x}$, and its restriction to a set $\mathbf{K} \subset \mathbb{R}^n$ is $\lambda_{\mathbf{K}} := \mathbb{1}_{\mathbf{K}} \lambda^n$, where $\mathbb{1}_{\mathbf{K}}$ denotes the indicator function equal to 1 in $\mathbf{K}$ and zero outside. In this paper, we focus on computing the volume or Lebesgue measure of $\mathbf{K}$, that we denote by $\operatorname{vol} \mathbf{K}$ or $\lambda^n(\mathbf{K})$ or $\lambda_{\mathbf{K}}(\mathbb{R}^n)$.

Given a Euclidean space $\mathbf{X}$ and a subspace $\mathbf{X}_i \subset \mathbf{X}$, the orthogonal projection map from $\mathbf{X}$ to $\mathbf{X}_i$ is denoted by $\pi_{\mathbf{X}_i}$. The m-dimensional subspace spanned by coordinates $x_{i_1}, \ldots, x_{i_m}$ is denoted $\langle x_{i_1}, \ldots, x_{i_m} \rangle$. Given a measure $\mu \in \mathscr{M}(\mathbf{X})$, its marginal with respect to $\mathbf{X}_i$ is denoted by $\mu^{\mathbf{X}_i} \in \mathscr{M}(\mathbf{X}_i)$. It is equal to the image or push-forward measure of μ through the map $\pi_{\mathbf{X}_i}$.

Let $\mathbb{R}[\mathbf{x}]$ be the ring of polynomials in the variables $\mathbf{x} = (x_1, \ldots, x_n)$ and let $\mathbb{R}[\mathbf{x}]_d$ be the vector space of polynomials of degree at most d, whose dimension is $s(d) := \binom{n+d}{n}$. For every $d \in \mathbb{N}$, let $\mathbb{N}^n_d := \{\alpha \in \mathbb{N}^n : |\alpha| (= \sum_i \alpha_i) \leq d\}$, and let $\mathbf{v}_d(\mathbf{x}) = (\mathbf{x}^\alpha)_{\alpha \in \mathbb{N}^n_d}$ be the vector of monomials of the canonical basis $(\mathbf{x}^\alpha)_{\alpha \in \mathbb{N}^n_d}$ of $\mathbb{R}[\mathbf{x}]_d$. A polynomial $p \in \mathbb{R}[\mathbf{x}]_d$ is written as

$$\mathbf{x} \mapsto p(\mathbf{x}) \;=\; \sum_{\alpha \in \mathbb{N}^n} p_\alpha \, \mathbf{x}^\alpha \;=\; \mathbf{p} \cdot \mathbf{v}_d(\mathbf{x})$$

for some vector of coefficients $\mathbf{p} = (p_\alpha)_{\alpha \in \mathbb{N}^n_d} \in \mathbb{R}^{s(d)}$, where the dot denotes the Euclidean inner product.

We say that $\mu \in \mathscr{M}(\mathbb{R}^n)$ is a representing measure of the sequence $\mathbf{m} = (m_\alpha)_{\alpha \in \mathbb{N}^n} \subset \mathbb{R}$ whenever

$$m_\alpha = \int \mathbf{x}^\alpha \, d\mu(\mathbf{x})$$

for all $\alpha \in \mathbb{N}^n$. Given a sequence $\mathbf{m} = (m_\alpha)_{\alpha \in \mathbb{N}^n}$, let $L_{\mathbf{m}} : \mathbb{R}[\mathbf{x}] \to \mathbb{R}$ be the linear functional

$$f\left(= \sum_\alpha f_\alpha \, \mathbf{x}^\alpha\right) \mapsto L_{\mathbf{m}}(f) := \sum_\alpha f_\alpha \, m_\alpha.$$

Given a sequence $\mathbf{m} = (m_\alpha)_{\alpha \in \mathbb{N}^n}$, and a polynomial $g := \sum_\gamma g_\gamma \, \mathbf{x}^\gamma \in \mathbb{R}[\mathbf{x}]$, the localizing moment matrix of order d associated with $\mathbf{m}$ and g is the real symmetric matrix $\mathbf{M}_d(g\,\mathbf{m})$ of size $s(d)$ with rows and columns indexed in $\mathbb{N}^n_d$ and with entries

$$\begin{aligned} \mathbf{M}_d(g\,\mathbf{m})(\alpha,\beta) &:= L_{\mathbf{m}}(g(\mathbf{x})\,\mathbf{x}^{\alpha+\beta}) \\ &= \sum_\gamma g_\gamma \, m_{\alpha+\beta+\gamma}, \quad \alpha,\beta \in \mathbb{N}^n_d. \end{aligned}$$

When $g \equiv 1$, the localizing moment matrix $\mathbf{M}_d(\mathbf{m})$ is called simply the moment matrix.

2.2 The Moment-SOS hierarchy for volume computation

Let $\mathbf{B} := [0,1]^n \subset \mathbf{X} := \mathbb{R}^n$ be the n-dimensional unit box, and let $\mathbf{K} \subset \mathbf{B}$ be a closed basic semialgebraic set defined by

$$\mathbf{K} := \{\mathbf{x} \in \mathbf{X} : \; g_i(\mathbf{x}) \geq 0, \; i = 1, \ldots, m\} = \{\mathbf{x} \in \mathbf{X} : \; \mathbf{g}(\mathbf{x}) \geq 0\}$$

where $\mathbf{g} = (g_i)_{i=1,\ldots,m} \in \mathbb{R}[\mathbf{x}]^m$ and the rightmost vector inequality is meant entrywise. As in [4] consider the infinite-dimensional linear program (LP) on measures

$$\begin{array}{rl} \max\limits_{\mu,\, \hat{\mu} \in \mathscr{M}_+(\mathbf{B})} & \int d\mu \\ \text{s.t.} & \mu + \hat{\mu} = \lambda_{\mathbf{B}} \\ & \text{spt}\, \mu \subset \mathbf{K} \\ & \text{spt}\, \hat{\mu} \subset \mathbf{B}. \end{array} \tag{1}$$

Its value is equal to $\text{vol}\,\mathbf{K}$ and the measures $\mu^* = \lambda_{\mathbf{K}}$, $\hat{\mu}^* := \lambda_{\mathbf{B}\setminus\mathbf{K}}$ are the unique optimal solutions of (1). The dual of (1) is the infinite-dimensional LP on continuous functions

$$\begin{array}{rl} \inf\limits_{v \in \mathscr{C}_+(\mathbf{B})} & \int v \, d\lambda_{\mathbf{B}} \\ \text{s.t.} & v \geq \mathbb{1}_{\mathbf{K}}. \end{array} \tag{2}$$

It turns out that there is no duality gap between (1) and (2), i.e., they both have the same optimal value. Notice that a minimizing sequence of (2) approximates the indicator function $\mathbb{1}_{\mathbf{K}}$ from above by polynomials of increasing degrees.

The LP (1) is a particular and simple instance of the Generalized Moment Problem (GMP). As described in [4] one may approximate its optimal value as closely as desired by using the following key result in [11]. Given an infinite dimensional LP on measures, one can construct a hierarchy of finite dimensional semidefinite programs[3] (SDP) whose associated sequence of optimal values converges monotonically to the optimal value of the original infinite dimensional LP. The basic idea is to represent a measure with the sequence $\mathbf{m}$ of its moments, and to formulate finite dimensional SDPs on truncations of the sequence $\mathbf{m}$. When this strategy is applied to LP (1), the step d of the moment-SOS hierarchy consists of solving the SDP relaxation

$$\begin{array}{rll} \mathbf{P}_d : & \max\limits_{\mathbf{m},\, \widehat{\mathbf{m}} \in \mathbb{R}^{s(d)}} & m_0 \\ & \text{s.t.} & m_\alpha + \widehat{m}_\alpha = \int_{\mathbf{B}} \mathbf{x}^\alpha \, d\mathbf{x}, \quad \alpha \in \mathbb{N}^n_{2d} \\ & & \mathbf{M}_d(\mathbf{m}) \succeq 0, \; \mathbf{M}_d(\widehat{\mathbf{m}}) \succeq 0 \\ & & \mathbf{M}_{d-d_i}(g_i\,\mathbf{m}) \succeq 0, \quad i = 1, \ldots, m \end{array} \tag{3}$$

[3] A semidefinite program is a convex conic optimization problem that can be solved numerically efficiently, e.g. by using interior point methods.

where $\mathbf{m} = (m_\alpha)_{\alpha \in \mathbb{N}^n_{2d}}$, $\widehat{\mathbf{m}} = (\widehat{m}_\alpha)_{\alpha \in \mathbb{N}^n_{2d}}$, and $d_i = \lceil (\deg g_i)/2 \rceil$, $i = 1, \ldots, m$.

The sequence of SDP problems $(\mathbf{P}_d)_{d \in \mathbb{N}}$ indexed by the relaxation order d is a hierarchy in the sense that its sequence of values converges monotonically from above to $\operatorname{vol} \mathbf{K}$ when d increases. Each SDP relaxation $\mathbf{P}_d$ has a dual formulated in terms of sums of squares (SOS) of polynomials, which leads to a dual SOS hierarchy, whence the name *moment-SOS hierarchy*. The basic moment-SOS hierarchy can be modeled using the GloptiPoly Matlab toolbox [16] and solved using any SDP solver, e.g. SeDuMi or Mosek. For more details on the moment-SOS hierarchy and some of its applications, the interested reader is referred to [11].

2.3 The sparsity pattern and its graph representation

Definition 1. A *sparse semi-algebraic set* has a description

$$\mathbf{K} := \{\mathbf{x} \in \mathbf{X} \ : \ \mathbf{g}_i(\pi_{\mathbf{X}_i}(\mathbf{x})) \geq 0, \ i = 1, \ldots, m\}$$

where each $\mathbf{g}_i$ is a vector of polynomials (inequalities are meant entrywise) and $\mathbf{X}_1, \ldots, \mathbf{X}_m$ are Euclidean vector spaces such that $\sum_{i=1}^m \mathbf{X}_i = \mathbf{X} := \mathbb{R}^n$.

A simple example of a sparse semi-algebraic set is

$$\mathbf{K} := \{\mathbf{x} = (x_1, x_2, x_3, x_4) \in \mathbb{R}^4 \ : \ \mathbf{g}_1(x_1, x_2) \geq 0, \ \mathbf{g}_2(x_2, x_3) \geq 0, \ \mathbf{g}_3(x_3, x_4) \geq 0\} \tag{4}$$

for $\mathbf{X} = \mathbb{R}^4$, $\mathbf{X}_1 = \langle x_1, x_2 \rangle$, $\mathbf{X}_2 = \langle x_2, x_3 \rangle$, $\mathbf{X}_3 = \langle x_3, x_4 \rangle$ and the projection maps are $\pi_{\mathbf{X}_1}(\mathbf{x}) = (x_1, x_2)$, $\pi_{\mathbf{X}_2}(\mathbf{x}) = (x_2, x_3)$, $\pi_{\mathbf{X}_3}(\mathbf{x}) = (x_3, x_4)$.

Our methodology is based on the classical theory of clique trees. We first construct a graph associated with the structure of the problem. For each variable in the definition of $\mathbf{K}$, we define a vertex of the graph. Then, two variables that interact within the same polynomial in the definition of $\mathbf{K}$ correspond to vertices connected by an edge. Up to a chordal extension (which is equivalent to slightly weakening the sparsity pattern), we suppose that the associated graph is *chordal* (i.e. every cycle of length greater than 3 has a chord, i.e. an edge linking two nonconsecutive vertices). Then we construct the *cliques* of the graph (a clique C is a subset of vertices such that every vertex of C is connected to all the other vertices of C). Figure 1 illustrates this constuction for the sparse set (4), the vertices are denoted by x_i and our cliques are denoted by C_j.

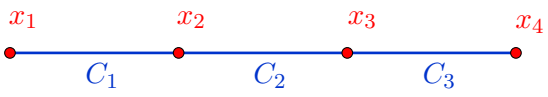


Figure 1: Graph associated to the sparse set (4).

Then, we construct a *clique tree* which is instrumental to the computer implementation. It is proved in [17] that if the graph is chordal, then its cliques can be organized within a tree satisfying the clique intersection property: for two cliques C and C' the intersection $C \cap C'$ is contained in every clique on the path from C to C'. Figure 2 represents the clique tree associated to the sparse set (4).

For a slightly more complicated illustration, consider the sparse set

$$\mathbf{K} := \{\mathbf{x} \in \mathbb{R}^6 \ : \ \mathbf{g}_1(x_1, x_2) \geq 0, \ \mathbf{g}_2(x_2, x_3, x_4) \geq 0, \ \mathbf{g}_3(x_3, x_5) \geq 0, \ \mathbf{g}_4(x_4, x_6) \geq 0\} \tag{5}$$

whose graph is represented on Figure 3 and whose clique tree is represented on Figure 4. The clique tree of Figure 2 is called *linear* because all cliques form a single chain (i.e. they are in a sequence) with no branching. In contrast, the clique tree of Figure 4 is called *branched*.

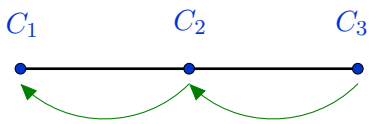


Figure 2: Linear clique tree associated to the sparse set (4).

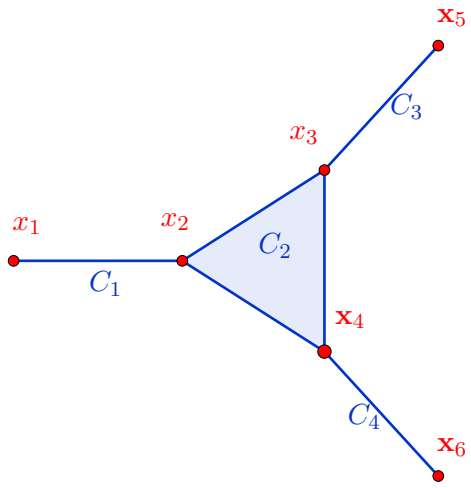


Figure 3: Graph associated to the sparse set (5).

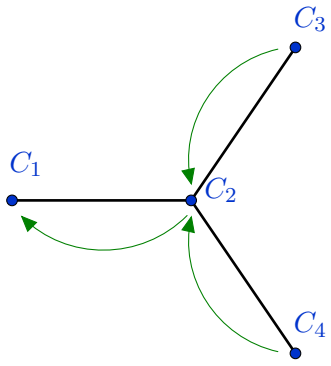


Figure 4: Branched clique tree associated to the sparse set (5).

Our method consists of conveniently rooting the clique tree and splitting the volume computation problem into lower-dimensional subproblems that are in correspondence with the cliques of the graph. The subproblem associated with a clique C takes as only input the solutions of the subproblems associated with the children of C in the clique tree. This way, one can compute in parallel the solutions of all the subproblems of a given generation, and then use their results to solve the subproblems of the parent generation. This is the meaning of the arrows in Figures 2 and 4. The volume of $\mathbf{K}$ is the optimal value of the (last) sub-problem associated with the root C_1 of the tree.

3 Linear sparse volume computation

In this section we describe the method in the prototype case of linear clique trees. The more general case of branched clique trees is treated later on.

3.1 An illustrative example: the bicylinder

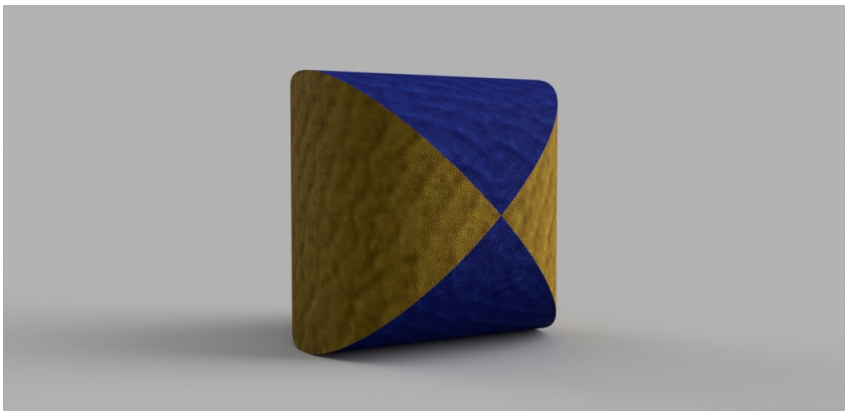


Figure 5: A representation of the bicylinder produced by the AutoDesk Fusion 360 software.

Before describing the methodology in the general case, we briefly explain the general underlying idea on a simple illustrative example. Consider the sparse semi-algebraic set

$$\mathbf{K} := \big\{\mathbf{x} \in \mathbb{R}^3 \ : \ \mathbf{g}_1(x_1, x_2) := 1 - x_1^2 - x_2^2 \geq 0,\ \mathbf{g}_2(x_2, x_3) := 1 - x_2^2 - x_3^2 \geq 0\big\} \tag{6}$$

modelling the intersection of two truncated cylinders $\mathbf{K}_1 := \{\mathbf{x} \in \mathbb{R}^3 \ : \ x_1^2 + x_2^2 \leq 1\}$ and $\mathbf{K}_2 := \{\mathbf{x} \in \mathbb{R}^3 \ : \ x_2^2 + x_3^2 \leq 1\}$, see Figure 3.1. The subspaces are $\mathbf{X}_1 = \langle x_1, x_2\rangle$ and $\mathbf{X}_2 = \langle x_2, x_3\rangle$ and the projection maps are $\pi_{\mathbf{X}_1}(\mathbf{x}) = (x_1, x_2)$ and $\pi_{\mathbf{X}_2}(\mathbf{x}) = (x_2, x_3)$. Let $\mathbf{U}_i := \pi_{\mathbf{X}_i}(\mathbf{K}_i)$ for $i = 1, 2$.

Following [4], computing $\operatorname{vol}\mathbf{K}$ is equivalent to solving the infinite-dimensional LP (1). Next observe that in the description (6) of $\mathbf{K}$ there is no direct interaction between variables x_1 and x_3, but this is neither exploited in the LP formulation (1) nor in the SDP relaxations (3) to solve (1). To exploit this sparsity pattern we propose the following alternative formulation

$$\begin{array}{rcl} \operatorname{vol}\mathbf{K} & = & \displaystyle\max_{\substack{\mu_i \in \mathscr{M}_+(\mathbf{X}_i)\\ i=1,2}} \int_{\mathbb{R}^2} d\mu_1 \\ & \text{s.t.} & \mu_2 \leq \lambda \otimes \lambda \\ & & \mu_1 \leq \lambda \otimes \mu_2^{\langle x_2\rangle} \\ & & \operatorname{spt}\mu_1 \subset \mathbf{U}_1,\ \operatorname{spt}\mu_2 \subset \mathbf{U}_2 \end{array} \tag{7}$$

where $\mu_2^{\langle x_2\rangle}$ denotes the marginal of μ_2 in the variable x_2.

In the sparse case, the basic idea behind our reformulation of the volume problem is as follows. We are interested in $\operatorname{vol}\mathbf{K}$. However, as the marginal of a measure has the same mass as the measure itself, instead of looking for the full measure μ in problem (1), we only look for its marginal on $\mathbf{X}_1$.

This marginal $\mu^{\mathbf{X}_1}$ is modeled by μ_1 in (7). In order to compute it, we need some additional information on μ captured by the measure μ_2 in (7). The unique optimal solution μ of (1) is

$$\begin{array}{rcl} d\mu(\mathbf{x}) & = & d\lambda_{\mathbf{K}}(\mathbf{x}) \\ & = & \mathbb{1}_{\mathbf{U}_1}(x_1, x_2)\ \mathbb{1}_{\mathbf{U}_2}(x_2, x_3)\ d\mathbf{x} \end{array}$$

and therefore its marginal $\mu_1 := \mu^{\mathbf{X}_1}$ on (x_1, x_2) is

$$\begin{aligned} d\mu_1(x_1,x_2) &= \int_0^1 d\mu(x_1,x_2,x_3) \\ &= \mathbb{1}_{\mathbf{U}_1}(x_1,x_2)\, dx_1 \underbrace{\left(\int_0^1 \mathbb{1}_{\mathbf{U}_2}(x_2,x_3)\, dx_3\right)}_{d\mu_2^{\langle x_2\rangle}(x_2)} dx_2 \end{aligned} \tag{8}$$

where

$$d\mu_2(x_2,x_3) = d\lambda_{\mathbf{U}_2}(x_2,x_3). \tag{9}$$

What is the gain in solving (7) when compared to solving (1) ? Observe that in (7) we have two unknown measures μ_1 and μ_2 on $\mathbb{R}^2$, instead of a single measure μ on $\mathbb{R}^3$ in (1). In the resulting SDP relaxations associated with (7) this translates into SDP constraints of potentially much smaller size. For instance, and to fix ideas, for the same relaxation degree d:

- The SDP relaxation $\mathbf{P}_d$ associated with (1) contains a moment matrix (associated with μ in (1)) of size $\binom{3+d}{d}$;
- The SDP relaxation $\mathbf{P}_d$ associated with (7) contains two moment matrices, one associated with μ_1 of size $\binom{2+d}{d}$, and one associated with μ_2 of size $\binom{2+d}{d}$, where μ_1 and μ_2 are as in (7).

As the size of those matrices is the crucial parameter for all SDP solvers, one can immediately appreciate the computational gain that can be expected from the formulation (7) versus the formulation (1) when the dimension is high or the relaxation order increases. Next it is not difficult to extrapolate that the gain can be even more impressive in the case where the sparsity pattern is of the form

$$\mathbf{K} = \{(\mathbf{x}_0,\ldots\mathbf{x}_m) \in \mathbf{X} :\ g_i(\mathbf{x}_{i-1},\mathbf{x}_i) \geq 0, \quad i = 1,\ldots,m\,\}, \tag{10}$$

with $\mathbf{x}_i \in \mathbb{R}^{n_i}$ and $n_i \ll n$ for $i = 0,\ldots,m$. In fact, it is straightforward to define examples of sets $\mathbf{K}$ of the form (10) where the first SDP relaxation associated with the original full LP formulation (1) cannot be even implemented on state-of-the-art computers, whereas the SDP relaxations associated with a generalization of the sparse LP formulation (10) can be easily implemented, at least for reasonable values of d.

3.2 Linear computation theorem

Let

$$\mathbf{K}_i := \{\mathbf{x} \in \mathbf{X} \ :\ \mathbf{g}_i(\pi_{\mathbf{X}_i}(\mathbf{x})) \geq 0\}$$

with $\mathbf{g}_i \in \mathbb{R}[\mathbf{x}_i]^{p_i}$, so that our sparse semi-algebraic set can be written

$$\mathbf{K} = \bigcap_{i=1}^m \mathbf{K}_i.$$

Moreover, let

$$\mathbf{U}_i := \{\mathbf{x}_i \in \mathbf{X}_i \ :\ \mathbf{g}_i(\mathbf{x}_i) \geq 0\} = \pi_{\mathbf{X}_i}(\mathbf{K}_i)$$

and let

$$\mathbf{Y}_i := \mathbf{X}_i \cap \mathbf{X}_{i+1}^{\perp}$$

be a subspace of dimension n_i for $i = 1,\ldots,m-1$ with $\mathbf{Y}_m = \mathbf{X}_m$. The superscript $\perp$ denotes the orthogonal complement.

Assumption 2. *For all* $i \in \{2, \dots, m\}$ *it holds* $\mathbf{X}_i \cap \sum_{j=1}^{i-1} \mathbf{X}_j \neq \{0\}$.

Assumption 3. *For all* $i \in \{2, \dots, m\}$ *it holds* $\mathbf{X}_i \cap \sum_{j=1}^{i-1} \mathbf{X}_j \subset \mathbf{X}_{i-1}$.

If Assumption 2 is violated then $\mathbf{K}$ can be decomposed as a Cartesian product, and one should just apply our methodology to each one of its factors. Assumption 3 ensures that the associated clique tree is linear.

Theorem 4. *If Assumptions 2 and 3 hold, then* $\operatorname{vol} \mathbf{K}$ *is the value of the LP problem*

$$\max_{\substack{\mu_i \in \mathscr{M}_+(\mathbf{X}_i) \\ i=1,\dots,m}} \int d\mu_1 \tag{11}$$

$$s.t. \quad \mu_i \leq \mu_{i+1}^{\mathbf{X}_i \cap \mathbf{X}_{i+1}} \otimes \lambda^{n_i} \quad i = 1, \dots, m-1 \tag{12}$$

$$\mu_m \leq \lambda^{n_m} \tag{13}$$

$$\operatorname{spt} \mu_i \subset \mathbf{U}_i \qquad i = 1, \dots, m. \tag{14}$$

Proof. Let us first prove that the value of the LP is larger than $\operatorname{vol} \mathbf{K}$. For $i = 1, \dots, m$, let $\mathbf{Z}_i := \mathbf{X}_i^\perp \cap \sum_{j=i+1}^{m} \mathbf{X}_j$ so that $\sum_{j=i}^{m} \mathbf{X}_j = \mathbf{X}_i \oplus \mathbf{Z}_i$. For $\mathbf{x}_i \in \mathbf{X}_i$ define

$$d\mu_i(\mathbf{x}_i) := \mathbb{1}_{\mathbf{U}_i}(\mathbf{x}_i) \left(\int_{\mathbf{Z}_i} \prod_{j=i+1}^{m} \mathbb{1}_{\mathbf{U}_j} \circ \pi_{\mathbf{X}_j}(\mathbf{x}_i + \mathbf{z}_i) \, d\mathbf{z}_i \right) d\mathbf{x}_i. \tag{15}$$

By construction $\mu_i \in \mathscr{M}_+(\mathbf{X}_i)$ and constraints (14) are enforced. In addition, one can check that, if $\mathbf{x}_{i,i+1} \in \mathbf{X}_i \cap \mathbf{X}_{i+1}$, then

$$\begin{aligned}
d\mu_{i+1}^{\mathbf{X}_i \cap \mathbf{X}_{i+1}}(\mathbf{x}_{i,i+1}) &\overset{\text{def}}{=} \int_{\mathbf{y}_{i,i+1} \in \mathbf{X}_i^\perp \cap \mathbf{X}_{i+1}} d\mu_{i+1}(\mathbf{x}_{i,i+1} + \mathbf{y}_{i,i+1}) \\
&\overset{(15)}{=} \left(\int_{\mathbf{X}_i^\perp \cap \mathbf{X}_{i+1}} \mathbb{1}_{\mathbf{U}_{i+1}}(\mathbf{x}_{i,i+1} + \mathbf{y}_{i,i+1}) \right. \\
&\qquad \left. \left(\int_{\mathbf{Z}_{i+1}} \prod_{j=i+2}^{m} \mathbb{1}_{\mathbf{U}_j} \circ \pi_{\mathbf{X}_j}(\mathbf{x}_{i,i+1} + \mathbf{y}_{i,i+1} + \mathbf{z}_{i+1}) \, d\mathbf{z}_{i+1} \right) d\mathbf{y}_{i,i+1} \right) d\mathbf{x}_{i,i+1} \\
&= \left(\int_{\mathbf{Z}_i} \prod_{j=i+1}^{m} \mathbb{1}_{\mathbf{U}_j} \circ \pi_{\mathbf{X}_j}(\mathbf{x}_{i,i+1} + \mathbf{z}_i) \, d\mathbf{z}_i \right) d\mathbf{x}_{i,i+1}
\end{aligned}$$

since $(\mathbf{X}_i^\perp \cap \mathbf{X}_{i+1}) \oplus \mathbf{Z}_{i+1} = \mathbf{Z}_i$.

Thus, constraints (12) are satisfied. Moreover, they are saturated on $\mathbf{U}_i$. Eventually, one has

$$\mathbf{X} = \mathbf{X}_1 \oplus \mathbf{Z}_1$$

and thus

$$\begin{aligned}
\int_{\mathbf{X}_1} d\mu_1 &= \int_{\mathbf{X}_1} \mathbb{1}_{\mathbf{U}_1}(\mathbf{x}_1) \left(\int_{\mathbf{Z}_1} \prod_{j=2}^{m} \mathbb{1}_{\mathbf{U}_j} \circ \pi_{\mathbf{X}_j}(\mathbf{x}_1 + \mathbf{z}_1) \, d\mathbf{z}_1 \right) d\mathbf{x}_1 \\
&= \int_{\mathbf{X}} \left(\prod_{j=1}^{m} \mathbb{1}_{\mathbf{U}_j} \circ \pi_{\mathbf{X}_j}(\mathbf{x}) \right) d\mathbf{x} \\
&= \int_{\mathbf{X}} \mathbb{1}_{\mathbf{K}}(\mathbf{x}) \, d\mathbf{x} \\
&= \operatorname{vol} \mathbf{K},
\end{aligned}$$

that is, we have just proved that the value of the LP is larger than or equal to vol $\mathbf{K}$.

To prove the converse inequality, observe that our previous choice $\mu_1, \ldots, \mu_m$ saturates the constraints (12) while enforcing the constraints (14). Any other feasible solution $\tilde{\mu}_1, \ldots, \tilde{\mu}_m$ directly satisfies the inequality $\tilde{\mu}_i \leq \mu_i$. In particular, $\tilde{\mu}_1 \leq \mu_1$ and thus

$$\int d\tilde{\mu}_1(\mathbf{x}_1) \leq \text{vol}\,\mathbf{K}.$$

□

Remark 5. The dual of the LP problem of Theorem 4 is the LP problem

$$\begin{array}{rll} \inf\limits_{\substack{v_i \in \mathscr{C}_+(\mathbf{X}_i) \\ i=1,\ldots,m}} & \displaystyle\int_{\mathbf{X}_m} v_m(\mathbf{x}_m)\, d\mathbf{x}_m & \\ \text{s.t.} & v_1(\mathbf{x}_1) \geq 1 & \forall\, \mathbf{x}_1 \in \mathbf{U}_1 \\ & v_{i+1}(\mathbf{x}_{i+1}) \geq \displaystyle\int_{\mathbf{Y}_i} v_i(\mathbf{y}_i, \pi_{\mathbf{X}_i}(\mathbf{x}_{i+1}))\, d\mathbf{y}_i & \forall\, \mathbf{x}_{i+1} \in \mathbf{U}_{i+1},\ i = 1, \ldots, m-1. \end{array}$$

According to [11], there is no duality gap, i.e. the value of the dual is vol $\mathbf{K}$. For example, in the case of the bicylinder treated in Section 3.1, the dual reads:

$$\begin{array}{rll} \inf\limits_{v_1,\, v_2 \in \mathscr{C}_+([0,1]^2)} & \displaystyle\int_0^1 \int_0^1 v_2(x_2, x_3)\, dx_2 dx_3 & \\ \text{s.t.} & v_1(x_1, x_2) \geq 1 & \forall\, (x_1, x_2) \in \mathbf{U}_1 \\ & v_2(x_2, x_3) \geq \displaystyle\int_0^1 v_1(x_1, x_2)\, dx_1 & \forall\, (x_2, x_3) \in \mathbf{U}_2. \end{array}$$

Thus, if $(v_1^k, v_2^k)_{k \in \mathbb{N}}$ is a minimizing sequence for this dual LP, then the sets

$$\mathbf{A}^k := \left\{ (x_1, x_2, x_3) \in [0,1]^3 \ : \ \frac{v_1^k(x_1, x_2)\, v_2^k(x_2, x_3)}{\int_0^1 v_1^k(x, x_2) dx} \geq 1 \right\}$$

are outer approximations of the set $\mathbf{K}$ and the sequence $(\text{vol}\,\mathbf{A}^k)_k$ decreases to vol $\mathbf{K}$. Similar statements can be made for the general dual problem.

Remark 6. The LP (11)-(14) is formulated as a single problem on m unknown measures. However, it is possible to split it in small chained subproblems to be solved in sequence. Each subproblem is associated with a clique (in the linear tree clique graph) and it takes as input the results of the subproblem associated with its parent clique. This way, the sparse volume computation is split into m linked low-dimension problems and solved sequentially. This may prove useful when m is large because when solving the SDP relaxations associated with the single LP (11)-(14), the SDP solver may encounter difficulties in handling a high number of measures simultaneously. It should be easier to sequentially solve a high number of low-dimensional problems with only one unknown measure.

3.3 Lower bounds for the volume

As explained in the introduction, the hierarchy of SDP relaxations associated with our infinite-dimensional LP provides us with a sequence of upper bounds on vol $\mathbf{K}$. One may also be interested in computing lower bounds on vol $\mathbf{K}$. In principle it suffices to apply the same methodology and approximate from above the volume of $\mathbf{B} \setminus \mathbf{K}$ since $\mathbf{K}$ is included in the unit box $\mathbf{B}$. However, it is unclear whether $\mathbf{B} \setminus \mathbf{K}$ has also a sparse description. We show that this is actually the case and so one may exploit sparsity to compute lower bounds although it is more technical. The following result is a consequence of Theorem 4:

Corollary 7. *If* $\mathbf{K}$ *is sparse, then* $\widehat{\mathbf{K}} := \mathbf{B} \setminus \mathbf{K}$ *is sparse as well, and* $\operatorname{vol} \widehat{\mathbf{K}}$ *is the value of the LP problem*

$$\begin{array}{ll} \max\limits_{\substack{\mu_{i,j} \in \mathscr{M}_+(\mathbf{X}_j) \\ 1 \le i \le j \le m}} & \sum\limits_{j=1}^{m} \int_{\mathbf{X}_j} d\mu_{1,j} \\ s.t. & \mu_{j,j} \le \lambda^{m_j} \\ & \mu_{i,j} \le \mu_{i+1,j}^{\mathbf{X}_i \cap \mathbf{X}_{i+1}} \otimes \lambda^{n_i} \quad i = 1, \ldots, j-1 \\ & \operatorname{spt} \mu_{i,j} \subset \mathbf{U}_i \qquad\qquad\quad i = 1, \ldots j-1 \\ & \operatorname{spt} \mu_{j,j} \subset \operatorname{cl} \widehat{\mathbf{U}}_j \end{array}$$

where $m_j := \dim \mathbf{X}_j$, $n_i := \dim \mathbf{X}_{i+1}^{\perp} \cap \mathbf{X}_i$, $\widehat{\mathbf{U}}_j := [0,1]^{m_j} \setminus \mathbf{U}_j$ *is open and* $\operatorname{cl} \widehat{\mathbf{U}}_j$ *denotes its closure*[4]*.*

Proof. The following description

$$\widehat{\mathbf{K}} = \bigsqcup_{j=1}^{m} \left[\bigcap_{i=1}^{j-1} \mathbf{U}_i \cap \widehat{\mathbf{U}}_j \right],$$

where $\bigsqcup$ stands for disjoint union, is sparse. Indeed the description of the basic semi-algebraic set

$$\mathbf{V}_j := \bigcap_{i=1}^{j-1} \mathbf{U}_i \cap \widehat{\mathbf{U}}_j$$

is sparse. In addition, by σ-additivity of the Lebesgue measure, it holds

$$\operatorname{vol} \widehat{\mathbf{K}} = \sum_{j=1}^{m} \operatorname{vol} \mathbf{V}_j.$$

Finally, by using Theorem 4 we conclude that $\operatorname{vol} \mathbf{V}_j$ is the value of LP consisting of maximizing $\int_{\mathbf{X}_j} d\mu_{1,j}$ subject to the same constraints as in the above LP problem. Summing up yields the correct value. □

4 Accelerating convergence

As already mentioned, the convergence of the standard SDP relaxations (3) for solving the GMP (1) is expected to be slow in general. To cope with this issue we introduce additional linear constraints that are redundant for the infinite dimensional GMP, and that are helpful to accelerate the convergence of the SDP relaxations. These constraints come from a specific application of Stokes' theorem.

4.1 Full Stokes constraints

We first focus on the full formulation (1). We know that the optimal measure of our infinite dimensional LP is $\mu = \lambda_{\mathbf{K}}$. Thus, one can put additional constraints in the hierarchy in order to give more information on the target sequence of moments, without increasing the dimension of the SDP relaxation. To keep the optimal value unchanged, such constraints should be redundant in the infinite dimensional LP. Ideally, we would like to characterize the whole set of polynomials p such that

$$\int_{\mathbf{K}} p(\mathbf{x})\, d\mu(\mathbf{x}) = 0.$$

[4]This is necessary since the support of a measure is a closed set by definition.

Indeed, given any such polynomial p, the moments $\mathbf{m}$ of μ necessarily satisfy the linear constraint $L_{\mathbf{m}}(p) = 0$. However, for a general semi-algebraic set $\mathbf{K}$, we do not have an explicit description of this set of polynomials. Nevertheless, we can generate many of them, and hence improve convergence of the SDP relaxations significantly, as it was done originally in [18] in another context. Let us explain how we generate these linear moment constraints.

We recall that Stokes' theorem states that if $\mathbf{O}$ is an open set of $\mathbb{R}^n$ with a boundary $\partial\mathbf{O}$ smooth almost everywhere, and ω is a $(n-1)$-differential form on $\mathbb{R}^n$, then one has

$$\int_{\partial\mathbf{O}} \omega = \int_{\mathbf{O}} d\omega.$$

A corollary to this theorem is obtained by choosing $\omega(\mathbf{x}) = \mathbf{h}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x})\ \sigma(d\mathbf{x})$, where the dot denotes the inner product between $\mathbf{h}$, a smooth (ideally polynomial) vector field, and $\mathbf{n}$ the outward pointing unit vector orthogonal to the boundary $\partial\mathbf{O}$, and σ denotes the $(n-1)$-dimensional Hausdorff measure on $\partial\mathbf{O}$. In this case, one obtains the Gauss formula [19]

$$\int_{\partial\mathbf{O}} \mathbf{h}(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x})\ \sigma(d\mathbf{x}) = \int_{\mathbf{O}} \operatorname{div} \mathbf{h}(\mathbf{x})\ d\mathbf{x}.$$

Choosing $\mathbf{h}(\mathbf{x}) = h(\mathbf{x})\,\mathbf{e}_i$, where h is a smooth function (ideally a polynomial) and $\mathbf{e}_i$ is the i-th vector of the canonical basis of $\mathbb{R}^n$, one obtains the following vector equality

$$\int_{\partial\mathbf{O}} h(\mathbf{x})\ \mathbf{n}(\mathbf{x})\ d\sigma(\mathbf{x}) = \int_{\mathbf{O}} \mathbf{grad}\, h(\mathbf{x})\ d\mathbf{x}.$$

Then, if we choose $\mathbf{O} = \mathbf{K} \setminus \partial\mathbf{K}$ and a polynomial h vanishing on $\partial\mathbf{K}$, the vector constraint

$$\int_{\mathbf{K}} \mathbf{grad}\, h(\mathbf{x})\ d\mathbf{x} = 0$$

is automatically satisfied and it can be added to the optimization problem (1) without changing its optimal value. Such constraints are redundant in the infinite-dimensional LP formulation (1) but not in the SDP relaxations (3). It has been numerically shown in [18] that adding these constraints dramatically increases the convergence rate of the hierarchy of SDP relaxations.

These so-called *Stokes constraints* can be added to the formulation of problem (1) to yield

$$\begin{array}{ll} \max\limits_{\mu,\,\hat{\mu}\in\mathscr{M}_+(\mathbf{B})} & \int d\mu \\ \text{s.t.} & \mu + \hat{\mu} = \lambda_{\mathbf{B}} \\ & (\mathbf{grad}\, h)\mu = \mathbf{grad}\,(h\mu) \\ & \operatorname{spt}\mu \subset \mathbf{K} \\ & \operatorname{spt}\hat{\mu} \subset \mathbf{B} \end{array} \tag{16}$$

where h is any polynomial vanishing on the boundary of $\mathbf{K}$, without changing its value $\operatorname{vol}\mathbf{K}$. The vector constraint $(\mathbf{grad}\, h)\mu = \mathbf{grad}\,(h\mu)$ should be understood in the sense of distributions, i.e. for all test functions $v \in \mathscr{C}^1(\mathbf{B})$ it holds

$$\int (\mathbf{grad}\, h) v\ d\mu = -\int (\mathbf{grad}\, v) h\ d\mu$$

or equivalently

$$\int ((\mathbf{grad}\, h)v + (\mathbf{grad}\, v)h)\, d\mu = \int \mathbf{grad}\,(hv)\ d\mu = 0$$

which becomes a linear moment constraint

$$L_{\mathbf{m}}(\mathbf{grad}\,(hv)) = 0$$

if v is polynomial. In practice, when implementing the SDP relaxation of degree d, we choose $h(\mathbf{x}) := \prod_{j=1} g_j(\mathbf{x})$ and $v(\mathbf{x}) = \mathbf{x}^\alpha$, $\alpha \in \mathbb{N}^n$, $|\alpha| \le d + 1 - \sum_{j=1}^m \deg g_j$.

Remark 8. The dual to the LP problem (16) is the LP problem

$$\begin{array}{rl} \inf\limits_{\substack{v\in\mathscr{C}_+(\mathbf{B})\\ \mathbf{w}\in\mathscr{C}(\mathbf{B})^n}} & \int v\,\lambda_{\mathbf{B}} \\ \text{s.t.} & v+\operatorname{div}(h\mathbf{w})\geq \mathbb{1}_{\mathbf{K}}. \end{array}$$

It follows that the function v is not required anymore to approximate from above the discontinuous indicator function $\mathbb{1}_{\mathbf{K}}$, so that the Gibbs effect is reduced.

4.2 Sparse Stokes constraints

We have designed efficient Stokes constraints for the full formulation of problem (1), at the price of introducing in problem (4) a polynomial h vanishing on the boundary of $\mathbf{K}$. However, in the sparse case (7), the polynomial h would destroy the sparsity structure, as it is the product of all polynomials defining $\mathbf{K}$. So we must adapt our strategy to introduce sparse Stokes constraints.

In this section, to keep the notations simple, we illustrate the ideas on our introductive bicylinder example of Section 3.1. Considering the optimal measures μ_1 and μ_2 defined in (8),(9), we can apply Stokes constraints derived from the Gauss formula, in the directions in which they are Lebesgue: for μ_1 in the x_1 direction and for μ_2 in the remaining directions. To see this, define

$$\begin{aligned} \mathbf{h}_1(x_1,x_2) &= g_1(x_1,x_2)\,x_1\,\mathbf{e}_1,\\ \mathbf{h}_2(x_2,x_3) &= g_2(x_2,x_3)\,x_2\,\mathbf{e}_2,\\ \mathbf{h}_3(x_2,x_3) &= g_2(x_2,x_3)\,x_3\,\mathbf{e}_3 \end{aligned}$$

where $g_i(x_i,x_{i+1}) = 1-x_i^2-x_{i+1}^2$, such that $\mathbf{h}_1\cdot\mathbf{n}_{\mathbf{U}_1}$ vanishes on the boundary of $\mathbf{U}_1$ and $\mathbf{h}_2\cdot\mathbf{n}_{\mathbf{U}_2}$ and $\mathbf{h}_3\cdot\mathbf{n}_{\mathbf{U}_2}$ vanish on the boundary of $\mathbf{U}_2$, where $\mathbf{n}_{\mathbf{U}_i}$ is the outward point vector orthogonal to the boundary of $\mathbf{U}_i$. For $i,j,k\in\mathbb{N}$, the Gauss formula yields

$$\begin{aligned} \int_{\mathbf{U}_1}\frac{\partial}{\partial x_1}(g_1(x_1,x_2)x_1^{i+1}x_2^j)\,d\mu_1 &= 0,\\ \int_{\mathbf{U}_2}\frac{\partial}{\partial x_2}(g_2(x_2,x_3)x_2^{j+1}x_3^k)\,d\mu_2 &= 0,\\ \int_{\mathbf{U}_2}\frac{\partial}{\partial x_3}(g_2(x_2,x_3)x_2^{j}x_3^{k+1})\,d\mu_2 &= 0. \end{aligned}$$

Hence, adding these constraints does not change the optimal value of the LP problem (7).

4.3 Example: bicylinder revisited

We refer to (1) as the full problem and to (7) as the sparse problem. For both problems, we consider instances with and without additional Stokes constraints. Note that for the bicylinder example of Section 3.1 the optimal value for both the full and the sparse problem is

$$\operatorname{vol}\mathbf{K} = \frac{16}{3} \approx 5.3333$$

since adding Stokes constraints does not change the optimal value.

We solve the SDP relaxations with Mosek on a standard laptop, for various relaxation orders and we report the bounds and the computation times in Table 1. We observe a slow convergence for the full and the sparse versions without Stokes constraints, and a much faster convergence with Stokes constraints. We also observe significantly smaller computation times when using the sparse formulation.

	full		sparse	
d	without Stokes	with Stokes	without Stokes	with Stokes
2	7.8232 (1.0s)	5,828 (1.1s)	7,7424 (1.1s)	5,4984 (1.1s)
3	7.2368 (0.9s)	5,4200 (1.3s)	7,1920 (0.9s)	5,3488 (1.1s)
4	7.0496 (1.4s)	5,3520 (2.2s)	7,0040 (1.2s)	5,3376 (1.2s)
5	6,8136 (3.1s)	5,3400 (4.4s)	6,7944 (1.8s)	5,3352 (1.8s)
6	6,7376 (7.2s)	5,3376 (8.2s)	6,6960 (2.1s)	5,3344 (2.3s)
7	6,6336 (12.8s)	5,3360 (18.3s)	6,6168 (2.6s)	5,3344 (3.2s)

Table 1: Bounds on the volume (and computation times in seconds) vs relaxation order for the bicylinder.

4.4 Example: nonconvex set

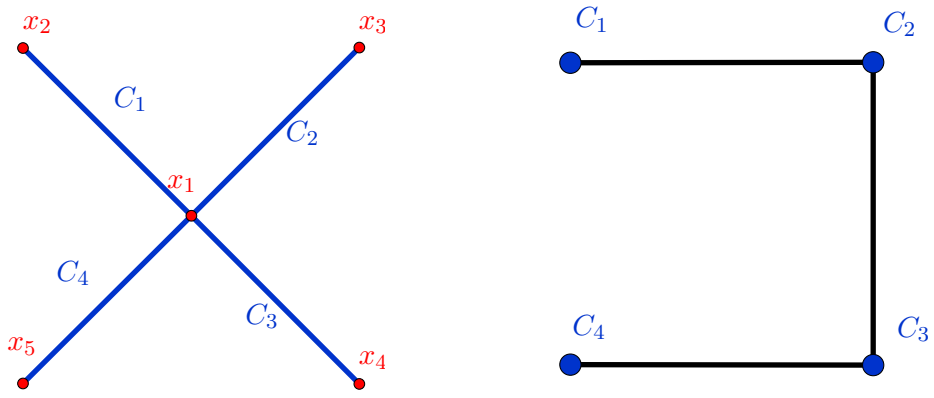
Let $\mathbf{X} := \mathbb{R}^5$, $\mathbf{X}_1 = \langle x_1, x_2 \rangle$, $\mathbf{X}_2 = \langle x_1, x_3 \rangle$, $\mathbf{X}_3 = \langle x_1, x_4 \rangle$, $\mathbf{X}_4 = \langle x_1, x_5 \rangle$ and

- $\mathbf{g}_i(x_1, x_{i+1}) := (2x_1^2 - x_{1+i}^2 - 1\,,\; x_1\,(1 - x_1)\,,\; x_{i+1}\,(1 - x_{i+1}))$, $\quad i = 1, \ldots, 4$
- $\mathbf{U}_i := \mathbf{g}_i^{-1}\left((\mathbb{R}_+)^3\right) = \{(x_1, x_{i+1}) \in [0,1]^2 : 2x_1^2 - x_{i+1}^2 \geq 1\}$, $i = 1, \ldots, 4$.

Let us approximate the volume of the sparse set

$$\mathbf{K} := \left\{(x_1, x_2, x_3, x_4, x_5) \in [0,1]^5 : 2x_i^2 - x_{i+1}^2 \geq 1,\; i = 1, \ldots, 4\right\} = \bigcap_{i=1}^{4} \pi_{\mathbf{X}_i}^{-1}(\mathbf{U}_i)\,.$$

Here the coordinates x_2, x_3, x_4 and x_5 do not interact: they are only linked with the coordinate x_1. The proper way to apply our linear computation Theorem 4 is to define a linear clique tree as shown in Figure 6.



(a) Variable graph (b) Clique tree

Figure 6: Graph with linear clique tree for the nonconvex set.

This yields the following formulation

$$\begin{aligned} \operatorname{vol}\mathbf{K} \quad = \quad & \max_{\substack{\mu_i \in \mathscr{M}_+(\mathbf{X}_i) \\ i=1,\ldots,4}} \int_{\mathbf{X}_1} d\mu_1 \\ \text{s.t.} \quad & d\mu_1(x_1,x_2) \leq d\mu_2^{\langle x_1 \rangle}(x_1)\, dx_2 \\ & d\mu_2(x_1,x_3) \leq d\mu_3^{\langle x_1 \rangle}(x_1)\, dx_3 \\ & d\mu_3(x_1,x_4) \leq d\mu_4^{\langle x_1 \rangle}(x_1)\, dx_4 \\ & d\mu_4(x_1,x_5) \leq dx_1\, dx_5 \\ & \operatorname{spt}\mu_i \subset \mathbf{U}_i \qquad i = 1,\ldots,4 \end{aligned} \tag{17}$$

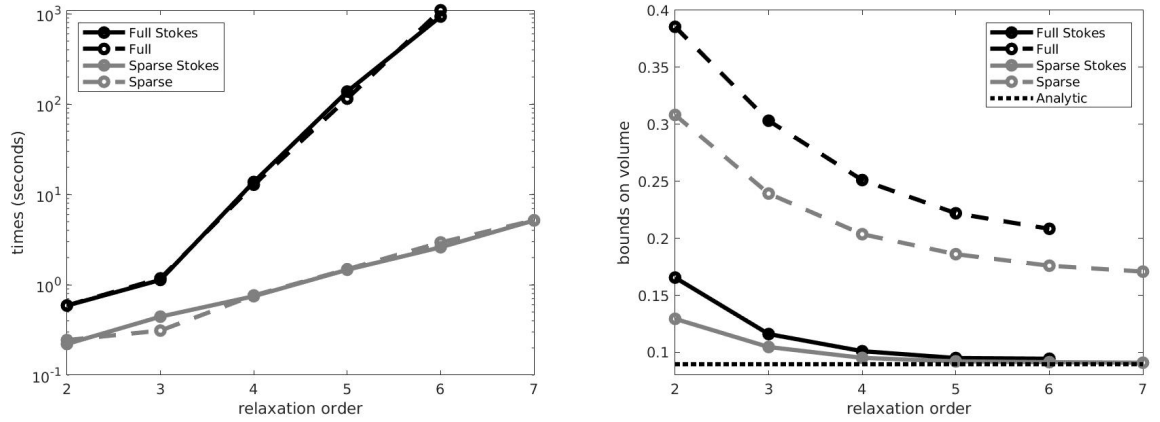
with Stokes constraints

$$\begin{aligned} \frac{\partial}{\partial x_2}\left[(2x_1^2 - x_2^2 - 1)\, x_2\, (1-x_2)\right] d\mu_1(x_1,x_2) &= \frac{\partial}{\partial x_2}\left[(2x_1^2 - x_2^2 - 1)\, x_2\, (1-x_2)\, d\mu_1(x_1,x_2)\right] \\ \frac{\partial}{\partial x_3}\left[(2x_1^2 - x_3^2 - 1)\, x_3\, (1-x_3)\right] d\mu_2(x_1,x_3) &= \frac{\partial}{\partial x_3}\left[(2x_1^2 - x_3^2 - 1)\, x_3\, (1-x_3)\, d\mu_2(x_1,x_3)\right] \\ \frac{\partial}{\partial x_4}\left[(2x_1^2 - x_4^2 - 1)\, x_4\, (1-x_4)\right] d\mu_3(x_1,x_4) &= \frac{\partial}{\partial x_4}\left[(2x_1^2 - x_4^2 - 1)\, x_4\, (1-x_4)\, d\mu_3(x_1,x_4)\right] \\ \frac{\partial}{\partial x_5}\left[(2x_1^2 - x_5^2 - 1)\, x_5\, (1-x_5)\right] d\mu_4(x_1,x_5) &= \frac{\partial}{\partial x_5}\left[(2x_1^2 - x_5^2 - 1)\, x_5\, (1-x_5)\, d\mu_4(x_1,x_5)\right] \\ \frac{\partial}{\partial x_1}\left[(2x_1^2 - x_5^2 - 1)\, x_1\, (1-x_1)\right] d\mu_4(x_1,x_5) &= \frac{\partial}{\partial x_1}\left[(2x_1^2 - x_5^2 - 1)\, x_1\, (1-x_1)\, d\mu_4(x_1,x_5)\right]. \end{aligned}$$

We can compute analytically

$$\operatorname{vol}\mathbf{K} = \frac{1}{15}\left(7 - 4\sqrt{2}\right) \simeq 0.0895.$$

On Figure 7 we show results from solving several relaxations via the full and the sparse approach, with and without Stokes constraints. While solving with Mosek the degree 12 full relaxation took about 1000 seconds, solving the degree 12 sparse relaxation took less than 10 seconds. With the sparse relaxations it was possible to go much higher in the hierarchy. Figure 7b shows convincingly how Stokes constraints accelerate the convergence of the hierarchy. We can also observe that the nonconvexity of $\mathbf{K}$ poses no special difficulty for the volume computation.



(a) Computation time vs relaxation order.

(b) Bounds on the volume vs relaxation order.

Figure 7: Performance for the nonconvex set.

4.5 Example: high dimensional polytope

Consider

$$\mathbf{K}_n := \{\mathbf{x} \in [0,1]^n \ : \ x_i + x_{i+1} \leq 1,\ i = 1, \ldots, n-1\}.$$

According to [20], for any $\theta \in]-\frac{\pi}{2}, \frac{\pi}{2}[$, one has the elegant formula :

$$\tan\theta + \sec\theta = 1 + \sum_{n=1}^{\infty} \mathrm{vol}\, \mathbf{K}_n\, \theta^n$$

which allows to compute analytically the volume for n arbitrarily large. For example when $n = 20$ we obtain

$$\mathrm{vol}\, \mathbf{K}_{20} = \frac{14814847529501}{97316080327065600} \approx 1.522 \cdot 10^{-4}.$$

From the SDP viewpoint, $\mathrm{vol}\, \mathbf{K}_n$ is computed by solving relaxations of the LP problem given in Theorem 4 where $m = n-1$, $\mathbf{X}_i = \langle x_i, x_{i+1} \rangle$ and $\mathbf{g}_i(x_i, x_{i+1}) = (x_i, x_{i+1}, 1 - x_i - x_{i+1})$, $i = 1, \ldots, n-1$.

We implemented the volume computation algorithm for $n = 20$, with Stokes contraints. This cannot be achieved without resorting to sparse computation as the dimension is too high for regular SDP solvers. With the sparse formulation however we could solve relaxations up to degree 28 in less than 100 seconds, see Figure 8. Note however, that the analytic volume is of the order of 10^{-4}. In consequence we observe a non monotonicity of the relaxation values which contradicts the theory. This issue is surprising as the Mosek SDP solver terminates without reporting issues. This indicates that computing small volumes in large dimension can be numerically sensitive.

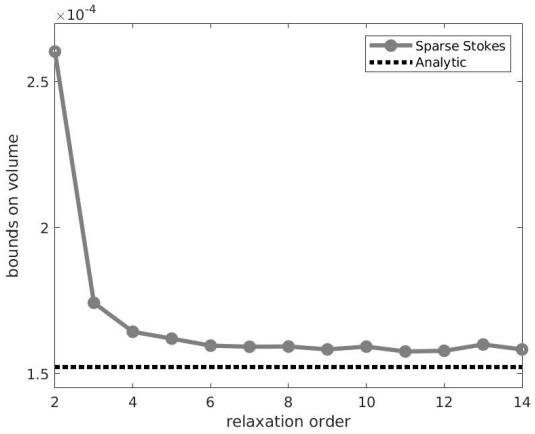


Figure 8: Bounds on the volume vs relaxation order for the high dimensional polytope.

In order to fix the monotonicity issue, we added a sparse rescaling to our problem. The idea is the following: at each step of the algorithm, the mass of the measure μ_i is less than the mass of the reference measure

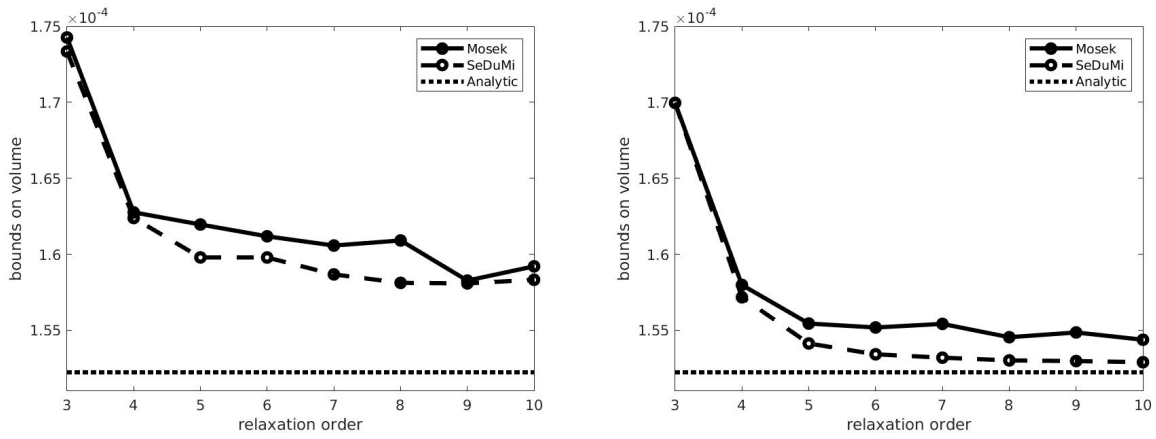
$$\rho_i := \mu_{i+1}^{\mathbf{X}_i \cap \mathbf{X}_{i+1}} \otimes \lambda^{n_i}.$$

Defining

$$\epsilon_i := \frac{|\mu_i|}{|\rho_i|} \in]0,1[,$$

we obtain that

$$\mathrm{vol}\, \mathbf{K} = \prod_{i=1}^{m} \epsilon_i$$



(a) Without rescaling.

(b) With rescaling.

Figure 9: Bounds on the volume vs relaxation order for the high dimensional polytope.

as a telescoping product, since $|\rho_m| = \operatorname{vol} \mathbf{B} = 1$. As a result, if m is large and the ϵ_i are small, one can expect the volume to be very small, which explains why the SDP solver encounters difficulties. Thus, a solution is to multiply each domination constraint by a well-chosen rescaling factor ϵ such that the mass of μ_i does not decrease too much. The resulting LP is as follows

$$\begin{aligned} \operatorname{vol} \mathbf{K} \quad = \quad & \epsilon^{m-1} \max_{\substack{\mu_i \in \mathscr{M}_+(\mathbf{X}_i) \\ i=1,\ldots,m}} \int_{\mathbf{X}_1} d\mu_1 \\ \text{s.t.} \quad & \epsilon\, \mu_i \le \mu_{i+1}^{\mathbf{X}_i \cap \mathbf{X}_{i+1}} \otimes \lambda^{n_i} \qquad i = 1, \ldots, m-1 \\ & \mu_m \le \lambda^{n_m} \\ & \operatorname{spt} \mu_i \subset \mathbf{U}_i \qquad\qquad i = 1, \ldots, m. \end{aligned} \tag{18}$$

Figure 9 gives a comparison between the results obtained with and without sparse rescaling, using the SDP Solvers SeDuMi and Mosek, for the choice $\epsilon = \frac{1}{2}$.

First, one can see that without rescaling (Figure 9a), both SeDuMi and Mosek have accuracy issues that make them lose monotonicity, while the rescaling (Figure 9b) allows to recover monotonicity at least when using SeDuMi (which is slower but more accurate than Mosek to our general experience). Second, it is clear that the relative approximation error is much smaller with scaling. This, combined to the fact that the error is relative (after rescaling, the error is much smaller), demonstrates the power of our rescaling method.

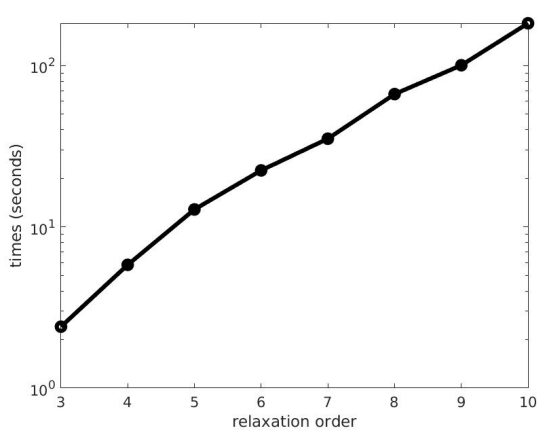
4.6 Example: nonconvex high dimensional set

Finally, we consider a set which is both nonconvex and high dimensional. Let

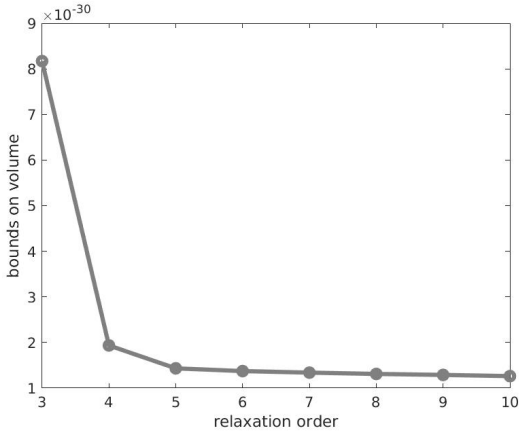
$$\mathbf{K} := \{\mathbf{x} \in [0,1]^n \ : \ x_{i+1} \le x_i^2\}$$

whose analytic volume, as a function of the dimension n, is not available to us, except for $n = 2, 3$. We approximated numerically the volume of $\mathbf{K}$ for $n = 2, 3, 100$. The lower dimension computations (not shown here) were done to check that the algorithm works normally. The $n = 100$ computation was done (with scaling and SeDuMi) to prove that our method allows to compute volumes of nonconvex sets in really high dimension for which it becomes difficult to use Monte-Carlo methods.

On Figure 10a we can see that the order 10 relaxation takes only 3 minutes to be solved. Moreover, Figure 10b gives us the upper bound $\operatorname{vol} \mathbf{K} \le 1.2624 \cdot 10^{-30}$.



(a) Computation time vs relaxation order.



(b) Bounds on the volume vs relaxation order.

Figure 10: Performance for the high dimensional nonconvex set.

5 General sparse volume computation

5.1 General sparsity pattern

Let us describe a general method to compute the volume of

$$\mathbf{K} := \bigcap_{i=1}^{m} \mathbf{K}_i.$$

For this we construct a graph $G = (V, E)$ as follows :

- $V = \{1 \dots, n\}$ represents the canonical basis $\{\mathbf{e}_1, \dots, \mathbf{e}_n\}$ of $\mathbf{X}$;
- $E = \{(i, j) \in \{1, \dots, n\}^2 \ : \ \mathbf{e}_i, \mathbf{e}_j \in \mathbf{X}_k \text{ for some } k \in \{1, \dots, m\}\}$.

In the following, we relax Assumption 3 with

Assumption 9. [Running Intersection Property (RIP)] *For all $i \in \{2, \dots, m\}$ it holds $\mathbf{X}_i \cap \sum_{j=1}^{i-1} \mathbf{X}_j \subset \mathbf{X}_k$ for some $k \in \{1, \dots, m\}$.*

This ensures that G is chordal[5], see [17]. Let $\mathcal{K}$ be the set of cliques of G. By construction, each clique is associated to a single subspace $\mathbf{X}_i$, which gives us a natural ordering on $\mathcal{K} = \{C_1, \dots, C_m\}$. We then make the following strong sparsity statement.

Proposition 10. [Disjoint Intersection Property (DIP)] *If Assumptions 2 and 9 hold, there is a clique tree $T = (\mathcal{K}, \mathcal{E})$, rooted in C_1, such that for all $i, j, k \in \{1, \dots, m\}$, if $(C_i, C_j) \in \mathcal{E}$ then $i < j$ and $C_i \cap C_j \neq \varnothing$, and if $(C_i, C_j) \in \mathcal{E}$ and $(C_i, C_k) \in \mathcal{E}$, then $j = k$ or $C_j \cap C_k = \varnothing$.*

In words, Proposition 10 means that each clique has a nonempty intersection with its parent clique, and an empty intersection with all its sibling cliques. The proof gives an algorithmic method to construct the clique tree. However it is quite technical, so we do not present it here. See Appendix A for details. Figure 11 illustrates the meaning of this proposition for $n = 12$ and $m = 8$. One can check that Assumptions 2 and 9 hold.

Remark 11. According to this proposition, the only possible clique trees for applying our method to the nonconvex example illustrated in Figure 6 are linear clique trees. Indeed, any branched clique tree would imply sibling cliques containing x_1.

[5]Our method is thus valid for any sparse structure, up to a chordal extension of the graph G. In particular, cyclic graphs can be tackled by adding empty interactions between well-chosen variables.

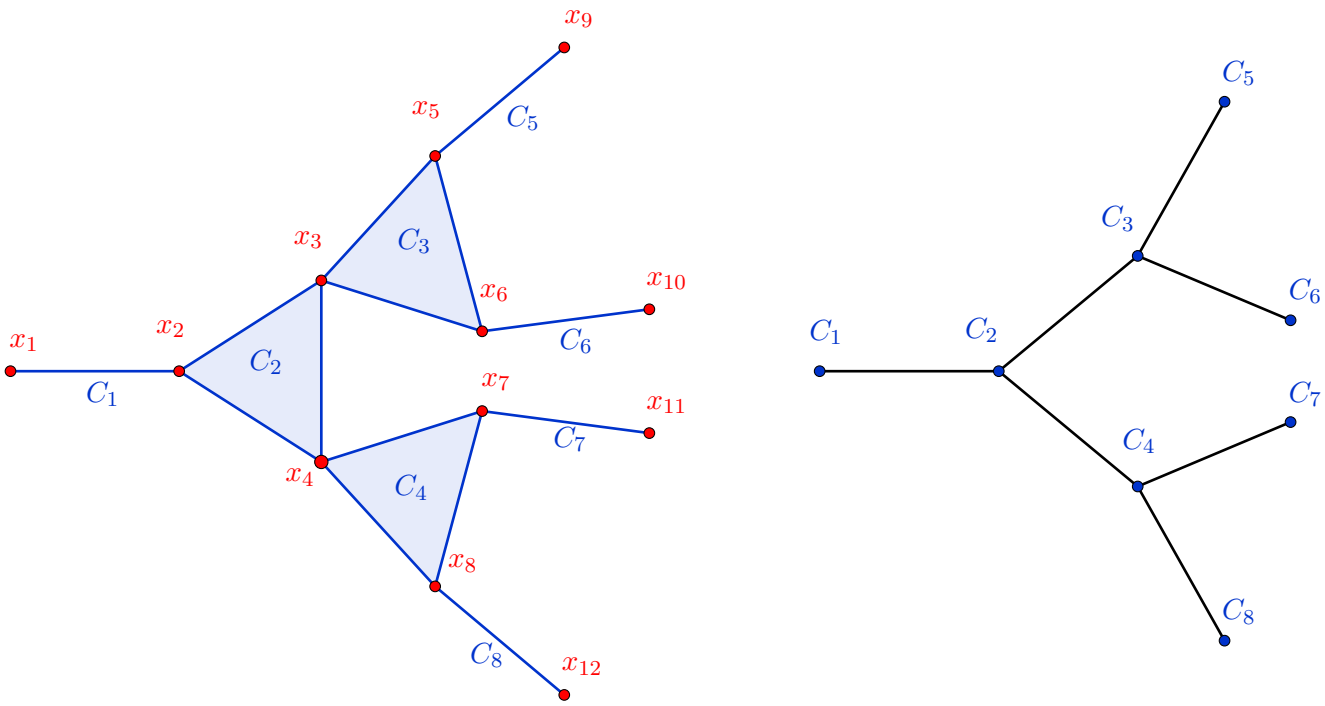


Figure 11: Chordal graph (left) with its clique tree (right).

5.2 Distributed computation theorem

One can formulate a simple generalization of Theorem 4 to our general sparsity pattern.

Theorem 12. *Let Assumption 2 and 9 hold. Let $T = (\mathcal{K}, \mathcal{E})$ be a clique tree as in Proposition 10. Then*

$$\begin{aligned} \operatorname{vol} \mathbf{K} \quad = \quad & \max_{\substack{\mu_i \in \mathscr{M}_+(\mathbf{X}_i) \\ i=1,\dots,m}} \int_{\mathbf{X}_1} d\mu_1 && (19) \\ s.t. \quad & \mu_i \leq \left(\bigotimes_{(C_i,C_j)\in\mathcal{E}} \mu_j^{\mathbf{X}_i \cap \mathbf{X}_j} \right) \otimes \lambda^{n_i} \qquad i = 1, \dots, m && (20) \\ & \operatorname{spt} \mu_i \subset \mathbf{U}_i \qquad\qquad\qquad\quad i = 1, \dots, m && (21) \end{aligned}$$

and $n_i = \dim \mathbf{X}_i \cap \left(\sum_{(C_i,C_j)\in\mathcal{E}} \mathbf{X}_j \right)^{\perp}$ *for* $i = 1, \dots, m$.

The proof of Theorem 12 is very similar in spirit to the proof of Theorem 4, but it is more technical and it requires some additional notations. For this reason, we postpone it to Appendix B.

The main difficulty in Theorem 12 is that problem (19) is not a linear problem on measures, since constraints (20) are multilinear in general. A way of solving this difficulty is to notice that the optimal measures are obtained by saturating constrains (20) while enforcing constraints (21). As a result, these optimal measures can be obtained by maximizing the mass of each one of the μ_i while enforcing constraints (20) and (21).

Then, the sparse volume computation problem can be separated into m linear subproblems as follows.

Corollary 13. *With the same notations as in Theorem 12 it holds* $\operatorname{vol}\mathbf{K} = \int_{\mathbf{X}_1} d\mu_1^*$ *where*

$$\begin{aligned} \mu_i^* \quad \in \quad & \underset{\substack{\mu_i \in \mathscr{M}_+(\mathbf{X}_i) \\ i=1,\ldots,m}}{\operatorname{argmax}} \int_{\mathbf{X}_i} d\mu_i \\ & s.t. \quad \mu_i \leq \left(\bigotimes_{(C_i,C_j)\in\mathcal{E}} \mu_j^{*\mathbf{X}_i\cap\mathbf{X}_j} \right) \otimes \lambda^{n_i} \\ & \qquad \operatorname{spt}\mu_i \subset \mathbf{U}_i, \qquad\qquad i = 1,\ldots,m \end{aligned} \tag{22}$$

and $n_i = \dim\ \mathbf{X}_i \cap \left(\sum_{(C_i,C_j)\in\mathcal{E}} \mathbf{X}_j \right)^\perp$ *for* $i = 1,\ldots,m$.

Therefore one obtains a sequence of infinite dimensional LPs on measures that can be algorithmically addressed using the usual SDP relaxations. The computations start from the leaves of the clique tree and proceed down to the root. It is worth noting that all the cliques of the same generation in the tree are totally independent, which allows to treat them simultaneously, *i.e.* to partially parallelize the computations.

Remark 14. As in Remark 5 we can derive a dual global multilinear sparse program

$$\begin{aligned} \operatorname{vol}\mathbf{K} \quad = \quad & \inf_{\substack{v_i \in \mathscr{C}_+(\mathbf{X}_i) \\ i=1,\ldots,m}} \int_{\mathbf{X}_1} v_1(\mathbf{x}_1)\, d\mathbf{x}_1 \\ & \text{s.t.} \quad v_i(\mathbf{x}_i) \geq \prod_{(C_i,C_j)\in\mathcal{E}} \int_{Z_{ij}} v_j(\pi_{\mathbf{X}_j}(\mathbf{x}_i), \mathbf{z}_{ij})\, d\mathbf{z}_{ij} \qquad \forall \mathbf{x}_i \in \mathbf{U}_i. \end{aligned}$$

The difference with the linear case is that here the dualization does not invert the direction of propagation of the information.

Remark 15. The convergence of the moment-SOS hierarchy associated to the sequence of problems (22), as well as their dual problems, is yet to be proved. Indeed, one cannot solve each problem (22) exactly and propagate the exact moments to solve the next problem when progressing in the clique tree. As numerical examples will show, the algorithm seems to give reasonably strong results. However, it is nontrivial to describe how the fact of transmiting approximate moments instead of exact moments can influence the final computation.

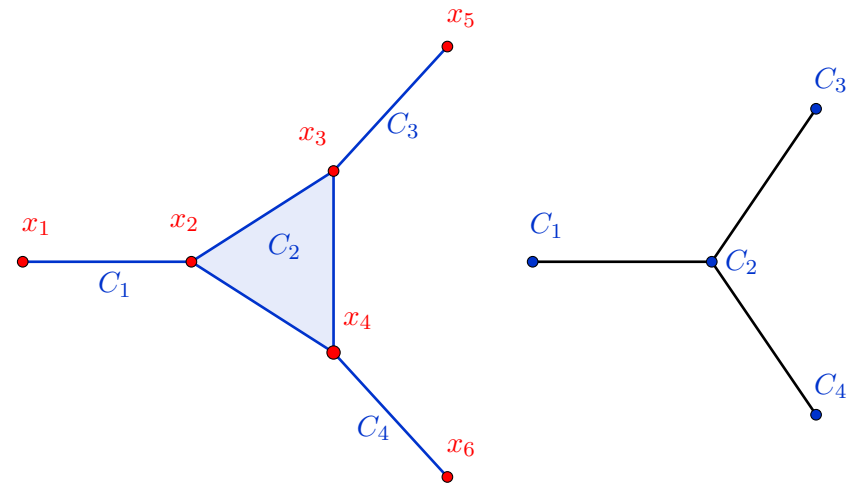
Remark 16. Stokes constraints can be implemented similarly to the linear case.

5.3 Example: 6D polytope

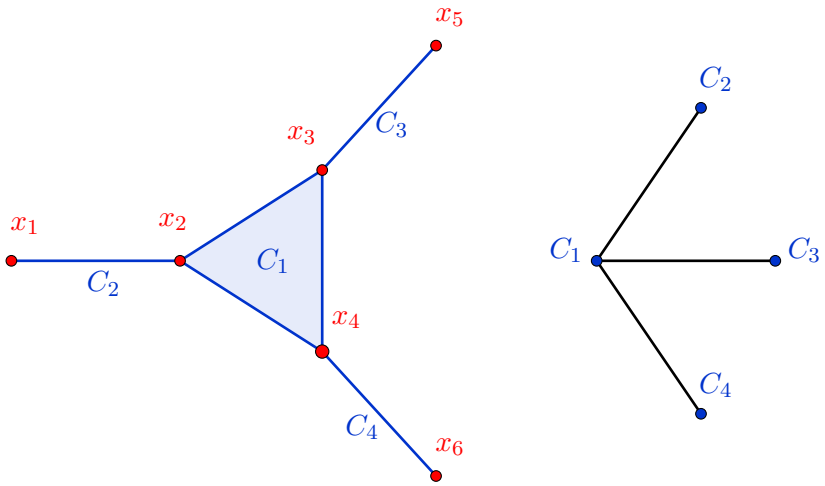
Let $\mathbf{X} := \mathbb{R}^6$ and $\mathbf{X}_1 = \langle x_1, x_2\rangle$, $\mathbf{X}_2 = \langle x_2, x_3, x_4\rangle$, $\mathbf{X}_3 = \langle x_3, x_5\rangle$, $\mathbf{X}_4 = \langle x_4, x_6\rangle$. For $i = 1, 3, 4$ let $\mathbf{g}_i(x, y) := (x, y, 1 - x - y)$ and $\mathbf{U}_i := \mathbf{g}_i^{-1}\left(\mathbb{R}^2_+\right) = \{(u, v) \in [0, 1]^2 \ : \ u + v \leq 1\}$. Let $\mathbf{g}_2(x, y, z) := (x, y, z, 1 - x - y - z)$ and $\mathbf{U}_2 := \mathbf{g}_2^{-1}\left(\mathbb{R}^3_+\right) = \{(x, y, z) \in [0, 1]^3 \mid x + y + z \leq 1\}$. Let us approximate the volume of the 6D polytope

$$\mathbf{K} := \{\mathbf{x} \in \mathbb{R}^6_+ \ : \ x_1 + x_2 \leq 1,\ x_2 + x_3 + x_4 \leq 1,\ x_3 + x_5 \leq 1,\ x_4 + x_6 \leq 1\} = \bigcap_{i=1}^4 \pi_{\mathbf{X}_i}^{-1}(\mathbf{U}_i)\,.$$

No linear clique tree is associated to this problem through Proposition 10. The only possible clique trees for applying our method are the two branched clique trees of Figure 12.



(a) 3 step clique tree



(b) 2 step clique tree

Figure 12: Two possible branched clique trees for the 6D polytope.

Let us compare the performance of the algorithms derived from the two possible clique tree configurations and with the full problem. For that, we first write the problem associated with the 3 step clique tree configuration of the top of Figure 12:

$$\text{vol}\,\mathbf{K} = \int_{\mathbf{X}_1} d\mu_1^* \tag{23}$$

where

$$\begin{aligned}
\mu_1^* \quad &= \quad \underset{\mu_1 \in \mathscr{M}_+(\mathbf{X}_1)}{\text{argmax}} \int_{\mathbf{X}_1} d\mu_1 \\
&\text{s.t.} \quad d\mu_1(x_1, x_2) \le dx_1 \; d\mu_2^{\langle x_2 \rangle}(x_2) \\
&\qquad \text{spt}\,\mu_1 \subset \mathbf{U}_1 \\
\mu_2^* \quad &= \quad \underset{\mu_2 \in \mathscr{M}_+(\mathbf{X}_2)}{\text{argmax}} \int_{\mathbf{X}_2} d\mu_2 \\
&\text{s.t.} \quad d\mu_2(x_2, x_3, x_4) \le dx_2 \; d\mu_3^{*\langle x_3 \rangle}(x_3) \; d\mu_4^{*\langle x_4 \rangle}(x_4) \\
&\qquad \text{spt}\,\mu_2 \subset \mathbf{U}_2
\end{aligned}$$

$$\begin{aligned}
\mu_i^* \quad = \quad & \underset{\substack{\mu_i \in \mathscr{M}_+(\mathbf{X}_i) \\ i=3,4}}{\operatorname{argmax}} \int_{\mathbf{X}_i} d\mu_i \\
\text{s.t.} \quad & \mu_i \leq \lambda^2 \\
& \operatorname{spt} \mu_i \subset \mathbf{U}_i, \qquad\qquad i = 3, 4.
\end{aligned}$$

This problem can be complemented with the following Stokes constraints:

$$\begin{aligned}
\frac{\partial}{\partial x_1}\left[x_1\,(1-x_1-x_2)\right]\,d\mu_1(x_1,x_2) &= \frac{\partial}{\partial x_1}\left[x_1\,(1-x_1-x_2)\,d\mu_1(x_1,x_2)\right] \\
\frac{\partial}{\partial x_2}\left[x_2\,(1-x_2-x_3-x_4)\right]\,d\mu_2(x_2,x_3,x_4) &= \frac{\partial}{\partial x_2}\left[x_2\,(1-x_2-x_3-x_4)\,d\mu_2(x_2,x_3,x_4)\right] \\
\frac{\partial}{\partial x_i}\left[x_i\,(1-x_i-x_{i+2})\right]\,d\mu_i(x_i,x_{i+2}) &= \frac{\partial}{\partial x_i}\left[x_i\,(1-x_i-x_{i+2})\,d\mu_i(x_i,x_{i+2})\right] \\
\frac{\partial}{\partial x_{i+2}}\left[x_{i+2}\,(1-x_i-x_{i+2})\right]\,d\mu_i(x_i,x_{i+2}) &= \frac{\partial}{\partial x_{i+2}}\left[x_{i+2}\,(1-x_i-x_{i+2})\,d\mu_i(x_i,x_{i+2})\right] \qquad i=3,4.
\end{aligned}$$

The 2 step clique tree of the bottom of Figure 12 yields the following formulation

$$\operatorname{vol}\mathbf{K} = \int_{\mathbf{X}_2} d\mu_2^* \tag{24}$$

where

$$\begin{aligned}
\mu_2^* \quad = \quad & \underset{\mu_2 \in \mathscr{M}_+(\mathbf{X}_2)}{\operatorname{argmax}} \int_{\mathbf{X}_2} d\mu_2 \\
\text{s.t.} \quad & d\mu_2(x_2,x_3,x_4) \leq d\mu_1^{*\langle x_2\rangle}(x_2)\,d\mu_3^{*\langle x_3\rangle}(x_3)\,d\mu_4^{*\langle x_4\rangle}(x_4) \\
& \operatorname{spt}\mu_2 \subset \mathbf{U}_2 \\
\mu_i^* \quad = \quad & \underset{\substack{\mu_i \in \mathscr{M}_+(\mathbf{X}_i) \\ i=1,3,4}}{\operatorname{argmax}} \int_{\mathbf{X}_i} d\mu_i \\
\text{s.t.} \quad & \mu_i \leq \lambda_{\mathbf{X}_i} \\
& \operatorname{spt}\mu_i \subset \mathbf{U}_i, \qquad\qquad i = 1, 3, 4
\end{aligned}$$

with Stokes constraints

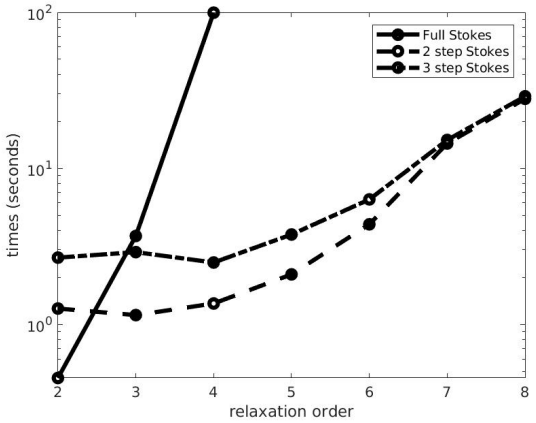
$$\begin{aligned}
\frac{\partial}{\partial x_i}\left[x_i\,(1-x_i-x_j)\right]\,d\mu_i(x_i,x_j) &= \frac{\partial}{\partial x_i}\left[x_i\,(1-x_i-x_j)\,d\mu_i(x_i,x_j)\right] \\
\frac{\partial}{\partial x_j}\left[x_j\,(1-x_i-x_j)\right]\,d\mu_i(x_i,x_j) &= \frac{\partial}{\partial x_j}\left[x_j\,(1-x_i-x_j)\,d\mu_i(x_i,x_j)\right]
\end{aligned}$$

for $(i,j) = (1,2),(3,5),(4,6)$. However, no Stokes constraints can be applied for the computation of μ_2 (there is no Lebesgue measure in the domination constraint, so the optimal measure is not uniform). For this reason one can expect a slower convergence than in the linear configuration.

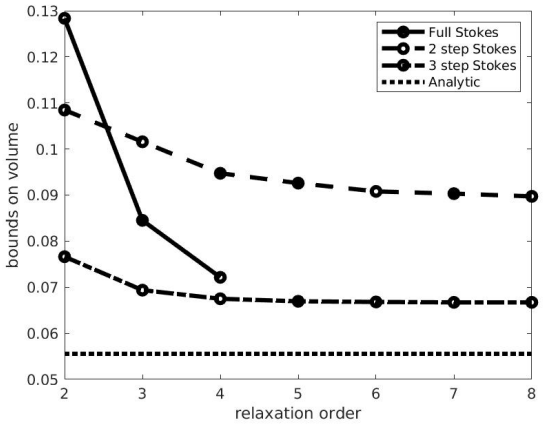
We implement the hierarchies associated to the 2 and 3 step sparse formulations, as well as the full problem hierarchy, and compare their performance in Figure 13. We can compute analytically

$$\operatorname{vol}\mathbf{K} = \frac{1}{18} \simeq 0.0556.$$

Both sparse formulations outperform the dense one in terms of computational time needed to solve the corresponding SDPs (Figure 13a). On the accuracy side however (Figure 13b), we observe that the 2 step formulation seems to be less efficient than the 3 step formulation. In



(a) Computation time vs relaxation order.



(b) Bounds on the volume vs relaxation order.

Figure 13: Performance for the 6D polytope.

particular when considering the accuracy/time effort relation at order 3 the full formulation provides a better value in almost the same time. We believe that this can be explained by the way Stokes constraints are added to the program. In the branched configuration, the last step of the optimization program cannot be accelerated by Stokes constraints. The gap between the optimal value and the analytic value for the branched formulation in Figure 13b could be explained by the Gibbs effect in the last optimization step.

As a consequence, in the following, one should avoid the branched hierarchies that cannot be accelerated at each step by Stokes constraints. Such a hierarchy appears when the root of the chosen clique tree shares all its vertices with its children cliques. It can be proved that such a configuration can always be avoided while implementing sparse volume computation, by choosing a leaf as the new root of the tree.

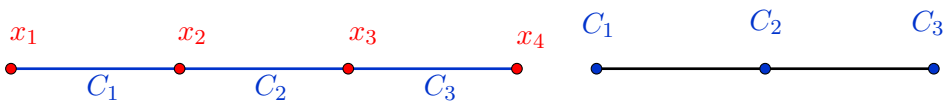
5.4 Example: 4D polytope

Let $\mathbf{X} := \mathbb{R}^4$, $\mathbf{X}_1 := \langle x_1, x_2 \rangle$, $\mathbf{X}_2 := \langle x_2, x_3 \rangle$, $\mathbf{X}_3 := \langle x_3, x_4 \rangle$, $\mathbf{g}_i(u, v) := (u, v, 1 - u - v)$, $i = 1, 2, 3$ and $\mathbf{U}_i := \mathbf{g}_i^{-1}\left(\mathbb{R}^3_+\right) = \{(u, v) \in [0, 1]^2 \ : \ u + v \leq 1\}$. Let us approximate the volume of the 4D polytope

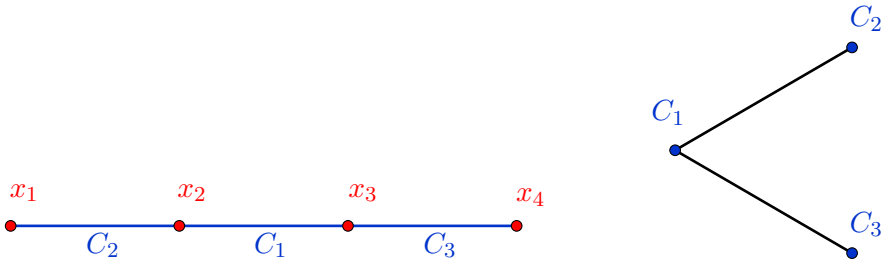
$$\mathbf{K} := \left\{(x_1, x_2, x_3, x_4) \in \mathbb{R}^4_+ \ : \ x_1 + x_2 \leq 1, \ x_2 + x_3 \leq 1, \ x_3 + x_4 \leq 1\right\} = \bigcap_{i=1}^{3} \pi_{\mathbf{X}_i}^{-1}\left(\mathbf{U}_i\right).$$

In such a case, there are two possible configurations for the associated clique tree of Proposition 10, see Figure 14. Accordingly, we can compute $\operatorname{vol} \mathbf{K}$ in two different ways. The first way

$$\begin{aligned} \operatorname{vol} \mathbf{K} \quad = \quad & \max_{\substack{\mu_1 \in \mathscr{M}_+(\mathbf{X}_1) \\ \mu_2 \in \mathscr{M}_+(\mathbf{X}_2) \\ \mu_3 \in \mathscr{M}_+(\mathbf{X}_3)}} \int_{\mathbf{X}_1} d\mu_1 \\ \text{s.t.} \quad & d\mu_1(x_1, x_2) \leq dx_1 \, d\mu_2^{\langle x_2 \rangle}(x_2) \\ & d\mu_2(x_2, x_3) \leq dx_2 \, d\mu_3^{\langle x_3 \rangle}(x_2) \\ & d\mu_3(x_3, x_4) \leq dx_3 \, dx_4 \\ & \operatorname{spt} \mu_i \subset \mathbf{U}_i, \qquad\qquad i = 1, 2, 3. \end{aligned} \tag{25}$$



(a) linear clique tree



(b) branched clique tree

Figure 14: Two possible clique trees for the 4D polytope.

is the linear formulation given by Corollary 4, which is under the form of a non-parallelizable single linear problem. The following additional Stokes constraints can be added:

$$\begin{aligned}
\frac{\partial}{\partial x_1}\Big[x_1\,(1-x_1-x_2)\Big]\;d\mu_1(x_1,x_2) &= \frac{\partial}{\partial x_1}\left[x_1\,(1-x_1-x_2)\;d\mu_1(x_1,x_2)\right]\\
\frac{\partial}{\partial x_2}\Big[x_2\,(1-x_2-x_3)\Big]\;d\mu_2(x_2,x_3) &= \frac{\partial}{\partial x_2}\left[x_2\,(1-x_2-x_3)\;d\mu_2(x_2,x_3)\right]\\
\frac{\partial}{\partial x_3}\Big[x_3\,(1-x_3-x_4)\Big]\;d\mu_3(x_3,x_4) &= \frac{\partial}{\partial x_3}\left[x_3\,(1-x_3-x_4)\;d\mu_3(x_3,x_4)\right]\\
\frac{\partial}{\partial x_4}\Big[x_4\,(1-x_3-x_4)\Big]\;d\mu_3(x_3,x_4) &= \frac{\partial}{\partial x_4}\left[x_4\,(1-x_3-x_4)\;d\mu_3(x_3,x_4)\right].
\end{aligned}$$

On the other hand, if one associates the clique C_1 to the subspace $\mathbf{X}_2$ and the clique C_2 to the subspace $\mathbf{X}_1$, one also has

$$\operatorname{vol}\mathbf{K} = \int_{\mathbf{X}_2} d\mu_2^* \tag{26}$$

where

$$\begin{aligned}
\mu_2^* \;=\; & \operatorname*{argmax}_{\mu_2\in\mathscr{M}_+(\mathbf{X}_2)} \int_{\mathbf{X}_2} d\mu_2\\
& \text{s.t.}\quad d\mu_2(x_2,x_3) \le \mu_1^{*\langle x_2\rangle}(dx_2)\;d\mu_3^{*\langle x_3\rangle}(x_3)\\
& \qquad \operatorname{spt}\mu_2 \subset \mathbf{U}_2\\
\mu_1^* \;=\; & \operatorname*{argmax}_{\mu_1\in\mathscr{M}_+(\mathbf{X}_1)} \int_{\mathbf{X}_1} d\mu_1\\
& \text{s.t.}\quad d\mu_1(x_1,x_2) \le dx_1\,dx_2\\
& \qquad \operatorname{spt}\mu_1 \subset \mathbf{U}_1\\
\mu_3^* \;=\; & \operatorname*{argmax}_{\mu_3\in\mathscr{M}_+(\mathbf{X}_3)} \int_{\mathbf{X}_3} d\mu_3\\
& \text{s.t.}\quad d\mu_3(x_3,x_4) \le dx_3\,dx_4\\
& \qquad \operatorname{spt}\mu_3 \subset \mathbf{U}_3
\end{aligned}$$

which is the branched formulation associated to Theorem 12. Here one can see that μ_1^* and μ_3^* can be computed independently in parallel, and then re-injected in the problem to which μ_2^* is

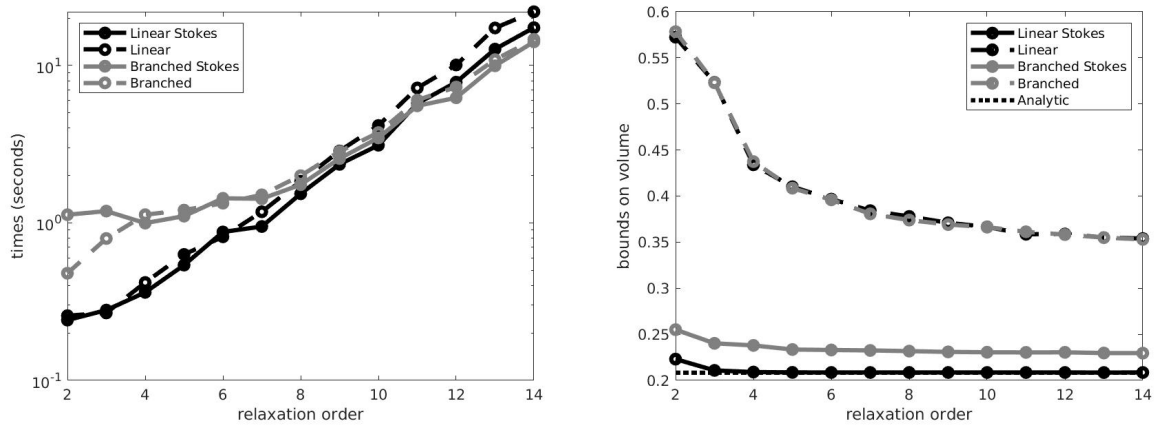
the solution. One can add the following Stokes constraints:

$$\begin{aligned}
\frac{\partial}{\partial x_1}\Big[x_1\,(1-x_1-x_2)\Big]\;d\mu_1(x_1,x_2) &= \frac{\partial}{\partial x_1}\left[x_1\,(1-x_1-x_2)\;d\mu_1(x_1,x_2)\right]\\
\frac{\partial}{\partial x_2}\Big[x_2\,(1-x_1-x_2)\Big]\;d\mu_1(x_1,x_2) &= \frac{\partial}{\partial x_2}\left[x_2\,(1-x_1-x_2)\;d\mu_1(x_1,x_2)\right]\\
\frac{\partial}{\partial x_3}\Big[x_3\,(1-x_3-x_4)\Big]\;d\mu_3(x_3,x_4) &= \frac{\partial}{\partial x_3}\left[x_3\,(1-x_3-x_4)\;d\mu_3(x_3,x_4)\right]\\
\frac{\partial}{\partial x_4}\Big[x_4\,(1-x_3-x_4)\Big]\;d\mu_3(x_3,x_4) &= \frac{\partial}{\partial x_4}\left[x_4\,(1-x_3-x_4)\;d\mu_3(x_3,x_4)\right].
\end{aligned}$$

We can compute analytically

$$\operatorname{vol}\mathbf{K} = \frac{5}{24} \simeq 0.2083.$$

In Figure 15 we compare the two sparse formulations with and without Stokes constraints. Surprisingly the linear formulation is faster than the branched one for small relaxation degrees. When going deeper in the hierarchy we see the advantage of the branched formulation where more computations are done in parallel. As observed in the previous example however, the branched formulation seems to have problems to converge to the optimal value on an early relaxation. While the values of both formulations without Stokes constraints almost coincide, the values of the linear formulation with Stokes are strictly better than the ones of the accelerated branched formulation. This further supports our conjecture that formulations where Stokes constraints can be added at every step of the optimization program are to be preferred.



(a) Computation time vs relaxation order.

(b) Bounds on the volume vs relaxation order.

Figure 15: Performance for the 4D polytope.

Conclusion

Our results

In this paper we addressed the problem of approximating the volume of sparse semi-algebraic sets with the moment-SOS hierarchy of SDP relaxations. As illustrated by our examples, our sparse formulation allows to dramatically decrease the computational time for each relaxation, and to tackle high dimensional volume computation problems that are not tractable with the usual SDP methods. By splitting the problems into low dimensional subproblems, one drastically reduces the dimension of each relaxation, without loss of precision. This reduction of complexity is due

to the correspondance between the structure of our algorithm and the sparsity pattern in the description of the semi-algebraic set.

We also showed that additional Stokes constraints have a huge effect on convergence and precision for volume computation, and that they can successfully be adapted to our sparse formulations. This yields a much better rate of convergence for the corresponding hierarchy. However, implementing these Stokes constraints leads to subtle constraints that have to be enforced if one wants to efficiently compute the volume:

- First, one should always prefer the linear formulation of Theorem 4 whenever possible, since this ensures that Stokes constraints can always be efficiently implemented.
- Then, in the more general case of Theorem 12, one should always avoid formulations in which the root at the computation tree has no Stokes constraint; fortunately, such configurations can always be avoided by chosing a leaf as the root of the clique tree.

Furthermore, in the branched case, one should be aware of the fact that each step of the algorithm introduces an approximation error, and the errors accumulate until the root is reached. Consequently, a formulation in which the clique tree has too many generations will lead to a larger global error than a formulation with less generations. For this reason, one should minimize the number of generations in the clique tree, which is equivalent to parallelizing as much as possible. In addition to that, when the problem has many dimensions and branches, parallelization can obviously drastically increase the speed of the computations.

Applications and future work

To the best of our knowledge, this sparse method for solving volume problems is new and full of promises for future applications. For instance, the problem of computing the mass of any compactly supported measure absolutely continuous (with respect to the Lebesgue measure) can be adressed using this sparsity method. Also, measures that are not compactly supported but have some decay properties (e.g. Gaussian measures) can also be handled by our method, which may prove useful in computations for probability and statistics. Also, specific constraints could probably be used in addition to Stokes constraints when the semi-algebraic set presents a specific structure (e.g. a polytope, a convex body).

Furthermore, the framework of exploiting sparsity can be applied to any method that relies on computations on measures, whether these measures are represented by their moments (as it is done in this paper), or by samples (as in the stochastic volume computation methods). In particular, we believe that this formalism could easily be extended to Monte-Carlo-based volume computations.

Finally, we also believe that this method can be adapted to the computation of regions of attraction, through the formalism developed in [13], for high dimensional differential-algebraic systems that present a network structure, such as power grids, distribution networks in general and possibly other problems. The main difficulty resides in taking sparsity into account when formulating the Liouville equation, and keeping uniqueness of the solution as in the non-controlled non-sparse framework.

A Disjoint Intersection Property

Definition 17. Let $G = (V, E)$ be a graph with vertices set V and edges set E. The following definitions can be found in e.g. [17]:

- A *clique* of G is a subset of vertices $C \subset V$ such that $u, v \in C$ implies $(u, v) \in E$.
- A graph G is *chordal* if every cycle of length greater than 3 has a chord, i.e. an edge connecting two nonconsecutive vertices on the cycle.

- A *tree* $T = (\mathcal{K}, \mathcal{E})$ is a graph without cycle.
- A *rooted tree* is a tree in which one vertex has been designated the *root*.
- In a rooted tree, the *parent* of a vertex v is the vertex w connected to it on the path to the root; v is then called a *child* of w; two vertices that have the same parent are called *siblings*; a *descendant* of a vertex v is any vertex which is either the child of v or is (recursively) the descendant of any of the children of v.
- The vertices of a rooted tree can be partitioned between the root, the *leaves* (the vertices that have parents but no children) and the *branches* (that have children and parents).
- Let $\mathcal{K}$ be the set of cliques of G. A *clique tree* $T = (\mathcal{K}, \mathcal{E})$ of G is a tree whose vertices are the cliques of G.
- A tree satisfies the clique intersection property (CIP) if for every pair of distinct cliques $C, C' \in \mathcal{K}$, the set $C \cap C'$ is contained in every clique on the path connecting C and C' in the tree. We denote by $\mathcal{T}^{ct}$ the set of clique trees of G that satisfy the CIP.
- The ordering $\mathcal{K} = \{C_1, \dots, C_m\}$ satisfies the *Running Intersection Property* (RIP) if $\forall i \in \{2, \dots, m\}$, $\exists\, k_i \in \{1, \dots, i-1\}$ such that $C_i \cap \bigcup_{j=1}^{i-1} C_j \subset C_{k_i}$.

Theorem 18. *A connected graph G is chordal if and only if $\mathcal{T}^{ct} \neq \varnothing$ if and only if $\mathcal{K}$ admits an ordering that satisfies the RIP.*

Definition 19. Let $G = (V, E)$ be chordal. Let $T = (\mathcal{K}, \mathcal{E}) \in \mathcal{T}^{ct}$ be a clique tree rooted in $C_1 \in \mathcal{K}$. T satisfies the *Disjoint Intersection Property* (DIP) if $\forall C, C', C'' \in \mathcal{K}$, if $(C, C') \in \mathcal{E}_T$, then $C \cap C' \neq \varnothing$, and if $(C, C') \in \mathcal{E}_T$ and $(C, C'') \in \mathcal{E}$ then $C' = C''$ or $C' \cap C'' = \varnothing$. In words, each clique has a nonempty intersection with its parent clique, and an empty intersection with all its siblings.

Theorem 20. *There is a clique tree $T^* = (\mathcal{K}, \mathcal{E}^*) \in \mathcal{T}^{ct}$ rooted in some $C_1 \in \mathcal{K}$ that satisfies the DIP. Moreover, such a tree can be constructed algorithmically by ordering the cliques and using the RIP.*

Proof. Since G is connected and chordal, there exists an ordering $\mathcal{K} = \{C_1, \dots, C_m\}$ such that $\forall i \in \{2, \dots, m\}$, it holds $C_i \cap \left(\bigcup_{j=1}^{i-1} C_j \right) \neq \varnothing$ and satisfying the RIP. We are going to construct the announced clique tree, by finite induction:

• By connectivity, $C_1 \cap C_2 \neq \varnothing$. We then define $\mathcal{E}_2 := \{(C_1, C_2)\}$. The tree $T_2 = (\{C_1, C_2\}, \mathcal{E}_2)$ trivially satisfies the DIP.

• Let $k \in \{3, \dots, m\}$. Suppose that we have constructed a tree $T_{k-1} = (\{C_1, \dots, C_{k-1}\}, \mathcal{E}_{k-1})$ satisfying the DIP, by successively adding the cliques $C_3, \dots, C_{k-1}$ to T_2. We want to add the clique C_k to our tree. We link it to a parent clique C_{j_*} of T_{k-1}. This C_{j_*} is a leaf or a node of the tree. We first notice that by construction $(C_i, C_j) \in \mathcal{E}_{k-1}$ implies $i < j$. Let $\mathcal{B}(T_{k-1}) := \{j \in \{1, \dots, k-2\} \ : \ C_j \text{ already has child cliques}\}$ be the set of the cliques to which other cliques have already been connected, i.e. the branch cliques.

- If $\exists\, i \in \mathcal{B}(T_{k-1})$ such that $C_i \cap C_k \neq \varnothing$ and $(C_i, C_j) \in \mathcal{E}_{k-1}$ implies $C_j \cap C_k = \varnothing$, then we can add (C_i, C_k) to $\mathcal{E}_{k-1}$. In words, C_k shares an available variable with a branch clique, which is therefore a valid choice for its parent clique. We define $\mathcal{E}_k := \mathcal{E}_{k-1} \cup \{(C_i, C_k)\}$.

- Else, $\forall i \in \mathcal{B}(T_{k-1})$ such that $C_i \cap C_k \neq \varnothing$, $\exists\, j \in \{i+1, \dots, k-1\}$ such that $(C_i, C_j) \in \mathcal{E}_{k-1}$ and $C_j \cap C_k \neq \varnothing$. In words, any branch clique of T_{k-1} cannot be chosed as a parent clique for C_k without violating the DIP.

Let $\mathcal{L}(T_{k-1}) := \{j \in \{2,\dots,k-1\} \mid C_j \text{ has no child clique yet}\}$ be the set of leaves of T_{k-1}. Then, C_k is connected to a leaf $C_{j_*} \in \mathcal{L}(T_{k-1})$ and $C_{j_*} \cap C_k \neq \varnothing$, see the proof of Lemma 21 below. We define $\mathcal{E}_k := \mathcal{E}_{k-1} \cup \{(C_j, C_k)\}$.

One can check that the graph $T_k = (\{C_1,\dots,C_k\}, \mathcal{E}_k)$ is a tree that satisfies the DIP.

By finite induction, we have constructed a clique tree $T^* = (\mathcal{K}, \mathcal{E}_m)$ that satisfies the DIP. □

Lemma 21. *With the notations of the proof of Theorem 20, if* $\forall i \in \mathcal{B}(T_{k-1})$ *such that* $C_i \cap C_k \neq \varnothing$, $\exists\, j \in \{i+1,\dots,k-1\}$ *such that* $(C_i, C_j) \in \mathcal{E}_{k-1}$ *and* $C_j \cap C_k \neq \varnothing$, *then* $\exists\, j_* \in \mathcal{L}(T_{k-1})$ *such that* $C_{j_*} \cap C_k \neq \varnothing$. *In words, if any branch clique of* T_{k-1} *cannot be chosed as a parent clique for* C_k *without violating the DIP, then* C_k *is connected to a leaf of* T_{k-1}.

Proof. We first notice that $\{1,\dots,k-1\} = \mathcal{B}(T_{k-1}) \sqcup \mathcal{L}(T_{k-1})$, i.e. any clique is either a node, or a leaf. Let $\mathcal{I}(C_k) := \{i \in \{1,\dots,k-1\} \mid C_i \cap C_k \neq \varnothing\}$. By connectivity, $\mathcal{I}(C_k) \neq \varnothing$. Thus, if $\mathcal{I}(C_k) \cap \mathcal{B}(T_{k-1}) = \varnothing$, then our first remark yields that $\mathcal{I}(C_k) \cap \mathcal{L}(T_{k-1}) \neq \varnothing$: $\exists\, j_* \in \mathcal{I}(C_k) \cap \mathcal{L}(T_{k-1})$ and $C_{j_*} \cap C_k \neq \varnothing$.

We now consider the case in which $\mathcal{I}(C_k) \cap \mathcal{B}(T_{k-1}) \neq \varnothing$. Let $i_* := \max(\mathcal{I}(C_k) \cap \mathcal{B}(T_{k-1}))$. In words, C_{i_*} is by construction the latest clique satisfying $C_{i_*} \cap C_k \neq \varnothing$ which is not a leaf of T_{k-1}. Since $i_* \in \mathcal{I}(C_k) \cap \mathcal{B}(T_{k-1})$, by assumption there is a $j_* \in \{i_*+1,\dots,k-1\}$ such that $(C_{i_*}, C_{j_*}) \in \mathcal{E}_{k-1}$ and $C_{j_*} \cap C_k \neq \varnothing$. By maximality of i_*, it is necessary that $j_* \notin \mathcal{B}(T_{k-1})$: $j_* \in \mathcal{L}(T_{k-1})$. □

B Proof of Theorem 12

Proof. We first observe that, according to Proposition 10, for any $i \in \{1,\dots,m\}$

$$\mathbf{X}_i = \left(\bigoplus_{(C_i,C_j)\in\mathcal{E}} \mathbf{X}_i \cap \mathbf{X}_j \right) \oplus \mathbf{Y}_i.$$

Thus, constraint (20) is well-posed.

Let us prove first that $p^* \geq \operatorname{vol} \mathbf{K}$.

For $i \in \{1,\dots,m\}$, let $\mathcal{D}(i) := \{j \neq i \ : \ \exists \text{ an oriented path from } C_i \text{ to } C_j \text{ in } T\}$, the set of descendants of C_i.

For $j \in \{2,\dots,m\}$, let $\mathbf{Z}_{ij} := \mathbf{X}_i^{\perp} \cap \mathbf{X}_j$ such that $\mathbf{X}_j = (\mathbf{X}_i \cap \mathbf{X}_j) \oplus \mathbf{Z}_{ij}$.

Let $i \in \{1,\dots,m\}$, $\mathbf{Z}_i := \sum_{j\in\mathcal{D}(i)} \mathbf{Z}_{ij}$. By definition, $\forall j \in \mathcal{D}(i)$, $\mathbf{X}_j \subset \mathbf{X}_i \oplus \mathbf{Z}_i$. Let

$$d\mu_i(\mathbf{x}_i) := \mathbb{1}_{\mathbf{U}_i}(\mathbf{x}_i) \left(\int_{\mathbf{Z}_i} \prod_{j\in\mathcal{D}(i)} \mathbb{1}_{\mathbf{U}_j}(\pi_{\mathbf{X}_j}(\mathbf{x}_i + \mathbf{z}_i))\, d\mathbf{z}_i \right) d\mathbf{x}_i.$$

By construction, $\mu_i \in \mathscr{M}_+(\mathbf{X}_i)$ and constraints (21) are enforced.

Moreover, if $(C_h, C_i) \in \mathcal{E}$, then

$$d\mu_i^{\mathbf{X}_h\cap\mathbf{X}_i}(\mathbf{x}_{hi}) = \left(\int_{\mathbf{Z}_{hi}} \mathbb{1}_{\mathbf{U}_i}(\mathbf{x}_{hi} + \mathbf{z}_{hi}) \left(\int_{\mathbf{Z}_i} \prod_{j\in\mathcal{D}(i)} \mathbb{1}_{\mathbf{U}_j}(\pi_{\mathbf{X}_j}(\mathbf{x}_{hi} + \mathbf{z}_{hi} + \mathbf{z}_i))\, d\mathbf{z}_i \right) d\mathbf{z}_{hi} \right) d\mathbf{x}_{hi}.$$

Let us prove that the family $\{\mu_1,\dots,\mu_m\}$ we have constructed saturates the domination constraints (20) on the supports of the μ_i.

Let $i \in \{1, \dots, m\}$, $A \in \mathcal{B}(\mathbf{X}_i)$. Then

$$\begin{aligned}
\left(\left(\bigotimes_{(C_i;C_j)\in\mathcal{E}} \mu_j^{\mathbf{X}_i\cap\mathbf{X}_j}\right) \otimes \lambda^{n_i}\right)(A \cap \mathbf{U}_i) &= \left(\prod_{(C_i,C_j)\in\mathcal{E}} \mu_j^{\mathbf{X}_i\cap\mathbf{X}_j}(A \cap \mathbf{U}_i \cap \mathbf{X}_j)\right) \times \lambda^{n_i}(A \cap \mathbf{U}_i \cap \mathbf{Y}_i) \\
&= \left(\prod_{(C_i,C_j)\in\mathcal{E}} \left(\int_{\mathbf{X}_i\cap\mathbf{X}_j} \mathbb{1}_{A\cap\mathbf{U}_i}(\mathbf{x}_{ij})\right.\right. \\
&\quad \times \left(\int_{\mathbf{Z}_{ij}} \mathbb{1}_{\mathbf{U}_j}(\pi_{\mathbf{X}_j}(\mathbf{x}_{ij} + \mathbf{z}_{ij}))\right. \\
&\quad \times \left(\int_{\mathbf{Z}_j} \prod_{k\in\mathcal{D}(j)} \mathbb{1}_{\mathbf{U}_k}(\pi_{\mathbf{X}_k}(\mathbf{x}_{ij} + \mathbf{z}_{ij} + \mathbf{z}_j))\, d\mathbf{z}_j\right) \\
&\quad \left.\left.\left. \times \quad d\mathbf{z}_{ij}\right) d\mathbf{x}_{ij}\right)\right) \times \int_{\mathbf{Y}_i} \mathbb{1}_{A\cap\mathbf{U}_i}(\mathbf{y}_i)\, d\mathbf{y}_i \\
&= \int_{\mathbf{X}_i} \mathbb{1}_{A\cap\mathbf{U}_i}(\mathbf{x}_i) \times \prod_{(C_i,C_j)\in\mathcal{E}} \left(\int_{\mathbf{Z}_{ij}} \mathbb{1}_{\mathbf{U}_j}(\pi_{\mathbf{X}_j}(\mathbf{x}_i + \mathbf{z}_{ij}))\right. \\
&\quad \times \left(\int_{\mathbf{Z}_j} \prod_{k\in\mathcal{D}(j)} \mathbb{1}_{\mathbf{U}_k}(\pi_{\mathbf{X}_k}(\mathbf{x}_i + \mathbf{z}_{ij} + \mathbf{z}_j))\, d\mathbf{z}_j\right) \\
&\quad \left. \times \quad d\mathbf{z}_{ij}\right) d\mathbf{x}_i \\
&= \int_{\mathbf{X}_i} \mathbb{1}_{A\cap\mathbf{U}_i}(\mathbf{x}_i) \left(\int_{\mathbf{Z}_i} \prod_{j\in\mathcal{D}(i)} \mathbb{1}_{\mathbf{U}_j}(\pi_{\mathbf{X}_j}(\mathbf{x}_i + \mathbf{z}_i))\, d\mathbf{z}_i\right) d\mathbf{x}_i \\
&= \int_{\mathbf{X}_i} \mathbb{1}_A(\mathbf{x}_i)\, d\mu_i(\mathbf{x}_i) \\
&= \mu_i(A).
\end{aligned}$$

Thus, the inequality constraints (20) are satisfied. In addition, they become equalities on $\mathbf{U}_i$ in the case of our choice of μ_i.

We have proved that our choice of $\{\mu_1, \dots, \mu_m\}$ is a feasible one. Before we compute $\int_{\mathbf{X}_1} d\mu_1$, we make the useful observation that, since T is rooted in C_1, it holds $\mathcal{D}(1) = \{2, \dots, m\}$ and $\mathbf{X} = \mathbf{X}_1 \oplus \mathbf{Z}_1$. Thus, we have

$$\begin{aligned}
\int_{\mathbf{X}_1} d\mu_1 &= \int_{\mathbf{X}_1} \mathbb{1}_{\mathbf{U}_1}(x_1) \left(\int_{\mathbf{Z}_1} \prod_{j=2}^{m} \mathbb{1}_{\mathbf{U}_j}(\pi_{\mathbf{X}_j}(x_1 + \mathbf{z}_1))\, d\mathbf{z}_1\right) dx_1 \\
&= \int_{\mathbf{X}} \left(\prod_{i=1}^{m} \mathbb{1}_{\mathbf{U}_i}(\pi_{\mathbf{X}_i}(\mathbf{x}))\right) d\mathbf{x} \\
&= \int_{\mathbf{X}} \mathbb{1}_{\mathbf{K}}(\mathbf{x})\, d\mathbf{x} \\
&= \operatorname{vol} \mathbf{K}.
\end{aligned}$$

Then we have proved that $p^* \geq \operatorname{vol} \mathbf{K}$.

The other direction $p^* \leq \mathrm{vol}\,\mathbf{K}$ is easier. Since our previous choice $\mu_1^*, \ldots, \mu_m^*$ saturates the constraints (20) while enforcing the constraints (21), any other feasible solution $\mu_1, \ldots, \mu_m$ directly satisfies $\mu_i \leq \mu_i^*$. In particular, $\mu_1 \leq \mu_1^*$, and thus $\int_{\mathbf{X}_1} d\mu_1(dx_1) \leq \mathrm{vol}\,\mathbf{K}$. □

References

[1] B. Bollobás. Volume estimates and rapid mixing. Pages 151–180 in Flavors of geometry, MSRI Publ. 31, Cambridge University Press, Cambridge, UK, 1997

[2] M. E. Dyer, A. M. Frieze. On the complexity of computing the volume of a polyhedron. SIAM J. Comput. 17:967–974, 1988.

[3] G. Elekes. A geometric inequality and the complexity of measuring the volume. Discrete Comput. Geom. 1:289–292, 1986.

[4] D. Henrion, J.B. Lasserre, C. Savorgnan. Approximate volume and integration for basic semialgebraic sets. SIAM Review 51(4):722–743, 2009.

[5] B. Büeler, A. Enge, K. Fukuda. Exact volume computation for polytopes: a practical study. Pages 131–154 in Polytopes: combinatorics and computation, G. Kalai and G. M. Ziegler (Eds.), Birkhäuser Verlag, Basel, 2000.

[6] M. E. Dyer, A. M. Frieze, R. Kannan. A random polynomial-time algorithm for approximating the volume of convex bodies, J. ACM 38:1–17, 1991.

[7] C. Belisle. Slow hit-and-run sampling, Statist. Probab. Lett. 47:33–43, 2000.

[8] C. Belisle, E. Romeijn, R. L. Smith. Hit-and-run algorithms for generating multivariate distributions. Math. Oper. Res. 18:255–266, 1993.

[9] R. L. Smith. Efficient Monte Carlo procedures for generating points uniformly distributed over bounded regions. Oper. Res. 32:1296–1308, 1984.

[10] B. Cousins, S. Vempala. A practical volume algorithm. Math. Program. Comput. 8:133–160, 2016.

[11] J. B. Lasserre. Moments, positive polynomials and their applications. Imperial College Press, London, UK, 2010.

[12] J.B. Lasserre. Computing Gaussian and exponential measures of semi-algebraic sets. Adv. Appl. Math. 91:137–163, 2017.

[13] D. Henrion, M. Korda. Convex computation of the region of attraction of polynomial control systems. IEEE Trans. Autom. Control 59(2):297-312, 2014.

[14] C. Josz, D.K. Molzahn, M. Tacchi, S. Sojoudi. Transient stability analysis of power systems via occupation measures. To be presented at Innovative Smart Grid Technologies, 2019.

[15] J.B. Lasserre. Convergent SDP relaxations in polynomial optimization with sparsity. SIAM J. Optim. 17:822–843, 2006.

[16] D. Henrion, J. B. Lasserre, J. Loefberg. GloptiPoly 3: moments, optimization and semidefinite programming. Optimization Methods and Software 24(4-5):761-779, 2009.

[17] J. R. S. Blair, B. Peyton. An introduction to chordal graphs and clique trees. Pages 1–29 in Graph Theory and Sparse Matrix Computation, Springer, New York, 1993.

[18] J. B. Lasserre. Representation of chance-constraints with strong asymptotic guarantees. IEEE Control Systems Letters 1(1):50–55, 2017.

[19] J. H. Hubbard, B. Burke Hubbard. Vector calculus, linear algebra, and differential forms - A unified approach. 2nd Ed., Prentice Hall, 2002.

[20] R. Stanley, I. G. Macdonald, R. B. Nelsen. Solution of elementary problem E2701. The American Mathematical Monthly 86(5):396, 1979.