Acceleration Technique for Multiple k-means Clustering based Locally Low-rank Approach to Nonlinear Matrix Completion

1st Ryohei Sasaki
School of Computer Science
Tokyo University of Technology
Tokyo, Japan
sasakirh@stf.teu.ac.jp

2nd Katsumi Konishi
Faculty of Computer and Information Science
Hosei University
Tokyo, Japan
konishi@hosei.ac.jp

***Abstract*—This paper deals with nonlinear matrix completion problem, which is a problem of estimating missing entries in a given matrix, where its column vectors belong to a low dimensional manifold. Authors have proposed the method which assumes that a low dimensional manifold can be approximated locally as a low dimensional linear subspace and iteratively solves low-rank matrix completion problems for submatrices generated by using the k-means clustering for several values of k and restores missing entries. To reduce the computational time, this paper a faster solving technique by alternating optimization using the gradient method for the low-rank submatrix completion problem. Numerical examples show that the proposed algorithm achieves better performance than other algorithms.**

***Index Terms*—nonlinear matrix completion, matrix rank minimization, compressed sensing**

I. Introduction

This paper deals with matrix completion problem, which is a problem of restoring missing entries in a given matrix. The low-rank matrix completion problem has various applications in the field of signal processing, including collaborative filtering [1], low-order model fitting and system identification [2], image inpainting [3], and human-motion recovery [4], all of which are formulated as signal-recovery or estimation problems. Several methods of estimating missing entries have been studied, and most of them solve matrix completion problems by assuming that column or row vectors of a matrix belong to a low dimensional linear subspace and formulating them as matrix rank minimization problems [5]–[8]. However, in most practical applications, the column vectors of a matrix belong to an low dimensional manifold, these classical methods do not achieve high performance.

To achieve higher recovery performance for real applications, authors have proposed some algorithms for the nonlinear matrix completion problem. One of the methods is based on the assumption that a local neighborhood of each vector on the manifold can be approximated as a low dimensional linear subspace [9]. This approach iteratively solves low-rank matrix completion problems for submatrices consisting of neighbor column vectors of the matrix and achieves higher recovery performance. However, it takes a lot of computing time to estimate missing entries because submatrices are generated for all column vectors using their neighbors and because each matrix completion problem is solved repeatedly until converge. To provide a faster algorithm based on the locally low-rank approach, authors have proposed a new locally low-rank approach, which iteratively solves low-rank submatrix completion problems generated by using the k-means clustering for several values of k [10]. To further reduce the computational time, this paper reformulates the submatrix completion problem used in [10] and propose a faster solving technique by alternating optimization using the gradient method. Numerical examples show that the proposed algorithm solves nonlinear matrix completion problems better than a classical low-rank approach and takes less computing time than the locally low-rank approach.

II. Matrix Completion and Low-rank Approach

A. Linear Matrix Completion

This section introduces the nonlinear matrix completion and its solution.

Firstly, this paper describes a general linear matrix completion problem. The matrix completion problem is a problem which estimate missing entries of a matrix $\boldsymbol{X} \in \mathbb{R}^{M \times N}$. Many classical methods assume that the column (or row) vectors of the matrix belong to a low dimensional linear subspace and solve the following matrix rank minimization problem,

$$\begin{array}{ll} \text{Minimize} & \text{rank}(\boldsymbol{X}) \\ \text{subject to} & \boldsymbol{P}_{\Omega}(\boldsymbol{X}) = \boldsymbol{P}_{\Omega}(\boldsymbol{X}_{true}) \end{array}, \tag{1}$$

where Ω denotes a given index set, $\boldsymbol{P}_{\Omega} : \mathbb{R}^{M \times N} \mapsto \mathbb{R}^{M \times N}$ denotes a linear operator that projects all entries except subscripts included in the set Ω to 0, and $\boldsymbol{X}_{true}$ denotes a true matrix to be recovered. Instead of ranks, a variety of objective functions are used since the problem (1) is known as NP-hard. This paper introduces a truncated nuclear norm minimization

approach, which relaxes the objective function of (1) by the truncated nuclear norm of $\boldsymbol{X}$ as follows,

$$\begin{array}{ll} \text{Minimize} & \|\boldsymbol{X}\|_{*,r} \\ \text{subject to} & \boldsymbol{P}_\Omega(\boldsymbol{X}) = \boldsymbol{P}_\Omega(\boldsymbol{X}_{true}) \end{array}, \tag{2}$$

where $\|\boldsymbol{X}\|_{*,r}$ with given constant r denotes the truncated nuclear norm defined by

$$\|\boldsymbol{X}\|_{*,r} = \sum_{i=r+1}^{M} \sigma_i \tag{3}$$

with respect to the i-th greatest singular value σ_i of $\boldsymbol{X}$. This problem can be solved by iterative partial matrix shrinkage algorithm (IPMS) [7] which iterates the following update schemes until converge,

$$\begin{array}{ll} \text{Step 1} & \boldsymbol{Z} \leftarrow \boldsymbol{\mathcal{T}}_{r,\lambda}(\boldsymbol{X}) \\ \text{Step 2} & \boldsymbol{X} \leftarrow \boldsymbol{P}^c_\Omega(\boldsymbol{Z}) + \boldsymbol{P}_\Omega(\boldsymbol{X}_{true}), \end{array}$$

where $\boldsymbol{\mathcal{T}}_{r,\lambda}(\boldsymbol{X})$ denotes a partial soft thresholding operator, which shrinks i-th greatest singular values by λ, that is, replaces σ_i by $\max(0, \sigma_i - \lambda)$ for $i = r+1, r+2, \cdots, M$. The update scheme achieves the best performance when the parameter r satisfies $r = \text{rank}(\boldsymbol{X}_{true})$. Although the best parameter r is usually unknown, a heuristic algorithm to search for r has been proposed in [7].

However, in most practical applications, the column vectors of a matrix do not belong to a low dimensional linear subspace. Therefore, the classical methods which estimate the low-rank matrix do not achieve high performance.

B. NONLINEAR MATRIX COMPLETION

In recent years, several methods which assume that the column vector of the matrix belong to a low dimensional manifold have been proposed. This paper introduces our approach based on the idea which local area of each vector on the manifold can be approximated by a low dimensional linear subspace.

Let us define the diagonal matrix $\boldsymbol{D}^{(i)} \in \{0,1\}^{N \times N}$ for $i = 1, \cdots, N$. [9] assumes that the submatrix $\boldsymbol{X}\boldsymbol{D}^{(i)}$ can be approximated by a low-rank matrix if the j-th diagonal element $\boldsymbol{D}^{(i)}_{j,j}$ is defined by

$$\boldsymbol{D}^{(i)}_{j,j} = \begin{cases} 1 & \text{if } \boldsymbol{x}_j \text{ is nearest neighbor of } \boldsymbol{x}_i \\ 0 & \text{otherwise} \end{cases}, \tag{4}$$

where $\boldsymbol{x}_i$ denotes i-th column vector of $\boldsymbol{X}$. Thus, the nonlinear matrix completion problem is formulated by the following local matrix completion problem,

$$\begin{array}{ll} \text{Minimize} & \displaystyle\sum_{i=1}^{N} \|\boldsymbol{X}\boldsymbol{D}^{(i)}\|_{*,r} \\ \text{subject to} & \boldsymbol{P}_\Omega(\boldsymbol{X}) = \boldsymbol{P}_\Omega(\boldsymbol{X}_{true}) \end{array}. \tag{5}$$

[9] has proposed an algorithm using the IPMS algorithm, and the algorithm iterates the following update scheme for each i,

$$\begin{array}{ll} \text{Step 1} & \text{Obtain } \boldsymbol{D}^{(i)} \text{ by (4)} \\ \text{Step 2} & \boldsymbol{Z} \leftarrow \boldsymbol{\mathcal{T}}_{r,\lambda}(\boldsymbol{X}\boldsymbol{D}^{(i)}) \\ \text{Step 3} & \boldsymbol{Y} \leftarrow \boldsymbol{Z}\boldsymbol{D}^{(i)} + \boldsymbol{X}(\boldsymbol{I} - \boldsymbol{D}^{(i)}) \\ \text{Step 4} & \boldsymbol{X} \leftarrow \boldsymbol{P}^c_\Omega(\boldsymbol{Y}) + \boldsymbol{P}_\Omega(\boldsymbol{X}_{true}), \end{array}$$

where $\boldsymbol{I}$ denotes the identity matrix and the nearest neighbor is provided w.r.t. Euclidean distance in Step 1.

However, this algorithm requires N singular value decompositions (SVDs) in one iteration and requires significant computing time. To reduce the number of submatrix completion problems, [10] has proposed a new algorithm, which applies the k-means clustering method to $\boldsymbol{x}_i$ w.r.t. Euclidean distance and gives k submatrix completion problems of k clusters. [10] defines diagonal matrix $\boldsymbol{D}^{(i)}$ whose j-th diagonal entries $\boldsymbol{D}^{(i)}_{j,j}$ is defined by

$$\boldsymbol{D}^{(i)}_{j,j} = \begin{cases} 1 & \text{if } \boldsymbol{x}_j \text{ is a member of the } i\text{-th clustering} \\ 0 & \text{otherwise} \end{cases}, \tag{6}$$

and then considers a relaxed submatrix completion problem with (6) as follows,

$$\begin{array}{ll} \text{Minimize} & \displaystyle\sum_{i=1}^{k} \|\boldsymbol{X}\boldsymbol{D}^{(i)}\|_{*,r} \\ \text{subject to} & \boldsymbol{P}_\Omega(\boldsymbol{X}) = \boldsymbol{P}_\Omega(\boldsymbol{X}_{true}) \end{array}. \tag{7}$$

Since the algorithm for (7) requires k SVDs in one iteration, the computational time cost can be significantly reduced when $k \ll N$. However, the algorithm corresponds with a low-rank approach based on the assumption that column vectors belong to multiple low dimensional linear subspaces, that is, a manifold is approximated by piece-wise linear subspaces, and the recovery accuracy decreases. To achieve high recovery accuracy, [10] has proposed multiple k-means clustering based algorithm, which uses k-means clustering for $k \in \{k_1, k_2, ..., k_K\}$, where k_i denotes the number of members in the i-th cluster. Based on the idea, the submatrix completion problem is formulated as follows,

$$\begin{array}{ll} \text{Minimize} & \displaystyle\sum_{l=1}^{K}\sum_{i=1}^{k_l} \|\boldsymbol{X}\boldsymbol{D}^{(l,i)}\|_{*,r} \\ \text{subject to} & \boldsymbol{P}_\Omega(\boldsymbol{X}) = \boldsymbol{P}_\Omega(\boldsymbol{X}_{true}) \end{array}, \tag{8}$$

where $\boldsymbol{D}^{(l,i)}$ is defined by

$$\boldsymbol{D}^{(l,i)}_{j,j} = \begin{cases} 1 & \text{if } \boldsymbol{x}_j \text{ is a member of the } i\text{-th cluster} \\ & \text{of the } l\text{-th clustering } (k = k_l) \\ 0 & \text{otherwise} \end{cases}, \tag{9}$$

A solution of this problem is obtained by the same update

Algorithm 1 Multiple k-means clustering based locally low-rank algorithm.

Require: $\boldsymbol{X}(0), \{k_l\}_{l=1}^K, \delta, \alpha_{min}, \eta_\alpha, \epsilon, t_{max}$
$\boldsymbol{X} \leftarrow \boldsymbol{X}^{(0)}, t \leftarrow 0, \alpha \leftarrow 1$
repeat
  $\boldsymbol{X}^{old} \leftarrow \boldsymbol{X}, t \leftarrow t+1, \alpha \leftarrow \max(\alpha/\eta_\alpha, \alpha_{min})$
  for $l = 1$ to K **do**
    Apply the k_l-means clustering to $\{\boldsymbol{x}_i\}_{i=1}^N$, obtain k_l clusters, and construct $\boldsymbol{D}^{(l,i)}$
    for $i = 1$ to k_l **do**
      $[\boldsymbol{U}, \sigma_1, \sigma_2, \cdots, \sigma_M, \boldsymbol{V}] \leftarrow \mathrm{SVD}(\boldsymbol{X}\boldsymbol{D}^{(l,i)})$
      $\hat{r} \leftarrow \underset{r}{\operatorname{argmin}}\, \sigma_r$ s.t. $\sigma_r \geq \alpha\sigma_1$
      $\lambda \leftarrow \delta\sigma_{\hat{r}}$
      $\boldsymbol{Z}^{(l,i)} \leftarrow \boldsymbol{\mathcal{T}}_{\hat{r},\lambda}(\boldsymbol{X}\boldsymbol{D}^{(l,i)})$
      $\boldsymbol{Y} \leftarrow \boldsymbol{Z}^{(l,i)}\boldsymbol{D}^{(l,i)} + \boldsymbol{X}(\boldsymbol{I} - \boldsymbol{D}^{(l,i)})$
      $\boldsymbol{X} \leftarrow \boldsymbol{P}_\Omega^c(\boldsymbol{Y}) + \boldsymbol{P}_\Omega(\boldsymbol{X}^{(0)})$
    end for
  end for
until $\|\boldsymbol{X}^{old} - \boldsymbol{X}\|_F / \|\boldsymbol{X}\|_F < \epsilon$ or $t_{max} < t$
Ensure: $\boldsymbol{X}$.

scheme for (5) replacing $\boldsymbol{D}^{(i)}$ with $\boldsymbol{D}^{(l,i)}$. Finally, this papar shows the multiple k-means clustering based locally low-rank algorithm as shown in Algorithm 1, where $\boldsymbol{X}$ is partially shrunk according to submatrix completion problems (8) after applying k_l-means clustering for $l \in \{1, 2, ..., K\}$ and constructing $\{\boldsymbol{D}^{(l,i)}\}_{l=1}^K$. This algorithm requires $\sum_{l=1}^K k_l$ SVDs in one iteration.

III. ACCELERATION FOR MULTIPLE k-MEANS CLUSTERING BASED LOCALLY LOW-RANK APPROACH

The major computational cost of Algorithm 1 is derived from computing the SVD. To reduce the cost, this paper reformulates (8) using substitute objective function and proposes an algorithm based on alternating optimization using the gradient method instead of SVD.

Firstly, this paper considers the standard linear matrix completion problem. If the rank of $\boldsymbol{X}$ is low, there exists a low rank matrix $\boldsymbol{W} \in \mathbb{R}^{M\times M}$ such that $\boldsymbol{X} = \boldsymbol{W}\boldsymbol{X}$ is satisfied. Based on the idea, [8] has formulated the low-rank matrix completion problem as follows,

$$\begin{array}{ll}\text{Minimize} & \|\boldsymbol{W}\|_F^2 \\ \text{subject to} & \boldsymbol{X} = \boldsymbol{W}\boldsymbol{X}, \boldsymbol{P}_\Omega(\boldsymbol{X}) = \boldsymbol{P}_\Omega(\boldsymbol{X}_{true})\end{array}, \tag{10}$$

where $\|\cdot\|_F$ denotes the Frobenius norm. In order to provide an update scheme, (11) is relaxed as follows,

$$\begin{array}{ll}\text{Minimize} & \|(\boldsymbol{I} - \boldsymbol{W})\boldsymbol{X}\|_F^2 + \gamma\|\boldsymbol{W}\|_F^2 \\ \text{subject to} & \boldsymbol{P}_\Omega(\boldsymbol{X}) = \boldsymbol{P}_\Omega(\boldsymbol{X}_{true})\end{array}, \tag{11}$$

where $\gamma > 0$ is given constant. Thus, this paper reformulates the problem (8) by substituting an objective function as follows,

$$\begin{array}{ll}\text{Minimize} & \displaystyle\sum_{l=1}^K \sum_{i=1}^{k_l} f_\gamma(\boldsymbol{W}^{(l,i)}, \boldsymbol{X}\boldsymbol{D}^{(l,i)}) \\ \text{subject to} & \boldsymbol{P}_\Omega(\boldsymbol{X}) = \boldsymbol{P}_\Omega(\boldsymbol{X}_{true})\end{array}, \tag{12}$$

where f_γ is defined by

$$f_\gamma(\boldsymbol{W}^{(l,i)}, \boldsymbol{X}\boldsymbol{D}^{(l,i)}) = \|(\boldsymbol{I} - \boldsymbol{W}^{(l,i)})\boldsymbol{X}\boldsymbol{D}^{(l,i)}\|_F^2 + \gamma\|\boldsymbol{W}^{(l,i)}\|_F^2 \tag{13}$$

Since this problem is the bi-convex quadratic problem for $\boldsymbol{X}$ and $\boldsymbol{W}^{(l,i)}$, the solution of this problem can be obtained by an alternating optimization for $\boldsymbol{X}$ and $\boldsymbol{W}^{(l,i)}$ for each (l,i). To reduce the computational time cost, this paper proposes an algorithm using the gradient descent method although the optimal solution $\boldsymbol{W}^{(l,i)}$ can be solved using SVD.

Let us focus on computing $\boldsymbol{W}^{(l,i)}$ with fixed $\boldsymbol{X}$. We have

$$\begin{aligned}\nabla_{\boldsymbol{W}^{(l,i)}} &= \frac{1}{2}\frac{\partial}{\partial \boldsymbol{W}^{(l,i)}} f_\gamma(\boldsymbol{W}^{(l,i)}, \boldsymbol{X}\boldsymbol{D}^{(l,i)}) \\ &= \gamma\boldsymbol{W}^{(l,i)} + \boldsymbol{W}^{(l,i)}\boldsymbol{X}\boldsymbol{D}^{(l,i)}\boldsymbol{X}^T - \boldsymbol{X}\boldsymbol{D}^{(l,i)}\boldsymbol{X}^T,\end{aligned} \tag{14}$$

and consider the step length μ for this gradient. Because $f_\gamma(\boldsymbol{W}^{(l,i)} - \mu\nabla_{\boldsymbol{W}^{(l,i)}}, \boldsymbol{X}\boldsymbol{D}^{(l,i)})$ is a convex quadratic function of μ for given $\boldsymbol{W}^{(l,i)}$ and $\boldsymbol{X}$ and because it holds that

$$\begin{aligned}&\frac{1}{2}\frac{\partial}{\partial\mu} f_\gamma(\boldsymbol{W}^{(l,i)} - \mu\nabla_{\boldsymbol{W}^{(l,i)}}, \boldsymbol{X}\boldsymbol{D}^{(l,i)}) \\ &= \mu\left(\|\nabla_{\boldsymbol{W}^{(l,i)}}\boldsymbol{X}\boldsymbol{D}^{(l,i)}\|_F^2 + \gamma\|\nabla_{\boldsymbol{W}^{(l,i)}}\|_F^2\right) \\ &\quad + \mathrm{trace}(\boldsymbol{X}\boldsymbol{D}^{(l,i)}\boldsymbol{X}^T\nabla_{\boldsymbol{W}^{(l,i)}}^T(\boldsymbol{W}^{(l,i)} - \boldsymbol{I}) + \gamma\nabla_{\boldsymbol{W}^{(l,i)}}^T\boldsymbol{W}^{(l,i)}),\end{aligned}$$

the step length minimizing $f_\gamma(\boldsymbol{W}^{(l,i)} - \mu\nabla_{\boldsymbol{W}^{(l,i)}}, \boldsymbol{X}\boldsymbol{D}^{(l,i)})$ is obtained as follows,

$$\mu = \frac{\mathrm{trace}(\boldsymbol{X}\boldsymbol{D}^{(l,i)}\boldsymbol{X}^T\nabla_{\boldsymbol{W}^{(l,i)}}^T(\boldsymbol{W}^{(l,i)} - \boldsymbol{I}) + \gamma\nabla_{\boldsymbol{W}^{(l,i)}}^T)}{\left(\|\nabla_{\boldsymbol{W}^{(l,i)}}\boldsymbol{X}\boldsymbol{D}^{(l,i)}\|_F^2 + \gamma\|\nabla_{\boldsymbol{W}^{(l,i)}}\|_F^2\right)}. \tag{15}$$

In the same way, the gradient for $\boldsymbol{X}$ with fixed $\boldsymbol{W}^{(l,i)}$ is obtained

$$\begin{aligned}\nabla_{\boldsymbol{X}} &= \frac{1}{2}\frac{\partial}{\partial\boldsymbol{X}} f_\gamma(\boldsymbol{W}^{(l,i)}, \boldsymbol{X}\boldsymbol{D}^{(l,i)}) \\ &= (\boldsymbol{I} - \boldsymbol{W}^{(l,i)})^T(\boldsymbol{I} - \boldsymbol{W}^{(l,i)})\boldsymbol{X}\boldsymbol{D}^{(l,i)},\end{aligned} \tag{16}$$

and the step length τ minimizing $f_\gamma(W^{(l,i)}, (\boldsymbol{X} - \tau\nabla_{\boldsymbol{X}})\boldsymbol{D}^{(l,i)})$ is obtained as follows,

$$\tau = \frac{\mathrm{trace}((\boldsymbol{I} - \boldsymbol{W}^{(l,i)})^T(\boldsymbol{I} - \boldsymbol{W}^{(l,i)})\nabla_{\boldsymbol{X}}\boldsymbol{D}^{(l,i)}\boldsymbol{X}^T)}{\|(\boldsymbol{I} - \boldsymbol{W}^{(l,i)})\nabla_{\boldsymbol{X}}\boldsymbol{D}^{(l,i)}\|_F^2}. \tag{17}$$

Algorithm 2 Multiple k-means clustering based locally low-rank algorithm using gradient descent.

Require: $\boldsymbol{X}^{(0)}, \{k_l\}_{l=1}^{K}, \gamma_{max}, \gamma_{min}, \eta_\gamma, \epsilon, t_{max}$

$\boldsymbol{X} \leftarrow \boldsymbol{X}^{(0)}, t \leftarrow 0, \gamma \leftarrow \gamma_{max}$

$\boldsymbol{W}^{(l,i)} \leftarrow 0$

repeat

  $\boldsymbol{X}^{old} \leftarrow \boldsymbol{X}, t \leftarrow t+1, \gamma \leftarrow \max(\gamma/\eta_\gamma, \gamma_{min})$

  for $l = 1$ to K **do**

    Apply the k_l-means clustering to $\{\boldsymbol{x}_i\}_{i=1}^{N}$, obtain k_l clusters, and construct $\boldsymbol{D}^{(l,i)}$

    for $i = 1$ to k_l **do**

      $\nabla_{\boldsymbol{W}^{(l,i)}} \leftarrow (14)$

      $\mu \leftarrow (15)$

      $\boldsymbol{W}^{(l,i)} \leftarrow \boldsymbol{W}^{(l,i)} - \mu \nabla_{\boldsymbol{W}^{(l,i)}}$

      $\nabla_{\boldsymbol{X}} \leftarrow (16)$

      $\tau \leftarrow (17)$

      $\boldsymbol{X} \leftarrow \boldsymbol{P}_{\Omega}^{c}(\boldsymbol{X} - \tau \nabla_{\boldsymbol{X}}) + P_{\Omega}(\boldsymbol{X}^{(0)})$

    end for

  end for

until $\|\boldsymbol{X}^{old} - \boldsymbol{X}\|_F / \|\boldsymbol{X}\|_F < \epsilon$ or $t_{max} < t$

Ensure: $\boldsymbol{X}$.

Finally, this papar shows the algorithm for (2) using the gradient descent as shown in Algorithm 2. In Algorithm 2, this paper utilizes a heuristic technique similar to [6] to give the the parameter γ which achieves the best performance.

IV. NUMERICAL EXAMPLE

This section presents several numerical examples for the nonlinear matrix completion. In this section, each i-th column vector of $\boldsymbol{X}_{true} \in \mathbb{R}^{M \times N}$ is generated by the following mapping function $\boldsymbol{\psi}_p : \mathbb{R}^r \mapsto \mathbb{R}^{\binom{r+p}{p}}$ defined as

$$\boldsymbol{\psi}_p(\boldsymbol{y}) = (\boldsymbol{y}^{\boldsymbol{\alpha}})_{|\boldsymbol{\alpha}| \le p} \in \mathbb{R}^{\binom{r+p}{p}}, \tag{18}$$

where $\boldsymbol{\alpha} = [\alpha_1 \ \cdots \ \alpha_r]$ denotes a multi-index of non-negative integers, $\boldsymbol{y}^{\boldsymbol{\alpha}}$ is defined as $\boldsymbol{y}^{\boldsymbol{\alpha}} = y_1^{\alpha_1} \cdots y_r^{\alpha_r}$, $|\boldsymbol{\alpha}| = \alpha_1 + \cdots + \alpha_r$. The matrix $\boldsymbol{X}_{true}$ is generated by $\boldsymbol{U}_p \left[\boldsymbol{\psi}_p(\boldsymbol{y}_1), \boldsymbol{\psi}_p(\boldsymbol{y}_2), \cdots, \boldsymbol{\psi}_p(\boldsymbol{y}_N)\right]$ using $\boldsymbol{U}_p \in \mathbb{R}^{M \times \binom{r+p}{p}}$ and $\boldsymbol{Y} = [\boldsymbol{y}_1 \ \boldsymbol{y}_2 \ \cdots \ \boldsymbol{y}_N] \in \mathbb{R}^{r \times N}$ generated by an i.i.d. continuous uniform distribution whose supports are $[-0.5, 0.5]$ and $[-1, 1]$, the elements of $\boldsymbol{Y}$ are normalized as $\max |(Y)_{i,j}| = 1$. The index set Ω is generated using the Bernoulli distribution with the given probability $q = 0.3$, for which an index (i, j) belongs to Ω. This paper uses relative recovery error as

$$\text{RE } [\%] = \frac{\|\boldsymbol{X}_{true} - \boldsymbol{X}\|_F}{\|\boldsymbol{X}_{true}\|_F} \times 100$$

to evaluate each algorithm. All numerical experiments were run in MATLAB 2020a on a PC with an AMD Ryzen 7 3700X 3.6 GHz CPU, 16GB of RAM, and no swap memory.

TABLE I
THE AVERAGE COMPUTATIONAL TIME COST (SECOND) OF THE ALGORITHMS

Algorithm	IPMS	Algorithm 1	Algorithm 2
	8.4 [s]	161.4[s]	88.4[s]

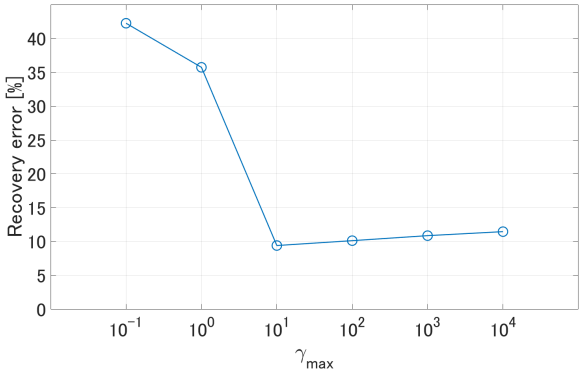

Fig. 1. Average RE of Algorithm 2 with $\gamma_{max} \in \{10^{-1}, \cdots, 10^{4}\}$ for 10 trials with $p = 5, r = 4$ for (18)

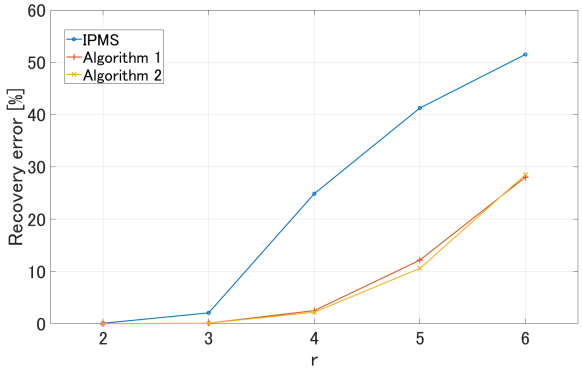

Fig. 2. Average RE for 10 trials with $p = 3$ for (18)

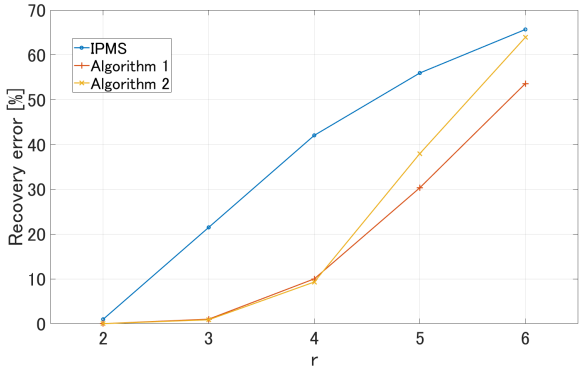

Fig. 3. Average RE for 10 trials with $p = 5$ for (18)

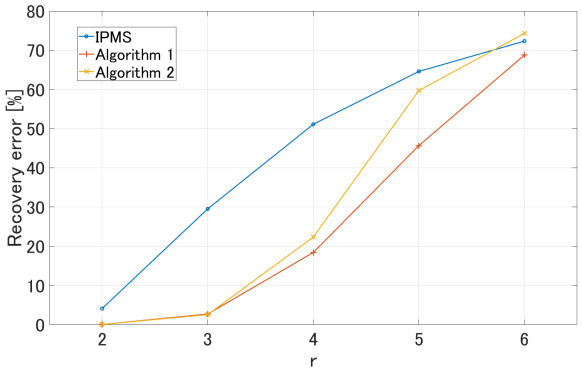

Fig. 4. Average RE for 10 trials with $p = 7$ for (18)

This paper applies the low-rank matrix-completion algorithm IPMS [7], the nonlinear matrix completion method Algorithm 1 [10]) and Algorithm 2 (proposed method) to several matrix completion problems with $M = 100$, $N = 4,000$, and $p = 3, 5, 7$ for (18). A maximum iteration number of $t_{max} = 1,000$ is used for IPMS, Algorithm 1, and Algorithm 2. The parameters for IPMS and Algorithm 1 are given as $\eta_\alpha = 10^{\frac{4}{t_{max}}}$, $\delta = 10^{-2}$, $K = 3$, and $(k_1, k_2, k_3) = [20, 30, 40]$, which are used for a numerical example in [10]. In Algorithm 2, the parameters in common with Algorithm 1 are used as same as those, and the other parameters are given as $\gamma_{max} = 10^1$ and $\eta_\gamma = 10^{\frac{5}{t_{max}}}$. The parameter $\gamma_{max} = 10^1$ in Algorithm 2 achieves the best performance in the result for $p = 5, r = 4$ (Fig. 1). The results are shown in Fig. 2-4 for $r \in \{2, 3, 4, 5, 6\}$. As can be seen, estimation accuracy of Algorithm 2 is almost same as that of Algorithm 1 for $r = 2, 3, 4$. The average computational time costs for all experiments are shown in Table I. This result indicates that Algorithm 2 is about 1.8 times faster.

Furthermore, this paper evaluates Algorithm 2 on the motion capture data, which consists of time-series trajectories of human motions such as running and jumping. This paper uses the trial #6 of subject #56 of the CMU motion capture dataset. The data has measurements from $M = 62$ sensors at 6784 time instants, which the data matrix is known as high rank matrix. In this experiment, the sequence is downsampled by factor 2, which the data matrix has $M = 62$ rows and $N = 3392$ columns. Then, the elements of the data matrix were randomly observed with the ratio $q = 0.3$, and this paper applied the matrix completion algorithms with the same parameters which is used in the previous simulation result. The average computational time costs and recovery errors for 10 trials are shown in Table II. Similar to results on synthetic data, the estimation accuracies of Algorithm 1 and 2 are better than that of IPMS, and the computational time of Algorithm 2 is less than half that of Algorithm 1.

TABLE II
THE AVERAGE COMPUTATIONAL TIME COST (SECOND) AND RECOVERY ERROR OF THE ALGORITHMSFOR CMU DATASET

Algorithm	IPMS	Algorithm 1	Algorithm 2
Time cost	4.7 [s]	97.5[s]	45.4[s]
RE	33.1 [%]	14.2[%]	17.4[%]

V. CONCLUSION

This paper deals with nonlinear matrix completion problem, which is a problem of restoring missing entries in a given matrix, where its column vectors belong to a low dimensional manifold. Although the conventional locally low-rank algorithm has a good recovery performance for this problem, it requires a lot of computing time. To reduce computational cost, this paper proposes an acceleration technique which iteratively solves low-rank submatrix completion problems using gradient descent instead of singular value decomposition. Numerical examples show that the proposed algorithm is about 1.8 times faster and has almost the same recovery performance comparing with the original locally low-rank algorithm. Because the convergence of the proposed algorithm is not guaranteed, further analysis of the algorithm would be required.

ACKNOWLEDGMENT

This work was supported by JSPS KAKENHI Grant Number JP19H02163.

REFERENCES

[1] X. Guan, C. T. Li, Y. Guan, "Matrix factorization with rating completion: An enhanced SVD model for collaborative filtering recommender systems", IEEE Access, vol. 5, pp. 27668–27678, 2017.
[2] M. Verhaegen, A. Hansson, "N2SID: Nuclear norm subspace identification of innovation models," Automatica, vol. 72, pp. 57–63, 2016.
[3] K. H. Jin, J. C. Ye, "Annihilating filter-based low-rank Hankel matrix approach for image inpainting," IEEE Trans. on Image Process., vol. 24, no. 11, pp. 3498–3511, 2015.
[4] Q. Zhao, D. Meng, Z. Xu, W. Zuo, Y. Yan, "L_1-norm low-rank matrix factorization by variational Bayesian method," IEEE Trans. Neural Netw. Learn. Syst., vol. 26, no. 4, pp. 825–839, 2015.
[5] J. F. Cai, E. J. Candés, Z. Shen, "A singular value thresholding algorithm for matrix completion," SIAM Journal on Optimization, vol. 20, no. 4, pp. 1956–1982, 2010.
[6] K. Mohan, M. Fazel, "Iterative reweighted algorithms for matrix rank minimization," The Journal of Machine Learning Research (JMLR), vol. 13, no. 1, pp. 3441–3473, 2012.
[7] K. Konishi, K. Uruma, T. Takahashi, T. Furukawa, "Iterative partial matrix shrinkage algorithm for matrix rank minimization," Signal Process., vol 100, pp. 124–131, 2014.
[8] K. Konishi, "Parallel GPU implementation of null space based alternating optimization algorithm for large-scale matrix rank minimization," IEEE Int. Conf. Acoustics, Speech and Signal Process. (ICASSP), pp. 3689-3692, 2012.
[9] R. Sasaki and K. Konishi and T. Takahashi and T. Furukawa, "Low-Rank and Locally Linear Embedding Approach to Image Inpainting," Proc. IEEE International Conference on Visual Communications and Image Processing, 2018.
[10] K. Konishi, T. Shise, R. Sasaki and T. Furukawa, "Multiple k-Means Clustering Based Locally Low-Rank Approach to Nonlinear Matrix Completion," 2019 27th European Signal Processing Conference (EUSIPCO), pp. 1–5, 2019.