Journal of Optimization Theory and Applications (2022) 194:465–491
https://doi.org/10.1007/s10957-022-02034-x

Simulated Annealing for Convex Optimization: Rigorous Complexity Analysis and Practical Perspectives

Riley Badenbroek[1] · **Etienne de Klerk**[2]

Received: 5 August 2021 / Accepted: 4 April 2022 / Published online: 5 May 2022

Abstract
We give a rigorous complexity analysis of the simulated annealing algorithm by Kalai and Vempala (Math Oper Res 31(2):253–266, 2006) using the type of temperature update suggested by Abernethy and Hazan (International Conference on Machine Learning, 2016). The algorithm only assumes a membership oracle of the feasible set, and we prove that it returns a solution in polynomial time which is near-optimal with high probability. Moreover, we propose a number of modifications to improve the practical performance of this method, and present some numerical results for test problems from copositive programming.

Keywords Simulated annealing · Convex optimization · Hit-and-run sampling · Semidefinite and copositive programming

Mathematics Subject Classification 90C25 · 90C59

1 Introduction

Let $K \subseteq \mathbb{R}^n$ be a convex body, and suppose that only a membership oracle of K is available. Let $\langle \cdot, \cdot \rangle$ be an inner product on $\mathbb{R}^n$, and fix a unit vector $c \in \mathbb{R}^n$. We are interested in the problem

Communicated by Luis F. Zuluaga.

✉ Etienne de Klerk
e.deklerk@uvt.nl

Riley Badenbroek
badenbroek@ese.eur.nl

1 Erasmus University Rotterdam, Rotterdam, The Netherlands

2 Tilburg University, Tilburg, The Netherlands

$$\min_{x \in K} \langle c, x \rangle. \tag{1}$$

One approach to solve problems of this type is using simulated annealing, a paradigm of randomized algorithms for general optimization introduced by Kirkpatrick et al. [14]. It features a so-called temperature parameter that decreases during the run of the simulated annealing algorithm. At high temperatures, the method explores the feasible set relatively freely, also moving to solutions that have worse objective values than the current point. As the temperature decreases, so does the probability that a solution with a worse objective value is accepted. Kalai and Vempala [12] showed that, for convex optimization, a polynomial-time simulated annealing algorithm exists. Specifically, their algorithm returns a feasible solution that is near-optimal with high probability in polynomial time. (A recent algorithm by Lee et al. [15] has an asymptotically better complexity.)

Abernethy and Hazan [1] recently clarified that Kalai and Vempala's algorithm is closely related to a specific interior point method. In general, interior point methods for convex bodies require a so-called self-concordant barrier of the feasible set. It was shown by Nesterov and Nemirovskii [23] that every open convex set that does not contain an affine subspace is the domain of a self-concordant barrier, known as the universal barrier. However, it is not known how to compute the gradients and Hessians of this barrier in general.

The interior point method to which Kalai and Vempala's algorithm corresponds uses the so-called entropic barrier over K, to be defined later. This barrier was introduced by Bubeck and Eldan [7], who also established the self-concordance of the barrier and analyzed its complexity parameter ϑ.

Drawing on the connection to interior point methods, Abernethy and Hazan [1] proposed a new temperature schedule for Kalai and Vempala's algorithm. This schedule does not depend on the dimension n of the problem, but on the complexity parameter ϑ of the entropic barrier. Our goal is to prove in detail that simulated annealing with this new temperature schedule returns a solution in polynomial time which is near-optimal with high probability. Moreover, we aim to investigate the practical applicability of this method. Our experiments suggest that it is very difficult to implement a practical simulated annealing algorithm for copositive programming using hit-and-run sampling. Although these are negative results, we find it important to communicate them, in order to stimulate research into algorithms for copositive programming that are not sampling-based, e.g., cutting plane methods.

1.1 Algorithm Statement

Central to simulated annealing is a family of exponential-type probability distributions known as Boltzmann distributions.

Definition 1.1 Let $K \subseteq \mathbb{R}^n$ be a convex body, and let $\theta \in \mathbb{R}^n$. Let $\langle \cdot, \cdot \rangle$ be an inner product. Then, the *Boltzmann distribution with parameter* θ is the probability distribution supported on K having density with respect to the Lebesgue measure proportional to $x \mapsto e^{\langle \theta, x \rangle}$.

Throughout this work, we will use $\Sigma(\theta)$ to refer to the covariance matrix of the Boltzmann distribution with parameter $\theta \in \mathbb{R}^n$. If $\langle \cdot, \cdot \rangle$ is some reference inner product, then $\langle x, y \rangle_\theta := \langle x, \Sigma(\theta) y \rangle$ for any $\theta \in \mathbb{R}^n$. Moreover, let $\| \cdot \|_\theta$ denote the norm induced by the inner product $\langle \cdot, \cdot \rangle_\theta$.

The procedure Kalai and Vempala [12] use to generate samples on K is called *hit-and-run* sampling. This Markov chain Monte Carlo method was introduced by Smith [26] to sample from the uniform distribution over a bounded convex set. Later, it was generalized to absolutely continuous distributions (see for example Bélisle et al. [4]). The details of hit-and-run sampling are given in Algorithm 1.

Algorithm 1 The hit-and-run sampling procedure

Input: Membership oracle for a convex body $K \subseteq \mathbb{R}^n$; probability distribution μ to sample from (i.e., the target distribution); covariance operator $\Sigma : \mathbb{R}^n \to \mathbb{R}^n$; starting point $x \in K$; number of hit-and-run steps $\ell \in \mathbb{N}$.

1: $X_0 \leftarrow x$
2: **for** $i \in \{1, ..., \ell\}$ **do**
3: Sample direction D_i from a $\mathcal{N}(0, \Sigma)$-distribution
4: Sample X_i from the univariate distribution μ restricted to $K \cap \{X_{i-1} + tD_i : t \in \mathbb{R}\}$
5: **end for**
6: **return** X_ℓ

Note that if $\langle \cdot, \cdot \rangle$ is the Euclidean inner product, the covariance operator Σ in Algorithm 1 can be represented as a matrix in $\mathbb{R}^{n \times n}$.

In each iteration k of Kalai and Vempala's algorithm [12], the temperature T_k is lowered. Then, hit-and-run samples are generated whose target distribution is the Boltzmann distribution with parameter $\theta_k = -c/T_k$. These random walks use an approximation $\widehat{\Sigma}(\theta_{k-1})$ of $\Sigma(\theta_{k-1})$ to generate search directions. With these samples, an approximation $\widehat{\Sigma}(\theta_k)$ of $\Sigma(\theta_k)$ is formed, which will then be used in the next iteration. As k grows sufficiently large, so does the norm of θ_k. The Boltzmann distributions with parameter θ_k will then concentrate more and more probability mass close to the set of optimal solutions to (1). For sufficiently large k, any sample from such a Boltzmann distribution is near-optimal with high probability.

One thing that needs further clarification is how to decrease the temperature in each iteration. In their original paper, Kalai and Vempala [12] show that the algorithm returns a near-optimal solution with high probability, for the temperature update (cf. line 5 in Algorithm 2)

$$T_k = \left(1 - \frac{1}{\sqrt{n}}\right) T_{k-1}, \tag{2}$$

in $m = O^*(\sqrt{n})$ iterations, where O^* suppresses polylogarithmic terms in the problem parameters. As mentioned above, Abernethy and Hazan's alternative temperature schedule depends on the complexity parameter of the entropic barrier. This function is defined as follows.

Definition 1.2 Let $K \subseteq \mathbb{R}^n$ be a convex body. Define the function $f : \mathbb{R}^n \to \mathbb{R}$ as $f(\theta) = \ln \int_K e^{\langle \theta, x \rangle} \, \mathrm{d}x$. Then, the convex conjugate f^* of f,

$$f^*(x) = \sup_{\theta \in \mathbb{R}^n} \{\langle \theta, x \rangle - f(\theta)\},$$

is called the *entropic barrier* for K.

Bubeck and Eldan [7] showed that f^* is a self-concordant barrier for K with complexity parameter $\vartheta \leq n + o(n)$. The complexity parameter of f^* is

$$\vartheta := \sup_{x \in K} \langle Df^*(x), [D^2 f^*(x)]^{-1} Df^*(x) \rangle = \sup_{\theta \in \mathbb{R}^n} \langle \theta, \Sigma(\theta)\theta \rangle = \sup_{\theta \in \mathbb{R}^n} \|\theta\|_\theta^2, \quad (3)$$

where we refer the reader to [3, 7] for more details.

Abernethy and Hazan [1] propose the temperature update

$$T_k = \left(1 - \frac{1}{4\sqrt{\vartheta}}\right) T_{k-1}, \quad (4)$$

which will lead to $m = O^*(\sqrt{\vartheta})$ iterations. As noted above, we have $\vartheta \leq n + o(n)$ in general, but it is not currently known if $\vartheta < n$ for any convex bodies. In particular, the temperature update (4) only improves on (2) if $\vartheta < n/16$, which is not known to hold for any convex body. We show in Appendix 1 to this paper that, for the Euclidean unit ball in $\mathbb{R}^n$, numerical evidence suggests that $\vartheta = (n+1)/2$. We therefore consider a variation on the temperature schedule (4) suggested by Abernethy and Hazan, namely

$$T_k = \left(1 - \frac{1}{\alpha\sqrt{\vartheta}}\right) T_{k-1} \text{ for some } \alpha > 1 + 1/\sqrt{\vartheta}, \quad (5)$$

which corresponds to (4) when $\alpha = 4$, but gives larger temperature reductions when $\alpha < 4$. We will refer to (5) as Abernethy–Hazan-type temperature updates. If $\vartheta < n$, this may result in a larger temperature decrease than the Kalai and Vempala scheme (2), for a suitable choice of the parameter α.

The algorithm by Kalai and Vempala [12] that uses a temperature schedule of the type introduced by Abernethy and Hazan [1] is now given in Algorithm 2.

1.2 Contributions and Outline of this Paper

Abernethy and Hazan [1] do not give a rigorous analysis of the temperature schedule (4) in their paper, only a sketch of the proof. In this paper, we provide the full details for the more general schedule (5). In doing so, we also provide some details that were omitted in the original work by Kalai and Vempala [12], that concern the application of a theorem by Rudelson [25]. Moreover, we discuss the perspectives for practical computation with Algorithm 2. Finally, we propose some heuristic improvements to Algorithm 2 to obtain speed-up. Many of these results also appear in the PhD thesis [2] of the first author.

Algorithm 2 Algorithm by Kalai and Vempala [12] using temperature schedule of type introduced by Abernethy and Hazan [1]

Input: normalized (with respect to $\|\cdot\|$) vector $c \in \mathbb{R}^n$; membership oracle of a convex body $K \subseteq \mathbb{R}^n$; radius $R \geq 1$ of a ball containing K; complexity parameter $\vartheta \leq n + o(n)$ of the entropic barrier over K; $x_0 \in K$ drawn randomly from the uniform distribution over K; update parameter $\alpha > 1 + 1/\sqrt{\vartheta}$; number of phases $m \in \mathbb{N}$; number of hit-and-run steps $\ell \in \mathbb{N}$; number of covariance update samples $N \in \mathbb{N}$; approximation $\widehat{\Sigma}(0)$ of $\Sigma(0)$ satisfying $\frac{1}{2}\widehat{\Sigma}(0) \preceq \Sigma(0) \preceq 2\widehat{\Sigma}(0)$.

Output: x_m such that $\langle c, x_m\rangle - \min_{x\in K}\langle c, x\rangle \leq \varepsilon$ at terminal iteration m.

$X_0 \leftarrow x_0$

$\theta_0 = 0$

$T_0 \leftarrow 2\alpha R$

for $k \in \{1, ..., m\}$ **do**

  $T_k \leftarrow \left(1 - \frac{1}{\alpha\sqrt{\vartheta}}\right) T_{k-1}$

  $\theta_k \leftarrow -c/T_k$

  Generate X_k by applying hit-and-run sampling to the Boltzmann distribution with parameter θ_k, starting the walk from X_{k-1}, taking ℓ steps, drawing directions from a $\mathcal{N}(0, \widehat{\Sigma}(\theta_{k-1}))$-distribution

  for $j \in \{1, ..., N\}$ **do**

    Generate Y_{jk} by applying hit-and-run sampling to the Boltzmann distribution with parameter θ_k, starting the walk from X_{k-1}, taking ℓ steps, drawing directions from a $\mathcal{N}(0, \widehat{\Sigma}(\theta_{k-1}))$-distribution

  end for

  $\widehat{\Sigma}(\theta_k)v \leftarrow \frac{1}{N}\sum_{j=1}^N \langle Y_{jk}, v\rangle Y_{jk} - \frac{1}{N}\sum_{j=1}^N \langle Y_{jk}, v\rangle \left(\frac{1}{N}\sum_{l=1}^N Y_{lk}\right)$ for all $v \in \mathbb{R}^n$

end for

return X_m

We start with a review of useful facts on probability distributions in Sect. 2. Then, Sect. 4 proves Algorithm 2 returns a solution that is near-optimal with high probability. In Sect. 5, we discuss the complexity of Algorithm 2. In Sect. 6, we look at the behavior of hit-and-run sampling for optimization over the doubly nonnegative cone and suggest some heuristic improvements. We then evaluate the resulting algorithm on problems from copositive programming (due to Berman et al. [5]) in Sect. 7.

2 Preliminaries

We will use the total variation distance to measure if the probability distribution of a hit-and-run sample is close to the target distribution.

Definition 2.1 Let $(K, \mathcal{F})$ be a measurable space. For two probability distributions μ and φ over this space, their *total variation distance* is

$$\|\mu - \varphi\|_{\mathrm{TV}} := \sup_{A\in\mathcal{F}} |\mu(A) - \varphi(A)|.$$

A useful property of the total variation distance is that it allows coupling of random variables, as the following lemma asserts.

Lemma 2.2 (e.g., [17, Proposition 4.7]) *Let X be a random variable on K with distribution μ, and let φ be a different probability distribution on K. If $\|\mu - \varphi\|_{\mathrm{TV}} = p$, we can construct another random variable Y on K distributed according to φ such that $\mathbb{P}\{X = Y\} = 1 - p$. Similarly, given two distributions μ and φ on K such that $\|\mu - \varphi\|_{\mathrm{TV}} = p$, there exists a joint distribution ν on $K \times K$, with marginals μ and φ, respectively, such that if $(X, Y) \sim (K \times K, \nu)$, one has $X \sim (K, \mu)$, $Y \sim (K, \varphi)$, and $\mathbb{P}_\nu\{X = Y\} \geq 1 - p$.*

We can now state the following mixing time result. It gives the number of hit-and-run steps one has to take before the distribution of the hit-and-run sample is sufficiently close to the target distribution. The result given here is a corollary of a result by Lovász and Vempala [19, Theorem 1.1].

Theorem 2.3 *Let $K \subset \mathbb{R}^n$ be a convex body. Suppose $\theta_0, \theta_1 \in \mathbb{R}^n$ satisfy $\Delta\theta := \|\theta_0 - \theta_1\|_{\theta_0} < 1$. Pick $p > 0$, and suppose we have an invertible matrix $\widehat{\Sigma}(\theta_0)$ such that*

$$\frac{1}{2}\widehat{\Sigma}(\theta_0)^{-1} \preceq \Sigma(\theta_0)^{-1} \preceq 2\widehat{\Sigma}(\theta_0)^{-1}. \tag{6}$$

Consider a hit-and-run random walk as in Algorithm 1 applied to the Boltzmann distribution μ_1 with parameter θ_1 from a random starting point drawn from a Boltzmann distribution μ_0 with parameter θ_0. Let μ^ℓ be the distribution of the hit-and-run point after ℓ steps of hit-and-run sampling applied to μ_1, where the directions are drawn from a $\mathcal{N}(0, \widehat{\Sigma}(\theta_0))$-distribution. Then, there exists an absolute constant $C > 0$, such that, after

$$\ell = C\frac{n^3}{(1 - \Delta\theta)^4}\log^5\left(\frac{n}{p^2(1 - \Delta\theta)^4}\right) \tag{7}$$

hit-and-run steps, we have $\|\mu^\ell - \mu_1\|_{\mathrm{TV}} \leq p$.

We omit the proof, since it is a straightforward consequence of Lovász and Vempala [19, Theorem 1.1], applied to the Boltzmann distribution. The interested reader may find a detailed proof in [2, Theorem 4.14].

Note that the theorem above requires an approximation of the covariance of a distribution that is in some sense 'close' to the target distribution. This is why Algorithm 2 approximates the covariance of every distribution that it encounters: This covariance is then used in the next iteration to generate hit-and-run directions.

One may reformulate the Assumption (6) in the theorem by using the following result from Horn and Johnson [11].

Lemma 2.4 ([11, Corollary 7.7.4(a)], see also [2, Lemma 2.4]) *Let A, B be positive definite, self-adjoint linear operators from $\mathbb{R}^n$ to $\mathbb{R}^n$. Then, $A \succeq B$ if and only if $B^{-1} \succeq A^{-1}$.*

Consequently, we have, for $k = 0, 1, \ldots,$

$$\frac{1}{2}\widehat{\Sigma}(\theta_k) \preceq \Sigma(\theta_k) \preceq 2\widehat{\Sigma}(\theta_k) \iff \frac{1}{2}\widehat{\Sigma}(\theta_k)^{-1} \preceq \Sigma(\theta_k)^{-1} \preceq 2\widehat{\Sigma}(\theta_k)^{-1}.$$

A key point of the analysis is to show that these conditions hold for each iteration k.

3 Approximation of the Covariance Matrix

To guarantee the required approximation of the covariance matrix, Kalai and Vempala [12] use the following corollary to a theorem by Rudelson [25].

Theorem 3.1 ([12, Theorem A.1]) *Let φ be a log-concave probability distribution over $\mathbb{R}^n$ with mean 0 and identity covariance (i.e., isotropic), and let $\rho \in (0, 1)$. Let $X_1, ..., X_N$ be independent samples from φ. Then, there exist absolute constants $C_1 > 0$ and $C_2 > 0$ such that, if*

$$N \geq C_1 n \log^{C_2}(n/\rho),$$

we have

$$\left\| \frac{1}{N} \sum_{j=1}^{N} X_j X_j^\top - I \right\| \leq \frac{1}{4},$$

with probability $1 - \rho$, where the norm is the spectral (operator) norm.

This theorem cannot be directly applied in the setting of Algorithm 2 for three reasons: The hit-and-run samples do not follow the target distribution φ, the samples are not independent, and φ does not have mean 0 and identity covariance, i.e., φ is not isotropic. While Kalai and Vempala (correctly) state that Theorem 3.1 can be extended to the hit-and-run setting without significantly changing the number of samples N, a formal proof is not given, and we will provide the missing details in this section, since this is not straightforward.

We first show how to extend Theorem 3.1 to the non-isotropic case. To this end, we will need two results on log-concave random variables. The first is that the sum of independent log-concave random variables is again log-concave.

Lemma 3.2 *Let X, Y be independent random variables in $\mathbb{R}^n$ with log-concave density functions f and g, respectively. Then, $X + Y$ is a log-concave random variable.*

Proof Theorem 7 in [24] shows that $x \mapsto \int_{\mathbb{R}^n} f(x - y)g(y)\,\mathrm{d}y$ is log-concave on $\mathbb{R}^n$. This function is precisely the density function of $X + Y$. □

The second result is a concentration result for log-concave distributions.

Lemma 3.3 (Lemma 3.3 from [18] (adapted from [20])) *Let X be a random variable with a log-concave distribution. Denote $\mathbb{E}(\|X - \mathbb{E}(X)\|_2^2) =: \sigma^2$. Then, for all $t > 1$,*

$$\mathbb{P}\{\|X - \mathbb{E}(X)\|_2 \leq t\sigma\} \geq 1 - e^{1-t}.$$

We now extend Theorem 3.1 to the non-isotropic case.

Corollary 3.4 *Let φ be a log-concave probability distribution over $\mathbb{R}^n$ with mean μ_φ and covariance Σ_φ, and let $\rho \in (0, 1)$. Let $X_1, ..., X_N$ be independent samples from φ, where*

$$N = \left\lceil \max \left\{ C_1 n \log^{C_2}(2n/\rho), 4n \left(1 + \ln\left(\frac{2}{\rho}\right)\right)^2 \right\} \right\rceil, \tag{8}$$

and C_1 and C_2 are the absolute constants from Theorem 3.1, and let

$$\hat{\mu} = \frac{1}{N}\sum_{i=1}^{N} X_i, \quad \hat{\Sigma} = \frac{1}{N}\sum_{i=1}^{N}(X_i - \hat{\mu})(X_i - \hat{\mu})^\top.$$

Then, we have

$$\frac{1}{2}\hat{\Sigma} \preceq \Sigma_\varphi \preceq \frac{3}{2}\hat{\Sigma}, \tag{9}$$

with probability $1 - \rho$.

Proof One may assume w.l.o.g. that $\Sigma_\varphi = I$, since the statement of the theorem is invariant under invertible linear transformations. Indeed, if a random variable X with distribution φ is replaced by AX for some linear transformation A, then the new covariance matrix and its approximation satisfy

$$\Sigma_\varphi^{(\text{new})} = A\Sigma_\varphi A^* \text{ and } \hat{\Sigma}^{(\text{new})} = A\hat{\Sigma}A^*.$$

By choosing $A = \Sigma_\varphi^{-1/2}$, we therefore get the identity covariance. Moreover,

$$\frac{1}{2}\hat{\Sigma} \preceq \Sigma_\varphi \preceq \frac{3}{2}\hat{\Sigma} \iff \frac{1}{2}A\hat{\Sigma}A^* \preceq A\Sigma_\varphi A^* \preceq \frac{3}{2}A\hat{\Sigma}A^*.$$

Assuming therefore w.l.o.g. that $\Sigma_\varphi = I$, the random variable $X - \mu_\varphi$ has an isotropic log-concave distribution, and after setting

$$\Sigma' = \frac{1}{N}\sum_{i=1}^{N}(X_i - \mu_\varphi)(X_i - \mu_\varphi)^\top,$$

Theorem 3.1 yields $\left\|\Sigma' - I\right\| \leq \frac{1}{4}$, with probability $1 - \rho/2$. It therefore suffices to show that Σ' and $\hat{\Sigma}$ are 'sufficiently close.' To this end, direct calculation yields

$$\hat{\Sigma} - \Sigma' = -(\mu_\varphi - \hat{\mu})(\mu_\varphi - \hat{\mu})^\top,$$

so that

$$\left\|\hat{\Sigma} - \Sigma'\right\| = \|\hat{\mu} - \mu_\varphi\|_2^2.$$

Note that $\hat{\mu} = \frac{1}{N}\sum_{i=1}^{N} X_i$ is a log-concave random vector, by Lemma 3.2. We may therefore bound right-hand side of the last equality via the concentration inequality of

Lemma 8 to obtain

$$\mathbb{P}\left\{\|\hat{\mu}-\mu_{\varphi}\|_2^2 \leq \frac{n}{N}\left(1+\ln\left(\frac{2}{\rho}\right)\right)^2\right\} \geq 1-\rho/2,$$

where we used that the variance of each component of X_i is 1 for all i. Thus, if $N \geq 4n\left(1+\ln\left(\frac{2}{\rho}\right)\right)^2$, one has

$$\|\hat{\Sigma}-I\| \leq \|\Sigma'-I\| + \left\|\hat{\Sigma}-\Sigma'\right\| \leq \frac{1}{4}+\frac{1}{4}=\frac{1}{2},$$

with probability at least $(1-\rho/2)^2 > 1-\rho$. This implies (9). □

We proceed to show that we may replace independent sampling by hit-and-run sampling in Corollary 3.4 in a well-defined sense. To this end, fix a starting vector X_0 and a tolerance $q > 0$, and consider the following two sequences of random variables:

- I.i.d. X_i ($i \in \{1,\ldots,N\}$) drawn independently from φ.
- Y_i obtained via ℓ steps of hit-and-run from starting point X_0 with ($i \in \{1,\ldots,N\}$) as in Step 9 of Algorithm 2. Here, we assume that the ℓ steps are sufficient to guarantee that the total variation distance between the distribution of Y_i and φ is at most a given $p \in (0,1)$ (with reference to Theorem 2.3).

A key observation is that we may assume, without loss of generality, that $X_i = Y_i$ for all ($i \in \{1,\ldots,N\}$) with high probability.

Proposition 3.1 *Let X_i, Y_i ($i \in \{1,\ldots,N\}$) be as above. Without loss of generality, one may assume that the joint distribution of X_i and Y_i satisfies $\mathbb{P}\{X_i = Y_i\} \geq 1-p$ for each $i \in \{1,\ldots,N\}$.*

Proof The X_i all have distribution φ on K and the Y_i all have the same distribution, say μ on K (that depends on X_0).

Moreover, by assumption $\|\mu-\varphi\|_{TV} \leq p$. By the second part of Lemma 2.2, there exists a joint distribution, say ν on $K \times K$ with marginals φ and μ, so that $(X,Y) \sim (\nu, K\times K)$ implies $\mathbb{P}_\nu\{X=Y\} \geq 1-p$, as well as $X \sim (\varphi, K)$ and $Y \sim (\mu, K)$.

We now replace (i.e., couple) the random variables Y_i with new random variables Y_i' so that $(X_i, Y_i') \sim (\nu, K\times K)$ for each $i \in \{1,\ldots,N\}$.

The important point is that we now have $\mathbb{P}[X_i = Y_i'] \geq 1-p$ for each $i \in \{1,\ldots,N\}$. Moreover, the random variables Y_i' will be indistinguishable from the hit-and-run random variables Y_i, in the sense that both $Y_i \sim (\mu, K)$ and $Y_i' \sim (\mu, K)$ for all i. □

As a result, Corollary 3.4 still holds if we replace the X_i by the Y_i for all i, but now with probability $(1-\rho)-Np$, by the union bound.

4 Proof of Convergence

We continue by proving that Algorithm 2 converges to the optimum in polynomial time. The following result was established by Kalai and Vempala [12] for linear functions, and extended from linear to convex functions $h : \mathbb{R}^n \to \mathbb{R}$ by De Klerk and Laurent [10].

Lemma 4.1 ([10, Corollary 1]) *Let $K \subseteq \mathbb{R}^n$ be a convex body. For any convex function $h : \mathbb{R}^n \to \mathbb{R}$ and temperature $T > 0$, we have*

$$\frac{\int_K h(x)e^{-h(x)/T}\,\mathrm{d}x}{\int_K e^{-h(x)/T}\,\mathrm{d}x} \leq nT + \min_{x\in K} h(x).$$

The main step in the analysis of Algorithm 2 is thus to show that we maintain a good approximation $\widehat{\Sigma}(\theta_k)$ of $\Sigma(\theta_k)$ for all k, to guarantee that the hit-and-run sampling continues to work in all iterations, as discussed in the previous section.

Theorem 4.2 *Consider the setting of Algorithm 2. Let $\alpha > 1$ be such that $\Delta\theta = \sqrt{\vartheta}/(\alpha\sqrt{\vartheta} - 1) < 1$, let $q \in (0, 1]$,*

$$m = \left\lceil \frac{\log\left(q\varepsilon/(4\alpha nR)\right)}{\log\left(1 - 1/(\alpha\sqrt{\vartheta})\right)} \right\rceil, \tag{10}$$

$$\rho = \frac{q}{2m}, \tag{11}$$

$$N \text{ as in (8)}, \tag{12}$$

$$p = \frac{q}{2Nm}, \tag{13}$$

and let ℓ be as in (7). (Note that ℓ depends on n, p, and $\Delta\theta$.) With these inputs, Algorithm 2 that returns a solution X_m with

$$\mathbb{P}\left\{\langle c, X_m\rangle - \min_{x\in K}\langle c, x\rangle \leq \varepsilon\right\} \geq 1 - q.$$

Proof First, let us show that the conditions of Theorem 2.3 are satisfied. Note that

$$\|c\|_0^2 = \langle c, \Sigma(0)c\rangle = \frac{\int_K \langle c, x - \mu_{\theta,K}\rangle^2\,\mathrm{d}x}{\int_K \mathrm{d}x},$$

because $\Sigma(0)$ is the covariance matrix of the uniform distribution. Since $\|c\| = 1$ and K is contained in a ball with radius R,

$$\|c\|_0^2 \leq \frac{\int_K \|c\|^2\|x - \mu_{\theta,K}\|^2\,\mathrm{d}x}{\int_K \mathrm{d}x} \leq \frac{\int_K 1^2(2R)^2\,\mathrm{d}x}{\int_K \mathrm{d}x} = (2R)^2.$$

Hence, for $k = 1$,

$$\|\theta_1 - \theta_0\|_{\theta_0} = \frac{\|c\|_0}{T_1} \leq \frac{2R}{T_1} = \frac{2R}{2\alpha R\left(1 - 1/(\alpha\sqrt{\vartheta})\right)} = \Delta\theta.$$

For all $k > 1$, our choice of θ_k and (3) yield

$$\|\theta_k - \theta_{k-1}\|_{\theta_{k-1}} = \left(\frac{T_{k-1}}{T_k} - 1\right)\|\theta_{k-1}\|_{\theta_{k-1}} \leq \left(\frac{1}{1 - 1/(\alpha\sqrt{\vartheta})} - 1\right)\sqrt{\vartheta} = \Delta\theta.$$

Throughout all iterations $k \leq m$ of Algorithm 2, we maintain

$$\tfrac{1}{2}\widehat{\Sigma}(\theta_k)^{-1} \preceq \Sigma(\theta_k)^{-1} \preceq 2\widehat{\Sigma}(\theta_k)^{-1}, \tag{14}$$

with probability $1 - m(\rho + Np) = 1 - q$, by the analysis in the previous section. Thus, the conditions of Theorem 2.3 are satisfied with high probability.

By the first part of Lemma 2.2, X_m is equal to a random variable drawn from a Boltzmann distribution with parameter θ_m with probability at least $1 - p$. Thus, Markov's inequality and Lemma 4.1 show that

$$\mathbb{P}\left\{\langle c, X_m\rangle - \min_{x\in K}\langle c, x\rangle > \varepsilon\right\} \leq \frac{\mathbb{E}\left[\langle c, X_m\rangle - \min_{x\in K}\langle c, x\rangle\right]}{\varepsilon} \leq \frac{nT_m}{\varepsilon} \leq \tfrac{1}{2}q, \tag{15}$$

where the final inequality uses the chosen value of m as follows:

$$T_m = 2\alpha R\left(1 - \frac{1}{\alpha\sqrt{\vartheta}}\right)^m \leq 2\alpha R\frac{q\varepsilon}{4\alpha nR} = \frac{q\varepsilon}{2n}.$$

□

5 Complexity Analysis and Discussion

We saw that for some combination of inputs, Algorithm 2 returns a solution which is near-optimal with high probability. Let us now consider the number of membership oracle calls required for this configuration. It was noted in, for example, [2, Section 4.1] that the number of oracle calls for one hit-and-run walk is $O^*(\ell)$. Hence, Algorithm 2 uses $O^*(mN\ell)$ oracle calls.

First, let us look at the number of iterations from (10). Since

$$\frac{-1}{\log\left(1 - 1/(\alpha\sqrt{\vartheta})\right)} = O(\alpha\sqrt{\vartheta}),$$

we have $m = O(\sqrt{\vartheta})$ for fixed α. Next, the number of samples from (8) satisfies

$$N = O(n \ln(mn/q)) = O(n \ln(\sqrt{\vartheta} n/q)) = O^*(n).$$

Next, we bound the number of hit-and-run steps. Recall (7) shows that

$$\ell = O\left(\frac{n^3}{(1-\Delta\theta)^4} \log^5\left(\frac{n}{p^2(1-\Delta\theta)^4}\right)\right). \tag{16}$$

For the $\Delta\theta$ from Theorem 4.2 and the value of p from (13), (16) shows

$$\ell = O\left(n^3 \log^5\left(\frac{n\vartheta}{q^2}\right)\right) = O^*\left(n^3\right).$$

Summarizing, the total number of oracle calls by Algorithm 2 is

$$O^*(mN\ell) = O^*\left(n^{4.5}\right),$$

where we used Bubeck and Eldan's result [7] that $\vartheta \leq n + o(n)$.

Thus, we have given a rigorous complexity analysis of the algorithm by Kalai and Vampala [12] using the temperature update of Abernethy and Hazan [1]. Our final bound for the number of oracle calls coincides with the $O^*(n^{4.5})$ bound claimed in [12].

Initialization

One might still wonder how to generate a good estimate $\widehat{\Sigma}(0)$ of the uniform covariance matrix $\Sigma(0)$ to start Algorithm 2. The 'rounding the body' procedure from Lovász and Vempala [18] is suitable for this purpose. This procedure returns a $\widehat{\Sigma}(0)$ for which

$$\mathbb{P}\left\{\tfrac{1}{2}\widehat{\Sigma}(0) \preceq \Sigma(0) \preceq 2\widehat{\Sigma}(0)\right\} \geq 1 - \frac{1}{n}.$$

By Lemma 2.4, $\frac{1}{2}\widehat{\Sigma}(0) \preceq \Sigma(0) \preceq 2\widehat{\Sigma}(0)$ if and only if $\frac{1}{2}\widehat{\Sigma}(0)^{-1} \succeq \Sigma(0)^{-1} \succeq 2\widehat{\Sigma}(0)^{-1}$, so the starting condition for Algorithm 2 can be satisfied by the 'rounding the body' procedure. The number of calls to the membership oracle for this procedure is $O^*(n^4)$, which is overshadowed by the complexity of Algorithm 2 itself.

6 Numerical Examples on the Doubly Nonnegative Cone

Having established the theoretical complexity of Algorithm 2, we now move to the second goal of this work: investigating the practical perspectives of this method. We will test Algorithm 2 on the problem of determining if a matrix is copositive, which is known to be a co-NP-complete problem [22].

To define this problem, let $\mathbb{S}^{m\times m}$ denote the space of real symmetric $m\times m$ matrices. A matrix $C\in\mathbb{S}^{m\times m}$ is called copositive if $x^\top Cx\geq 0$ for all $x\in\mathbb{R}^m_+$ (see Bomze [6] for a survey on copositive programming and its applications).

The standard SDP relaxation of the problem of checking for copositivity of C is the following:

$$\inf\left\{\langle C,X\rangle : \sum_{i=1}^m\sum_{j=1}^m X_{ij}\leq 1, X\geq 0, X\succeq 0\right\}, \tag{17}$$

where $\langle\cdot,\cdot\rangle$ is the trace inner product. If the value of (17) is nonnegative, the matrix C is copositive. However, since we place a nonnegativity constraint on every element of the matrix X, the Newton system in every interior point iteration is of size $O(m^2\times m^2)$, which quickly leads to impractical computation times (see, e.g., Burer [8]).

Before we can apply Algorithm 2, we need to translate (17) to a problem over $\mathbb{R}^{m(m+1)/2}$. The approach is standard: for any $A=[A_{ij}]_{ij}\in\mathbb{S}^{m\times m}$, define

$$\operatorname{svec}(A):=(A_{11},\sqrt{2}A_{12},\ldots,\sqrt{2}A_{1m},A_{22},\sqrt{2}A_{23},\ldots,\sqrt{2}A_{2m},\ldots,A_{mm})^\top,$$

such that $\operatorname{svec}(A)\in\mathbb{R}^{m(m+1)/2}$. If $\mathbb{R}^{m(m+1)/2}$ is endowed with the Euclidean inner product, the adjoint of svec is defined for every $a\in\mathbb{R}^{m(m+1)/2}$ as

$$\operatorname{smat}(a)=\begin{bmatrix} a_1 & a_2/\sqrt{2} & \ldots & a_m/\sqrt{2}\\ a_2/\sqrt{2} & a_{m+1} & \ldots & a_{2m-1}/\sqrt{2}\\ \vdots & \vdots & \ddots & \vdots\\ a_m/\sqrt{2} & a_{2m-1}/\sqrt{2} & \ldots & a_{m(m+1)/2}\end{bmatrix},$$

such that $\operatorname{smat}(a)\in\mathbb{S}^{m\times m}$. Moreover, $\operatorname{smat}(\operatorname{svec}(A))=A$ and $\operatorname{svec}(\operatorname{smat}(a))=a$ for all A and a. Now let $c=\operatorname{svec}(C)$. Problem (17) is equivalent to the following problem over $\mathbb{R}^{m(m+1)/2}$:

$$\inf\left\{\langle c,x\rangle : \sum_{i=1}^m\sum_{j=1}^m(\operatorname{smat}(x))_{ij}\leq 1, x\geq 0, \operatorname{smat}(x)\succeq 0\right\}. \tag{18}$$

Note that membership of the feasible set of (18) can be determined in $O(m^3)$ operations. Let $n=\frac{1}{2}m(m+1)$ be the number of variables in problem (18).

6.1 Covariance Approximation

First, we investigate how the quality of the covariance approximation depends on the walk length for problem (18). We take 20,000 hit-and-run samples from the uniform distribution over the feasible set of (18) with walk length 50,000 (directions are drawn from $\mathcal{N}(0,I)$ and the starting point is $\operatorname{svec}(mI+J)/(2e^\top\operatorname{svec}(mI+J))$, where J

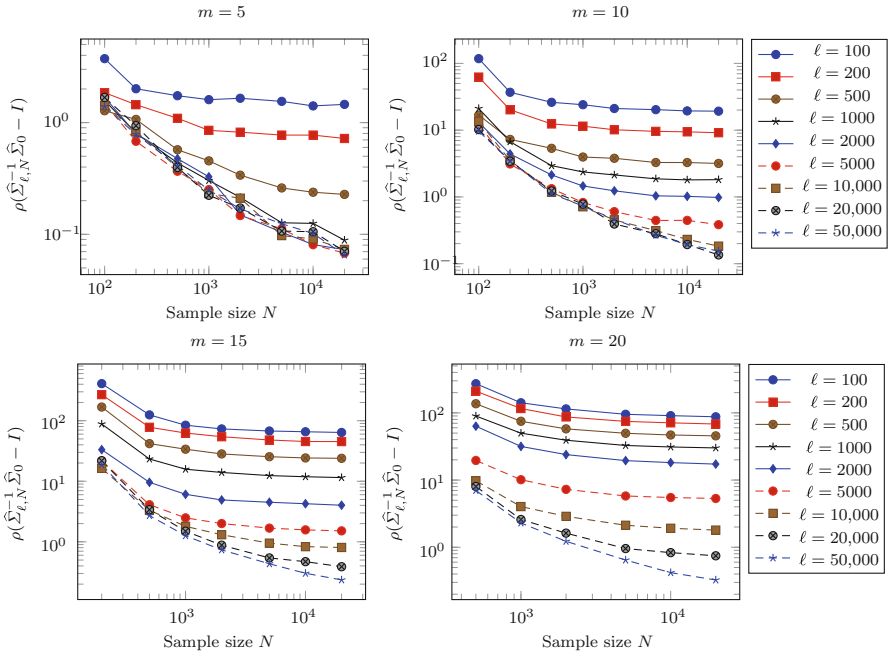

Fig. 1 Effect of sample size N and walk length ℓ on quality of uniform covariance approximation

is the all-ones matrix). These samples are used to create the estimate $\widehat{\Sigma}_0$. Then, the experiment is repeated for walk lengths $\ell \leq 50{,}000$ and sample sizes $N \leq 20{,}000$. We refer to these new estimates as $\widehat{\Sigma}_{\ell,N}$. We would like

$$-\epsilon y^\top \widehat{\Sigma}_{\ell,N} y \leq y^\top (\widehat{\Sigma}_0 - \widehat{\Sigma}_{\ell,N}) y \leq \epsilon y^\top \widehat{\Sigma}_{\ell,N} y \qquad \forall y \in \mathbb{R}^n,$$

for some small $\epsilon > 0$. This is equivalent to

$$-\epsilon x^\top x \leq x^\top (\widehat{\Sigma}_{\ell,N}^{-1/2} \widehat{\Sigma}_0 \widehat{\Sigma}_{\ell,N}^{-1/2} - I) x \leq \epsilon x^\top x \qquad \forall x \in \mathbb{R}^n,$$

i.e., we would like the spectral radius of $\widehat{\Sigma}_{\ell,N}^{-1/2} \widehat{\Sigma}_0 \widehat{\Sigma}_{\ell,N}^{-1/2} - I$ to be at most ϵ. Because the spectra of $\widehat{\Sigma}_{\ell,N}^{-1/2} \widehat{\Sigma}_0 \widehat{\Sigma}_{\ell,N}^{-1/2} - I$ and $\widehat{\Sigma}_{\ell,N}^{-1} \widehat{\Sigma}_0 - I$ are the same, we focus on the spectral radius $\rho(\widehat{\Sigma}_{\ell,N}^{-1} \widehat{\Sigma}_0 - I)$.

The result is shown in Fig. 1, where m refers to the size of the matrices in (17). Hence, the covariance matrices in question are of size $\frac{1}{2}m(m+1) \times \frac{1}{2}m(m+1)$.

One major conclusion from Fig. 1 is that the trajectory toward zero is relatively slow. To show that simply adding more samples with higher walk lengths will in practice not be feasible, we present the running times required to estimate a covariance matrix at the desired accuracy in Fig. 2. Specifically, this figure shows the running times of the 'efficient' combinations of N and ℓ: these are the combinations of N and ℓ plotted in Fig. 1 for which there are no N' and ℓ' such that $N'\ell' \leq N\ell$ and $\rho(\widehat{\Sigma}_{\ell',N'}^{-1} \widehat{\Sigma}_0 - I) <$

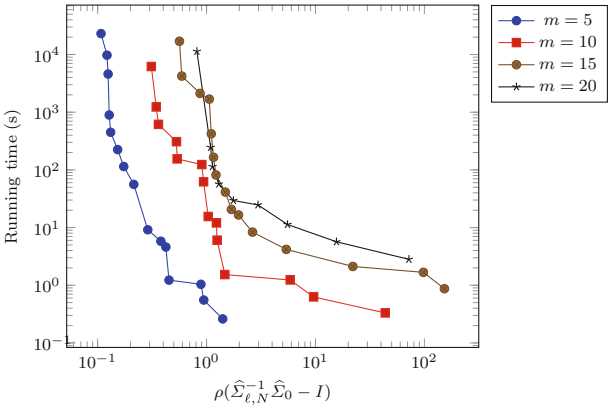

Fig. 2 Running times required to find an approximation $\widehat{\Sigma}_{\ell,N}$ of the desired quality

$\rho(\widehat{\Sigma}_{\ell,N}^{-1}\widehat{\Sigma}_0 - I)$. (The computer used has an Intel i7-6700 CPU with 16 GB RAM, and the code used six threads.) Figure 2 shows that, even at low dimensions, approximating the covariance matrix to high accuracy will take an unpractical amount of time, which approximately shows the time required to approximate the covariance matrix up to desired accuracy.

To show that the slow trajectory toward zero in Fig. 1 is a result of covariance estimation's fundamental difficulty, we consider a simpler problem. We want to approximate the covariance matrix of the uniform distribution over the hypercube $[0, 1]^n$ in $\mathbb{R}^n$. Note that the true covariance matrix of this distribution is known to be $\Sigma := \frac{1}{12}I$.

Again, we will use hit-and-run with varying walk lengths and sample sizes to generate samples from the uniform distribution over $[0, 1]^n$, and compare the resulting covariance matrices $\widehat{\Sigma}_{\ell,N}$ with $\Sigma = \frac{1}{12}I$. (Comparing against a covariance estimate based on hit-and-run samples as in Fig. 1 yields roughly the same image.) The result is shown in Fig. 3.

Figure 3 shows a pattern similar to that of Fig. 1: As the problem size increases, the walk length should increase with the sample size to ensure the estimate is as good as the sample size can guarantee. While this progression toward zero may appear as slow, we do not have to know every eigenvalue and eigenvector of the covariance to high accuracy. Recall that we only use this covariance estimate in Algorithm 2 to generate hit-and-run directions. As such, it may suffice to have an estimate that roughly shows which directions are 'long,' and which ones are 'short.'

6.2 Mean Approximation

Next, we consider the problem of approximating the mean. Although it is not required for Algorithm 2 to approximate the mean of a Boltzmann distribution, such a mean does lie on the central path of the interior point method proposed by Abernethy and Hazan [1, Appendix D].

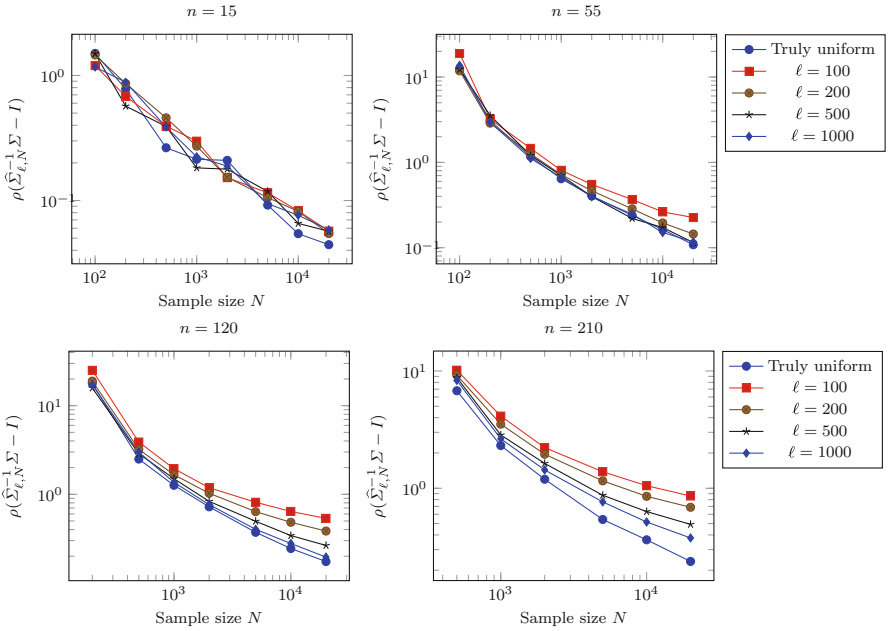

Fig. 3 Effect of sample size N and walk length ℓ on quality of uniform covariance matrix approximation over the hypercube in $\mathbb{R}^n$

We again take 20,000 hit-and-run samples from the uniform distribution over the feasible set of (18) with walk length 50,000. (Directions are drawn from $\mathcal{N}(0, I)$ and the starting point is $\operatorname{svec}(mI + J)/(2e^\top \operatorname{svec}(mI + J))$, where J is the all-ones matrix.) These samples are used to create the mean estimate $\widehat{x}_0$. Then, the experiment is repeated for walk lengths $\ell \leq 50{,}000$ and sample sizes $N \leq 20{,}000$. We refer to these new estimates as $\widehat{x}_{\ell,N}$. Using the approximation $\widehat{\Sigma}_0$ of the uniform covariance matrix from the previous section, we compute $\|\widehat{x}_0 - \widehat{x}_{\ell,N}\|_{\widehat{\Sigma}_0^{-1}}$ and plot the results in Fig. 4.

The results are comparable to those in Figs. 1 and 2. It will take an impractical amount of time before the mean estimate approximates the true mean well enough for practical purposes.

6.3 Kalai–Vempala Algorithm

The results from the previous two sections show that we should not hope to approximate the covariance matrix and sample mean with high accuracy in high dimensions. However, it is still insightful to verify if this is really required for Algorithm 2 to work in practice.

We therefore generated a random vector $c \in \mathbb{R}^{m(m+1)/2}$ as follows: If $C \in \mathbb{R}^{m\times m}$ is a matrix with all elements belonging to a standard normal distribution, then $C + C^\top + (\sqrt{2} - 2)\operatorname{Diag}(C)$ is a symmetric matrix whose elements all have variance 2.

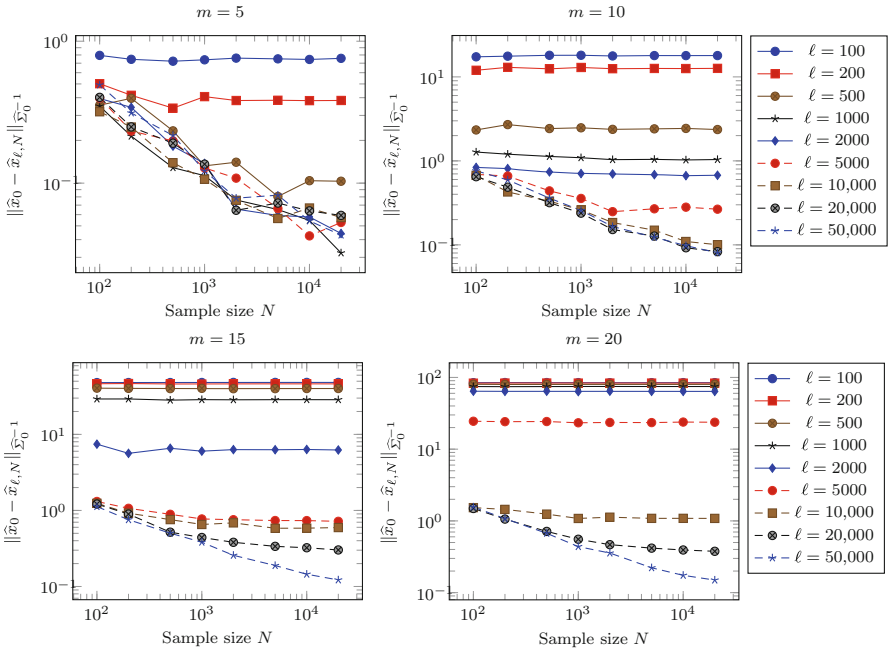

Fig. 4 Effect of sample size N and walk length ℓ on quality of uniform mean approximation

We then let

$$c = \frac{\operatorname{svec}(C + C^\top + (\sqrt{2} - 2)\operatorname{Diag}(C))}{\|\operatorname{svec}(C + C^\top + (\sqrt{2} - 2)\operatorname{Diag}(C))\|},$$

serve as the objective of our optimization problem (18). We can find the optimal solution x_* with MOSEK 8.0 [21] and then run Algorithm 2 with $\varepsilon = 10^{-3}$ and $p = 10^{-1}$. The final gap $\langle c, x_{\text{final}} - x_* \rangle$ is shown in Fig. 5. One can see that for practical sample sizes and walk lengths, the method does not converge to the optimal solution.

6.4 Kalai–Vempala Algorithm with Acceleration Heuristic

Keeping our findings above in mind, we propose the heuristic adaption of Algorithm 2 presented in Algorithm 3. The main modifications we suggest to make to Algorithm 2 are:

1. Use the (centered) samples generated in the previous iteration as directions for hit-and-run in the current iteration. This would eliminate the need to estimate the covariance matrix of a distribution, only to then draw samples from that same distribution. Instead, we can also draw directions directly from the centered samples (cf. line 10 in Algorithm 3). Thus, each sample is used to generate a hit-and-run direction with uniform probability.

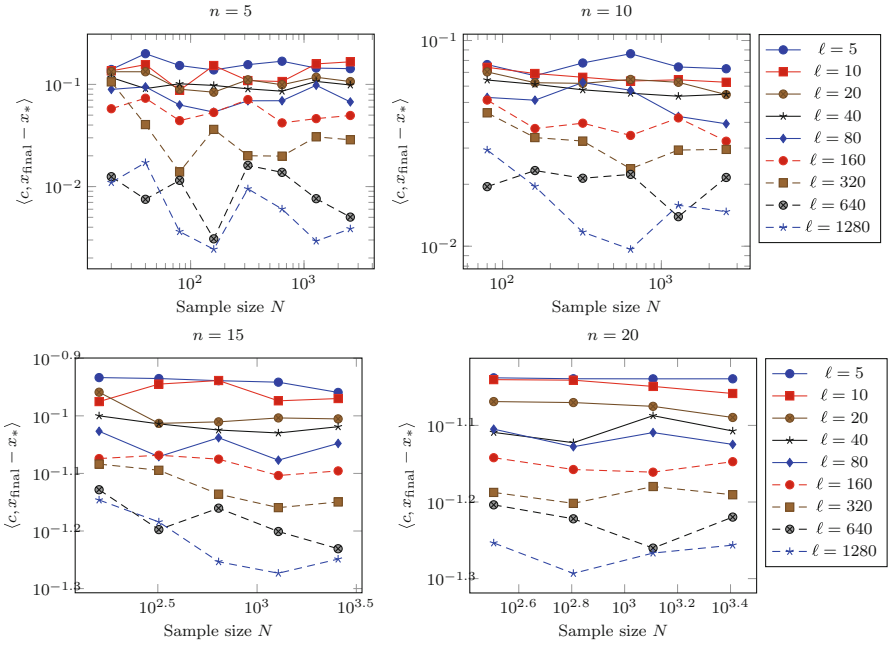

Fig. 5 Effect of sample size N and walk length ℓ on the final gap of Algorithm 2

2. As a starting point for the first random walk in some iteration k, use the sample mean from iteration $k - 1$. While this does significantly change the distribution of the starting point, it concentrates more probability mass around the mean of the Boltzmann distribution with parameter θ_{k-1}, such that the starting point of the random walk is likely already close to the mean of the Boltzmann distribution with parameter θ_k. In a similar vein, we return the mean of the samples in the final iteration, not just one sample. This will not change the expected objective value of the final result, and will therefore also not affect the probabilistic guarantee that we derived in (15) by Markov's inequality. However, using the mean does reduce the variance in the final solution.
3. Start all except the first random walk in some iteration k from the end point of the previous random walk, rather than from a common starting point. The idea here is that the random samples as a whole will then exhibit less dependence, thus improving the approximation quality of the empirical distribution.

With these modifications implemented, we can no longer study the quality of the covariance matrix. Therefore, we will simply consider if the resulting optimization algorithm leads to a small error in the objective value. We solve the problem from Sect. 6.3 with Algorithm 3. The results are shown in Fig. 6.

For low dimensions in particular, the proposed changes seem to have a positive effect.

It can be seen from Fig. 6 that—roughly speaking—the final gap $\langle c, x_{\text{final}} - x_* \rangle$ takes values between two extremes. At one end, the method does not converge and

Algorithm 3 Heuristic adaptation of Algorithm 2

Input: unit vector $c \in \mathbb{R}^n$; membership oracle of a convex body $K \subseteq \mathbb{R}^n$; radius R of Euclidean ball containing K; complexity parameter $\vartheta \leq n + o(n)$ of the entropic barrier over K; update parameter $\alpha > 1 + 1/\sqrt{\vartheta}$; error tolerance $\varepsilon > 0$; failure probability $p \in (0, 1)$; number of hit-and-run steps $\ell \in \mathbb{N}$; number of samples $N \in \mathbb{N}$; $y_1, ..., y_N \in K$ drawn randomly from the uniform distribution over K.

$Y_{j0} \leftarrow y_j$ for all $j \in \{1, ..., N\}$
$X_0 \leftarrow \frac{1}{N} \sum_{j=1}^{N} Y_{j0}$
$\theta_0 \leftarrow 0$
$T_0 \leftarrow \infty$, $T_1 \leftarrow R$
$k \leftarrow 1$
while $nT_{k-1} > \varepsilon p$ **do**
  $\theta_k \leftarrow -c/T_k$
  $Y_{0k} \leftarrow X_{k-1}$
  for $j \in \{1, ..., N\}$ **do**
    Generate Y_{jk} by applying hit-and-run sampling to the Boltzmann distribution with parameter θ_k, starting the walk from $Y_{j-1,k}$, taking ℓ steps, drawing directions uniformly from $\{Y_{1,k-1} - X_{k-1}, ..., Y_{N,k-1} - X_{k-1}\}$
  end for
  $X_k \leftarrow \frac{1}{N} \sum_{j=1}^{N} Y_{jk}$
  $T_{k+1} \leftarrow \min\{R(1 - \frac{1}{\alpha\sqrt{\vartheta}})^k, R(1 - \frac{1}{\sqrt{n}})^k\}$
  $k \leftarrow k + 1$
end while
return X_{k-1}

the final gap is still of the order 10^{-1}. At the other end, the method does converge to the optimum, such that the gap is of the order $10^{-4} = \varepsilon p$. Note that εp is exactly the size we would like the expected gap to have to guarantee that the gap is smaller than ε with probability $1 - p$ by Markov's inequality. Whether we are at one end or the other depends on N and ℓ being large enough compared to m. As a heuristic, we propose that

$$N = \lceil n\sqrt{n} \rceil, \qquad \ell = \lceil n\sqrt{n} \rceil, \tag{19}$$

where $n = m(m + 1)/2$ is the number of variables, are generally sufficient to ensure that the final gap is of the order εp.

7 Numerical Examples on the Copositive Cone

We now turn our attention away from the doubly nonnegative cone, and toward the copositive cone. Although—as mentioned earlier—deciding if a matrix is copositive is a co-NP-complete problem [22], there are a number of procedures to test for copos-

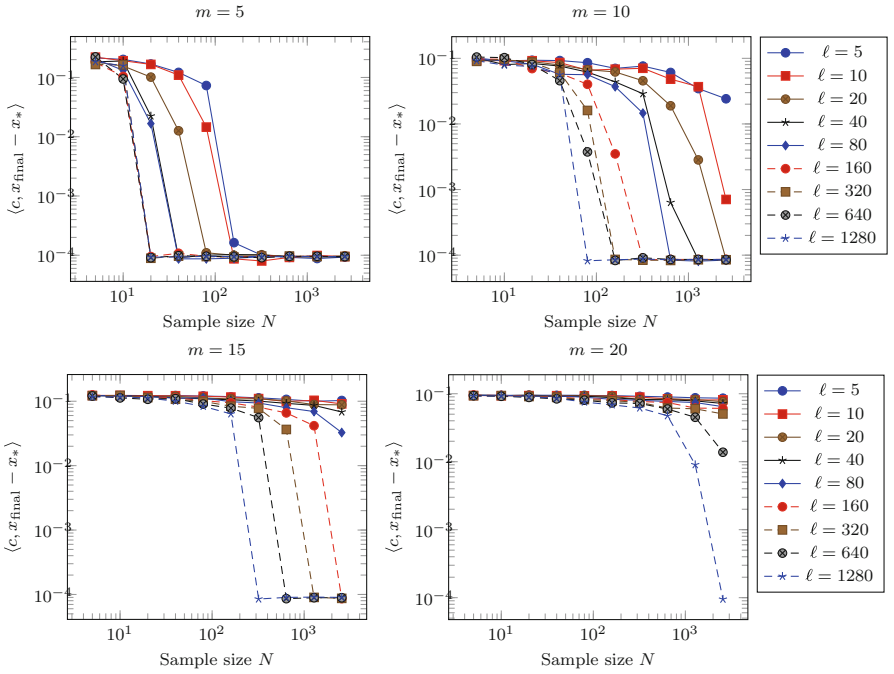

Fig. 6 Effect of sample size N and walk length ℓ on the final gap of Algorithm 3

itivity. Clearly, $A = [A_{ij}]_{ij} \in \mathbb{S}^{m\times m}$ is copositive if and only if

$$\min \left\{ a^\top A a : e^\top a = 1, a \geq 0 \right\}, \tag{20}$$

is nonnegative, where e is the all-ones vector. Xia et al. [28] show that solving (20) is equivalent to solving

$$\begin{aligned} \min \ & -\nu \\ \text{s.t. } & Aa + \nu e - \eta = 0 \\ & e^\top a = 1 \\ & 0 \leq a \leq b \\ & 0 \leq \eta \leq M(e - b) \\ & b \in \{0, 1\}^n, \end{aligned} \tag{21}$$

where $M = 2m \max_{i,j\in\{1,\ldots,m\}} |A_{ij}|$. (To be precise, every optimal solution (a, ν, η) to (21) gives an optimal solution a to (20), and these two problems have the same optimal values.) Note that we are generally not interested in solving (21) to optimality: it suffices to find a feasible solution of (21) with a negative objective value, or confirm that no such solution exists. For the majority of the matrices encountered by Algorithm 3 applied to our test sets described below, this could be checked quickly.

7.1 Separating from the Completely Positive Cone

Recall that a matrix $A \in \mathbb{S}^{m\times m}$ is completely positive if $A = BB^\top$ for some $B \geq 0$. It is easily seen that optimization problems over the completely positive cone can be relaxed as optimization problems over the doubly nonnegative cone. To strengthen this relaxation, one could add a cutting plane separating the optimal solution Y of the doubly nonnegative relaxation from the completely positive cone. This is listed as an open (computational) problem by Berman et al. [5, Section 5], who note that the problem of generating such a cut has only been answered for specific structures of Y, including 5×5 matrices [9]. In general, such a cut could be generated for a doubly nonnegative matrix Y by the copositive program

$$\inf \{\langle Y, X\rangle : \langle X, X\rangle \leq 1, X \text{ copositive}\}. \tag{22}$$

Below, we solve this problem for 6×6 matrices, by way of example.

To generate test instances, we are interested in matrices on the boundary of the 6×6 doubly nonnegative cone. The extreme rays of this cone are described by Ycart [29, Proposition 6.1]. We generate random instances from the class of matrices described under case 3, graph 4 in Proposition 6.1 in [29]. These matrices are (up to permutation of the indices) doubly nonnegative matrices $Y = [Y_{ij}]_{ij}$ with rank 3 satisfying $Y_{i,i+1} = 0$ for $i = 1, ..., 5$. To generate such a matrix, we draw the elements of two vectors $v_1, v_2 \in \mathbb{R}^6$ and the first element $(v_3)_1 \in \mathbb{R}$ of a vector $v_3 \in \mathbb{R}^6$ from a Poisson distribution with rate 1, and multiply each of these elements with -1 with probability $\frac{1}{2}$.

The remaining elements of v_3 are then chosen such that $Y = \sum_{k=1}^{3} v_k v_k^\top$ satisfies $Y_{i,i+1} = 0$ for $i = 1, ..., 5$. This procedure is repeated if the matrix Y is not doubly nonnegative, or if BARON 15 [27] could find a nonnegative matrix $B \in \mathbb{R}^{6\times 9}$ such that $Y = BB^\top$ in less than 30 s. (For the cases where such a decomposition could be found, BARON terminated in less than a second.) Thus, we are left with doubly nonnegative matrices for which it cannot quickly be shown that they are completely positive.

For ten of such randomly generated matrices (see Appendix 1), the optimal value of Algorithm 3 applied to (22) is given in Table 1. This table shows the normalized objective value $\langle Y/\|Y\|, X^*\rangle$, where Y is a doubly nonnegative matrix as described above, and X^* is the final solution returned by Algorithm 3.

Note that in all cases, Algorithm 3 succeeds in finding a copositive matrix X^* such that $\langle Y, X^*\rangle < 0$, which means a cut separating Y from the completely positive matrices was found.

Note that solving the MILP (21) for a matrix A that is not copositive yields a hyperplane separating A from the copositive cone. Thus, we can also solve problem (22) with the Ellipsoid method of Yudin and Nemirovski [30], for example. For the sake of comparison, the results of the Ellipsoid method are also included in Table 1. Note, in particular, that the number of oracle calls in Table 1 is several orders of magnitude smaller for the Ellipsoid method.

Table 1 Objective values returned by Algorithm 3 and by the Ellipsoid method, applied to (22)

	Final objective value (normalized)		Oracle calls	
Name	Algorithm 3	Ellipsoid method	Algorithm 3	Ellipsoid method
extremal_rand_1	− 7.626893e−03	− 7.667645e−03	8.766473e+06	3.152000e+03
extremal_rand_2	− 1.983630e−02	− 1.987634e−02	9.073317e+06	3.412000e+03
extremal_rand_3	− 3.591875e−02	− 3.596345e−02	9.334264e+06	3.835000e+03
extremal_rand_4	− 9.937402e−03	− 9.980087e−03	8.830209e+06	3.147000e+03
extremal_rand_5	− 5.897273e−03	− 5.940056e−03	8.628287e+06	2.957000e+03
extremal_rand_6	− 4.303956e−02	− 4.307761e−02	9.438518e+06	4.024000e+03
extremal_rand_7	− 2.411010e−02	− 2.415651e−02	9.179767e+06	3.708000e+03
extremal_rand_8	− 6.822593e−02	− 6.826558e−02	9.641288e+06	4.277000e+03
extremal_rand_9	− 4.232229e−02	− 4.236829e−02	9.416909e+06	3.981000e+03
extremal_rand_10	− 2.962993e−02	− 2.967333e−02	9.236507e+06	3.743000e+03

Algorithm 3 was run with $\varepsilon = 10^{-3}$ and $p = 0.1$, and N and ℓ as in (19). The Ellipsoid method was run with error tolerance 10^{-4}

8 Conclusion

We have shown that Kalai and Vempala's algorithm [12] returns a solution which is near-optimal for (1) with high probability in polynomial time, when the temperature update (5) is used. The main drawback in using the algorithm in practice is that a large number of samples (i.e., membership oracle calls) are required. As a result, in our tests the Ellipsoid method outperformed Algorithm 3 by a large margin. Thus, based on our experiments, one would favor polynomial-time cutting plane methods like the Ellipsoid method, or more sophisticated alternatives as described, for example, in [16]. In order to obtain a practically viable variant of the Kalai–Vempala algorithm, one would have to improve the sampling process greatly, or utilize massive parallelism to speed up the hit-and-run sampling.

Acknowledgements The authors would like to thank Sébastien Bubeck, Osman Güler, and Levent Tunçel for valuable discussions about the complexity parameter of the entropic barrier. Also, the authors are deeply indebted to an anonymous referee, who generously suggested the outline of the analysis in Sect. 3, which improved the complexity analysis in an earlier version of this paper.

Appendix A: The Complexity Parameter of the Entropic Barrier for the Euclidean Ball

Let $B_n := \{x \in \mathbb{R}^n : \|x\|^2 \leq 1\}$ be the unit ball in $\mathbb{R}^n$ with respect to the Euclidean inner product $\langle \cdot, \cdot \rangle$. Let X_θ be a random variable following a Boltzmann distribution over B_n with parameter $\theta \in \mathbb{R}^n$, and denote the expectation operator by $\mathbb{E}$. We are interested in the complexity parameter ϑ of the entropic barrier on B_n.

Recall from (3) that

$$\begin{aligned}\vartheta &= \sup_{\theta \in \mathbb{R}^n} \langle \theta, \Sigma(\theta)\theta \rangle = \sup_{\theta \in \mathbb{R}^n} \mathbb{E}[\langle X_\theta - \mathbb{E}[X_\theta], \theta \rangle^2] \\ &= \sup_{\theta \in \mathbb{R}^n} \left\{ \mathbb{E}[\langle \theta, X_\theta \rangle^2] - \langle \theta, \mathbb{E}[X_\theta] \rangle^2 \right\}.\end{aligned}$$

For every $\theta \in \mathbb{R}^n$, there exists a rotation matrix Q with $|\det Q| = 1$ such that $\langle \theta, Qy \rangle = \|\theta\| y_1$ for all $y \in \mathbb{R}^n$. Using the fact that the volume of an $(n-1)$-dimensional ball with radius r is r^{n-1} times some factor depending only on n, we see that

$$\langle \theta, \mathbb{E}[X_\theta] \rangle = \frac{\int_{B_n} \langle \theta, x \rangle e^{\langle \theta, x \rangle} \, \mathrm{d}x}{\int_{B_n} e^{\langle \theta, x \rangle} \, \mathrm{d}x} = \frac{\int_{B_n} \|\theta\| y_1 e^{\|\theta\| y_1} \, \mathrm{d}y}{\int_{B_n} e^{\|\theta\| y_1} \, \mathrm{d}y}$$

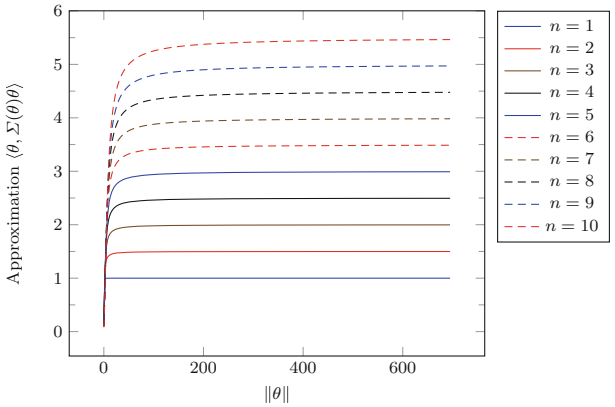

Fig. 7 Numerical approximation of $\langle \theta, \Sigma(\theta)\theta \rangle$

$$= \frac{\int_{-1}^{1} \|\theta\| y_1 (\sqrt{1-y_1^2})^{n-1} e^{\|\theta\| y_1} \, \mathrm{d}y_1}{\int_{-1}^{1} (\sqrt{1-y_1^2})^{n-1} e^{\|\theta\| y_1} \, \mathrm{d}y_1}.$$

The final expression cannot be computed in closed form, but it can be approximated numerically for fixed $\|\theta\|$. Similarly,

$$\mathbb{E}[\langle \theta, X_\theta \rangle^2] = \frac{\int_{B_n} \langle \theta, x \rangle^2 e^{\langle \theta, x \rangle} \, \mathrm{d}x}{\int_{B_n} e^{\langle \theta, x \rangle} \, \mathrm{d}x} = \frac{\int_{B_n} \|\theta\|^2 y_1^2 e^{\|\theta\| y_1} \, \mathrm{d}y}{\int_{B_n} e^{\|\theta\| y_1} \, \mathrm{d}y}$$
$$= \frac{\int_{-1}^{1} \|\theta\|^2 y_1^2 (\sqrt{1-y_1^2})^{n-1} e^{\|\theta\| y_1} \, \mathrm{d}y_1}{\int_{-1}^{1} (\sqrt{1-y_1^2})^{n-1} e^{\|\theta\| y_1} \, \mathrm{d}y_1}.$$

Numerical approximation of $\mathbb{E}[\langle \theta, X_\theta \rangle^2] - \langle \theta, \mathbb{E}[X_\theta] \rangle^2$ for different values of n and $\|\theta\|$ yields Fig. 7. This figure suggests that $\vartheta = \frac{1}{2}(n+1)$.

Appendix B: Extremal Doubly Nonnegative Matrix Examples

Below are the ten randomly generated extreme points of the 6×6 doubly nonnegative cone that are used in Sect. 7.1. These matrices can be strictly separated from the completely positive cone.

$$\texttt{extremal\_rand\_1} = \begin{bmatrix} 2 & 0 & 6 & 0 & 1 & 2 \\ 0 & 6 & 0 & 8 & 1 & 2 \\ 6 & 0 & 18 & 0 & 3 & 6 \\ 0 & 8 & 0 & 11 & 0 & 3 \\ 1 & 1 & 3 & 0 & 6 & 0 \\ 2 & 2 & 6 & 3 & 0 & 3 \end{bmatrix}$$

$$\texttt{extremal\_rand\_2} = \begin{bmatrix} 2 & 0 & 2 & 3 & 1 & 3 \\ 0 & 2 & 0 & 3 & 1 & 1 \\ 2 & 0 & 3 & 0 & 2 & 1 \\ 3 & 3 & 0 & 18 & 0 & 12 \\ 1 & 1 & 2 & 0 & 2 & 0 \\ 3 & 1 & 1 & 12 & 0 & 9 \end{bmatrix}$$

$$\texttt{extremal\_rand\_3} = \begin{bmatrix} 12 & 0 & 4 & 2 & 0 & 2 \\ 0 & 2 & 0 & 2 & 1 & 2 \\ 4 & 0 & 2 & 0 & 1 & 0 \\ 2 & 2 & 0 & 3 & 0 & 3 \\ 0 & 1 & 1 & 0 & 2 & 0 \\ 2 & 2 & 0 & 3 & 0 & 3 \end{bmatrix}$$

$$\texttt{extremal\_rand\_4} = \begin{bmatrix} 2 & 0 & 2 & 2 & 2 & 4 \\ 0 & 8 & 0 & 4 & 4 & 8 \\ 2 & 0 & 3 & 0 & 4 & 0 \\ 2 & 4 & 0 & 8 & 0 & 16 \\ 2 & 4 & 4 & 0 & 8 & 0 \\ 4 & 8 & 0 & 16 & 0 & 32 \end{bmatrix}$$

$$\texttt{extremal\_rand\_5} = \begin{bmatrix} 5 & 0 & 5 & 0 & 5 & 3 \\ 0 & 6 & 0 & 10 & 1 & 18 \\ 5 & 0 & 5 & 0 & 5 & 3 \\ 0 & 10 & 0 & 20 & 0 & 42 \\ 5 & 1 & 5 & 0 & 6 & 0 \\ 3 & 18 & 3 & 42 & 0 & 99 \end{bmatrix}$$

$$\texttt{extremal\_rand\_6} = \begin{bmatrix} 3 & 0 & 3 & 4 & 0 & 4 \\ 0 & 6 & 0 & 2 & 6 & 2 \\ 3 & 0 & 11 & 0 & 4 & 0 \\ 4 & 2 & 0 & 8 & 0 & 8 \\ 0 & 6 & 4 & 0 & 8 & 0 \\ 4 & 2 & 0 & 8 & 0 & 8 \end{bmatrix}$$

$$\texttt{extremal\_rand\_7} = \begin{bmatrix} 14 & 0 & 4 & 8 & 2 & 16 \\ 0 & 6 & 0 & 4 & 2 & 8 \\ 4 & 0 & 8 & 0 & 8 & 0 \\ 8 & 4 & 0 & 8 & 0 & 16 \\ 2 & 2 & 8 & 0 & 9 & 0 \\ 16 & 8 & 0 & 16 & 0 & 32 \end{bmatrix}$$

$$\texttt{extremal\_rand\_8} = \begin{bmatrix} 6 & 0 & 4 & 2 & 0 & 2 \\ 0 & 5 & 0 & 2 & 2 & 2 \\ 4 & 0 & 6 & 0 & 2 & 0 \\ 2 & 2 & 0 & 2 & 0 & 2 \\ 0 & 2 & 2 & 0 & 2 & 0 \\ 2 & 2 & 0 & 2 & 0 & 2 \end{bmatrix}$$

$$\texttt{extremal\_rand\_9} = \begin{bmatrix} 2 & 0 & 2 & 0 & 4 & 2 \\ 0 & 2 & 0 & 2 & 2 & 0 \\ 2 & 0 & 2 & 0 & 4 & 2 \\ 0 & 2 & 0 & 3 & 0 & 2 \\ 4 & 2 & 4 & 0 & 14 & 0 \\ 2 & 0 & 2 & 2 & 0 & 6 \end{bmatrix}$$

$$\texttt{extremal\_rand\_10} = \begin{bmatrix} 2 & 0 & 2 & 2 & 0 & 2 \\ 0 & 2 & 0 & 2 & 2 & 2 \\ 2 & 0 & 3 & 0 & 1 & 0 \\ 2 & 2 & 0 & 8 & 0 & 8 \\ 0 & 2 & 1 & 0 & 3 & 0 \\ 2 & 2 & 0 & 8 & 0 & 8 \end{bmatrix}$$

References

1. Abernethy, J., Hazan, E.: Faster convex optimization: simulated annealing with an efficient universal barrier. In: Proceedings of the 33rd International Conference on Machine Learning
2. Badenbroek, R.: Interior point methods and simulated annealing for nonsymmetric conic optimization. PhD thesis, CentER, Center for Economic Research, Tilburg University (2021). https://pure.uvt.nl/ws/portalfiles/portal/48162364/200375_PhD_Riley_Badenbroek_digitaal.pdf
3. Badenbroek, R., de Klerk, E.: Complexity analysis of a sampling-based interior point method for convex optimization. Math. Oper. Res. **47**(1), 779–811 (2022)
4. Bélisle, C.J., Romeijn, H.E., Smith, R.L.: Hit-and-run algorithms for generating multivariate distributions. Math. Oper. Res. **18**(2), 255–266 (1993)
5. Berman, A., Dur, M., Shaked-Monderer, N.: Open problems in the theory of completely positive and copositive matrices. Electron. J. Linear Algebra **29**(1), 46–58 (2015)
6. Bomze, I.M.: Copositive optimization–recent developments and applications. Eur. J. Oper. Res. **216**(3), 509–520 (2012)
7. Bubeck, S., Eldan, R.: The entropic barrier: exponential families, log-concave geometry, and self-concordance. Math. Oper. Res. **44**(1), 264–276 (2018)
8. Burer, S.: Optimizing a polyhedral-semidefinite relaxation of completely positive programs. Math. Program. Comput. **2**(1), 1–19 (2010)
9. Burer, S., Dong, H.: Separation and relaxation for cones of quadratic forms. Math. Program. **137**(1–2), 343–370 (2013)
10. de Klerk, E., Laurent, M.: Comparison of Laserre's measure-based bounds for polynomial optimization to the bounds obtained by simulated annealing. Math. Oper. Res. **43**, 1317–1325 (2018)
11. Horn, R.A., Johnson, C.R.: Matrix Analysis. Cambridge University Press, Cambridge (2012)
12. Kalai, A.T., Vempala, S.: Simulated annealing for convex optimization. Math. Oper. Res. **31**(2), 253–266 (2006)
13. Kannan, R., Lovász, L., Simonovits, M.: Random walks and an $O^*(n^5)$ volume algorithm for convex bodies. Random Struct. Algorithms **11**(1), 1–50 (1997)
14. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P., et al.: Optimization by simulated annealing. Science **220**(4598), 671–680 (1983)

15. Lee, Y.T., Sidford, A., Vempala, S.S.: Efficient convex optimization with Oracles. In: Bárány, I., Katona, G., Sali, A. (eds.) Building Bridges II. Bolyai Society Mathematical Studies, vol. 28. Springer, Berlin, Heidelberg (2019). https://doi.org/10.1007/978-3-662-59204-5_10
16. Lee, Y.T., Sidford, A., Wong, S.C.-W.: A faster cutting plane method and its implications for combinatorial and convex optimization. In: 2015 IEEE 56th Annual Symposium on Foundations of Computer Science, pp. 1049–1065. IEEE (2015)
17. Levin, D.A., Peres, Y., Wilmer, E.L.: Markov Chains and Mixing Times, vol. 107. American Mathematical Society, Providence (2017)
18. Lovász, L., Vempala, S.: Simulated annealing in convex bodies and an $O^*(n^4)$ volume algorithm. J. Comput. Syst. Sci. **72**(2), 392–417 (2006)
19. Lovász, L., Vempala, S.: Fast algorithms for logconcave functions: Sampling, rounding, integration and optimization. In: 47th Annual IEEE Symposium on Foundations of Computer Science, 2006. FOCS'06, pp. 57–68. IEEE (2006)
20. Lovász, L., Vempala, S.: The geometry of logconcave functions and sampling algorithms. Random Struct. Algorithms **30**(3), 307–358 (2007)
21. MOSEK ApS. The MOSEK optimization toolbox for MATLAB manual. Version 8.1. (2017)
22. Murty, K.G., Kabadi, S.N.: Some np-complete problems in quadratic and nonlinear programming. Math. Program. **39**(2), 117–129 (1987)
23. Nesterov, Y., Nemirovskii, A.: Interior-Point Polynomial Algorithms in Convex Programming. SIAM, Philadelphia (1994)
24. Prékopa, A.: Logarithmic concave measures and functions. Acta Sci. Math. **34**(1), 334–343 (1973)
25. Rudelson, M.: Random vectors in the isotropic position. J. Funct. Anal. **164**(1), 60–72 (1999)
26. Smith, R.L.: Efficient Monte Carlo procedures for generating points uniformly distributed over bounded regions. Oper. Res. **32**(6), 1296–1308 (1984)
27. Tawarmalani, M., Sahinidis, N.V.: A polyhedral branch-and-cut approach to global optimization. Math. Program. **103**(2), 225–249 (2005)
28. Xia, W., Vera, J., Zuluaga, L.F.: Globally solving non-convex quadratic programs via linear integer programming techniques. INFORMS J. Comput. **32**(1), 40–56 (2020)
29. Ycart, B.: Extrémales du cône des matrices de type non négatif, à coefficients positifs ou nuls. Linear Algebra Appl. **48**, 317–330 (1982)
30. Yudin, D.B., Nemirovski, A.S.: Informational complexity and efficient methods for solving complex extremal problems. Matekon **13**(25–45), 6 (1977)