Optimizing over the Growing Spectrahedron

Joachim Giesen[1], Martin Jaggi[2], and Sören Laue[1]

[1] Friedrich-Schiller-Universität Jena, Germany
[2] CMAP, École Polytechnique, Palaiseau, France

Abstract We devise a framework for computing an approximate solution path for an important class of parameterized semidefinite problems that is guaranteed to be ε-close to the exact solution path. The problem of computing the entire regularization path for matrix factorization problems such as maximum-margin matrix factorization fits into this framework, as well as many other nuclear norm regularized convex optimization problems from machine learning. We show that the combinatorial complexity of the approximate path is independent of the size of the matrix. Furthermore, the whole solution path can be computed in *near linear* time in the size of the input matrix.
The framework employs an approximative semidefinite program solver for a fixed parameter value. Here we use an algorithm that has recently been introduced by Hazan. We present a refined analysis of Hazan's algorithm that results in improved running time bounds for a single solution as well as for the whole solution path as a function of the approximation guarantee.

1 Introduction

We provide an algorithm for tracking an approximate solution of a parameterized semidefinite program (SDP) along the parameter path. The algorithm is very simple and comes with approximation quality- and running time guaranties. It computes at some parameter value a slightly better approximate solution than required, and keeps this solution along the path as long as the required approximation quality can be guaranteed. Only when the approximation quality is no longer sufficient, a new solution of the SDP is computed. Hence, the complexity of the algorithm is determined by the time to compute a single approximate SDP solution and the number of solution updates along the path. We show that, if an approximation guarantee of $\varepsilon > 0$ is required, then the number of updates is in $O(1/\varepsilon)$, independent of the size of the problem.

Any SDP solver can be used within the framework to compute an approximate solution of the SDP at fixed parameter values. Here we use Hazan's algorithm [8], which is a Frank-Wolfe type algorithm (also known as conditional gradient descent) applied to SDPs. Hazan's algorithm scales well to large inputs, provides low-rank approximate solutions with guarantees, and only needs a simple approximate eigenvector computation in each of its iterations. A refined analysis of this algorithm, that we present here, shows that its running time can be improved to $\tilde{O}(N/\varepsilon^{1.5})$, where N is the number of non-zeros in the input problem. In the $\tilde{O}$ notation we are ignoring polylogarithmic factors in N.

Motivation. Our work is motivated by the problem of completing a matrix $Y \in \mathbb{R}^{m \times n}$ from which only a small fraction of the entries is known. This problem has been approached in a number of ways in recent years, see for example [2,16,10]. Here we will consider the maximum-margin matrix factorization approach that computes a completion X of Y as a solution of the following optimization problem:

$$\min_{X \in \mathbb{R}^{m \times n}} \sum_{(i,j) \in \Omega} (X_{ij} - Y_{ij})^2 + \lambda \cdot \|X\|_* \tag{1}$$

where Ω is the set of indices at which Y has been observed, and $\lambda > 0$ is the regularization parameter. The solution X whose nuclear norm (also known as trace norm) is $\|X\|_*$ does not need to coincide with Y on the index set Ω. It can be seen as a low rank approximation of Y, where the regularization parameter λ controls the rank of X.

We consider the following equivalent constraint formulation of Problem (1),

$$\begin{array}{ll} \min_X & \sum_{(i,j) \in \Omega} (X_{ij} - Y_{ij})^2 \\ \text{s.t.} & \|X\|_* \leq t \\ & X \in \mathbb{R}^{m \times n}, \end{array} \tag{2}$$

such that there is a one-to-one mapping between the solutions at the parameters t and λ. In fact, Problem (1) is the Lagrangian of Problem (2). The latter is a convex optimization problem that can be solved for fixed values of t using standard convex optimization techniques. By the properties of the nuclear norm, it is known that Problem (2) can be equivalently re-formulated as

$$\begin{array}{ll} \min_X & f(X) \\ \text{s.t.} & X \succeq 0 \\ & \mathrm{Tr}(X) \leq t \\ & X \in \mathbb{R}^{(m+n) \times (m+n)}, \end{array} \tag{3}$$

where f has to be chosen properly, and $\mathrm{Tr}(X)$ denotes the trace of X. We call the set defined by the constraints of Problem (3) a spectrahedron that is growing with t.

It remains to choose a good value for t. We use the simple algorithmic framework that we have described above to track an ε-approximate solution along the whole parameter range. It is interesting to note that in doing so, we follow the standard approach for choosing a good value for t, namely, computing an approximate solution for Problem (2) at a finite number of values for t, and then picking the best out of these finitely many solutions. However, in contrast to previous work, we automatically pick the values for t such that the whole parameter range is covered by an ε-approximate solution, for a specified accuracy ε. We show that the number t-values at which a solution needs to be computed such that the ε-approximation guarantee holds, is in $O(1/\varepsilon)$, independent of the size of the matrix Y. At the chosen t-values we use Hazan's algorithm [8] to compute an approximate solution for Problem (3).

Related Work There have been algorithms proposed for computing the regularization path of matrix factorization problems. d'Aspremont et al. [4] consider the regularization path for sparse principal component analysis. However, here it is known at which parameters one should compute a solution, i.e., all integral values from 1 to n, where n is the number of variables.

Mazumder et al. [12] consider a very similar problem (regularized matrix factorization) and provide an algorithm for computing a solution at a fixed parameter t. They suggest to approximate the regularization path by computing solutions at various values for t. However, the parameter values at which the solutions are computed are chosen heuristically, e.g. on a uniform grid, and therefore no continuous approximation guarantee can be given.

Our solution path algorithm is motivated by the approach of [6] for parameterized convex optimization problems over the simplex, such as e.g., support vector machines. A direct but rather ad hoc extension of this approach to parameterized SDPs has appeared in [7].

Computing an approximate solution for a fixed parameter t simplifies to solving an SDP. The most widely-known implementations of SDP solvers are interior point methods, which provide high-accuracy solutions. However, with a running time that is a low-order polynomial in the number of variables, they do not scale to medium/large problems. Proximal methods have been proposed to overcome the disadvantages of interior point methods. These methods achieve better running times at the expense of less accurate solutions [13,1,14]. Alternating direction methods form yet another approach. On specific, well-structured SDP problems, they achieve very good speed in practice [17].

Notation. For arbitrary real matrices, the standard *inner product* is defined as $A \bullet B := Tr(A^T B)$, and the (squared) *Frobenius matrix norm* $\|A\|_{Fro}^2 := A \bullet A$ is the sum of all squared entries in the matrix. By $\mathbb{S}^{n\times n}$ we denote the set of *symmetric* $n \times n$ matrices. $A \in \mathbb{S}^{n\times n}$ is called *positive semidefinite* (PSD), written as $A \succeq 0$, iff $v^T A v \geq 0 \; \forall v \in \mathbb{R}^n$. Note that $v^T A v = A \bullet v v^T$. $\lambda_{\max}(A) \in \mathbb{R}$ denotes the largest eigenvalue of A. $\|A\|_*$ is the *nuclear norm* of the matrix A, also known as the *trace norm* (sum of the singular values).

2 Convex optimization over the spectrahedron

We consider convex optimization problems of the form

$$\min_{X \in \mathcal{S}_t} \; f(X) \tag{4}$$

where $f : \mathbb{S}^{n\times n} \to \mathbb{R}$ is symmetric, convex, and continuously differentiable such that $-\nabla f(X)$ is not negative definite for all $X \in \mathcal{S}_t$, and the domain $\mathcal{S}_t := \{X \in \mathbb{S}^{n\times n} \mid X \succeq 0, \; \mathrm{Tr}(X) \leq t\}$ is the set of symmetric PSD matrices whose trace is at most t. This set generalizes the set of all symmetric PSD matrices of trace 1, i.e., the convex hull of all rank-1 matrices of unit trace, which is also known as the *spectrahedron*. The spectrahedron can be seen as a generalization of the unit simplex to the space of symmetric matrices.

By the convexity of f, we have the following linearization, for any $X, X_0 \in \mathcal{S}_t$:

$$f(X) \;\geq\; \nabla f(X_0) \bullet (X - X_0) + f(X_0).$$

This allows us to define a lower bound function on (4) for any fixed matrix $X_0 \in \mathcal{S}_t$ as follows,

$$\omega_t(X_0) \;:=\; \min_{X \in \mathcal{S}_t} \nabla f(X_0) \bullet (X - X_0) + f(X_0) \;=\; f(X_0) - \max_{X \in \mathcal{S}_t} -\nabla f(X_0) \bullet (X - X_0).$$

The function $\omega_t(X_0)$ will be called here the Wolfe dual function[1]. Hence we can define the duality gap as

$$g_t(X_0) \;:=\; f(X_0) - \omega_t(X_0) \;=\; \max_{X \in \mathcal{S}_t} -\nabla f(X_0) \bullet (X - X_0) \;\geq\; 0,$$

where non-negativity holds because of

$$f(X_0) \;\geq\; \min_{X \in \mathcal{S}_t} f(X) \;\geq\; \min_{X \in \mathcal{S}_t} \nabla f(X_0) \bullet (X - X_0) + f(X_0) \;=\; \omega_t(X_0).$$

By the definition of the objective function f, the gradient $\nabla f(X_0)$ is always a symmetric matrix (not necessarily PSD), and therefore has real eigenvalues, which is important in the following. Furthermore, this allows to equip any iterative SDP solver with guarantees for the duality gap by running until the gap is smaller than a prescribed bound. The latter can be easily checked using the following lemma.

Lemma 1. *The Wolfe dual of Problem (4) can be written as*

$$\omega_t(X_0) \;=\; f(X_0) - t \cdot \lambda_{\max}\left(-\nabla f(X_0)\right) - \nabla f(X_0) \bullet X_0,$$

and the duality gap can be written as

$$g_t(X_0) \;=\; t \cdot \lambda_{\max}\left(-\nabla f(X_0)\right) + \nabla f(X_0) \bullet X_0.$$

Proof. It is well-known that any matrix $X \in \mathbb{S}^{n \times n}$ can be written as $X = \sum_{i=1}^{n} \lambda_i u_u u_i^T$ with the orthonormal system of eigenvectors u_i, $\|u_i\| = 1$ of X and corresponding eigenvalues λ_i. Moreover $\mathrm{Tr}(X) = \sum_{i=1}^{n} \lambda_i$. So we have

$$\begin{aligned}
\max_{X \in \mathcal{S}_t} G \bullet X &= \max \sum_{i=1}^{n} \alpha_i (G \bullet u_i u_i^T) \;=\; \max \sum_{i=1}^{n} \alpha_i u_i^T G u_i \\
&= t \cdot \max_{v \in \mathbb{R}^n, \|v\|=1} v^T G v \quad \text{[since } G \text{ is not negative definite]} \\
&= t \cdot \lambda_{\max}(G),
\end{aligned}$$

where the last equality is the variational characterization of the largest eigenvalue. The equality above can be interpreted as, any linear function must attain its maximum at a "vertex" (extreme point) of $\mathcal{S}_t$. Finally, both claims follow by plugging in $-\nabla f(X_0)$ for G. □

[1] Strictly speaking, this is not a proper "dual" function. However, we follow here the terminology of Clarkson [3].

Remark 1. If we alternatively define $\mathcal{S}_t := \{X \in \mathbb{S}^{n\times n} \mid X \succeq 0,\ \mathrm{Tr}(X) = t\}$, i.e., replacing "$\leq t$" by "$= t$", then Lemma 1 also holds even if $-\nabla f(X)$ can become negative definite.

From the definition of the duality gap we derive the definition of an approximate solution.

Definition 1. *A matrix $X \in \mathcal{S}_t$ is an* ε-approximation *to Problem (4) if $g_t(X) \leq \varepsilon$.*

Obviously, the duality gap is always an upper bound for the *primal error* $h_t(X)$ of the approximate solution X, where $h_t(X) := f(X) - f(X^*)$ and X^* is an optimal solution of Problem (4) at parameter value t.

3 Hazan's algorithm revisited

We adapt the algorithm by Hazan [8] for approximating SDPs over the spectrahedron to our needs, i.e., approximating SDPs over $\mathcal{S}_t$. The proofs that we provide in the following fix some minor errors in the original paper. We also tighten the analysis and get improved bounds on the running time, $\tilde{O}(N/\varepsilon^{1.5})$ instead of $\tilde{O}(N/\varepsilon^2)$. In particular, we show that the eigenvector computation that is employed by the algorithm only needs to be computed up to an accuracy of ε instead of ε^2.

We start with the simple observation that every SDP in the form of Problem (4) be converted into an equivalent SDP where the inequality constraint $\mathrm{Tr}(X) \leq t$ is replaced by the equality constraint $\mathrm{Tr}(X) = t$, i.e., the optimization problem $\min_{X\in\mathcal{S}_t} f(X)$ is equivalent to the optimization problem

$$\begin{array}{ll} \min_{\hat{X}} & \hat{f}(\hat{X}) \\ \text{s.t.} & \hat{X} \succeq 0 \\ & \mathrm{Tr}(\hat{X}) = t \\ & \hat{X} \in \mathbb{S}^{(n+1)\times(n+1)}, \end{array} \tag{5}$$

where $\hat{f}$ is the same function as f but defined on the larger set of symmetric matrices,

$$\hat{f}(\hat{X}) \;=\; \hat{f}\left(\begin{pmatrix} X & X_2 \\ X_2^T & X_3 \end{pmatrix}\right) \;:=\; f(X)$$

for $X \in \mathbb{S}^{n\times n}$, $X_2 \in \mathbb{R}^{n\times 1}$, $X_3 \in \mathbb{R}^{1\times 1}$. Every feasible solution X for Problem (4) can be converted into a feasible solution $\hat{X}$ for Problem (5) with the same objective function value and vice versa.

Hazan's algorithm for computing an ε-approximate solution for Problem (5) is summarized in pseudo-code in Algorithm 1.

The function ApproxEV used in Algorithm 1 returns an approximate eigenvector to the largest eigenvalue: given a square matrix M it returns a vector

Algorithm 1 ApproxSDP

Input: Convex function f, trace t, k
Output: Approximate solution of Problem 5.
Initialize $X_0 = t \cdot v_0 v_0^T$ for some arbitrary rank-1 PSD matrix $t \cdot v_0 v_0^T$ with trace t and $\|v_0\| = 1$.
for $i = 0$ **to** k **do**
  Let $\alpha_i = \frac{2}{i+2}$.
  Let $\varepsilon_i = \frac{\alpha_i \cdot C_f}{t}$.
  Compute $v_i = \text{ApproxEV}(-\nabla f(X_i), \varepsilon_i)$.
  Set $X_{i+1} = X_i + \alpha_i(t \cdot v_i v_i^T - X_i)$.
end for

$v = \text{ApproxEV}(M, \varepsilon)$ of unit length that satisfies $v^T M v \geq \lambda_{\max}(M) - \varepsilon$. The *curvature constant* C_f used in the algorithm is defined as

$$C_f := \sup_{\substack{X, Z \in \mathcal{S}_t, \\ \alpha \in [0,1], \\ Y = X + \alpha(Z - X)}} \frac{1}{\alpha^2} \left(f(Y) - f(X) - (Y - X) \bullet \nabla f(X) \right).$$

The curvature constant is a measure of how much the function $f(X)$ deviates from a linear approximation in X. The boundedness of this curvature is a widely used in convex optimization, and can be upper bounded by the Lipschitz constant of the gradient of f and the diameter of $\mathcal{S}_t$. Now we can prove the following theorem.

Theorem 1. *For each $k \geq 1$, the iterate X_k of Algorithm 1 satisfies $f(X_k) - f(X^*) \leq \varepsilon$, where $f(X^*)$ is the optimal value for the minimization Problem (5), and $\varepsilon = \frac{8C_f}{k+2}$.*

Proof. For each iteration of the algorithm, we have that

$$\begin{aligned} f(X_{i+1}) &= f(X_i + \alpha_i(t \cdot v_i v_i^T - X_i)) \\ &\leq f(X_i) + \alpha_i(t \cdot v_i v_i^T - X_i) \bullet \nabla f(X_i) + \alpha_i^2 C_f \ , \end{aligned} \tag{6}$$

where the last inequality follows from the definition of the curvature C_f. Furthermore,

$$\begin{aligned} (t \cdot v_i v_i^T - X_i) \bullet \nabla f(X_i) &= (X_i - t \cdot v_i v_i^T) \bullet (-\nabla f(X_i)) \\ &= X_i \bullet (-\nabla f(X_i)) - t \cdot v_i^T (-\nabla f(X_i)) v_i \\ &\leq -X_i \bullet \nabla f(X_i) - t \cdot (\lambda_{\max}(-\nabla f(X_i)) - \varepsilon_i) \\ &= -g_t(X_i) + t \cdot \varepsilon_i \\ &= -g_t(X_i) + \alpha_i \cdot C_f \ . \end{aligned}$$

The last two equalities follow from the remark after Lemma 1 and from setting $\varepsilon_i = \frac{\alpha_i \cdot C_f}{t}$ within Algorithm 1. Hence, Inequality (6) evaluates to

$$\begin{aligned} f(X_{i+1}) &\leq f(X_i) - \alpha_i g_t(X_i) + \alpha_i^2 C_f + \alpha_i^2 C_f \\ &= f(X_i) - \alpha_i g_t(X_i) + 2\alpha_i^2 C_f \ . \end{aligned} \tag{7}$$

Subtracting $f(X^*)$ on both sides of Inequality (7), and denoting the current primal error by $h(X_i) = f(X_i) - f(X^*)$, we get

$$h(X_{i+1}) \;\leq\; h(X_i) - \alpha_i g_t(X_i) + 2\alpha_i^2 C_f \;, \tag{8}$$

which by using the fact that $g_t(X_i) \geq h(X_i)$ gives

$$h(X_{i+1}) \;\leq\; h(X_i) - \alpha_i h(X_i) + 2\alpha_i^2 C_f \;. \tag{9}$$

The claim of this theorem is that the primal error $h(X_i) = f(X_i) - f(X^*)$ is small after a sufficiently large number of iterations. Indeed, we will show by induction that $h(X_i) \leq \frac{8C_f}{i+2}$. In the first iteration ($i = 0$), we know from (9) that the claim holds, because of the choice of $\alpha_0 = 1$.

Assume now that $h(X_i) \leq \frac{8C_f}{i+2}$ holds. Using $\alpha_i = \frac{2}{i+2}$ in our inequality (9), we can now bound $h(X_{i+1})$ as follows,

$$\begin{aligned} h(X_{i+1}) &\leq h(X_i)(1-\alpha_i) + 2\alpha_i^2 C_f \\ &\leq \frac{8C_f}{i+2}\left(1 - \frac{2}{i+2}\right) + \frac{8C_f}{(i+2)^2} \\ &\leq \frac{8C_f}{i+2} - \frac{8C_f}{(i+2)^2} \\ &\leq \frac{8C_f}{i+1+2} \;, \end{aligned}$$

which proves the theorem. □

Theorem 1 states that after k iterations of Algorithm 1, the approximate solution is within $\varepsilon = \frac{8C_f}{k+2}$ of the optimum. However, for our framework of computing approximate solution paths we need a stronger theorem, namely, we also need the duality gap to be small. Though this cannot be guaranteed after k iterations, it can be guaranteed after at most $2k+1$ iterations as the next theorem states (whose proof follows along the lines of [3]). Note that the theorem does not guarantee that the solution after $2k+1$ iterations has a small enough duality gap, but only that the gap is small enough after one of the iterations between k and $2k+1$. In practice one can run the algorithm for $2k+1$ iterations and keep the solution with the smallest duality gap after any iteration.

Theorem 2. *After at most $2k+1$ iterations, Algorithm 1 has computed an approximate solution X_i whose duality gap $g_t(X_i)$ is at most $\frac{8C_f}{k+2}$.*

Proof. Theorem 1 shows that k iterations suffice for Algorithm 1 to provide an approximate solution X_k with $h(X_k) \leq \frac{8C_f}{k+2}$. For the subsequent $k+1$ iterations we change Algorithm 1 slightly by fixing the step-length α_i to $\frac{2}{k+2}$ for all $i : k \leq i \leq 2k+1$, i.e., we do not decrease the step-size anymore.

For a contradiction assume that $g_t(X_i) > \frac{8C_f}{k+2}$ for all $i : k \leq i \leq 2k+1$. As we have seen in the proof of Theorem 1 (Equation (8)), the bound $h(X_{i+1}) \leq$

$h(X_i) - \alpha_i g_t(X_i) + 2\alpha_i^2 C_f$ in fact holds for any step size $\alpha_i \in [0,1]$. Using our assumption, we get

$$\begin{aligned} h(X_{i+1}) &\leq h(X_i) - \alpha_i g_t(X_i) + 2\alpha_i^2 C_f \\ &< h(X_i) - \frac{2}{k+2} \cdot \frac{8C_f}{k+2} + 2C_f \frac{2^2}{(k+2)^2} \\ &\leq h(X_i) - \frac{16C_f}{(k+2)^2} + \frac{8C_f}{(k+2)^2} \\ &= h(X_i) - \frac{8C_f}{(k+2)^2} \ . \end{aligned}$$

Since $h(X_k) \leq \frac{8C_f}{k+2}$ and $h(X_i) \geq 0$ for all $i \geq 0$, we must have that $g_t(X_i) \leq \frac{8C_f}{k+2}$ holds after at most $k+1$ such additional iterations, since otherwise the inequality from above for $h(X_{i+1})$ implies $h(X_{2k+1}) < 0$, which is a contradiction. □

Note that matrix X_i is stored twice in Algorithm 1, once as a low rank factorization and once as a sparse matrix. The low rank factorization of X_i is obtained in the algorithm via $X_i = \sum_i \beta_i v_i v_i^T$, where v_i is the eigenvector chosen in iteration i, and β_i are the appropriate factors computed via all α_i. Collecting all $O(\frac{1}{\varepsilon})$ eigenvectors amounts to $O(\frac{n}{\varepsilon})$ time and space in total. The sparse representation of X_i stores only the N entries of X_i that are necessary for computing $f(X_i)$ and $\nabla f(X_i)$. Depending on the application, we often have $N \ll n^2$, as for instance in the case of the matrix completion problem (cf. Section 5). The sparse representation of X_i is computed in a straightforward way from the sparse representation of X_{i-1} and the current sparse representation of $v_i v_i^T$, where only the N necessary entries of $v_i v_i^T$ are computed. Hence, computing the sparse representation for all X_i amounts to $O(N)$ operations per iteration and $O(\frac{N}{\varepsilon})$ operations in total.

Furthermore, it is known that the function ApproxEV(M, ε) that computes an approximate eigenvector with guarantee ε can be implemented using the Lanczos method that runs in time $\tilde{O}\left(\frac{N\sqrt{L}}{\sqrt{\varepsilon}}\right)$ and returns a valid approximation with high probability, where N is the number of non-zero entries in the matrix $M \in \mathbb{R}^{n \times n}$ and L is a bound on the largest eigenvalue of M, see [11]. Hence, we can conclude with the following corollary.

Corollary 1. *Algorithm 1 computes an approximate solution X for Problems (5) whose duality gap $g_t(X)$ is at most ε, with high probability, and its running time is in $\tilde{O}\left(\frac{N}{\varepsilon^{1.5}}\right)$.*

4 Optimizing over the growing spectrahedron

We are interested in ε-approximations for Problem (4) for all parameter values of $t \in \mathbb{R}, t \geq 0$.

Definition 2. *The* ε-approximation path complexity *of Problem (4) is the minimum number of sub-intervals over all possible partitions of the parameter range*

$t \in \mathbb{R}, t \geq 0$, *such that for each individual sub-interval there is a single solution of Problem (4) which is an* ε*-approximation for that entire sub-interval.*

The following simple stability lemma is at the core of our discussion and characterizes all parameter values t' such that a given $\frac{\varepsilon}{\gamma}$-approximate solution X (for $\gamma > 1$) at t is at least an ε-approximate solution at t'.

Lemma 2. *Let* $X \in \mathcal{S}_t$ *be an* $\frac{\varepsilon}{\gamma}$*-approximate solution of Problem (4) for some fixed parameter value* t*, and for some* $\gamma > 1$*. Then for all* $t' \geq t \in \mathbb{R}$ *that satisfy*

$$(t' - t) \cdot \lambda_{\max}\left(-\nabla f(X)\right) \;\leq\; \varepsilon \left(1 - \tfrac{1}{\gamma}\right), \tag{10}$$

the solution X *is still an* ε*-approximation to Problem (4) at the parameter value* t'*.*

Proof. Any feasible solution X for the problem at parameter t is also a feasible solution at parameter t' since $\mathcal{S}_t \subseteq \mathcal{S}_{t'}$. We have to show that

$$g_{t'}(X) = t' \cdot \lambda_{\max}\left(-\nabla f(X)\right) + \nabla f(X) \bullet X \;\leq\; \varepsilon.$$

To do so, we add to Inequality (10) the inequality stating that X is an $\frac{\varepsilon}{\gamma}$-approximate solution at value t, i.e.,

$$t \cdot \lambda_{\max}\left(-\nabla f(X)\right) + \nabla f(X) \bullet X \;\leq\; \frac{\varepsilon}{\gamma}\,,$$

and obtain

$$t' \cdot \lambda_{\max}\left(-\nabla f(X)\right) + \nabla f(X) \bullet X \leq \varepsilon\,,$$

which is the claimed bound on the duality gap at the parameter value t'. □

We assume that the unconstrained minimization problem $\min_{X \in \mathbb{S}^{n\times n}} f(X)$ has a bounded solution X^*. This assumption holds in general for all practical problems and the applications that we will consider later.

Theorem 3. *The path complexity of Problem (4) over the parameter range* $t \in \mathbb{R}, t \geq 0$ *is in* $O\left(\frac{1}{\varepsilon}\right)$*.*

Proof. From Lemma 2, we obtain intervals of constant ε-approximate solutions, where each interval is of length as least $\frac{\varepsilon\left(1-\frac{1}{\gamma}\right)}{\lambda_{\max}(-\nabla f(X))}$. Let $t_{\max}$ be the trace of the optimal solution X^* to the unconstrained minimization problem $\min_{X \in \mathbb{S}^{n\times n}} f(X)$. Then X^* has a zero duality gap for Problem (4) for all parameters $t \geq t_{\max}$. On the other hand, for $t < t_{\max}$, we have $\mathcal{S}_t \subseteq \mathcal{S}_{t_{\max}}$ and can therefore bound the number of intervals by $\left\lceil \frac{\gamma_f}{\varepsilon} \right\rceil$, where

$$\gamma_f \;:=\; t_{\max} \cdot \max_{X \in \mathcal{S}_{t_{\max}}} \lambda_{\max}\left(-\nabla f(X)\right) \frac{\gamma}{\gamma - 1}$$

is an absolute constant depending only on the function f, like its Lipschitz constant. □

Lemma 2 immediately suggests a simple algorithm (Algorithm 2) to compute an ε-approximate solution path for Problem (4). The running time of this algorithm is by Theorem 3 in $O(T(\varepsilon)/\varepsilon)$, where $T(\varepsilon)$ is the time to compute a single ε-approximate solution for Problem (4) at a fixed parameter value.

Algorithm 2 SDP-Path

Input: convex function $f, t_{\min}, t_{\max}, \varepsilon, \gamma$
Output: ε-approximate solution path for Problem (4).
Set $t = t_{\min}$.
repeat
  Compute approximate solution $X = \text{ApproxSDP}(f, t, \varepsilon/\gamma)$.
  Update $t = t + \frac{\varepsilon(1-1/\gamma)}{\lambda_{\max}(-\nabla f(X))}$
until $t > t_{\max}$

5 Applications

5.1 Nuclear norm regularized problems and matrix factorization

A nuclear norm regularized problem

$$\min_{X \in \mathbb{R}^{m \times n}} f(X) + \lambda \|X\|_* \tag{11}$$

for *any* loss function f in its constrained formulation

$$\min_{X \in \mathbb{R}^{m \times n},\, \|X\|_* \leq t/2} f(X)$$

is by a straightforward transformation, see [5,9,16], equivalent to the optimization problem

$$\begin{array}{ll} \min_X & \hat{f}(\hat{X}) \\ \text{s.t.} & \hat{X} \in \mathcal{S}_t \end{array} \tag{12}$$

over the scaled spectrahedron $\mathcal{S}_t \subseteq \mathbb{S}^{(m+n)\times(m+n)}$, which is our original problem (4). Here $\hat{f}$ is defined using f as

$$\hat{f}(\hat{X}) \;=\; \hat{f}\left(\begin{pmatrix} X_1 & X_2 \\ X_2^T & X_3 \end{pmatrix}\right) \;:=\; f(X_2)$$

for $\hat{X} \in \mathbb{S}^{(m+n)\times(m+n)}$, $X_2 \in \mathbb{R}^{m \times n}$. Observe that this is now a convex problem whenever the loss function f is convex.

The gradient of the transformed function $\hat{f}$ is

$$\nabla \hat{f}(\hat{X}) \;=\; \begin{pmatrix} 0 & \nabla f(X) \\ (\nabla f(X))^T & 0 \end{pmatrix}, \tag{13}$$

which is not negative definite because if $(v, w)^T$ is an eigenvector of $\nabla \hat{f}(\hat{X})$ for the eigenvalue λ, then $(-v, w)^T$ is an eigenvector for the eigenvalue $-\lambda$. We can now use this gradient in Algorithm 2 to compute the entire regularization path for Problem (12).

A *matrix factorization* can be obtained directly from the low-rank representation of X_i of Algorithm 1.

5.2 Matrix completion

The matrix completion Problem (1) from the introduction is the probably the most commonly used instance of Problem (11) with the standard squared loss function

$$f(X) \;=\; \frac{1}{2} \sum_{(i,j)\in\Omega} (X_{ij} - Y_{ij})^2. \tag{14}$$

The gradient of this loss function is

$$(\nabla f(X))_{ij} \;=\; \begin{cases} X_{ij} - Y_{ij} & : (i,j) \in \Omega, \\ 0 & : \text{otherwise.} \end{cases}$$

Using the notation $(A)_\Omega$ for the matrix that coincides with A on the indices Ω and is zero otherwise, $\nabla f(X)$ can be written as $\nabla f(X) \;=\; (X - Y)_\Omega$. This implies that the square gradient matrix $\nabla \hat{f}(\hat{X})$ that we use in our algorithm (see Equation (13)) is also of this simple form. As this matrix is sparse–it has only $N = |\Omega|$ non-zero entries, storage and approximate eigenvector computations can be performed much more efficiently than for dense problems. Also note that the curvature constant $C_{\hat{f}}$ that appears in the bound for the running time of Hazan's algorithm equals t^2 for the squared loss function from Equation (14), see [9]. Hence, the full regularization path can be computed in time $\tilde{O}\left(\frac{N}{\varepsilon^{2.5}}\right)$.

Finally, note that our framework applies to matrix completion problems with any convex differentiable loss function, such as the smoothed hinge loss or the standard squared loss, and includes the classical maximum-margin matrix factorization variants [16].

6 Conclusion

We have presented a simple and efficient framework that allows to approximate solution paths for parameterized semidefinite programs with guarantees. Many well known regularized matrix factorization and completion problems from machine learning fit into this framework. Even weighted nuclear norm regularized convex optimization problems, see e.g., [15], fit into the framework though we have not shown this here for lack of space. We also have not shown experimental results, and just want to state here that they support the theory , i.e., the running time is near linear in the size of the input matrix. Finally, we have also improved the running time of Hazan's algorithm by a refined analysis.

Acknowledgments

The authors would like to thank the reviewers for their useful comments. The research of J. Giesen and S. Laue has been supported by the Deutsche Forschungsgemeinschaft (DFG) under grant GI-711/3-2. The research of M. Jaggi has been supported by ERC starting grant SIPA, a Google Research Award and by the Swiss National Science Foundation (SNF grant 20PA21-121957).

References

1. Sanjeev Arora, Elad Hazan, and Satyen Kale. Fast Algorithms for Approximate Semidefinite Programming using the Multiplicative Weights Update Method. In *Proceedings of the Annual IEEE Symposium on Foundations of Computer Scienc (FOCS)*, pages 339–348, 2005.
2. Emmanuel J. Candès and Terence Tao. The Power of Convex Relaxation: Near-Optimal Matrix Completion. *IEEE Transactions on Information Theory*, 56(5):2053–2080, 2010.
3. Kenneth L. Clarkson. Coresets, sparse greedy approximation, and the frank-wolfe algorithm. *ACM Transactions on Algorithms*, 6(4), 2010.
4. Alexandre d'Aspremont, Francis R. Bach, and Laurent El Ghaoui. Full Regularization Path for Sparse Principal Component Analysis. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 177–184, 2007.
5. Maryam Fazel, Haitham Hindi, and Stephen P Boyd. A Rank Minimization Heuristic with Application to Minimum Order System Approximation. In *Proceedings of the American Control Conference*, volume 6, pages 4734–4739, 2001.
6. Joachim Giesen, Martin Jaggi, and Sören Laue. Approximating Parameterized Convex Optimization Problems. In *Proceedings of Annual European Symposium on Algorithms (ESA)*, pages 524–535, 2010.
7. Joachim Giesen, Martin Jaggi, and Sören Laue. Regularization Paths with Guarantees for Convex Semidefinite Optimization. In *Proceedings International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2012.
8. Elad Hazan. Sparse Approximate Solutions to Semidefinite Programs. In *Proceedings of Theoretical Informatics, 8th Latin American Symposium (LATIN)*, pages 306–316, 2008.
9. Martin Jaggi and Marek Sulovský. A Simple Algorithm for Nuclear Norm Regularized Problems. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 471–478, 2010.
10. Yehuda Koren, Robert M. Bell, and Chris Volinsky. Matrix Factorization Techniques for Recommender Systems. *IEEE Computer*, 42(8):30–37, 2009.
11. Jacek Kuczyński and Henryk Woźniakowski. Estimating the Largest Eigenvalue by the Power and Lanczos Algorithms with a Random Start. *SIAM Journal on Matrix Analysis and Applications*, 13(4):1094–1122, 1992.
12. Rahul Mazumder, Trevor Hastie, and Robert Tibshirani. Spectral Regularization Algorithms for Learning Large Incomplete Matrices. *Journal of Machine Learning Research*, 11:2287–2322, 2010.
13. Arkadi Nemirovski. Prox-method with Rate of Convergence O(1/T) for Variational Inequalities with Lipschitz Continuous Monotone Operators and Smooth Convex-concave Saddle Point Problems. *SIAM Journal on Optimization*, 15:229–251, 2004.
14. Yurii Nesterov. Smoothing Technique and its Applications in Semidefinite Optimization. *Math. Program.*, 110(2):245–259, 2007.
15. Ruslan Salakhutdinov and Nathan Srebro. Collaborative Filtering in a Non-Uniform World: Learning with the Weighted Trace Norm. In *Proceedings of Advances in Neural Information Processing Systems 23 (NIPS)*, 2010.
16. Nathan Srebro, Jason D. M. Rennie, and Tommi Jaakkola. Maximum-Margin Matrix Factorization. In *Proceedings of Advances in Neural Information Processing Systems 17 (NIPS)*, 2004.
17. Zaiwen Wen, Donald Goldfarb, and Wotao Yin. Alternating Direction Augmented Lagrangian Methods for Semidefinite Programming. Technical report, 2009.