Low rank matrix approximation is a popular topic in machine learning. In this paper, we propose a new algorithm for this topic by minimizing the least-squares estimation over the Riemannian manifold of fixed-rank matrices. The algorithm is an adaptation of classical gradient descent within the framework of optimization on manifolds. In particular, we reformulate an unconstrained optimization problem on a low-rank manifold into a differential dynamic system. We develop a splitting numerical integration method by applying a splitting integration scheme to the dynamic system. We conduct the convergence analysis of our splitting numerical integration algorithm. It can be guaranteed that the error between the recovered matrix and true result is monotonically decreasing in the Frobenius norm. Moreover, our splitting numerical integration can be adapted into matrix completion scenarios. Experimental results show that our approach has good scalability for large-scale problems with satisfactory accuracy.
matrix completion scenarios, where the matrix $M$ is partially observed. Empirical results are also encouraging, especially on large-scale datasets.

The remainder of the paper is organized as follows. Section 2 presents the notation frequently used in this paper and the problem formulation. Section 3 describes our algorithm and theoretical analysis. Empirical results are given in Section 4.

## 2 Notation and Preliminaries

First of all, we present the notation used in this paper. Let $I_m$ be the $m \times m$ identity matrix. Given a matrix $Y \in \mathbb{R}^{m \times n}$, $\|Y\|_F$ denotes the Frobenius norm of $Y$ and $\|Y\|_2$ denotes the spectral norm.

It is well established that every rank-$r$ matrix $Y \in \mathbb{R}^{m \times n}$ can be written in the form

$$ Y = USV^T, $$

where $U \in \mathbb{R}^{m \times r}$ and $V \in \mathbb{R}^{n \times r}$ are column orthonormal, i.e., $U^T U = I_r$ and $V^T V = I_r$, and $S \in \mathbb{R}^{r \times r}$ is nonsingular. Notice that here we do not require $S$ to be the diagonal matrix of the singular values. The representation in Eqn. (3) is not unique because $Y = USV^T$ is another representation where $U = UP$, $V = VQ$, and $S = P^T SQ$ whenever $P, Q \in \mathbb{R}^{r \times r}$ are any orthonormal matrices.

As a substitute for the non-uniqueness in Eqn. (3), we will use a unique decomposition in the tangent space. Let $\mathcal{V}_{m,r}$ represent the Stiefel manifold of real column orthonormal matrices of size $m \times r$ ($m > r$). The tangent space at the point $U \in \mathcal{V}_{m,r}$ is defined as:

$$ \mathcal{T}_U \mathcal{V}_{m,r} = \{ \delta U \in \mathbb{R}^{m \times r} : \delta U^T U + U^T \delta U = 0 \} = \{ \delta U \in \mathbb{R}^{m \times r} : U^T \delta U \in so(r) \}, $$

where $so(r)$ denotes the space of skew-symmetric real $r \times r$ matrices. Consider the extended tangent map of $(S, U, V) \mapsto Y = USV^T$:

$$ \mathbb{R}^{r \times r} \times \mathcal{T}_U \mathcal{V}_{m,r} \times \mathcal{T}_V \mathcal{V}_{n,r} \rightarrow \mathcal{T}_Y \mathcal{M}_r \times so(r) \times so(r), $$

$$(\delta S, \delta U, \delta V) \mapsto (\delta USV^T + U\delta SV^T + US\delta V^T, U^T \delta U, V^T \delta V).$$

The manifold of rank-$r$ matrices is denoted by $\mathcal{M}_r$. The tangent space at any $Y \in \mathcal{M}_r$ is denoted by $\mathcal{T}_Y \mathcal{M}_r$, which is defined as follows. Every $\delta Y \in \mathcal{T}_Y \mathcal{M}_r$ can be written into the following form:

$$ \delta Y = \delta USV^T + U\delta SV^T + US\delta V^T, $$

where $\delta S \in \mathbb{R}^{r \times r}$, $\delta U \in \mathcal{T}_U \mathcal{V}_{m,r}$, and $\delta V \in \mathcal{T}_V \mathcal{V}_{n,r}$. Furthermore, $\delta S$, $\delta U$, and $\delta V$ are uniquely determined by $\delta Y$ if we impose the orthogonality constraints:

$$ U^T \delta U = 0, $$

$$ V^T \delta V = 0. $$

The projection operators onto the spaces spanned by the columns of $U$ and $V$, and their orthogonal complements are defined as

$$ P_U = UU^T, $$

$$ P_V = VV^T, $$

$$ P_U^\perp = I_m - UU^T, $$

$$ P_V^\perp = I_n - VV^T. $$

Then $\delta S$, $\delta U$ and $\delta V$ are uniquely determined by $\delta Y$ as follows:

$$ \delta S = U^T \delta Y V, $$

$$ \delta U = P_U^\perp \delta Y VS^{-1}, $$

$$ \delta V = P_V^\perp \delta Y^T US^{-T}. $$

## 2.1 Dynamic Low Rank Approximation

Let us return to Problem 2 and let

$$ f(Y) = \frac{1}{2} \|Y - M\|_F^2 $$

denote the objective function. The gradient of $f(Y)$ in $Y$ can be written as:

$$ \nabla f(Y) = Y - M. $$
Recall that the constraint $Y \in M_r$, Riemannian gradient, denoted $\nabla Y$, is used instead of $\nabla f(Y)$, which is a specific tangent vector corresponding to the direction of steepest ascent of $f(Y)$ but restricted to the tangent space $T_Y M_r$. It can be solved via the following optimization problem:

$$
\arg \min_{Y \in \mathbb{M} \times n} \frac{1}{2} \| \nabla f(Y) - \nabla A \|_F^2
$$

(6)

s.t. $\nabla Y \in T_Y M_r$,

where we denote $\nabla A \triangleq -\nabla f(Y)$ for notational simplicity and $\nabla A$ can be seen as a given constant at each iteration. The solution of Problem (6) is well-studied in the differential geometry literature. That is,

Proposition 1. Koch and Lubich [2007] Let $Y = USV^T \in M_r$, where $S \in \mathbb{R}^{r \times r}$ is nonsingular, and $U \in \mathbb{R}^{m \times r}$ and $V \in \mathbb{R}^{n \times r}$ are column orthonormal. Then the solution to Problem (6) can be written in the following form:

$$
\nabla Y = \nabla USV^T + U\nabla SV^T + US\nabla V^T,
$$

(7)

where

$$
\nabla S = U^T \nabla AV,
$$

$$
\nabla U = P_U^V \nabla AV S^{-1},
$$

$$
\nabla V = P_V^U \nabla A^T US^{-T}.
$$

(8)

The resulting algorithm is a simple gradient descent procedure. That is,

$$
U \leftarrow U + \epsilon \nabla U, \quad S \leftarrow S + \epsilon \nabla S, \quad \text{and} \quad V \leftarrow V + \epsilon \nabla V,
$$

(9)

where $\epsilon$ is stepsize. The algorithm was firstly proposed in Koch and Lubich [2007] and called dynamical low-rank approximation, a landmark in solving SVD in the view of dynamic system. Prior work of solving SVD in dynamic system is about full SVD, without considering truncated SVD.

According to the low rank assumption, the matrix $Y$ does not come up explicitly for ease of computation. Instead, $Y$ is represented via the product of $U$, $S$ and $V^T$ (at order of $O(mr + nr)$), as shown in Eqn. (3). Similarly, the Riemannian gradient $\nabla Y$ can be also represented by using relatively small matrices (at order of $O(mr + nr)$) as shown in Eqn. (7). However, for notational convenience, we use $Y$ and $\nabla Y$ instead of $USV^T$ and $(\nabla U)SV^T + U(\nabla S)V^T + US(\nabla V)^T$, respectively.

Though the extant dynamical low rank approximation is a benchmark work in solving low rank approximation in view of dynamic system, it is not competitive compared with state-of-the-art low rank approximation approaches due to the following issues.

First, this framework fails to exploit geometric information sufficiently. Specifically, on the fixed-rank manifold, from $Y_i = U_i S_i V_i^T$ to $Y_{i+1} = U_{i+1} S_{i+1} V_{i+1}^T$, the algorithm only makes use of the Riemannian gradient of $Y_i$. Second, since the discretization error of the dynamic system is proportional to the stepsize, the stepsize is required to be extremely small, which limits the convergence rate. Third, some extra operations are introduced to ensure the column orthogonality of $U$ and of $V$, which is deemed to be a brutal strategy, often leading to loss of information.

To address these issues, we seek a widely used scheme in differential systems to compute gradient descent. Specifically, we use a splitting integration technique to update three components $(U, S, V)$ step-by-step, making a more sufficient use of geometric information.

### 3 Methodology

In this section we present our method. We first give a novel view of dynamic system for Problem (6). With this view, we use a splitting integration scheme to devise a dynamic flow subspace method for solving Problem (6). Finally, we give the convergence analysis of the method and extend it into full SVD and low rank matrix completion.

#### 3.1 Dynamic System

Departing from a perspective of differential systems, we study the solution of the optimization problem in (6). In particular, we consider an alternative formulation for Problem (6) in the form of a dynamic system:

$$
\arg \min_{Y \in \mathbb{M} \times n} \frac{1}{2} \| \dot{Y} - \dot{A} \|_F^2
$$

(10)

s.t. $\dot{Y} \in T_Y M_r$,
where $Y$ is regarded as a time-dependent matrix such that $Y = Y(t)$ and $\dot{Y}$ denotes the derivative of $Y$ w.r.t. time, and $\dot{A} \triangleq \nabla A = -\nabla f(Y)$ in our case. Notice that it is the continuous version of Problem (6).

According to the Galerkin condition on the tangent space $T_Y M_r$ in numerical analysis Hairer et al. [2006], Problem (10) is equivalent to the following projection:

finding $\dot{Y} \in T_Y M_r$ such that

$$\langle \dot{Y} - \dot{A}, \delta Y \rangle = 0 \text{ for all } \delta Y \in T_Y M_r.$$  

(11)

Furthermore, Problem (11) can be transformed into the following form:

$$\dot{Y} = \tilde{P}_Y (\dot{A}),$$

(12)

where $\tilde{P}_Y(\cdot)$ is a projection operator, defined as

$$\tilde{P}_Y(B) = B - \tilde{P}_Y^\perp(B)$$

with $\tilde{P}_Y^\perp(B) = P^\perp U B P^\perp V \forall B \in \mathbb{R}^{m \times n}$.  

(13)

Substituting Eqn. (13) into Eqn. (12), we have the following dynamic system:

$$\dot{Y} = U U^T \dot{A} - U U^T \dot{A} V V^T + \dot{A} V V^T.$$  

(14)

### 3.2 Dynamic Flow Subspace Method

Our current concern is to solve the differential equation in (14). We resort to a splitting scheme Hairer et al. [2006]. In particular, let $L$ be a local generator Leimkuhler and Matthews [2013] corresponding to the exact solution to Problem (14) and separate it into several sub-generators as follows:

$$L = L_A + L_B + L_O,$$

where

$$L_A : \dot{Y} = U U^T \dot{A},$$

$$L_B : \dot{Y} = -U U^T \dot{A} V V^T,$$

$$L_O : \dot{Y} = \dot{A} V V^T.$$  

(15)

**Theorem 1.** For $Y$ defined in Eqn. (3) and $\dot{Y}$ defined in Eqn. (4), assume that the condition described in Eqn. (5) is satisfied. Then the analytical solution to the sub-generator $L_A$ described in Eqn. (15) is

$$\dot{S} V^T = U^T \dot{A},$$

$$\dot{U} = 0.$$  

The analytical solution to the sub-generator $L_B$ is

$$\dot{S} = -U^T \dot{A} V,$$

$$\dot{U} = 0,$$

$$\dot{V} = 0.$$  

And the analytical solution to the sub-generator $L_O$ is

$$\dot{U} S = \dot{A} V,$$

$$\dot{V} = 0.$$  

Based on Theorem 1, we devise a novel method for Problem (2). The splitting integration scheme also allows us to alternatively update $U$, $V$ and $S$, rather than directly update $Y$. Owing to the Markovian property of the Kolmogorov operator, different orders of sub-generators $L_A$, $L_B$ and $L_O$ are equivalent Leimkuhler and Matthews [2013]. In our work, we restrict our interest on ‘OBA’ scheme: $L_O + L_B + L_A$.

We call our method the splitting numerical integration method. The detail is given in Algorithm 1. Here $\sigma_{\min}(V_{i-1}^T V_i)$ measures the distance between column spaces of $V_{i-1}$ and $V_i$, and it is employed as the stopping criteria.

Compared with dynamical low rank approximation in Eqn. (10) which directly updates $Y$ and the stepsize must be small enough (approaching to 0), our splitting numerical integration method updates the three components ($U$, $S$ and $V$) in a “finer-grained” manner. For instance, more specifically, in Step 7 of Algorithm 1, we adopt the fresh $U_i$ rather than $U_{i-1}$. Notice that in Algorithm 1 the stepsize is implicitly set to 1. In summary, our approach is able to address the issues mentioned in the end of Section 2.1.
### 3.3 Convergence Analysis

In this section we study convergence properties of our splitting numerical integration algorithm.

**Theorem 2.** Let \( \{ Y_i : i = 0, 1, \ldots \} \) denote the sequence generated by Algorithm 1. Then

\[
\| Y_{i-1} - M \|_F > \| Y_i - M \|_F.
\]

Furthermore, it can be also shown that our splitting numerical integration method converges in terms of the subspace estimation.

**Theorem 3.** Let \( \{ U_i : i = 0, 1, \ldots \} \) be the sequence generated by splitting numerical integration in Algorithm 1. Then the subspace error \( \| U_i U_i^T M - M \|_F \) decreases monotonically until convergence. In particular, we have

\[
\| U_{i-1} U_{i-1}^T M - M \|_F > \| U_i U_i^T M - M \|_F.
\]

The similar results hold for \( \{ V_i : i = 0, 1, \ldots \} \). That is,

\[
\| M V_{i-1} V_{i-1}^T M - M \|_F > \| M V_i V_i^T M - M \|_F.
\]

**Algorithm 1** splitting numerical integration

**Input:** matrix \( M \in \mathbb{R}^{m \times n} \), target rank \( r \), initial value \( Y_0 = U_0 S_0 V_0^T \in M_r \), tolerance for stopping criteria \( \tau < 1 \), maximal iteration \( T \).

**Output:** truncated SVD of \( M \).

1: for \( i = 1 : T \) do
2: Compute the gradient \( \hat{A} = \nabla f(X_{i-1}) \). # O(mn) flops
3: \( Q = \hat{A} V_{i-1} \in \mathbb{R}^{m \times r} \) # O(mnr) flops
4: \( K = U_{i-1} S_{i-1} + Q \in \mathbb{R}^{m \times r} \) # O(mr^2) flops
5: Perform QR-factorization to \( K : [U_i, S_{i-1+1/3}] = QR(K) \) # O(mr^2) flops
6: \( S_{i-1+2/3} = S_{i-1+1/3} - U_i^T Q \in \mathbb{R}^{r \times r} \) # O(mr^2) flops
7: \( L = V_{i-1} S_{i-1+2/3} + \hat{A}^T U_i \in \mathbb{R}^{n \times r} \) # O(mnr) flops
8: Perform QR-factorization to \( L : [V_i, S_i] = QR(L) \) # O(mr^2) flops
9: if \( (\sigma_{min}(V_{i-1}^T V_i) > \tau) \) then
10: break.
11: end if
12: end for
13: Perform SVD on the matrix \( S_T = U_s D V_s^T \), then we have that \( M_r = (U_T U_s) D (V_T V_s)^T \).

### 3.4 Application in low-rank matrix completion

The matrix completion problem is to recover a low-rank matrix from a few observations of this matrix. In fixed-rank formulation of matrix completion, we modify Problem 2 into

\[
\arg \min_{Y} \frac{1}{2} \| P_{\Omega}(M) - P_{\Omega}(Y) \|_F^2
\]

s.t. \( Y \in M_r \),

where \( \Omega \) represents the index of observations. Naturally, our splitting numerical integration method applies to this scenario with only a modification that the objective function becomes

\[
f_1(Y) = \frac{1}{2} \| P_{\Omega}(M) - P_{\Omega}(Y) \|_F^2.
\]

Then the convergence property can be extended into the partial observation case. In the following theorem, we prove that the objective function is monotonically decreasing until reaching the convergence condition.

**Theorem 4.** Let \( \{ Y_i : i = 0, 1, \ldots \} \) be the sequence generated by splitting numerical integration in Algorithm 1 under any observation index \( \Omega \). Then splitting numerical integration decreases monotonically in the objective function \( f_1 \) defined in Eqn. (16); that is,

\[
f_1(Y_{i-1}) > f_1(Y_i).
\]
Table 1: Performance of all methods on low rank matrix approximation, DLRA denotes dynamical low rank approximation and RSVD denotes randomized SVD. Note that “(20K,20K,20K)” corresponds to number of row, column and rank, respectively.

| Method   | A:(20K,20K,20K) | B:(20K,20K,2K) | C:(20K,20K,200) |
|----------|-----------------|----------------|-----------------|
| DLRA     | 2.8e-03         | 1.0e-03        | 1.2e-03         |
| RSVD     | 9.3e-02         | 6.0e-02        | 7.2e-02         |
| power    | 2.5e-08         | 4.7e-07        | 7e-08           |
| SNI      | 2.3e-11         | 2.1e-10        | 1.4e-10         |

Table 2: Results of recommendation systems measured in terms of the RMSE. ‘-’ represents the absence of results, which means that corresponding algorithm fails on this task due to memory or running time issue.

| Data set       | Soft-Impute | ALS  | GECO  | LMaFit | RP   | ScGrass | LRGoemCG | SNI   |
|----------------|--------------|------|-------|--------|------|---------|----------|-------|
| Movielens 100K| 0.9026       | 0.9696 | 0.9528 | 1.0821 | 0.9508 | 0.9502  | 0.9643   | 0.9501 |
| Movielens 1M  | 0.9127       | 0.9159 | 0.8601 | 0.8972 | 0.8590 | 0.8723  | 0.8934   | 0.8612 |
| Movielens 10M | 0.8915       | 0.8726 | 0.8241 | 0.8921 | 0.8290 | 0.8991  | 0.8779   | 0.823  |
| Netflix       | 0.9356       | 0.9501 | 0.8738 | 0.9247 | 0.8601 | 0.9232  | 0.8723   | 0.8612 |
| Yahoo Music   | 24.77        | 24.59 | -      | 26.43  | 23.93 | -       | 24.09    | 22.86 |

4 Empirical Evaluation

In this section, we conduct the empirical analysis of the splitting numerical integration method. First we analyze the performance of splitting numerical integration for low rank matrix approximation on simulated datasets. Then we validate the performance of splitting numerical integration for low rank matrix completion on a set of real data [Hoang et al. 2019; Xiao et al. 2020].

4.1 Low rank matrix approximation

We evaluate splitting numerical integration on low rank matrix approximation with comparison with some popular baseline methods. The baseline methods contain the power method [Gu 2015; Fu and others 2021], randomized SVD [Halko et al. 2011] and dynamical low rank approximation [Koch and Lubich 2007]. The primary goal is to illustrate the approximate accuracy on three simulated datasets.

In particular, the three target matrices with different ranks are randomly generated. The error is measured by $\|\mathbf{USV} - \mathbf{M}_r\|_F / \|\mathbf{M}_r\|_F$. Since the complexity of these approaches are all $O(mnr)$, the runtime is similar for all methods and not reported. Each trial is conducted three independent times and average error are reported in Table 1. We observe that our splitting numerical integration method owns obvious advantage over other baseline method in accuracy. In addition, we use a special initialization by letting the column spaces of $\mathbf{U}_0$ and $\mathbf{V}_0$ lie in the orthogonal complements of $\mathbf{U}_r$ and $\mathbf{V}_r$, respectively. The similar accuracy can be obtained, which means that our splitting numerical integration method is insensitive to initialization.

4.2 Low rank matrix completion

We now conduct the empirical analysis of our splitting numerical integration method for the low rank matrix completion (LRMC) problem. To show the efficiency and effectiveness of our splitting numerical integration-LRMC, we compare it with a bunch of baseline methods, including Soft-Impute [Mazumder et al. 2010], ALS (Soft-Impute Alternating Least Squares) [Hastie et al. 2014], GECO (Greedy Efficient Component Optimization) [Shalev-Shwartz et al. 2011], LMaFit (Low Rank Matrix Fitting) [Wen et al. 2010], RP (Riemann Pursuit for matrix recovery) [Tan et al. 2014], ScGrass (Scaled Gradient on Grassmann Manifold) [Ngo and Saad 2012], and LRGeomCG (Low rank Geometric Conjugate Gradient) [Vandereycken 2013]. The codes

Table 3: Running time (in seconds) of all methods on recommendation systems.

| Data set       | Soft-Impute | ALS  | GECO  | LMaFit | RP   | ScGrass | LRGoemCG | SNI   |
|----------------|--------------|------|-------|--------|------|---------|----------|-------|
| Movielens 100K| 2.56         | 0.49 | 2.90  | 0.230  | 0.21 | 0.92    | 0.99    | 0.094 |
| Movielens 1M  | 22.81        | 5.61 | 176.11| 1.412  | 1.00 | 50.11   | 15.23   | 0.94  |
| Movielens 10M | 675.11       | 88.40| > 10^4| 159.80 | 147.34| > 10^4  | 313.30  | 47.93 |
| Netflix       | > 5 × 10^4  | 1189.47| > 10^4| 345.00 | 744.43| > 5 × 10^4| 3823.13 | 350.38|
| Yahoo Music   | > 5 × 10^4  | 8522.23| -    | 1239.56| 1858.43| -       | 4043.22 | 236.32|


of all the methods can be available online, e.g., Soft-Impute and ALS\footnote{http://web.stanford.edu/~hastie/pub.htm}, GECO\footnote{http://www.cs.huji.ac.il/~shais/code/index.html}, LMaFit\footnote{http://lmafit.blogs.rice.edu/}, RP and LRGoemCC\footnote{http://www-users.cs.umn.edu/~thango/}, and ScGrass\footnote{http://www-users.cs.umn.edu/~thango/}. These algorithms have been proved to be state-of-the-art algorithms in low rank matrix completion.

We compare these methods on several popular recommendation systems. It is worth mentioning that large-scale recommendation systems (say, Yahoo Music and Netflix) are used to evaluate the scalability of our method. We use five publicly available datasets: Movielens 100K, 1M, 10M, NetFlix, Yahoo Music Track 1 to evaluate both the effectiveness and efficiency of our method.

Testing error in terms of RMSE (Root-Mean-Square Error) and computational efficiency measured by running time are shown in Table\ref{tab:RMSE} and Table\ref{tab:time} respectively. From Table\ref{tab:RMSE} we can observe that our method can achieve better performance than most of the baseline methods in terms of RMSE\footnote{http://web.stanford.edu/~hastie/pub.htm}. Whilst Table\ref{tab:time} shows that our method can achieve great speedup compared with almost all baseline methods under the same setting. It is worth mentioning that in large scale tasks such as Yahoo Music dataset, some results are not listed, which means that the corresponding algorithm can not handle these cases in limited time or simply fail in these situations.

References

Petros Drineas, Michael W Mahoney, and S Muthukrishnan. Subspace sampling and relative-error matrix approximation: Column-based methods. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 316–326. Springer, 2006.

Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

Tianfan Fu et al. $\alpha$-mop: Molecule optimization with $\alpha$-divergence. In *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 240–244. IEEE, 2020.

Tianfan Fu et al. Probabilistic and dynamic molecule-disease interaction modeling for drug discovery. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 404–414, 2021.

Tian Gao, Tengfei Ma, and Jimeng Sun. Pearl: Prototype learning via rule learning. In *Proceedings of the 10th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics*, pages 223–232, 2019.

Gene H Golub and Charles F Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.

Ming Gu. Subspace iteration randomization and singular value problems. *SIAM Journal on Scientific Computing*, 37(3):A1139–A1173, 2015.

Ernst Hairer, Marlis Hochbruck, Arieh Iserles, and Christian Lubich. Geometric numerical integration. *Oberwolfach Reports*, 3(1):805–882, 2006.

Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

Trevor Hastie, Rahul Mazumder, Jason Lee, and Reza Zadeh. Matrix completion and low-rank svd via fast alternating least squares. *arXiv preprint arXiv:1410.2596*, 2014.

Trong Nghia Hoang, Cao Xiao, and Jimeng Sun. Ddl: Deep dictionary learning for predictive phenotyping. In *IJCAI: proceedings of the conference*, volume 2019, page 5857, 2019.

Kexin Huang et al. Deeppurpose: A deep learning library for drug-target interaction prediction. *Bioinformatics*, 2020.

Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

Othmar Koch and Christian Lubich. Dynamical low-rank approximation. *SIAM Journal on Matrix Analysis and Applications*, 29(2):434–454, 2007.

Benedict Leimkuhler and Charles Matthews. Rational construction of stochastic numerical methods for molecular sampling. *Applied Mathematics Research eXpress*, 2013(1):34–56, 2013.

Rahul Mazumder, Trevor Hastie, and Robert Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *The Journal of Machine Learning Research*, 11:2287–2322, 2010.
Neil Muller, Lourenço Magaia, and Ben M. Herbst. Singular value decomposition, eigenfaces, and 3D reconstructions. *SIAM Review*, 46(3):518–545, 2004.

Thi N. Ngo and Yousef Saad. Scaled gradients on Grassmann manifolds for matrix completion. In *Advances in Neural Information Processing Systems*, pages 1412–1420, 2012.

Shai Shalev-Shwartz, Alon Gonen, and Ohad Shamir. Large-scale convex minimization with a low-rank constraint. *arXiv preprint arXiv:1106.1622*, 2011.

Mingkui Tan, Ivor W. Tsang, Li Wang, Bart Vandereycken, and Sinno J. Pan. Riemannian pursuit for big matrix recovery. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1539–1547, 2014.

Bart Vandereycken. Low-rank matrix completion by Riemannian optimization. *SIAM Journal on Optimization*, 23(2):1214–1236, 2013.

Z Wen, W. Yin, and Y. Zhang. Low-rank factorization model for matrix completion by a non-linear successive over-relaxation algorithm. Technical report, Rice CAAM Tech Report 10-07, University of Rice, 2010.

Cao Xiao, Jimeng Sun, and Xinhao Li. MIMOSA: Multi-constraint molecule sampling for molecule optimization. *the Association for the Advancement of Artificial Intelligence*, 2020.
This figure "acoustic_power.png" is available in "png" format from:

http://arxiv.org/ps/2202.06482v1
This figure "mnist_power.png" is available in "png" format from:

http://arxiv.org/ps/2202.06482v1