Manifold Proximal Point Algorithms for Dual Principal Component Pursuit and Orthogonal Dictionary Learning

Shixiang Chen
ISE
Texas A&M University
College Station, TX, USA
sxchen@tamu.edu

Zengde Deng
SEEM
The Chinese University of Hong Kong
Hong Kong, China
zddeng@se.cuhk.edu.hk

Shiqian Ma
Mathematics
UC Davis
Davis, CA, USA
sqma@ucdavis.edu

Anthony Man-Cho So
SEEM
The Chinese University of Hong Kong
Hong Kong, China
manchoso@se.cuhk.edu.hk

***Abstract*—Dual principal component pursuit and orthogonal dictionary learning are two fundamental tools in data analysis, and both of them can be formulated as a manifold optimization problem with nonsmooth objective. Algorithms with convergence guarantees for solving this kind of problems have been very limited in the literature. In this paper, we propose a novel manifold proximal point algorithm for solving this nonsmooth manifold optimization problem. Numerical results are reported to demonstrate the effectiveness of the proposed algorithm.**

***Index Terms*—Manifold Optimization, Proximal Point Algorithm, Subspace Clustering, Dictionary Learning**

I. Introduction

In this paper, we focus on the nonsmooth and nonconvex optimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \ f(\mathbf{x}) := \|Y^\top \mathbf{x}\|_1, \ \text{s.t.}, \ \|\mathbf{x}\|_2 = 1, \tag{1}$$

where $Y \in \mathbb{R}^{n\times p}$ is a given matrix and $\|\cdot\|_1$ denotes the ℓ_1 norm of a vector. The set $\{\mathbf{x}\in\mathbb{R}^n \mid \|\mathbf{x}\|_2 = 1\}$ is known as a sphere constraint and is a special case of the Stiefel manifold. Note that (1) has a nonconvex constraint and a nonsmooth objective, which make it very challenging to solve. Though manifold optimization has been a very active area [1], most existing algorithms require computing the derivative of the objective function and are therefore not applicable to (1). The problem (1), which should be contrasted with the ℓ_1-PCA problem considered in [2], has recently received much attention because of its many interesting applications, among which are two representative ones: orthogonal dictionary learning (ODL) and dual principal component pursuit (DPCP). In the following we briefly survey the literature on these two problems.

In ODL, one is given a set of p ($p \gg n$) data points $\mathbf{y}_1, \ldots, \mathbf{y}_p \in \mathbb{R}^n$ and aims to find an orthonormal basis of $\mathbb{R}^n$ to represent them compactly. In other words, by letting $Y = [\mathbf{y}_1, \ldots, \mathbf{y}_p] \in \mathbb{R}^{n\times p}$, we want to find an orthogonal matrix $X \in \mathbb{R}^{n\times n}$ and a sparse matrix $A \in \mathbb{R}^{n\times p}$, such that $Y = XA$. Since X is orthogonal, we know that $A = X^\top Y$. Recently, Spielman, Wang, and Wright [3] have suggested to find the rows of A by looking for sparse vectors in the row space of Y, which leads to the following optimization problem:

$$\min_{\mathbf{x}} \ \|Y^\top \mathbf{x}\|_0, \ \text{s.t.}, \ \mathbf{x} \neq 0, \tag{2}$$

where $\|\mathbf{z}\|_0$ counts the number of nonzeros of vector $\mathbf{z}$. In fact, if $Y = X_0 A_0$ and A_0 follows the Bernoulli-Gaussian (BG) model with rate γ (i.e., $[A_0]_{ij} = \Omega_{ij} V_{ij}$, with $\Omega_{ij} = \text{Ber}(\gamma)$ and $V_{ij} \sim \mathcal{N}(0,1)$), then the rows of A_0 are the n sparsest vectors in row(Y) with high probability whenever $p \geq \Omega(n\log n)$ (see [3]). Since (2) is numerically intractable, the following relaxation of (2) is proposed in [3]:

$$\min_{\mathbf{x}\in\mathbb{R}^n} \ \|Y^\top \mathbf{x}\|_1, \ \text{s.t.}, \ \mathbf{b}^\top \mathbf{x} = 1, \tag{3}$$

where $\mathbf{b}$ denotes one column of Y. Note that (3) is a linear program (LP) and can be solved by off-the-shelf LP solvers. The authors of [3] suggest to solve (3) for p times, where in the j-th time $\mathbf{b}$ is set to be the j-th column of Y. It is proved in [3] that the p sparse vectors obtained from this process cover the n row vectors of A with high probability, given that $p \geq \Omega(n\log n)$. However, the result breaks down when each column of A_0 has more than $O(\sqrt{n})$ nonzeros. Under a planted sparse model, a similar sequential LP approach was developed by Demanet and Hand [4] to find sparse vectors with sparsity up to $O(p/\sqrt{n})$. Qu, Sun, and Wright [5] have recently improved this threshold to $O(p)$ by considering (1) with the ℓ_1 norm replaced by the (smooth) Huber function. In [6], [7], Sun, Qu, and Wright proved the same result – $O(p)$ sparsity – for the BG model by considering (1) with ℓ_1 norm replaced by another smooth function. One possible drawback of the approaches in [5]–[7] is that the smoothed function might not render the desired sparsity structure.

Another representative application of (1) is known as dual principal component pursuit (DPCP) for robust subspace recovery (RSR). RSR aims to fit a linear subspace to a dataset corrupted by outliers, which is a fundamental problem in

The work of S. Ma was supported in part by a startup package in the Department of Mathematics at UC Davis. The work of A. M.-C. So was supported in part by the Hong Kong Research Grants Council (RGC) General Research Fund (GRF) project CUHK 14203218, and in part by the CUHK Research Sustainability of Major RGC Funding Schemes project 3133236.

machine learning and data mining. RSR can be described as follows. Given a dataset $Y = [\mathcal{X}, \mathcal{O}]\Gamma \in \mathbb{R}^{n\times(p_1+p_2)}$, where $\mathcal{X} \in \mathbb{R}^{n\times p_1}$ are inlier points spanning a d-dimensional subspace $\mathcal{S}$ of $\mathbb{R}^n$ ($d < p_1$), $\mathcal{O} \in \mathbb{R}^{n\times p_2}$ are outlier points without linear structure, and $\Gamma \in \mathbb{R}^{(p_1+p_2)\times(p_1+p_2)}$ is an unknown permutation, the goal is to recover the inlier space $\mathcal{S}$, or equivalently, to cluster the points into inliers and outliers. For a more comprehensive review of RSR, see the recent survey paper by Lerman and Maunu [8]. DPCP is a recently proposed approach to RSR that seeks to learn recursively a basis for the orthogonal complement $\mathcal{S}$ by solving (1). The idea of DPCP is to first compute a normal vector $\mathbf{x}$ to a hyperplane $\mathcal{H}$ that contains all inliers $\mathcal{X}$. As outliers are not orthogonal to $\mathbf{x}$ and the number of outliers is known to be small, the normal vector $\mathbf{x}$ can be found by solving (1). If d is known, then one can recover $\mathcal{S}$ as the intersection of the $n-d$ orthogonal hyperplanes that contain $\mathcal{X}$. It is shown in [9], [10] that under certain conditions, solving (1) indeed yields a vector that is orthogonal to $\mathcal{S}$, given that the number of outliers p_2 is at most on the order of $O(p_1^2)$. Sequentially solving (1) thus gives all $n-d$ orthogonal vectors that are orthogonal to $\mathcal{S}$.

Our contributions. In this paper, we propose a novel manifold proximal point algorithm (ManPPA) for solving (1). We conduct extensive numerical comparisons of ManPPA with existing methods for solving (1). The results demonstrate that ManPPA is a robust and efficient algorithm comparing to the state-of-the-art.

Organization. The rest of the paper is organized as follows. In Section II we survey the existing methods for solving (1). In Section III we propose our ManPPA algorithm and show in detail how to apply the inexact augmented Lagrangian method and the semi-smooth Newton method to solve the subproblems. In Section IV we apply ManPPA to solve the DPCP and ODL problems and compare its performance with some existing algorithms.

II. Existing Algorithms for ODL and DPCP

In view of the nonsmooth objective and sphere constraint in (1), a natural idea for tackling it is to replace the ℓ_1 norm by a smooth function, so that existing manifold optimization algorithms designed for smooth objective functions can be applied. For example, Sun et al. [6], [7] and Gilboa et al. [11] suggested to replace the absolute value function $|t|$ by $h_\mu(t) = \mu\log(\cosh(t/\mu))$, where $\mu > 0$ is a smooth parameter, and then solve the smoothed problem by either Riemannian trust-region method [6], [7] or Riemannian gradient descent method [11]. In [5], Qu et al. considered the following relaxation of (1):

$$\min_{\mathbf{x},\mathbf{z}} \ \lambda\|\mathbf{z}\|_1 + \frac{1}{2}\|Y^\top\mathbf{x} - \mathbf{z}\|_2^2, \ \text{s.t.}, \ \|\mathbf{x}\|_2 = 1, \tag{4}$$

where $\lambda > 0$ is a weighting parameter. An alternating minimization algorithm is then employed to solve (4), which updates the iterates in the following way:

$$\begin{aligned} \mathbf{x}^{k+1} &:= \operatorname{argmin}_{\mathbf{x}} \ \tfrac{1}{2}\|Y^\top\mathbf{x} - \mathbf{z}^k\|_2^2, \ \text{s.t.}, \ \|\mathbf{x}\|_2 = 1, \\ \mathbf{z}^{k+1} &:= \operatorname{argmin}_{\mathbf{z}} \ \lambda\|\mathbf{z}\|_1 + \tfrac{1}{2}\|Y^\top\mathbf{x}^{k+1} - \mathbf{z}\|_2^2. \end{aligned} \tag{5}$$

Note that both subproblems in (5) admit easily computable closed-form solutions. However, solving (4) does not yield a sparse $Y^\top\mathbf{x}$, because there is always some error between $Y^\top\mathbf{x}$ and $\mathbf{z}$. To remedy this, Qu et al. [5] proposed an LP rounding procedure to enhance the sparsity of $Y^\top\mathbf{x}$.

Recently, projected subgradient method (PSGM) and Riemannain subgradient method have been proposed to solve (1) directly without smoothing the ℓ_1 norm. In [12], Bai et al. proposed to solve (1) by a Riemannian subgradient method, which updates $\mathbf{x}$ via

$$\mathbf{x}^{k+1} := \frac{\mathbf{x}^k - \eta_k\mathbf{v}^k}{\|\mathbf{x}^k - \eta_k\mathbf{v}^k\|_2}, \quad \text{for } \mathbf{v}^k \in \partial_R f(\mathbf{x}^k), \tag{6}$$

where $\partial_R f(\cdot)$ denotes the set of Riemannian subgradients of f. Bai et al. [12] suggested to solve (1) for $O(n\log n)$ times via (6), each time with an independent random initialization. Bai et al. [12] proved the following result: for the BG model, if the sample size $p \geq Cn^4\theta^{-2}\log^3 n$, by setting $\eta_k = O(k^{-3/8}/\sqrt{n})$ and under certain randomness hypothesis, among the $O(n\log n)$ obtained vectors, n of them will recover the columns of the dictionary X up to permutation and sign change with high probability. The iterates of the PSGM for solving (1) are generated via

$$\mathbf{x}_{k+1} := \frac{\mathbf{x}^k - \eta_k\mathbf{v}^k}{\|\mathbf{x}^k - \eta_k\mathbf{v}^k\|_2}, \quad \text{for } \mathbf{v}^k \in \partial f(\mathbf{x}^k). \tag{7}$$

The difference between (6) and (7) is that $\mathbf{v}^k$ is a Riemannian subgradient in (6) and an Euclidean subgradient in (7). It is easy to verify the following connection between them [13]:

$$\partial_R f(\mathbf{x}) = (I - \mathbf{x}\mathbf{x}^\top)\partial f(\mathbf{x}).$$

In [10], Zhu et al. proposed a PSGM with piecewise geometrically diminishing step size (we denote it by PSGM-Geo in this paper), and they proved that PSGM-Geo converges linearly under certain randomness hypothesis. The piecewise geometrically diminishing step size can be chosen as $\beta_k = \beta^{\lfloor(k-K_0)/k\rfloor+1}$, where $\beta \in (0,1)$ and K_0 is an integer. In practice, the parameters β and K_0 are difficult to determine. Therefore, Zhu et al. also incorporated a modified backtracking line search technique to PSGM (PSGM-MBLS). Though it works well in practice, there is no convergence guarantee for PSGM-MBLS. Hosseini and Uschmajew [13] studied a gradient sampling (GS) algorithm for nonsmooth optimization on manifold, which can be used to solve (1). GS needs to solve a quadratic program over simplex each time to determine an appropriate descent direction. However, the collection of Riemannian gradient samples needs an operation called vector transport, which could be expensive.

III. ManPPA: A Manifold Proximal Point Algorithm

In this section, we propose our ManPPA (Manifold Proximal Point Algorithm) for solving the vector problem (1). Our ManPPA differs from existing proximal point algorithms for Riemannian optimization in the literature. PPA is a classical method for continuous optimization in Euclidean setting [14]–[18]. Ferreira and Oliveira [19] extended PPA to manifold optimization, which in each iteration needs to minimize the original function plus a proximal term over the manifold. However, there are two issues that limit its applicability. First, the subproblem can be as difficult as the original problem. For example, Bačák et al. [20] suggested to use the subgradient method to solve the subproblem, but they require the subproblem to be in the form of the pointwise maximum of smooth functions tackled in [21]. Second, the discussions in the literature mainly focus on the Hadamard manifold and exploit heavily the convexity assumption of the objective function. Thus, they do not apply to compact manifolds such as the sphere and orthogonal group. Bento et al. [22] aimed to resolve the second issue and proved the convergence of the PPA for more general Riemannian manifolds under the assumption that the KŁ inequality holds for the objective function. In [23], Bento et al. analyzed the convergence of some inexact descent methods based on the KŁ inequality, including the PPA and steepest descent method. In a more recent work [24], Bento et al. studied the iteration complexity of PPA under the assumption that the constraint set is the Hadamard manifold and the objective function is convex. Nevertheless, the results in [19], [22]–[24] seem to be of theoretical interest only because no numerical results were presented. As mentioned earlier, this could be due to the difficulty in solving the PPA subproblems.

Our ManPPA resolves the above-mentioned issues because in each iteration it solves a convex subproblem. This convex subproblem has a special structure that allows efficient solvers to be applied. In this section we introduce ManPPA for solving (1) in detail. ManPPA is motived by our recent work ManPG [25]. A typical iteration of our ManPPA for solving (1) is:

$$\mathbf{d}_k := \operatorname*{argmin}_{\mathbf{d}} \ \|Y^\top(\mathbf{x}_k+\mathbf{d})\|_1 + \frac{1}{2t}\|\mathbf{d}\|_2^2, \ \text{s.t.}, \ \mathbf{d}^\top\mathbf{x}_k = 0, \tag{8}$$

$$\mathbf{x}_{k+1} := (\mathbf{x}_k + \alpha_k\mathbf{d}_k)/\|\mathbf{x}_k+\alpha_k\mathbf{d}_k\|_2, \tag{9}$$

where $t>0$ and $\alpha_k>0$ are step sizes. The constraint in (8) means that $\mathbf{d}$ lies in the tangent space of the sphere $\|\mathbf{x}\|_2=1$. As a result, (8) can be viewed as a PPA restricted on the tangent space of the sphere, because it reduces to PPA on the Euclidean setting if the tangent space constraint is ignored. The update (9) is a retraction step that brings the iterate back to the sphere. The details of ManPPA is summarized in Algorithm 1.

The efficiency of ManPPA depends on whether one can solve the subproblem (8) efficiently. Fortunately, due to its special structure, we can apply an inexact augmented Lagrangian method (iALM) [26] combined with a semi-smooth Newton method (SSN) to solve it.

Algorithm 1 ManPPA: Manifold Proximal Point Algorithm

1: Input: Initial point $\mathbf{x}_0$, $\|\mathbf{x}_0\|=1$, $\beta\in(0,1)$.
2: **while** not converge **do**
3: Solve subproblem (8) with $t>0$ to obtain $\mathbf{d}_k$.
4: Let j be the smallest integer such that $f(\mathbf{x}_k+\beta^j\mathbf{d}_k)\le f(\mathbf{x}_k)-\frac{\beta^j}{2t}\|\mathbf{d}_k\|_2^2$.
5: Set $\mathbf{x}_{k+1}=(\mathbf{x}_k+\beta^j\mathbf{d}_k)/\|\mathbf{x}_k+\beta^j\mathbf{d}_k\|_2$.
6: Set $k:=k+1$.
7: **end while**

A. An Inexact Augmented Lagrangian Method for Solving (8)

The iALM and SSN for solving (8) are motivated by [26]. For ease of presentation, in this subsection, we drop the subindex of $\mathbf{x}_k$ and let $\mathbf{x}:=\mathbf{x}_k$, $\mathbf{c}:=Y^\top\mathbf{x}_k$. The subproblem (8) can be equivalently written as

$$\begin{aligned}\min_{\mathbf{d},\mathbf{u}} \ & \frac{1}{2}\|\mathbf{d}\|_2^2 + t\|\mathbf{u}\|_1 \\ \text{s.t.} \ & Y^\top\mathbf{d}+\mathbf{c}=\mathbf{u}, \\ & \mathbf{d}^\top\mathbf{x}=0.\end{aligned} \tag{10}$$

The augmented Lagrangian function of (10) is

$$\begin{aligned}\mathcal{L}_\sigma(\mathbf{d},\mathbf{u};y,\mathbf{z}) :=& \frac{1}{2}\|\mathbf{d}\|_2^2 + t\|\mathbf{u}\|_1 + y\cdot\mathbf{d}^\top\mathbf{x} + \langle\mathbf{z}, Y^\top\mathbf{d}+\mathbf{c}-\mathbf{u}\rangle \\ & + \frac{\sigma}{2}(\mathbf{d}^\top\mathbf{x})^2 + \frac{\sigma}{2}\|Y^\top\mathbf{d}+\mathbf{c}-\mathbf{u}\|^2,\end{aligned}$$

where $y\in\mathbb{R}$ and $\mathbf{z}\in\mathbb{R}^m$ are Lagrange multipliers (dual variables) and $\sigma>0$ is a penalty parameter. The iALM for solving (10) is detailed in Algorithm 2, where in the j-th iteration we define, for fixed σ_j, y^j, and $\mathbf{z}^j$,

$$\Psi_j(\mathbf{d},\mathbf{u}) := \mathcal{L}_{\sigma_j}(\mathbf{d},\mathbf{u};y^j,\mathbf{z}^j). \tag{11}$$

In each step of iALM, we need to solve the minimization problem (12) inexactly, followed by an update to the dual variables.

Algorithm 2 An Inexact Augmented Lagrangian Method (iALM) for subproblem (10)

1: Input: $(\mathbf{d}^0,\mathbf{u}^0,y^0,\mathbf{z}^0)$.
2: **for** $j=0,1,\dots$ **do**
3: Compute

$$(\mathbf{d}^{j+1},\mathbf{u}^{j+1}) \approx \operatorname*{argmin}_{\mathbf{d},\mathbf{u}} \ \Psi_j(\mathbf{d},\mathbf{u}). \tag{12}$$

4: Update dual variables:

$$\begin{aligned}y^{j+1} &= y^j + \sigma_j\mathbf{d}^\top\mathbf{x}, \\ \mathbf{z}^{j+1} &= \mathbf{z}^j + \sigma_j(Y^\top\mathbf{d}^{j+1}+\mathbf{c}-\mathbf{u}^{j+1}).\end{aligned}$$

5: Update $\sigma_{j+1}\uparrow\sigma_\infty\le\infty$.
6: **end for**

B. Solving the Augmented Lagrangian Subproblem (12)

In this subsection we discuss how to apply the semi-smooth Newton algorithm to solve the augmented Lagrangian subproblem (12). For ease of presentation, we drop the index j here. First, we rewrite (12) as

$$\begin{aligned}
&\min_{\mathbf{d},\mathbf{u}} \Psi(\mathbf{d},\mathbf{u}) := \ \mathcal{L}_\sigma(\mathbf{d},\mathbf{u};y,\mathbf{z}) \\
&= \frac{1}{2}\|\mathbf{d}\|_2^2 + t\|\mathbf{u}\|_1 - \sigma(Y^\top \mathbf{d} + \mathbf{c} + \mathbf{z}/\sigma)^\top \mathbf{u} \\
&\quad + \frac{\sigma}{2}\left\|Y^\top \mathbf{d} + \mathbf{c} + \frac{\mathbf{z}}{\sigma}\right\|_2^2 + \frac{\sigma}{2}\|\mathbf{u}\|_2^2 \\
&\quad + \frac{\sigma}{2}\left(\mathbf{d}^\top \mathbf{x} + \frac{y}{\sigma}\right)^2 - \frac{y^2}{2\sigma} - \frac{\|\mathbf{z}\|_2^2}{2\sigma}.
\end{aligned}$$

Let $h(\mathbf{u}) = t\|\mathbf{u}\|_1$. Then, its conjugate function is given by $h^*(\mathbf{u}) = \mathbb{I}_{\{\|\mathbf{u}\|_\infty \le t\}}$. Using the Moreau decomposition identity $\mathbf{d} = \text{prox}_{h/\sigma}(\mathbf{d}) + (1/\sigma)\text{prox}_{\sigma h^*}(\sigma \mathbf{d}), \forall \mathbf{d}$, we have

$$\begin{aligned}
&\psi(\mathbf{d}) := \ \inf_{\mathbf{u}} \Psi(\mathbf{d},\mathbf{u}) \\
&= \frac{1}{2}\|\mathbf{d}\|_2^2 + h(\text{Prox}_{h/\sigma}(Y^\top \mathbf{d} + \mathbf{c} + \frac{\mathbf{z}}{\sigma})) \\
&\quad + \frac{1}{2\sigma}\|\text{Prox}_{\sigma h^*}(\sigma Y^\top \mathbf{d} + \sigma \mathbf{c} + \mathbf{z})\|^2 \\
&\quad + \frac{\sigma}{2}\left(\mathbf{d}^\top \mathbf{x} + \frac{y}{\sigma}\right)^2 - \frac{y^2}{2\sigma} - \frac{\|\mathbf{z}\|^2}{2\sigma}.
\end{aligned}$$

Therefore, if $(\bar{\mathbf{d}}, \bar{\mathbf{u}}) = \text{argmin}\, \Psi(\mathbf{d},\mathbf{u})$, we can compute them simultaneously as follows:

$$\begin{cases}
\bar{\mathbf{d}} = \text{argmin}\, \psi(\mathbf{d}), \\
\bar{\mathbf{u}} = \text{Prox}_{h/\sigma}(Y^\top \bar{\mathbf{d}} + \mathbf{c} + \frac{\mathbf{z}}{\sigma}).
\end{cases}$$

Notice that $\Psi(\mathbf{d},\mathbf{u})$ is strongly convex with respect to $\mathbf{u}$ and thus $\psi(\mathbf{d})$ is continuously differentiable with gradient

$$\nabla\psi(\mathbf{d}) = \mathbf{d} + Y\text{Prox}_{\sigma h^*}(\sigma Y^\top \mathbf{d} + \sigma \mathbf{c} + \mathbf{z}) + \sigma\left(\mathbf{d}^\top \mathbf{x} + \frac{y}{\sigma}\right)\mathbf{x}.$$

We also know $\psi(\mathbf{d})$ is strongly convex, so we can obtain $\bar{\mathbf{d}}$ by solving the following nonsmooth equation:

$$\nabla\psi(\mathbf{d}) = 0. \tag{13}$$

Now, solving the augmented Lagrangian subproblem (12) reduces to solving the equation (13). We found that this equation can be solved efficiently by the semi-smooth Newton method. Details of the semi-smooth Newton method for solving (13) is described in Algorithm 3.

IV. Numerical experiments

In this section, we compare our proposed algorithm ManPPA with existing method PSGM-MBLS. We do not include the results of the Riemannian subgradient method with diminishing step sizes (such as $1/k$ or $1/\sqrt{k}$), because the numerical results in [27] showed that they are very slow compared to PSGM-MBLS and cannot achieve high accuracy. For ManPPA, we set the step size $t = 0.1$ and $\beta = 0.5$ in all the tests, and the maximum number of iterations of ManPPA (Algorithm 1), iALM (Algorithm 2), and SSN (Algorithm 3) are set to 100, 30 and 20, respectively. In all tests conducted in this section, we stopped the ManPPA if the relative change of objective values satisfies

$$|f(\mathbf{x}_k) - f(\mathbf{x}_{k-1})|/f(\mathbf{x}_{k-1}) \le 10^{-9}.$$

For Algorithm 3, the linear equation (14) is solved by Cholesky decomposition, as the matrix V_j is positive definite. The parameters are set to $\mu = 0.1$, $\bar{\eta} = 0, \tau = 1$, and $\delta = 0.5$.

Algorithm 3 Semi-smooth Newton method for solving equation (13)

1: Input: $\mu \in (0, 1/2), \bar{\eta} \in [0, 1), \tau \in (0, 1]$ and $\delta \in (0, 1)$.
2: **for** $j = 0, 1, \ldots$ **do**
3: Choose $U^j \in \partial\text{Prox}_{\sigma p^*}(\sigma Y^\top d + \sigma \mathbf{c} + \mathbf{z})$. Let $V_j = I + \sigma Y U^j Y^\top + \sigma \mathbf{x}\mathbf{x}^\top$. Solve the linear system

$$V_j \mathbf{w}^j = -\nabla\psi(\mathbf{d}^j) \tag{14}$$

exactly for $\mathbf{w}^j$ or by the CG algorithm to find an approximate solution $\mathbf{w}^j$ such that

$$\|V_j \mathbf{w}^j + \nabla\psi(\mathbf{d}^j)\| \le \min(\bar{\eta}, \|\nabla\psi(\mathbf{d}^j)\|^{1+\tau}).$$

4: (Line search) Set $\alpha_j = \delta^{m_j}$, where m_j is the first nonnegative integer m for which

$$\psi(\mathbf{d}^j + \delta^m \mathbf{w}^j) \le \psi(\mathbf{d}^j) + \mu\delta^m \langle \nabla\psi(\mathbf{d}^j), \mathbf{w}^j \rangle.$$

5: Set $\mathbf{d}^{j+1} = \mathbf{d}^j + \alpha_j \mathbf{w}^j$.
6: **end for**

A. Numerical Results on DPCP

In this section, we report numerical results on the DPCP problem. We generated the data $Y = [\mathcal{X}, \mathcal{O}] \in \mathbb{R}^{n\times(p_1+p_2)}$ in the following manner. The inlier data $\mathcal{X}$ was generated as $\mathcal{X} = QC$, where $Q \in \mathbb{R}^{n\times(n-1)}$ is an orthogonal matrix and $C \in \mathbb{R}^{(n-1)\times p_1}$ is a coefficient matrix, whose elements are independent standard Gaussian random variables. The matrix Q was firstly generated according to the standard Gaussian distribution, then orthonormalized using QR decomposition. The outlier $\mathcal{O} \in \mathbb{R}^{n\times p_2}$ is a standard Gaussian matrix. Finally, the columns of matrix Y were normalized. We used the same initialization as that in [27]. We set the initial point $\mathbf{x}_0$ as the eigenvector of $YY^\top$ corresponding to the eigenvalue with minimum magnitude.

In the first figure of Figure 1 we report the quadratic fitting curves of different algorithms. As shown in [27], the DPCP model can tolerate $O((\#\text{inliers})^2)$ outliers; i.e., $p_2 = O(p_1^2)$. For different $p_2 \in \{40, 80, 120, \ldots, 600\}$, we find the smallest $p_1 \in \{60, 70, 80, \ldots, 260\}$ such that the principal angle $\theta < 10^{-1}$. Note that θ is the principal angle between $\mathbf{x}_k$ and $\mathcal{S}^\perp$, where $\mathbf{x}_k = \sin(\theta)\mathbf{n} + \cos(\theta)\mathbf{s}$, and $\mathbf{n} = \text{Proj}_{\mathcal{S}}(\mathbf{x}_k)/\|\text{Proj}_{\mathcal{S}}(\mathbf{x}_k)\|$, $\mathbf{s} = \text{Proj}_{\mathcal{S}^\perp}(\mathbf{x}_k)/\|\text{Proj}_{\mathcal{S}^\perp}(\mathbf{x}_k)\|$, and $\mathcal{S}$ is the column space of $\mathcal{X}$. Here the principal angle θ is the mean value of 10 trials; i.e., we find pairs (p_1, p_2) such that for fixed p_1, p_2 is the largest number of outliers that can be tolerated. We then use a quadratic function to fit these pairs (p_1, p_2). A higher curve indicates that the algorithm is more robust, because for fixed p_1, more outliers

can be tolerated. From the first figure of Figure 1, we see that the curve corresponding to PSGM-MBLS is lower than that corresponding to ManPPA, which indicates that ManPPA is more robust. In the second and third figures of Figure 1, we report the CPU time versus p_1 and p_2. We see that PSGM-MBLS is usually faster than ManPPA. In summary, Figure 1 suggests that ManPPA is usually slower than PSGM-MBLS, but it is more robust.

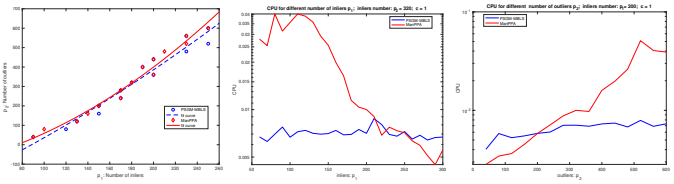

Fig. 1. Comparison on solving DPCP (1) with $n = 30$. Left: Quadratic fitting curves; Middle: CPU time versus the number of inliers p_1 ($p_2 = 320$); Right: CPU time versus the number of outliers p_2 ($p_1 = 200$).

B. Numerical Results on ODL

For the ODL problem, we randomly generated an orthonormal matrix $\hat{X} \in \mathcal{O}(n)$ and a Bernoulli-Guassian matrix $\hat{A} \in \mathbb{R}^{n\times p}$. We then set $Y = \hat{X}\hat{A}$. In Figure 2, we report the linear fitting curves for $\log(n)$ and $\log(p)$ and CPU time for results of ManPPA and PSGM-MBLS. Note that for the DL problem, it has been found empirically that the sample size and dimension satisfy $p = O(n^2)$ [12]. The linear fitting curves were found in the following manner. For different dimensions $n \in \{5, 10, 15 \ldots, 50\}$, we find the smallest sample number $p \in 2n + \{10, 20, 30, \ldots, 800\}$ such that $\theta < 10^{-1}$, where θ is the principal angle between $\mathbf{x}_k$ and the column space of $\hat{X}$, and it is the mean value of 10 trials. We then use a linear function to fit the points $(\log(n), \log(p))$. From Figure 2 we again find that PSGM-MBLS is usually faster than ManPPA but the latter is more robust, because the lower the fitting curve, the fewer the samples needed.

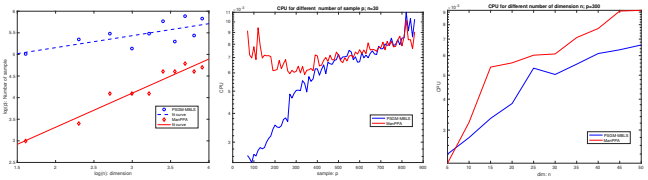

Fig. 2. Fitting curves and CPU time: comparison on solving ODL (1) with $n = 30, \gamma = 0.1$.

REFERENCES

[1] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

[2] P. Wang, H. Liu, and A. M.-C. So, "Globally convergent accelerated proximal alternating maximization method for L1–principal component analysis," in *Proceedings of the 2019 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2019)*, 2019, pp. 8147–8151.

[3] D. Spielman, H. Wang, and J. Wright, "Exact recovery of sparsely-used dictionaries," in *COLT*, 2012.

[4] L. Demanet and P. Hand, "Scaling law for recovering the sparsest element in a subspace," *Information and Inference*, vol. 3, no. 4, pp. 295–309, 2014.

[5] Q. Qu, J. Sun, and J. Wright, "Finding a sparse vector in subspace: linear sparsity using alternating directions," *IEEE Trans. Information Theory*, vol. 62, no. 10, pp. 5855–5880, 2016.

[6] J. Sun, Q. Qu, and J. Wright, "Complete dictionary recovery over the sphere I: Overview and the geometric picture," *IEEE Trans. Information Theory*, vol. 63, no. 2, pp. 853–884, 2017.

[7] J. Sun, Q. Qu, and J. Wright, "Complete dictionary recovery over the sphere II: Recovery by Riemannian trust-region method," *IEEE Trans. Information Theory*, vol. 63, no. 2, pp. 885–914, 2017.

[8] G. Lerman and T. Maunu, "An overview of robust subspace recovery," *Proceedings of the IEEE*, vol. 106, no. 8, pp. 1380–1410, 2018.

[9] M. C. Tsakiris and R. Vidal, "Dual principal component pursuit, 2018," *Journal of Machine Learning Research*, 2018.

[10] Z. Zhu, Y. Wang, D. P. Robinson, D. Q. Naiman, R. Vidal, and M. C. Tsakiris, "Dual principal component pursuit: probability analysis and efficient algorithms," *https://arxiv.org/pdf/1812.09924.pdf*, 2018.

[11] D. Gilboa, S. Buchanan, and J. Wright, "Efficient dictionary learning with gradient descent," *https://arxiv.org/abs/1809.10313*, 2018.

[12] Y. Bai, Q. Jiang, and J. Sun, "Subgradient descent learns orthogonal dictionaries," *ICLR 2019*, 2019.

[13] S. Hosseini and A. Uschmajew, "A Riemannian gradient sampling algorithm for nonsmooth optimization on manifolds," *SIAM J. Optim.*, vol. 27, no. 1, pp. 173–189, 2017.

[14] R. T. Rockafellar, "Augmented Lagrangians and applications of the proximal point algorithm in convex programming," *Math. Oper. Res.*, vol. 1, no. 2, pp. 97–116, 1976.

[15] R. T. Rockafellar, "Monotone operators and the proximal point algorithm," *SIAM Journal on Control and Optimization*, vol. 14, pp. 877–898, 1976.

[16] A. Iusem, "Augmented Lagrangian methods and proximal point methods for convex optimization," *Investigación Operativa*, vol. 8, pp. 11–49, 1999.

[17] B. Lemaire, "The proximal algorithm," *International Series of Numerical Mathematics*, pp. 73–87, 1989.

[18] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends in Optimization*, 2013.

[19] O. P. Ferreira and P. R. Oliveira, "Proximal point algorithm on Riemannian manifold," *Optimization*, vol. 51, pp. 257–270, 2002.

[20] M. Bačák, R. Bergmann, G. Steidl, and A. Weinmann, "A second order nonsmooth variational model for restoring manifold-valued images," *SIAM Journal on Scientific Computing*, vol. 38, pp. A567–A597, 2016.

[21] P. B. Borckmans, S. Easter Selvan, N. Boumal, and P.-A. Absil, "A Riemannian subgradient algorithm for economic dispatch with valve-point effect," *J. Comput. Appl. Math.*, vol. 255, pp. 848–866, 2014.

[22] G. C. Bento, J. X. Cruz Neto, and P. R. Oliveira, "A new approach to the proximal point method: convergence on general Riemannian manifolds," *J. Optim Theory Appl*, vol. 168, pp. 743–755, 2016.

[23] G. C. Bento, J. X. Cruz Neto, and P. R. Oliveira, "Convergence of inexact descent methods for nonconvex optimization on Riemannian manifolds," *https://arxiv.org/abs/1103.4828v1*, 2011.

[24] G. C. Bento, O. P. Ferreira, and J. G. Melo, "Iteration-complexity of gradient, subgradient and proximal point methods on Riemannian manifolds," *J. Optim Theory Appl*, vol. 173, pp. 548–562, 2017.

[25] S. Chen, S. Ma, A. M.-C. So, and T. Zhang, "Proximal Gradient Method for Manifold Optimization," *arXiv preprint arXiv:1811.00980*, 2018.

[26] X. Li, D. Sun, and K.-C. Toh, "A highly efficient semi-smooth Newton augmented Lagrangian method for solving Lasso problems," *SIAM J. Optim.*, vol. 28, pp. 433–458, 2018.

[27] Z. Zhu, Y. Wang, D. P. Robinson, D. Naiman, R. Vidal, and M. C. Tsakiris, "Dual principal component pursuit: Improved analysis and efficient algorithms," *NIPS*, 2018.