Accepted for publication in **Journal of Optimization Theory and Applications**
https://doi.org/10.1007/s10957-021-01820-3

Quartic First-Order Methods for Low-Rank Minimization

Radu-Alexandru Dragomir · Alexandre d'Aspremont · Jérôme Bolte

Last revised on January 18, 2021

Abstract We study a general nonconvex formulation for low-rank minimization problems. We use recent results on non-Euclidean first-order methods to provide efficient and scalable algorithms. Our approach uses the geometry induced by the Bregman divergence of well-chosen kernel functions; for unconstrained problems we introduce a novel family of Gram quartic kernels that improve numerical performance.

Numerical experiments on Euclidean distance matrix completion and symmetric nonnegative matrix factorization show that our algorithms scale well and reach state of the art performance when compared to specialized methods.

Keywords Bregman first-order methods · Low-rank minimization · Burer-Monteiro · Matrix factorization · Euclidean Distance matrix completion

Mathematics Subject Classification (2000) 90C06 · 90C26

1 Introduction

We consider the problem of minimizing a smooth convex function over the set of low-rank positive semidefinite matrices. Fundamental applications of this problem arise in various areas of data analysis including matrix completion [1–3], matrix sensing [4], Euclidean matrix completion [5,6], phase retrieval [7], robust principal component analysis [8], to name a few.

A popular approach to low-rank semidefinite minimization, known as the Burer-Monteiro formulation [9], consists in explicitly modeling the rank constraint by writing the matrix in a factorized form. This method is especially appealing for large-scale instances, since it requires storing much less variables than the standard semidefinite programming approaches; see [8, 10–15] and references therein.

This formulation comes however with an important drawback, as the problem becomes nonconvex, even if the original objective is convex. Therefore, local optimization methods can generally only hope to find a stationary point, or at best a local minimum. Nevertheless, recent work shows convergence towards a global optimum for a close enough initialization [10, 11, 15], or under additional statistical assumptions about the problem [8,16,17]. Although these global optimality results often impose restrictive assumptions that may not be satisfied in practice, they help to explain why using local algorithms to solve Burer-Monteiro problem formulations often leads to satisfactory solutions in practice.

The most commonly used algorithm to solve these problem formulations is some variant of the proximal gradient method. However, a critical issue with gradient schemes is the choice of step sizes,

Radu-Alexandru Dragomir, corresponding author
Université Toulouse 1 Capitole & D.I. École Normale Supérieure, Paris, France
radu-alexandru.dragomir@inria.fr

Alexandre d'Aspremont
CNRS & D.I. École Normale Supérieure, Paris, France
aspremon@ens.fr.

Jérôme Bolte
Université Toulouse 1 Capitole, Toulouse, France
jerome.bolte@ut-capitole.fr

which significantly impacts performance. This step size choice is closely related to the smoothness of the objective. In particular, when it has a L-Lipschitz continuous gradient with respect to the Euclidean norm, standard gradient methods can be applied with a step size lying in $]0, 1/L]$. This smoothness assumption is used in the broad majority of theoretical analyses of gradient algorithms, yet there are many cases where it is not satisfied [18,19]. In particular, it does not hold for the general Burer-Monteiro low-rank problem, as we will show in what follows.

Of course, there is a way to circumvent this issue in classical Euclidean methods, by using an Armijo line search [20]. However, in some cases, this naive line search strategy generates very small step sizes which in turn involve costly subroutines. Other approaches impose a step size that is only proven to be valid in a small neighborhood of the optimum [11, 15].

Non-Euclidean gradient methods. We adopt an original approach based on a recent line of work on non-Euclidean gradient methods [18, 19, 21] and subsequent work [22]. Unlike standard gradient descent that uses the uniform Euclidean geometry, the NoLips method, also known as Bregman/Mirror descent, uses the Bregman divergence induced by a well-chosen convex *kernel* function. This allows the algorithm to take gradient steps that are more adapted to the geometry of the problem, advancing faster in directions where the gradient of the objective changes slowly, thus improving convergence speed. The kernel function is chosen so that the objective function satisfies a compatibility condition called *relative smoothness* [18, 22], which is a generalization of the usual smoothness assumption mentioned earlier.

In our setting, the objective has a quartic growth, hence choosing the geometry induced by a quartic polynomial will prove to be efficient.

Contributions. In this work, we focus on deriving efficient algorithms to find stationary points of non-convex low-rank problems. Our main contribution is to identify favorable non-Euclidean geometries for these problems, induced by well-chosen quartic kernels.

We first study a simple quartic *norm* kernel that is compatible with various regularization terms. We then introduce a novel family of quartic kernels that we call *Gram kernels*, which can be applied to unregularized problems. They provide richer geometries which greatly improve convergence speed with little impact on the iteration complexity. We also extend the NoLips scheme to Dyn-NoLips, allowing for adaptive step size strategies.

To highlight the benefits of our approach, we study applications to symmetric nonnegative matrix factorization and Euclidean distance matrix completion and show competitive numerical performance compared to specialized algorithms for these problems.

Notations For a square matrix M, we denote its trace $\mathbf{Tr}\, M = \sum_{i=1}^n M_{ii}$. For two matrices X and Y of same size, we denote the standard Euclidean inner product and norm by $\langle X, Y\rangle = \mathbf{Tr}(X^T Y)$ and $\|X\| = \sqrt{\mathbf{Tr}(X^T X)}$. For a function $f : \mathbb{R}^{n\times r} \to \mathbb{R}$, we denote by $\nabla F(X)$ its gradient matrix $\nabla F(X)_{ij} = \frac{\partial F(X)}{\partial x_{ij}}$ and by $\nabla^2 F(X)[U, V]$ the second derivative at X in the directions $U, V \in \mathbb{R}^{n\times r}$. I_r denotes the identity matrix of size $r \times r$. For two square matrices X, Y, we write $X \preceq Y$ if the matrix $Y - X$ is positive semidefinite. We write $\|\mathcal{A}\|_{\text{op}}$ for the operator norm of a linear application $\mathcal{A}$.

2 Quartic Geometries for Low-Rank Minimization

2.1 Problem Setup

Let $n \geq 1$ and consider a low-rank semidefinite program, written

$$\min F(Y) \quad \text{subject to } Y \succeq 0,\ \text{rank}(Y) \leq r \tag{SDP-r}$$

in the variable $Y \in \mathbb{R}^{n\times n}$, where F is a smooth convex function and $r \leq n$ is the target rank. The Burer-Monteiro formulation [9] consists in representing Y as $Y = XX^T$ to solve instead

$$\min \Psi(X) := F(XX^T) + g(X) \tag{P}$$

in the variable $X \in \mathbb{R}^{n\times r}$, where g is a *simple* convex regularization function. F is typically a quadratic loss function, and g enforces penalties on the factor X such as sparsity when choosing the ℓ_1 norm, or

nonnegativity when choosing the indicator function of the nonnegative orthant. We will write $f : \mathbb{R}^{n\times r} \to \mathbb{R}$ the factorized function defined by

$$f(X) := F(XX^T).$$

Throughout the paper, we make the following standing assumptions.

Assumption 2.1 (a) *$F : \mathbb{R}^{n\times n} \to \mathbb{R}$ is a twice continuously differentiable function which is μ_F-strongly convex and L_F-smooth, i.e.,*

$$\begin{aligned}\langle \nabla F(X) - \nabla F(Y), X - Y\rangle &\geq \mu_F \|X - Y\|^2,\\ \|\nabla F(X) - \nabla F(Y)\| &\leq L_F\|X - Y\| \quad \forall X, Y \in \mathbb{R}^{n\times n},\end{aligned}$$

(b) *$g : \mathbb{R}^{n\times n} \to \mathbb{R} \cup \{+\infty\}$ is a closed convex proper function,*
(c) $\min_{\mathbb{R}^{n\times r}} \Psi > -\infty$.

Our analysis will involve the following lemma.

Lemma 2.1 *Let $F : \mathbb{R}^{n\times n} \to \mathbb{R}$ be a twice differentiable μ_F-strongly convex and L_F-smooth function. Then, the function $G := F - \frac{\mu_F}{2}\|\cdot\|^2$ is convex and $(L_F - \mu_F)$-smooth.*

Proof It suffices to use the second-order characterization [23] and notice that, for $Y, U \in \mathbb{R}^{n\times n}$, we have $\nabla^2 G(Y)[U, U] = \nabla^2 F(Y)[U, U] - \mu_F\|U\|^2$ and hence

$$\mu_F\|U\|^2 \leq \nabla^2 F(Y)[U, U] \leq L_F\|U\|^2 \implies 0 \leq \nabla^2 G(Y)[U, U] \leq (L_F - \mu_F)\|U\|^2.$$

□

2.2 Relative Smoothness and the Bregman Iteration Map

In this section, we recall the framework of [18, 19] to derive non-Euclidean gradient methods.

The first essential step is the choice of a *distance kernel*. In our context, we choose a differentiable strictly convex function $h : \mathbb{R}^{n\times r} \to \mathbb{R}$, with $\mathbf{dom}\, h = \mathbb{R}^{n\times r}$ (although more general distance kernels can be used). The distance kernel h induces in turn a *Bregman distance*

$$D_h(X, Y) = h(X) - h(Y) - \langle \nabla h(Y), X - Y\rangle. \tag{1}$$

Note that D_h is not a proper distance, it is sometimes referred to as a *Bregman divergence*. However D_h enjoys a distance-like separation property: $D_h(X, X) = 0$ and $D_h(X, Y) > 0$ for $X \neq Y$. The choice of a distance kernel suited to the function f is guided by the following relative smoothness condition, also called generalized Lipschitz property.

Definition 2.1 (Relative smoothness [18]) We say that a differentiable function $f : \mathbb{R}^{n\times r} \to \mathbb{R}$ is L-smooth relatively to the distance kernel h if there exists $L > 0$ such that for every $X, Y \in \mathbb{R}^{n\times r}$,

$$f(X) \leq f(Y) + \langle \nabla f(Y), X - Y\rangle + L\, D_h(X, Y). \tag{RelSmooth}$$

For twice differentiable functions, relative smoothness has an elementary characterization: f is L-smooth relatively to h if and only if

$$\nabla^2 f(X)[U, U] \leq L\, \nabla^2 h(X)[U, U], \quad \forall X, U \in \mathbb{R}^{n\times r} \tag{2}$$

where $\nabla^2 f(X)[U, U]$ denotes the second derivative of f at X in the direction U. Notice that if $h(X) = \frac{1}{2}\|X\|^2$, then $D_h(X, Y) = \frac{1}{2}\|X - Y\|^2$ and we recover the standard Euclidean descent lemma that would be implied by Lipschitz continuity of the gradient of f.

Bregman iteration map Now that we are equipped with a non-Euclidean geometry generated by h, we define the Bregman proximal iteration map with step size λ as follows.

$$T_\lambda(X) = \operatorname*{argmin}_{U\in\mathbb{R}^{n\times r}} \left\{ g(U) + f(X) + \langle \nabla f(X), U - X\rangle + \frac{1}{\lambda} D_h(U, X) \right\}, \tag{3}$$

which consists in minimizing a surrogate for Ψ where f has been replaced by the upper approximation given by (RelSmooth) and the nonsmooth part g is kept intact, generalizing thus the approach used in the proximal gradient method. The relative smoothness condition ensures that this operation decreases the objective Ψ when $\lambda \in]0, 1/L]$. This iteration map is the basic brick for non-Euclidean methods à la Bregman. The simplest method is NoLips [18] and its extension Dyn-NoLips (Algorithm 1), which simply amounts to iterating $X^{k+1} = T_{\lambda_k}(X^k)$, but other possibilities exist using momentum ideas [24–26].

2.3 The Quartic Geometry

In order to provide some insight into the quartic geometry of our problem, let us consider the example where F is a *quadratic* function, i.e.,

$$F(Y) = \frac{1}{2}\langle \mathcal{A}Y, Y\rangle + \langle B, Y\rangle \quad \forall\, Y \in \mathbb{R}^{n\times n}, \tag{4}$$

where $B \in \mathbb{R}^{n\times n}$ and $\mathcal{A} : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$ is some linear map. Then, f writes

$$f(X) = F(XX^T) = \frac{1}{2}\langle \mathcal{A}(XX^T), XX^T\rangle + \langle BX, X\rangle \quad \forall\, X \in \mathbb{R}^{n\times r}.$$

Clearly, f is a *quartic* function and its gradient is not Lipschitz continuous on $\mathbb{R}^{n\times r}$, as the Hessian "grows" to infinity when $\|X\| \to \infty$. In other words, (RelSmooth) does not hold with the Euclidean kernel $h = \frac{1}{2}\|\cdot\|^2$. We now show that relative smoothness holds with a family of well-chosen quartic kernels, which are more adapted to the geometry of f.

2.3.1 The Quartic Norm Kernel

We begin with the simplest quartic kernel, which depends solely on the Frobenius norm of X. Define the *norm kernel* h_N as

$$h_N(X) = \frac{\alpha}{4}\|X\|^4 + \frac{\sigma}{2}\|X\|^2 \quad \forall\, X \in \mathbb{R}^{n\times r}, \tag{5}$$

where $\alpha, \sigma > 0$ are fixed parameters. Note that this kernel is not new by itself, as it has been already studied in [19] for vectors in $\mathbb{R}^n$. Our first contribution is to show that it is adapted to every function of our class of problems.

Proposition 2.1 (Norm kernel) *The function f is 1-smooth relative to the norm kernel h_N for $\alpha \geq 6L_F$ and $\sigma \geq 2\|\nabla F(0)\|$.*

Proof As F is twice differentiable, then so if f and we can use the Hessian characterization (2). For $X, U \in \mathbb{R}^{n\times r}$, the second derivative of h_N is written

$$\begin{aligned} \nabla^2 h_N(X)[U, U] &= \alpha\left(\|X\|^2\|U\|^2 + 2\langle X, U\rangle^2\right) + \sigma\|U\|^2 \\ &\geq \alpha\|X\|^2\|U\|^2 + \sigma\|U\|^2. \end{aligned} \tag{6}$$

On the other hand, the second derivative of f is

$$\nabla^2 f(X)[U, U] = \nabla^2 F(XX^T)[UX^T + XU^T, UX^T + XU^T] + 2\langle \nabla F(XX^T), UU^T\rangle. \tag{7}$$

Since F has a Lipschitz continuous gradient, the standard second derivative inequality yields

$$\nabla^2 F(XX^T)[UX^T + XU^T, UX^T + XU^T] \leq L_F\|UX^T + XU^T\|^2.$$

Now, the second term can be bounded by using the triangle inequality, the Cauchy-Schwarz inequality and the gradient Lipschitz property, to get

$$\begin{aligned}\langle \nabla F(XX^T), UU^T\rangle &= \langle \nabla F(0), UU^T\rangle + \langle \nabla F(XX^T) - \nabla F(0), UU^T\rangle \\ &\leq \|\nabla F(0)\|\,\|U\|^2 + \|\nabla F(XX^T) - \nabla F(0)\|\,\|UU^T\| \\ &\leq \Big(\|\nabla F(0)\| + L_F\|XX^T\|\Big)\|U\|^2\end{aligned}$$

hence

$$\begin{aligned}\nabla^2 f(X)[U,U] &\leq L_F\|UX^T + XU^T\|^2 + 2\left(L_F\|XX^T\| + \|\nabla F(0)\|\right)\|U\|^2 \\ &\leq 2L_F\left(\|UX^T\|^2 + \|XU^T\|^2\right) + 2\left(L_F\|XX^T\| + \|\nabla F(0)\|\right)\|U\|^2 \\ &\leq 6L_F\|X\|^2\|U\|^2 + 2\|\nabla F(0)\|\|U\|^2 \\ &\leq \alpha\|X\|^2\|U\|^2 + \sigma\|U\|^2\end{aligned} \tag{8}$$

where we used the submultiplicative property of the Frobenius norm, and our choice of parameters α, σ. Combining (6) and (8) gives that

$$\nabla^2 f(X)[U,U] \leq \nabla^2 h_N(X)[U,U]$$

for all $X, U \in \mathbb{R}^{n\times r}$, hence that f is 1-smooth relatively to h [18]. □

The Bregman iteration map (3) associated with the kernel h_N can be computed easily in closed form. We give its expression in the unconstrained case [19].

Proposition 2.2 (Bregman iteration map for h_N, unconstrained case) *Assume that there is no penalty term, i.e., that $g \equiv 0$. The Bregman iteration map of the norm kernel h_N with step size $\lambda > 0$ is given by*

$$T_\lambda(X) = \frac{1}{\tau_\sigma(\alpha\|U\|^2)}U$$

where

$$U = \nabla h_N(X) - \lambda\nabla f(X) = (\alpha\|X\|^2 + \sigma)X - \lambda\nabla f(X)$$

and $\tau_\sigma(c)$ denotes the unique real solution z to the cubic equation $z^2(z-\sigma) = c$.

Note that $\tau_\sigma(c)$ can be computed in closed form using Cardano's method

$$\tau_\sigma(c) = \frac{\sigma}{3} + \sqrt[3]{\frac{c+\sqrt{\Delta}}{2} + \frac{\sigma^3}{27}} + \sqrt[3]{\frac{c-\sqrt{\Delta}}{2} + \frac{\sigma^3}{27}} \text{ where } \Delta = c^2 + \frac{4}{27}c\sigma^3. \tag{9}$$

Compared to a standard gradient iteration, the additional operations are elementary and have a minimal impact on the arithmetic complexity.

Constraints and regularization terms. Following the ideas in [19], the Bregman iteration map of h_N can also be easily computed in closed form when g is the ℓ_1 norm or the ℓ_0 pseudonorm. As we will show in Section 4.1, this is also elementary when g is the indicator function of the nonnegative orthant.

2.3.2 A More Refined Kernel for Unregularized Problems: the Gram Kernel

While the kernel h_N is simple and compatible with many penalties g, a better kernel can be derived for unconstrained instances by considering a richer geometry involving the Gram matrix. Define the *Gram kernel* as

$$h_G(X) = \frac{\alpha}{4}\|X\|^4 + \frac{\beta}{4}\|X^TX\|^2 + \frac{\sigma}{2}\|X\|^2 \quad \forall\, X \in \mathbb{R}^{n\times r}, \tag{10}$$

where $\alpha, \beta \geq 0$, $\sigma > 0$ are given parameters. The Gram kernel is more refined than the previous norm kernel since it incorporates some nonisotropic information with the $\|X^TX\|^2$ term. To show where this

term stems from, observe that following Lemma 2.1, F can be decomposed as $F = \frac{\mu_F}{2}\|\cdot\|^2 + \tilde{F}$ where $\tilde{F}$ is $(L_F - \mu_F)$-smooth. Hence f writes

$$f(X) = F(XX^T) = \frac{\mu_F}{2}\|XX^T\|^2 + \tilde{F}(XX^T). \tag{11}$$

Since $\|XX^T\|^2 = \|X^TX\|^2$, the first term can be directly incorporated into the kernel, which allows to prove a tighter relative smoothness inequality.

Proposition 2.3 (Gram kernel) *f is 1-smooth relatively to the Gram kernel h_G for $\alpha \geq 2(L_F - \mu_F)$, $\beta \geq 2L_F$ and $\sigma \geq 2\|\nabla F(0)\|$.*

Proof This amounts to refine the analysis of the proof of Proposition 2.1. Let $X, U \in \mathbb{R}^{n\times r}$. The second derivative of h_G at X in the direction U writes

$$\begin{aligned}\nabla^2 h_G(X)[U,U] &= \alpha\left(\|X\|^2\|U\|^2 + 2\langle X,U\rangle^2\right)\\ &\quad + \beta\left(\frac{1}{2}\|UX^T + XU^T\|^2 + \|U^TX\|^2\right) + \sigma\|U\|^2\\ &\geq \alpha\|X\|^2\|U\|^2 + \beta\left(\frac{1}{2}\|UX^T + XU^T\|^2 + \|U^TX\|^2\right) + \sigma\|U\|^2.\end{aligned} \tag{12}$$

On the other hand, following (7) the second derivative of f satisfies

$$\nabla^2 f(X)[U,U] \leq L_F\|UX^T + XU^T\|^2 + 2\langle \nabla F(XX^T), UU^T\rangle$$

To bound the second term, we use Lemma 2.1 which states that the function $G(Y) := F(Y) - \mu_F\|Y\|^2/2$ is convex and smooth with constant $L_F - \mu_F$. Using the gradient Lipschitz property of G yields

$$\begin{aligned}\langle \nabla F(XX^T), UU^T\rangle &= \langle \nabla F(0), UU^T\rangle + \mu_F\langle XX^T, UU^T\rangle\\ &\quad + \langle \nabla F(XX^T) - \nabla F(0) - \mu_F(XX^T - 0), UU^T\rangle\\ &= \langle \nabla F(0), UU^T\rangle + \mu_F\|U^TX\|^2 + \langle \nabla G(XX^T) - \nabla G(0), UU^T\rangle\\ &\leq \|\nabla F(0)\|\,\|U\|^2 + \mu_F\|U^TX\|^2 + (L_F - \mu_F)\|XX^T\|\,\|UU^T\|\\ &\leq \|\nabla F(0)\|\,\|U\|^2 + L_F\|U^TX\|^2 + (L_F - \mu_F)\|X\|^2\|U\|^2,\end{aligned}$$

using that $\mu_F \leq L_F$, and so we have

$$\begin{aligned}\nabla^2 f(X)[U,U] &\leq 2(L_F - \mu_F)\|X\|^2\|U^2\| + L_F\|UX^T + XU^T\|^2 + 2L_F\|U^TX\|^2\\ &\quad + 2\|\nabla F(0)\|\,\|U\|^2\\ &\leq \alpha\|X\|^2\|U\|^2 + \frac{\beta}{2}\|UX^T + XU^T\|^2 + \beta\|U^TX\|^2 + \sigma\|U\|^2\\ &\leq \nabla^2 h_G(X)[U,U]\end{aligned}$$

which shows that, for the prescribed choice of α, β, σ, the function f is 1-smooth relatively to h_G. □

Approximation quality for well-conditioned F. Let us illustrate the advantage of the Gram kernel when F is well-conditioned. For simplicity, assume here that F is a quadratic function, as in (4), i.e., $F(Y) = \frac{1}{2}\langle \mathcal{A}Y, Y\rangle + \langle B, Y\rangle$ where $\mathcal{A}$ is a positive semidefinite linear operator on $\mathbb{R}^{n\times r}$, and hence f has a *quartic* and a *quadratic* term

$$f(X) = \frac{1}{2}\langle \mathcal{A}(XX^T), XX^T\rangle + \langle B, XX^T\rangle.$$

The gap between f and h_G with the choice of coefficients prescribed by Proposition 2.3 writes, for $X \in \mathbb{R}^{n\times r}$,

$$\begin{aligned} h_G(X) - f(X) &= \frac{(L_F-\mu_F)}{2}\|X\|^4 + \frac{L_F}{2}\|X^TX\|^2 + \|\nabla F(0)\|\,\|X\|^2 \\ &\quad - \frac{1}{2}\langle \mathcal{A}(XX^T), XX^T\rangle - \langle BX, X\rangle \\ &= \underbrace{\frac{(L_F-\mu_F)}{2}\|X\|^4 + \frac{1}{2}\langle (L_FI - \mathcal{A})(XX^T), XX^T\rangle}_{d_4(X)} \\ &\quad + \underbrace{\langle (\|\nabla F(0)\|I - B)X, X\rangle}_{d_2(X)} \end{aligned} \tag{13}$$

where we separated the gap into a quartic term d_4 and a quadratic term d_2. It can be seen from (2) that the quality of approximation of the kernel is given by the difference of the Hessians. Focusing on the quartic part, the Hessian difference is

$$\begin{aligned} \nabla^2 d_4(X)[U,U] &= 2(L_F-\mu_F)\left(\|X\|^2\|U\|^2 + 2\langle X, U\rangle^2\right) + 2\langle (L_FI-\mathcal{A})(XX^T), UU^T\rangle \\ &\quad + \langle (L_FI-\mathcal{A})(UX^T + XU^T), UX^T + XU^T\rangle \\ &\le 6(L_F-\mu_F)\|X\|^2\|U\|^2 \\ &\quad + \|L_FI - \mathcal{A}\|_{\mathrm{op}}\left(2\|XX^T\|\,\|UU^T\| + \|UX^T + XU^T\|^2\right) \end{aligned}$$

for $X, U \in \mathbb{R}^{n\times r}$. Recalling that F is L_F-smooth and μ_F-strongly convex, we have that $\|L_FI - \mathcal{A}\|_{\mathrm{op}} \le (L_F - \mu_F)$, therefore

$$\begin{aligned} \nabla^2 d_4(X)[U,U] &\le (L_F-\mu_F)\left(6\|X\|^2\|U\|^2 + 2\|XX^T\|\,\|UU^T\| + \|UX^T + XU^T\|^2\right) \\ &\le 12L_F(1 - \frac{\mu_F}{L_F})\|X\|^2\|U\|^2 \end{aligned}$$

which shows that the quality of approximation of the quartic part of f by the Gram kernel depends on the condition number $\kappa_F := L_F/\mu_F$ of F. Note that one could actually refine the analysis by replacing κ_F with the condition number of F restricted to the set of matrices of rank at most $2r$, which can be much smaller. This is the case when the linear map $\mathcal{A}$ satisfies the *restricted isometry property* (RIP), which occurs with high probability in matrix sensing applications with a sufficiently large number n of samples [3, 4, 27].

Computing the iteration map. We show now that, when there is no penalty term g, the Bregman iteration map of h_G can be computed efficiently, as it involves solving an easy quartic minimization subproblem of size r.

Proposition 2.4 (Gram's iteration map) *Assume that $g \equiv 0$. For $X \in \mathbb{R}^{n\times r}$, the Bregman iteration map of f for the Gram kernel h_G with step size $\lambda > 0$, called Gram's iteration map, is given by*

$$T_\lambda(X) = V\left[\alpha\,\mathbf{Tr}(Z)I_r + \beta Z + \sigma I_r\right]^{-1}$$

where the matrices V, Z are computed through the routine:

- *Set $V = \nabla h_G(X) - \lambda\nabla f(X)$,*
- *diagonalize V^TV as $V^TV = P^TDP$ where $P \in \mathcal{O}_r$ and $D = \mathbf{diag}(\eta_1^2, \dots, \eta_r^2)$,*
- *let $\mu = (\mu_1, \dots, \mu_r)$ be the unique solution of the convex minimization problem*

$$\min_{x\in\mathbb{R}^r} \phi(x) := \frac{\alpha}{4}\|x\|^4 + \frac{\beta}{4}\sum_{i=1}^r x_i^4 + \frac{\sigma}{2}\|x\|^2 - \sum_{i=1}^r \eta_i x_i,$$

- *finally set $Z = P^T\,\mathbf{diag}\left[\mu_1^2, \dots, \mu_r^2\right]P$.*

Proof When $g \equiv 0$, The Bregman iteration map of h_G writes, for $X \in \mathbb{R}^{n\times r}$,

$$\begin{aligned} T_\lambda(X) &= \operatorname*{argmin}_{U\in\mathbb{R}^{n\times r}} \left\{ \langle \nabla f(X), U - X\rangle + \frac{1}{\lambda} D_{h_G}(U, X) \right\} \\ &= \operatorname*{argmin}_{U\in\mathbb{R}^{n\times r}} \{h_G(U) - \langle V, U\rangle\} \end{aligned} \tag{14}$$

where we remove constant terms and defined $V := \nabla h_G(X) - \lambda \nabla f(X)$. Write for the sake of clarity $U^\star := T_\lambda(X)$. The optimization problem (14) is strictly convex and the unique solution $U^\star$ satisfies $\nabla h_G(U^\star) = V$, meaning that

$$U^\star \left(\alpha \|U^\star\|^2 I_r + \beta U^{\star T} U^\star + \sigma I_r\right) = V. \tag{15}$$

Define $Z := U^{\star T} U^\star \in \mathbb{R}^{r\times r}$. Then, the knowledge of Z determines $U^\star$, since $\|U^\star\|^2 = \mathbf{Tr}(Z)$ and therefore $U^\star = V(\alpha \mathbf{Tr}(Z) I_r + \beta Z + \sigma I_r)^{-1}$.

Now, taking (15) and multiplying by its transpose implies that

$$(\alpha \|U^\star\|^2 I_r + \beta Z + \sigma I_r)^2 Z = V^T V. \tag{16}$$

This shows that $V^T V$ is a polynomial in Z, and therefore that they admit the same eigenvectors. Write the diagonalization $V^T V = P^T \mathbf{diag}(\eta_1^2, \dots, \eta_r^2) P$ and $Z = P^T \mathbf{diag}(\mu_1^2, \dots, \mu_r^2) P$ where $P \in \mathcal{O}_r$ and $\mu_i, \eta_i \geq 0$ for $i = 1 \dots r$. It follows from diagonalizing (16) and taking the square root that

$$\left(\alpha \left(\sum_{j=1}^r \mu_j^2\right) + \beta \mu_i^2 + \sigma\right) \mu_i = \eta_i \quad \forall i = 1, \dots, r \tag{17}$$

This is exactly the first-order optimality condition on $\mu = (\mu_1, \dots, \mu_r)$ for the problem

$$\mu = \operatorname*{argmin}_{x\in\mathbb{R}^r} \frac{\alpha}{4}\|x\|^4 + \frac{\beta}{4}\sum_{i=1}^r x_i^4 + \frac{\sigma}{2}\|x\|^2 - \sum_{i=1}^r \eta_i x_i. \tag{18}$$

Note that we do not need to enforce the nonnegativity constraint on x, since we chose $\eta_i \geq 0$ it follows that the optimal solution will be nonnegative. Hence, we can reconstruct Z from the diagonalization of $V^T V$ and the solution of Problem (18), and thus we get the procedure described in the theorem for computing $U^* = T_\lambda(X)$. □

Complexity Note that the order of multiplication is important: we only need to compute the eigendecomposition of $V^T V$, which is of size $r \times r$. We additionally need to solve a small minimization problem of size r, which can be done efficiently using the quartic NoLips algorithm with norm kernel (see Appendix A for implementation details). Due to this, the complexity of computing the Bregman iteration map of h_G is $O(nr^2 + r^3 + Kr)$, where K is the number of iterations needed to solve the subproblem. Since r is usually much smaller than n by several orders of magnitude, the main computational bottleneck remains in most applications computing the gradient $\nabla f(X)$.

2.3.3 Comparison: how to choose the most appropriate kernel

In order to devise efficient methods, one should search for the kernel h such that the upper approximation of f in (RelSmooth) is *as tight as possible*, or, equivalently, such that the Hessian of the residual $Lh - f$ is small. On the other hand, h has to be simple enough so that the iteration map (3) is *easy to compute* (which precludes choosing $h = f$, as the iteration would be as hard to solve as the initial problem). This trade-off is key in choosing the appropriate kernel. Let us review these two conflicting criteria in our situation.

Complexity of the Bregman iteration map. For the norm kernel h_N, one iteration involves computing the gradient of f, then solving a simple scalar equation. The Gram kernel h_G involves solving a subproblem which requires $O(nr^2 + r^3)$ additional operations. This overhead is negligible for the typical regime where $r \ll n$; however, the iterate can be computed easily only for unconstrained problems.

Quality of Hessian approximation. We showed in Section 2.3.2 that the quality of the approximation of the quartic component of f by the Gram kernel is bounded by $O(1 - \mu_F/L_F)$. Therefore, it is expected to show good performance when F is sufficiently well-conditioned. The norm kernel, however, has no such property, as its approximation of f is much coarser. The difference stems from the supplementary $\|X^T X\|^2$ term, which can be much smaller than $\|X\|^4$, especially when the columns of X are nearly orthogonal.

Note that even if F is not globally strongly convex or μ_F is unknown, the Gram kernel can take advantage of local strong convexity through adaptive step sizes, as we show in the sequel.

3 Algorithms for Quartic Low-Rank Minimization

Now that we are equipped with a non-Euclidean geometry induced by one of the kernels h_N and h_G, we are ready to define the minimization scheme Dyn-NoLips in Algorithm 1. It extends the NoLips algorithm from [19] to allow step sizes larger than the theoretical value $1/L$.

Algorithm 1 Dyn-NoLips

Input: A distance kernel h such that f is smooth relatively to h and a maximal step size $\lambda_{\max}$
Initialize $X^0 \in \mathbb{R}^{n\times r}$ such that $\Psi(X^0) < \infty$.
for k = 1,2,... **do**
  Choose a step size $\lambda_k \leq \lambda_{\max}$ such that the sufficient decrease condition (19) holds
  Set $X^k = T_{\lambda_k}(X^{k-1})$
end for

Step size choice The step size λ_k is chosen so that the new iterate $X^k = T_{\lambda_k}(X^{k-1})$ satisfies

$$f(X^k) \leq f(X^{k-1}) + \langle \nabla f(X^{k-1}), X^k - X^{k-1} \rangle + \frac{1}{\lambda_k} D_h(X^k, X^{k-1}). \tag{19}$$

There are two ways to ensure this condition holds.

- **Fixed step size.** Since f is L-smooth relatively to h, (19) holds as soon as $0 < \lambda_k \leq 1/L$.
- **Dynamical step size.** In some cases, the relative Lipschitz constant might be too conservative, and better numerical performance can be achieved by taking larger steps. We therefore can use a dynamical strategy for extending the step size, ensuring that (19) holds at each iteration. There are many strategies to efficiently adjust the step size; see, e.g., [28]. In our case, we choose a simple strategy similar in spirit to the Armijo line search: at iteration k, start with a tentative step size λ_k, then find the smallest integer j such that (19) is satisfied with step size $2^{-j}\lambda_k$. Then, set $\lambda_{k+1} = 2^{-j+1}\lambda_k$.

Convergence to a stationary point. We now extend the theoretical convergence results from [19] to handle the dynamical step size strategy.

Theorem 3.1 (Convergence results) *Let $\{X^k\}_{k\geq 0}$ be the sequence generated by Algorithm 1. Assume that*

1. *f is L-smooth relatively to a distance kernel h such that h is strongly convex and twice continuously differentiable on $\mathbb{R}^{n\times r}$, and the penalty function g is convex.*
2. *The function $\Psi = f + g$ is coercive (meaning that $\Psi(X) \to +\infty$ when $\|X\| \to +\infty$) and semialgebraic.*

Then, the sequence $\{\Psi(X^k)\}_{k\geq 0}$ is nonincreasing, and the sequence $\{X^k\}_{k\geq 0}$ converges towards a critical point X^* of problem (P).

Proof First, the step size λ_k can be bounded for $k \geq 0$ as

$$\frac{1}{2L} \leq \lambda_k \leq \lambda_{\max}. \tag{20}$$

Indeed, the upper bound holds by construction of the algorithm. The lower bound comes from the relative smoothness property: condition (19) is true for every $\lambda \in (0, \frac{1}{L}]$, so the inner loop will stop whenever λ gets below $1/L$.

Let us now prove the result. Since Condition (19) holds at each iteration k, we can write

$$f(X^{k+1}) \leq f(X^k) + \langle \nabla f(X^k), X^{k+1} - X^k \rangle + \frac{1}{\lambda_k} D_h(X^{k+1}, X^k). \tag{21}$$

On the other hand, the optimality condition characterizing $X^{k+1} = T_{\lambda_k}(X^k)$ writes

$$0 \in \lambda_k \left(\partial g(X^{k+1}) + \nabla f(X^k) \right) + \nabla h(X^{k+1}) - \nabla h(X^k), \tag{22}$$

where ∂g denotes the subdifferential of the convex function g. Combining (22) with the subgradient inequality for g yields

$$g(X^{k+1}) \leq g(X^k) + \frac{1}{\lambda_k} \langle \nabla h(X^k) - \nabla h(X^{k+1}), X^{k+1} - X^k \rangle \\ - \langle \nabla f(X^k), X^{k+1} - X^k \rangle. \tag{23}$$

Summing (21) and (23) gives

$$\Psi(X^{k+1}) \leq \Psi(X^k) + \frac{1}{\lambda_k} \big[D_h(X^{k+1}, X^k) + \langle \nabla h(X^k) - \nabla h(X^{k+1}), X^{k+1} - X^k \rangle \big],$$

which yields

$$\Psi(X^{k+1}) \leq \Psi(X^k) - \frac{1}{\lambda_k} D_h(X^k, X^{k+1}). \tag{24}$$

From this inequality, we can now prove the same convergence properties as for the standard NoLips scheme. Indeed, the monotonicity of the sequence $\{\Psi(X^k)\}_{k \geq 0}$ is a direct consequence of the above. Since $\lambda_k \leq \lambda_{\max}$, it follows that at every iteration $k \geq 0$,

$$\Psi(X^k) - \Psi(X^{k+1}) \geq \frac{1}{\lambda_{\max}} D_h(X^k, X^{k+1}).$$

Now, this inequality is the same as the one needed to prove convergence in the case of the fixed step size in [19]. Thus, global convergence towards a critical point is a consequence of [19, Th. 4.1], since all the assumptions are met: the kernel h is defined over the entire space $\mathbb{R}^{n \times r}$, it is strongly convex, and ∇h is Lipschitz continuous on bounded subsets of $\mathbb{R}^{n \times r}$ (because we assumed it is C^2). We also need the fact that the sequence $\{X^k\}_{k \geq 0}$ is bounded, which is a consequence of the monotonicity of $\{\Psi(X^k)\}_{k \geq 0}$ and the fact that the function Ψ is coercive. □

The semialgebraicity assumption is needed to establish the crucial nonsmooth Lojasiewicz property [29], required to show convergence to a critical point. It holds for all the applications we cited, since the class of semialgebraic functions includes polynomial functions, ℓ_1 and ℓ_2 norms, the ℓ_0 seminorm and indicators of polynomial sets.

4 Applications

We now illustrate applications of our methodology to two different low-rank problems, symmetric nonnegative matrix factorization and Euclidean distance matrix completion. We show that good numerical performance can be reached using the dynamical step strategy, and that, for Euclidean matrix completion, it can be further improved by using the Gram kernel.

4.1 Symmetric Nonnegative Matrix Factorization

Symmetric Nonnegative Matrix Factorization (SymNMF) is the task of finding, given a symmetric nonnegative matrix $M \in \mathbb{R}^{n \times n}$, a nonnegative matrix $X \in \mathbb{R}^{n \times r}$ such that $M \approx XX^T$. This is done by solving

$$\begin{array}{ll} \min & \frac{1}{2} \|M - XX^T\|_F^2 \\ \text{subject to} & X \geq 0 \end{array} \tag{SymNMF}$$

in the variable $X \in \mathbb{R}^{n \times r}$, where the inequality constraint is meant componentwise and $r \leq n$ is the target rank.

(SymNMF) is used as a probabilistic clustering or graph clustering technique [30, 31]. Numerical experiments by [32] have shown that it achieves state-of-the-art clustering accuracy on several text and image datasets.

4.1.1 Solving SymNMF.

While (SymNMF) looks similar to the well-known asymmetric NMF problem $\min_{X,Y} \frac{1}{2}\|M - XY^T\|$, it is actually harder. This is because NMF has a favorable block structure that allows the application of efficient alternating algorithms [33, 34]. SymNMF, however, does not enjoy the same block structure. Current solvers fall into two categories:

Direct solvers. There have been several attempts at solving the original problem, including multiplicative update rules [31], projected gradient algorithm quasi-Newton schemes [32], and coordinate descent [35].

Nonsymmetric relaxations. Another idea is to use a mere penalty method [32,36,37], relaxing (SymNMF) to the following penalized nonsymmetric problem

$$\begin{array}{l} \text{minimize} \quad \frac{1}{2}\|M - XY^T\|_F^2 + \mu\|X - Y\|_F^2 \\ \text{subject to } X, Y \geq 0, \end{array} \tag{P-NMF}$$

in the variables $X, Y \in \mathbb{R}^{n\times r}$, with parameter $\mu \geq 0$. This formulation is very similar to asymmetric NMF and can be solved by the same fast alternating algorithms that exploit the block structure, such as Alternating Nonnegative Least Squares (ANLS) and Hierarchical Alternating Least Squares [37] (HALS), which are arguably the fastest SymNMF solvers.

Applying NoLips We propose to apply NoLips for optimizing the original objective function. Problem (SymNMF) falls within our framework with $F(Y) = \frac{1}{2}\|M - Y\|^2$, which has a Lipschitz gradient with constant 1, and $g(X) = i_{\{X\geq 0\}}$ the indicator function of the nonnegative orthant. Therefore, Proposition 2.1 implies that $f(X) := \frac{1}{2}\|M - XX^T\|^2$ is 1-smooth relatively to the kernel h_N with $\alpha = 6$ and $\sigma = 2\|\nabla F(0)\| = 2\|M\|$. Since, in addition, f is polynomial and g is the indicator of a polynomial set, $f + g$ is semialgebraic, and it is also coercive, so Theorem 3.1 guarantees that NoLips will converge towards a stationary point of problem (SymNMF).

In this problem, the Bregman iteration map is solved by simply adding a projection step

$$T_\lambda(X) = \frac{1}{\tau_\sigma(\alpha\|\Pi_+(U)\|^2)}\Pi_+(U),$$

where $U = \nabla h_N(X) - \lambda\nabla f(X)$, τ_σ has been defined in Proposition 2.2 and Π_+ is the projection on the nonnegative orthant: $\Pi_+(U) = \max(U, 0)$ (entrywise).

Computational complexity for NoLips. The computational complexity of an iteration is dominated by gradient computations and objective function evaluations, as all other operations are linear in the size of the variable.

If M is a $n \times n$ **dense** matrix, each gradient and function evaluation uses $O(n^2r + nr^2)$ floating point operations. If M is represented as a **sparse** matrix with $p << n^2$ nonzero elements, then we can take advantage of this structure [35, Rmk. 2] by using

$$\begin{aligned} f(X) &= \frac{1}{2}\|XX^T - M\|^2 = \frac{1}{2}\|M\|^2 + \frac{1}{2}\|X^TX\|^2 - \langle MX, X\rangle \\ \nabla f(X) &= 2X(X^TX) - 2MX \end{aligned} \tag{25}$$

which yields a much improved $O\left((r^2 + p)n\right)$ complexity per iteration.

4.1.2 Numerical experiments

We implemented the following algorithms: Algorithm 1 with dynamical step size and the norm kernel (`Dyn-NoLips`), the β-`SNMF` scheme from [31], where we set $\beta = 0.99$ as advised by the authors, the projected gradient algorithm (`PG`) with Armijo line search from [32], where we use the line search parameters $\beta = 0.1$ and $\sigma = 0.01$, the coordinate descent scheme (`CD`) from [35], the `ADMM` algorithm [36], and the two fast algorithms from [37] for solving the penalized problem (P-NMF): `SymANLS` and `SymHALS`. For the last two, we tuned the μ penalization parameter for best performance. We left out the quasi-Newton algorithm from [32] because of its prohibitive $O(n^3)$ complexity for large datasets.

Table 1 CPU time (in seconds) needed to reach a decrease of $\epsilon = 10^{-3}$ in projected gradient norm (see (26) for definition). Results have been averaged over 10 random initializations. Hyperparameters for SymHALS, SymANLS and ADMM have been tuned for best performance. Missing values indicate failure of convergence.

Dataset	r	NoLips	PG	Beta	CD	SymHALS	SymANLS	ADMM
Coil-20	10	24.7	51.4	-	26.2	7.0	32.3	-
	20	23.7	36.8	-	21.3	4.0	18.2	-
	30	20.7	40.8	-	35.4	6.5	20.2	-
	40	21.7	49.5	-	57.6	7.5	28.4	-
CBCL	10	38.2	42.7	44.0	35.6	13.6	35.2	42.8
	20	57.7	88.4	-	93.9	17.8	47.8	-
	30	60.9	134.3	-	135.0	15.1	43.4	-
	40	50.8	126.4	-	90.0	23.7	52.5	-
TDT2	10	35.2	54.2	-	97.5	11.0	-	-
	20	52.4	76.1	-	109.9	20.1	-	-
	30	29.4	45.1	-	-	12.1	-	-
	40	28.0	49.8	-	-	17.7	-	-
Reuters	10	6.5	10.0	-	33.0	3.0	54.2	-
	20	28.7	32.8	-	71.7	9.5	74.7	-
	30	24.3	45.5	-	69.4	6.5	91.0	-
	40	40.2	68.5	-	83.2	10.6	108.3	-

All algorithms were implemented in Julia [38] which is a highly-optimized numerical computing language. Since our algorithms have different complexity per iteration, it is essential to compare them in terms of running time, and Julia provides a fairly accurate way to do so as there is little interpreter overhead in loops.[1]

We used two image and two text datasets.

- **Image.**
  - **CBCL**[2]: 2,429 images of faces of size 19×19
  - **Coil-20**[3]: 1440 images of size 128×128 representing 20 objects under various angles.
- **Text.**
  - **TDT2**[4]: dataset of 11,201 news articles classified in 96 semantic categories. We used the version provided by Cai et al. [39–42], which has been restricted to the largest 30 categories, leaving a total of 9,394 documents.
  - **Reuters**[4]: dataset of news articles, which we restricted to the largest 25 categories, leaving a total of 7,963 documents.

For all image and text datasets, we construct a sparse similarity matrix M following the procedure described in [32, Section 7.1]. We begin by computing the similarity graph between data points, using cosine similarity on term frequency vectors for text, and a Gaussian kernel for image (with the self-tuning method for the scale). The graph obtained is *sparsified* by keeping only the edges connecting the k-nearest neighbors, with $k = \lfloor \log_2 n \rfloor + 1$. Then, M is taken as a normalized version of the graph adjacency matrix.

We use the usual convergence criterion for constrained nonconvex problems

$$\frac{\|\nabla^P f(X^k)\|}{\|\nabla^P f(X^0)\|} \leq \epsilon \tag{26}$$

where $\nabla^P f(X)$ is the projected gradient defined as

$$(\nabla^P f(X))_{ij} = \begin{cases} \nabla f(X)_{ij} & \text{if } X_{ij} > 0, \\ \min\left(\nabla f(X)_{ij}, 0\right) & \text{if } X_{ij} = 0. \end{cases}$$

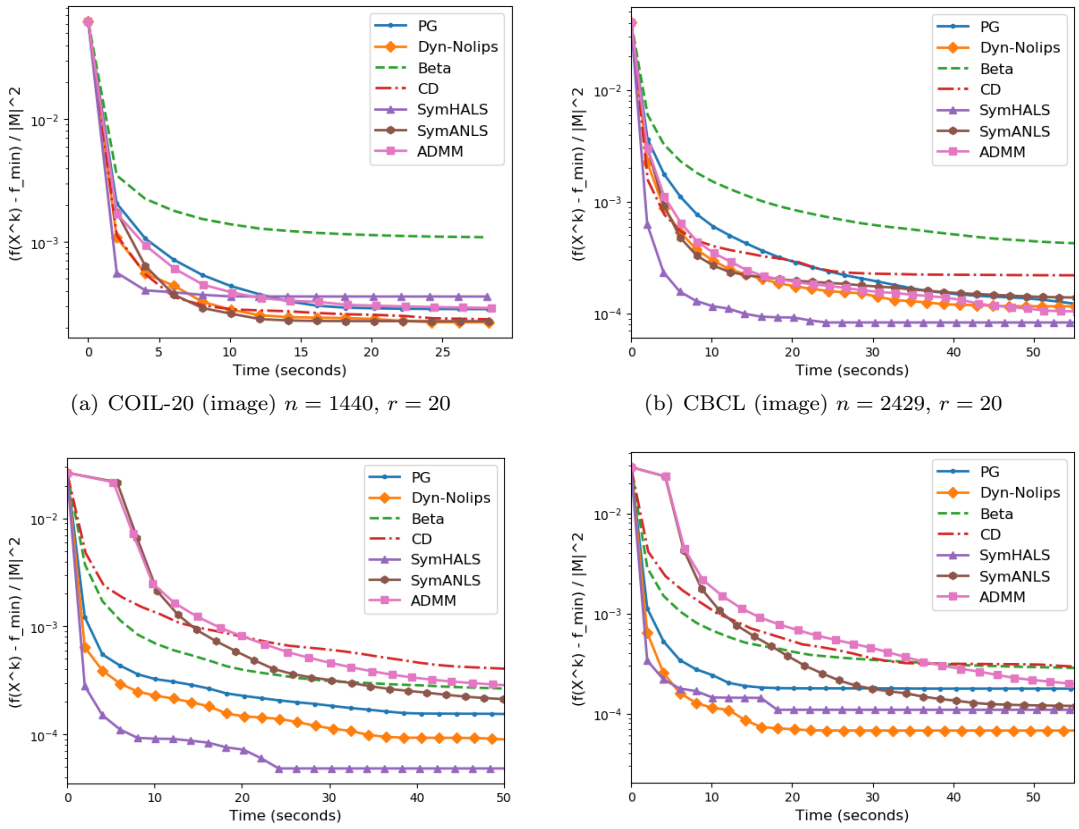
Table 1 reports the average time needed to reach a convergence criterion of $\epsilon = 10^{-3}$, for 10 random initializations. For each dataset, we test several values for the rank parameter r. In addition, Figure 1

[1] Tests were run on a PC Intel CORE i7-4910MQ CPU @ 2.90 GHz x 8 with 32 Go RAM.

[2] http://cbcl.mit.edu/software-datasets/FaceData2.html

[3] http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php

[4] http://www.cad.zju.edu.cn/home/dengcai/Data/TextData.html



(a) COIL-20 (image) $n = 1440$, $r = 20$

(b) CBCL (image) $n = 2429$, $r = 20$

(c) TDT2 (text) $n = 9394$, $r = 30$

(d) Reuters (text) $n = 7963$, $r = 30$

Fig. 1 SymNMF normalized objective gap $\left(f(X^k) - f_{\min}\right)/\|M\|^2$ averaged over 10 random initializations, for various sparse similarity matrices $M \in \mathbb{R}^{n\times n}$. Hyperparameters for SymHALS, SymANLS were tuned for best performance, while Dyn-NoLips is parameter-free.

shows the average evolution of the normalized objective gap $\left(f(X^k) - f_{min}\right)/\|M\|^2$, where f_{min} is the minimal objective value encountered in all initializations.

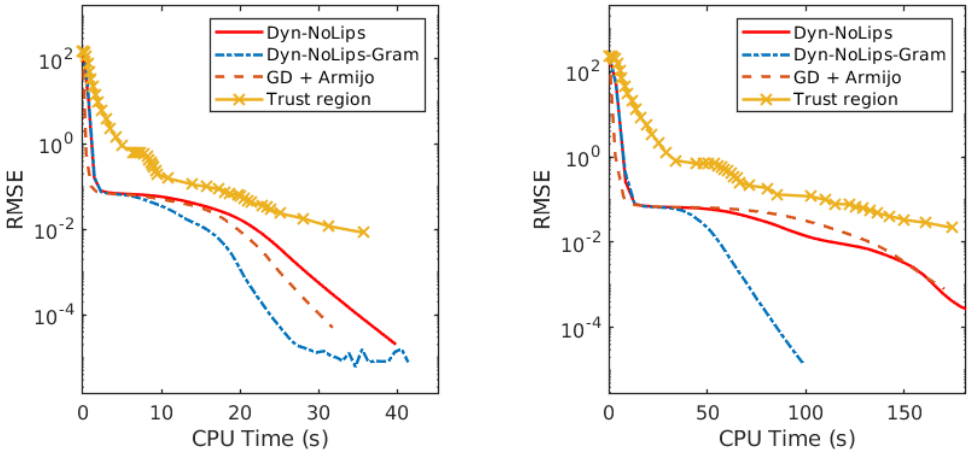
Overall, the algorithm that shows the best convergence speed is `SymHALS`, but it has the disadvantage of needing to tune the penalization parameter μ. In the experiments we report, small values of μ yielded optimal performance, while the convergence theory of [43] only holds for large values for which the algorithm is much slower. By contrast, `Dyn-NoLips` is hyperparameter-free and has the second best overall performance. The gap with the other methods is particularly significant on the larger `TDT2` and `Reuters` datasets, showing that the method scales well with problem dimension.

4.2 Euclidean Distance Matrix Completion

Euclidean distance matrix completion (EDMC) is the task of recovering the position of n points $x_1^*, \ldots, x_n^* \in \mathbb{R}^r$, given the knowledge of a partial set of pairwise distances $d_{ij} = \|x_i^* - x_j^*\|^2$ for $(i,j) \in \Omega$, where $\Omega \subset [1,n] \times [1,n]$. It is a fundamental problem with applications in sensor network localization and the study of conformation of molecules; see [6,44,45] and references therein. The Burer-Monteiro nonconvex formulation for solving this problem writes

$$\min f(X) := \frac{1}{2} \sum_{(i,j)\in\Omega} \left(\|X_i - X_j\|^2 - d_{ij}\right)^2 \qquad \text{(EDMC)}$$

in the variable $X \in \mathbb{R}^{n\times r}$. It can be rewritten $f(X) = \frac{1}{2}\|\mathcal{P}_\Omega(\kappa(XX^T) - D)\|^2$ where D is the matrix of known distances, $\mathcal{P}_\Omega$ denotes the projection operator such that $\mathcal{P}_\Omega(Y)_{ij} = Y_{ij}$ if $(i,j) \in \Omega$, and



(a) $n = 2000, r = 3$ (b) $n = 5000, r = 3$

Fig. 2 Euclidean matrix completion problems on the `Helix` dataset, with 10% known distances and two different problem sizes. We present the normalized RMSE over the full distance matrix versus CPU time. The results are averaged over 10 random initializations.

$\mathcal{P}_\Omega(Y)_{ij} = 0$ elsewhere, and κ is the linear operator defined for $Y \in \mathbb{R}^{n\times n}$ by

$$\kappa(Y)_{ij} = Y_{ii} + Y_{jj} - 2Y_{ij} \text{ for } 1 \leq i, j \leq n \tag{27}$$

Applying NoLips with the norm kernel. Problem (EDMC) falls within our framework with $F(Y) = \frac{1}{2}\|\mathcal{P}_\Omega(\kappa(Y) - D)\|^2$, which can be shown to have a Lipschitz gradient with constant

$$L_{EDM} := 9 \max_{i=1\ldots n} |\{j|(i,j) \in \Omega\}|.$$

Therefore, as in the case of SymNMF, the norm kernel h_N can be used with an initial step size 1 and parameters $\alpha = 6L_{EDM}$ and $\sigma = \frac{1}{3}\|\nabla F(0)\| = 2\|\mathcal{P}_\Omega(D)\|$.

Using the Gram kernel As the problem is unconstrained, we can also apply minimization using the Gram kernel h_G. We use the parameters $\alpha = 2L_{EDM}$, $\beta = L_{EDM}$ and $\sigma = 2\|\mathcal{P}_\Omega(D)\|$, which ensure that f is 1-smooth relatively to h_G by Proposition 2.3.

Computational complexity for NoLips. As before, the main computational bottleneck for an iteration consists in computing the value and gradient of the objective function. If $p = |\Omega|$ denotes the number of known distances, then the computational complexity is $O(pr)$. If the Gram kernel is used, each iteration requires an additional $O(nr^2 + r^3)$ flops (see Section 2.3.2), which is negligible compared to the latter in the usual setting where $p >> n$ and r is small.

Numerical experiments We implement the following algorithms: NoLips with a dynamical step size and the norm kernel (`Dyn-NoLips`), NoLips with a dynamical step size and the Gram kernel (`Dyn-NoLips-Gram`), gradient descent with Armijo line search (`GD`), the Riemannian trust region algorithm from [5] (`TR`). We leave out semidefinite relaxations because of their memory requirement which is prohibitive on large data. As the implementation for `TR` is provided in Matlab, we run our experiments on Matlab as well, with the same setup as in Section 4.1.

We try the algorithms on a standard EDMC problem, the 3-dimensional `Helix` dataset [5] which is generated as $X_i = (\cos(3t_i), \sin(3t_i), 2t_i)$ where $\{t_i\}_{i=1}^n$ are sampled uniformly in $[0, 2\pi]$. We randomly keep only 10 % on the pairwise distances, and test on two different problem sizes: $n = 2000$ and $n = 5000$. Figure 2 reports the normalized root mean squared error (RMSE) over *all distances* (known and unknown) averaged on 10 random initializations. All the algorithms manage to recover the ground truth; the `Dyn-NoLips-Gram` algorithm shows the best numerical performance, which demonstrates the advantage of using the Gram geometry.

5 Conclusion

We proposed a generic approach for solving Burer-Monteiro formulations of low-rank minimization problems using the methodology of Bregman gradient methods and relative smoothness. We studied two quartic kernels, including a new Gram kernel, and demonstrated their benefits on numerical experiments. In future work, performance could be improved further by studying inertial variants [25, 26]. New kernels could also be explored beyond the class of quartic functions to tackle other problems with inherent non-Euclidean geometries.

Code

The code for reproducing experiments for SymNMF and Euclidean Distance Matrix Completion can be downloaded from the public repository
`https://github.com/RaduAlexandruDragomir/QuarticLowRankOptimization`

Acknowledgments

The authors would like to thank the anonymous reviewers for their insightful comments.

Radu-Alexandru Dragomir would like to acknowledge support from an AMX fellowship, the Air Force Office of Scientific Research, Air Force Material Command, USAF, under grant number FA9550-18-1-0226, as well as from Sébastien Gadat.

Jérôme Bolte was partially supported by ANR-3IA Artificial and Natural Intelligence Toulouse Institute, and Air Force Office of Scientific Research, and Air Force Material Command, USAF, under grant numbers FA9550-18-1-0226 & FA9550-19-1-7026.

AA is at CNRS & département d'informatique, École normale supérieure, UMR CNRS 8548, 45 rue d'Ulm 75005 Paris, France, INRIA and PSL Research University. AA acknowledges support from the French government under management of Agence Nationale de la Recherche as part of the "Investissements d'avenir" program, reference ANR-19-P3IA-0001 (PRAIRIE 3IA Institute), the ML & Optimisation joint research initiative with the fonds AXA pour la recherche and Kamet Ventures, as well as a Google focused award.

Appendix

A Solving the Subproblem for Computing the Bregman Iteration Map of the Gram Kernel

While it seems that computing the Bregman iteration map of the Gram kernel involves solving another difficult quartic subproblem, it is actually of small size (r is typically not larger than a few dozens) and can be solved efficiently with the NoLips scheme.

Indeed, the objective function ϕ of problem (18) is 1-smooth relatively to the norm kernel in $\mathbb{R}^r$ $h_N(x) = \frac{\alpha_u}{4}\|x\|^4 + \frac{\sigma_u}{2}\|x\|^2$ with a choice of parameters $\alpha_u = \alpha + 3\beta$ and $\sigma_u = \sigma$.

Algorithm 2 details the procedure. We initialize μ with the values for the previous iteration of the outer procedure. This proves to be efficient as the values will not vary much from one iteration to another. For the stopping criterion, we use the scaled gradient norm $\|\nabla\phi(v)\|/\|\eta\|$ and a tolerance value $\epsilon = 10^{-6}$.

The subproblem being very well conditionned, it is minimized easily; in numerical experiments, it usually convergences in no more than 20 iterations.

Algorithm 2 Computing the Bregman iteration map of the Gram kernel

Input: Matrix $X \in \mathbb{R}^{n\times r}$, gradient of the objective $\nabla f(X)$, step size $\lambda > 0$, parameters $\alpha, \beta, \sigma > 0$, subproblem tolerance ϵ, and (optionally), values μ^- of μ computed at the previous iteration.

Form $V = \nabla h_G(X) - \lambda \nabla f(X) = \left(\alpha \|X\|^2 I_r + \beta X^T X + \sigma I_r\right) X - \lambda \nabla f(X)$
Compute $V^T V$
Form the eigendecomposition of $V^T V = P^T D P$ where $P \in \mathcal{O}_r$ and $D = \mathbf{diag}(\eta_1^2, \ldots, \eta_r^2)$
Initialize μ as μ^- if provided, and as $(0, \ldots, 0)$ otherwise.
repeat
  Compute $\nabla\phi(\mu)$ where $\nabla\phi(\mu)_i = \alpha\|\mu\|^2\mu_i + \beta\mu_i^3 + \sigma\mu_i - \eta_i$
  Compute $\nabla h_N(\mu)$ where $\nabla h_N(\mu)_i = (\alpha + 3\beta)\|\mu\|^2\mu_i + \sigma\mu_i$
  Form $v = \nabla h_N(\mu) - \nabla\phi(u)$
  Set $\mu \leftarrow \left[\tau_\sigma\left((\alpha + 3\beta)\|v\|^2\right)\right]^{-1} v$ where τ_σ has been defined in Proposition 2.2
until stopping criterion has been satisfied, i.e., $\|\nabla\phi(v)\|/\|\eta\| < \epsilon$
Form $Z = P^T \mathbf{diag}(\mu_1^2, \ldots, \mu_r^2) P$
Compute $T_\lambda(X) = V\left[\alpha\,\mathbf{Tr}(Z) I_r + \beta Z + \sigma I_r\right]^{-1}$
Output: Bregman gradient iterate $T_\lambda(X)$

References

1. Emmanuel J. Candès and Benjamin Recht. Exact Matrix Completion Via Convex Optimization. *Found Comput Math*, 9(6), 2009.
2. Jian-Feng Cai, Emmanuel J. Candès, and Zuowei Shen. A Singular Value Tresholding Algorithm for Matrix Completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.
3. Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. Low-rank Matrix Completion using Alternating Minimization. In *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing*, pages 665—-674, 2013.
4. Benjamin Recht, Maryam Fazel, and Pablo A. Parrilo. Guaranteed Minimum-Rank Solutions of Linear Matrix Equations via Nuclear Norm Minimization. *SIAM Review*, 52(3):471–501, 2007.
5. Bandev Mishra, Gilles Meyer, and Rodolphe Sepulchre. Low-rank optimization for distance matrix completion. In *Proceedings of the IEEE Conference on Decision and Control*, pages 4455–4460, 2011.
6. Haw Ren Fang and Dianne P. O'Leary. Euclidean distance matrix completion problems. *Optimization Methods and Software*, 27(4):695–717, 2012.
7. Emmanuel J. Candès, Xiaodong Li, and Mahdi Soltanolkotabi. Phase retrieval via wirtinger flow: Theory and algorithms. *IEEE Transactions on Information Theory*, 61(4):1985–2007, 2015.
8. Yudong Chen and Martin J. Wainwright. Fast low-rank estimation by projected gradient descent: General statistical and algorithmic guarantees. *arXiv preprint arXiv:1509.03025*, 2015.
9. Samuel Burer and Renato D C Monteiro. Local Minima and Convergence in Low-Rank Semidefinite Programming. *Mathematical Programming*, 103(3):427–444, 2005.
10. Stephen Tu, Ross Boczar, Max Simchowitz, Mahdi Soltanolkotabi, and Benjamin Recht. Low-rank Solutions of Linear Matrix Equations via Procrustes Flow. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning*, pages 964–973, 2016.
11. Srinadh Bhojanapalli, Anastasios Kyrillidis, and Sujay Sanghavi. Dropping Convexity for Faster Semi-definite Optimization. *JMLR: Workshop and Conference Proceedings*, 40:1–53, 2016.
12. Tuo Zhao, Zhaoran Wang, and Han Liu. A Nonconvex Optimization Framework for Low Rank Matrix Estimation. In *Advances in Neural Information Processing Systems 28*, pages 559–567, 2015.
13. Ruoyu Sun and Zhi-Quan Luo. Guaranteed Matrix Completion via Nonconvex Factorization. *IEEE Transactions on Information Theory*, 62(11):6535–6579, 2016.
14. Qinqing Zheng and John Lafferty. Convergence Analysis for Rectangular Matrix Completion Using Burer-Monteiro Factorization and Gradient Descent. *arXiv preprint arXiv:1605.07051*, 2016.
15. Dohyung Park, Anastasios Kyrillidis, Constantine Caramanis, and Sujay Sanghavi. Finding low-rank solutions to matrix problems , efficiently and provably. *arXiv preprint arXiv:1606.03168v1*, 2016.
16. Qinqing Zheng and John Lafferty. A Convergent Gradient Descent Algorithm for Rank Minimization and Semidefinite Programming from Random Linear Measurements. In *Advances in Neural Information Processing Systems 28*, 2015.
17. Rong Ge, Jason D. Lee, and Tengyu Ma. Matrix Completion has No Spurious Local Minimum. *Advances in Neural-Information Processing Systems*, pages 2973–2981, 2016.
18. Heinz H. Bauschke, Jérôme Bolte, and Marc Teboulle. A Descent Lemma Beyond Lipschitz Gradient Continuity: First-Order Methods Revisited and Applications. *Mathematics of Operations Research*, 42(2):330–348, 2017.
19. Jérôme Bolte, Shoham Sabach, Marc Teboulle, and Yakov Vaisbourd. First order methods beyond convexity and lipschitz gradient continuity with applications to quadratic inverse problems. *SIAM Journal on Optimization*, 28(3):2131–2151, 2018.
20. Chih-Jen Lin. Projected Gradient Methods for Nonnegative Matrix Factorization. *Neural Computation*, 2007.
21. Quang Van Nguyen. Forward-backward splitting with bregman distances. *Vietnam Journal of Mathematics*, 45(3):519–539, 2017.
22. Haihao Lu, Robert M. Freund, and Yurii Nesterov. Relatively-Smooth Convex Optimization by First-Order Methods, and Applications. *SIAM Journal on Optimization*, 28(1):333–354, 2018.
23. Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*. Springer US, 2003.
24. Alfred Auslender and Marc Teboulle. Interior gradient and proximal methods for convex and conic optimization. *SIAM Journal on Optimization*, 16(3):697–725, 2006.

25. Filip Hanzely, Peter Richt, and Lin Xiao. Accelerated Bregman proximal gradient methods for relatively smooth convex optimization. *ArXiv preprint arXiv:1808.03045v1*, 2018.
26. Mahesh Chandra Mukkamala, Peter Ochs, Thomas Pock, and Shoham Sabach. Convex-Concave Backtracking for Inertial Bregman Proximal Gradient Algorithms in Non-Convex Optimization. *arXiv preprint arXiv:1904.03537*, 2019.
27. Raghu Meka, Prateek Jain, and Inderjit S. Dhillon. Guaranteed Rank Minimization via Singular Value Projection. *NIPS*, 2010.
28. Yurii Nesterov. Gradient methods for minimizing composite objective function. *CORE Report*, 2007.
29. Jérôme Bolte, Aris Daniilidis, and Adrian Lewis. The Łojasiewicz Inequality for Nonsmooth Subanalytic Functions with Applications to Subgradient Dynamical Systems. *SIAM Journal on Optimization*, 17(4):1205–1223, 2007.
30. Chris Ding, Xiaofeng He, and Horst Simon. On the Equivalence of Nonnegative Matrix Factorization and Spectral Clustering. In *Proceedings of the 2005 SIAM ICDM*, number 4, pages 126–135, 2005.
31. Zhaoshui He, Shengli Xie, Rafal Zdunek, Guoxu Zhou, and Andrzej Cichocki. Symmetric nonnegative matrix factorization: Algorithms and applications to probabilistic clustering. *IEEE Transactions on Neural Networks*, 22(12):2117–2131, 2011.
32. Da Kuang, Sangwoon Yun, and Haesun Park. SymNMF: nonnegative low-rank approximation of a similarity matrix for graph clustering. *Journal of Global Optimization*, 62(3):545–574, 2015.
33. Jingu Kim and Haesun Park. Fast Nonnegative Matrix Factorization: An Active-set-like Method and Comparisons. *SIAM Journal on Scientific Computing*, 33(6):3261–3281, 2013.
34. Andrzej Cichocki and Anh Huy Phan. Fast local algorithms for large scale nonnegative matrix and tensor factorizations. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 2009.
35. Arnaud Vandaele, Nicolas Gillis, Qi Lei, Kai Zhong, and Inderjit Dhillon. Efficient and non-convex coordinate descent for symmetric nonnegative matrix factorization. *IEEE Transactions on Signal Processing*, 64(21):5571–5584, 2016.
36. Songtao Lu, Mingyi Hong, and Zhengdao Wang. A Nonconvex Splitting Method for Symmetric Nonnegative Matrix Factorization : Convergence Analysis and Optimality. *IEEE Transactions on Signal Processing*, 65(12):2572–2576, 2017.
37. Zhihui Zhu, Xiao Li, Kai Liu, and Qiuwei Li. Dropping Symmetry for Fast Symmetric Nonnegative Matrix Factorization. In *Advances in Neural Information Processing Systems 31*, 2018.
38. Stefan Karpinski Jeff Bezanson, Alan Edelman and Viral B. Shah. Julia : A Fresh Approach to Numerical Computing. *SIAM Review*, 59(1):65–98, 2017.
39. Deng Cai, Xuanhui Wang, and Xiaofei He. Probabilistic dyadic data analysis with local and global consistency. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML'09)*, pages 105–112, 2009.
40. Deng Cai, Qiaozhu Mei, Jiawei Han, and Chengxiang Zhai. Modeling hidden topics on document manifold. In *Proceeding of the 17th ACM conference on Information and knowledge management (CIKM'08)*, pages 911–920, 2008.
41. Deng Cai, Xiaofei He, Wei Vivian Zhang, and Jiawei Han. Regularized locality preserving indexing via spectral regression. In *Proceedings of the 16th ACM conference on Conference on information and knowledge management (CIKM'07)*, pages 741–750, 2007.
42. Deng Cai, Xiaofei He, and Jiawei Han. Document clustering using locality preserving indexing. *IEEE Transactions on Knowledge and Data Engineering*, 17(12):1624–1637, December 2005.
43. Zhihui Zhu, Xiao Li, Kai Liu, and Qiuwei Li. Dropping Symmetry for Fast Symmetric Nonnegative Matrix Factorization. *NIPS*, 2018.
44. Hou Duo Qi and Xiaoming Yuan. Computing the nearest Euclidean distance matrix with low embedding dimensions. *Mathematical Programming*, 147(1-2):351–389, 2013.
45. Ivan Dokmanic, Reza Parhizkar, Juri Ranieri, and Martin Vetterli. Euclidean Distance Matrices: Essential theory, algorithms, and applications. *IEEE Signal Processing Magazine*, 32(6):12–30, 2015.