In-depth Analysis of Low-rank Matrix Factorisation in a Federated Setting

Constantin Philippenko, Kevin Scaman, Laurent Massoulié

Inria Paris - Département d'informatique de l'ENS, PSL Research University
firstname.lastname@inria.fr

Abstract

We analyze a distributed algorithm to compute a low-rank matrix factorization on N clients, each holding a local dataset $\mathbf{S}^i \in \mathbb{R}^{n_i \times d}$, mathematically, we seek to solve $min_{\mathbf{U}^i \in \mathbb{R}^{n_i \times r}, \mathbf{V} \in \mathbb{R}^{d \times r}} \frac{1}{2} \sum_{i=1}^N \|\mathbf{S}^i - \mathbf{U}^i \mathbf{V}^\top\|_{\mathrm{F}}^2$. Considering a power initialization of $\mathbf{V}$, we rewrite the previous smooth non-convex problem into a smooth strongly-convex problem that we solve using a parallel Nesterov gradient descent potentially requiring a single step of communication at the initialization step. For any client i in $\{1, \dots, N\}$, we obtain a global $\mathbf{V}$ in $\mathbb{R}^{d \times r}$ common to all clients and a local variable $\mathbf{U}^i$ in $\mathbb{R}^{n_i \times r}$. We provide a linear rate of convergence of the excess loss which depends on $\sigma_{\max}/\sigma_r$, where σ_r is the r^{th} singular value of the concatenation $\mathbf{S}$ of the matrices $(\mathbf{S}^i)_{i=1}^N$. This result improves the rates of convergence given in the literature, which depend on $\sigma_{\max}^2/\sigma_{\min}^2$. We provide an upper bound on the Frobenius-norm error of reconstruction under the power initialization strategy. We complete our analysis with experiments on both synthetic and real data.

Code — https://github.com/philipco/matrix_factorization

1 Notation

For $\mathbf{A}$ a matrix in $\mathbb{R}^{n \times d}$, we note $\mathrm{rank}\,(\mathbf{A})$ its rank, $\lambda_1(\mathbf{A}) \geq \cdots \geq \lambda_{\mathrm{rank}(\mathbf{A})}(\mathbf{A})$ are its decreasing eigenvalues, $\sigma_1(\mathbf{A}) \geq \cdots \geq \sigma_{\mathrm{rank}(\mathbf{A})}(\mathbf{A}) \geq 0$ are its positive and decreasing singular values. More specifically, we note $\lambda_{\max}(\mathbf{A}), \lambda_{\min}(\mathbf{A})$ (resp. $\sigma_{\max}(\mathbf{A}), \sigma_{\min}(\mathbf{A})$) the biggest/smallest eigenvalue (resp. singular value) of $\mathbf{A}$. We define the condition number of $\mathbf{A}$ as $\kappa(\mathbf{A}) := \sigma_{\max}(\mathbf{A})/\sigma_{\min}(\mathbf{A})$. Furthermore, we note $\|\mathbf{A}\|_{\mathrm{F}}^2 := \sqrt{\mathrm{Tr}\,(\mathbf{A}\mathbf{A}^\top)} = (\sum_{i=1}^{\min\{n,d\}} \sigma_i^2(\mathbf{A}))^{1/2}$ the Frobenius norm and $\|\mathbf{A}\|_2 := \sqrt{\lambda_{\max}(\mathbf{A}^\top \mathbf{A})} = \sigma_{\max}(\mathbf{A})$ the operator norm induced by the 2-norm. The group of orthogonal matrices is denoted $\mathcal{O}_d(\mathbb{R})$. For $a, b \in \mathbb{R}$, we also denote $a \wedge b := \min(a, b)$.

2 Introduction

The problem of low-rank matrix factorization is widely analyzed in machine learning (e.g. Deshpande and Vempala 2006; Achlioptas and McSherry 2007; Liberty et al. 2007; Nguyen, Do, and Tran 2009; Rokhlin, Szlam, and Tygert 2010; Halko, Martinsson, and Tropp 2011; Witten and Candes 2015; Tropp et al. 2019; Tropp and Webber 2023). Indeed, several key challenges can be reduced to it, for instance: clustering (Li and Ding 2006), features learning (Bisot et al. 2017), dictionary learning (Mensch et al. 2016), anomaly detection (Tong and Lin 2011), denoising (Wilson et al. 2008), or matrix completion (Jain, Netrapalli, and Sanghavi 2013). Let $\mathbf{S}$ in $\mathbb{R}^{n \times d}$ and $r \in \mathbb{N}^*$, the study of the low-rank matrix factorization (MF) problem corresponds to find $\mathbf{U} \in \mathbb{R}^{n \times r}$ and $\mathbf{V} \in \mathbb{R}^{d \times r}$ minimizing:

$$\frac{1}{2} \left\|\mathbf{S} - \mathbf{U}\mathbf{V}^\top\right\|_{\mathrm{F}}^2 := F(\mathbf{U}, \mathbf{V}) \leq \epsilon_{\min} + \epsilon\,, \qquad \text{(MF)}$$

where $\epsilon_{\min}$ is the minimal achievable error w.r.t. the Forbenius-norm, ϵ is the error induced by the algorithm, and r denote the *latent dimension*. The Eckart and Young (1936) theorem shows that $\sum_{i>r} \sigma_i^2$ (we omit to indicate the matrix $\mathbf{S}$ for its eigen/singular values) is the minimal Frobenius-norm error when approximating $\mathbf{S}$ with a rank-r matrix, and it is σ_{r+1} for the 2-norm (Mirsky 1960).

Proposition 1. *We have for the Frobenius-norm* $\min_{\mathbf{U},\mathbf{V} \in \mathbb{R}^{n \times r} \times \mathbb{R}^{d \times r}} \left\|\mathbf{S} - \mathbf{U}\mathbf{V}^\top\right\|_{\mathrm{F}}^2 = \sum_{i>r} \sigma_i^2$ *and for the* 2*-norm* $\min_{\mathbf{U},\mathbf{V} \in \mathbb{R}^{n \times r} \times \mathbb{R}^{d \times r}} \|\mathbf{S} - \mathbf{U}\mathbf{V}^\top\|_2 = \sigma_{r+1}$.

Proposition 1 corresponds to the thin SVD decomposition (Eckart and Young 1936) and to keeping the r largest singular values, the error is thus determined by the $n \wedge d - r$ smallest singular values of the spectrum of $\mathbf{S}$.

Contemporary machine learning problems frequently involve data spread across multiple clients, each holding a subset of the dataset, thereby introducing an additional layer of complexity. In this paper, we consider the federated setting (Konečný et al. 2016; McMahan et al. 2017) where a client i holds a dataset $\mathbf{S}^i$ in $\mathbb{R}^{n_i \times d}$ with n_i in $\mathbb{N}$ rows and d constant features across the N in $\mathbb{N}$ clients. We want to factorize for any client i its dataset based on a global shared variable $\mathbf{V}$ in $\mathbb{R}^{d \times r}$ and a local personalized variable $\mathbf{U}^i$ in $\mathbb{R}^{n_i \times r}$, Equation (MF) can thus be rewritten as minimizing:

$$\frac{1}{2} \sum_{i=1}^N \left\|\mathbf{S}^i - \mathbf{U}^i \mathbf{V}^\top\right\|_{\mathrm{F}}^2 := \sum_{i=1}^N F^i(\mathbf{U}^i, \mathbf{V})\,. \qquad \text{(Dist. MF)}$$

We note $\mathbf{S}$ and $\mathbf{U}$ the vertical concatenation of matrices $(\mathbf{S}^i, \mathbf{U}^i)_{i=1}^N$. We write the SVD of $\mathbf{S} = \mathbf{U}_* \mathbf{\Sigma} \mathbf{V}_*^\top$ where

$\mathbf{U}_*$ in $\mathbb{R}^{n\times n}$ ($n = \sum_{i=1}^N n_i$) is the left basis vectors s.t. $\mathbf{U}_*^\top \mathbf{U}_* = \mathbf{I}_r$, $\mathbf{V}_*$ in $\mathbb{R}^{d\times d}$ is the right basis vectors s.t. $\mathbf{V}_*^\top \mathbf{V}_* = \mathbf{I}_r$, and $\mathbf{\Sigma} = \mathrm{Diag}\left(\lambda_1(\mathbf{S}), \ldots, \lambda_{n\wedge d}(\mathbf{S})\right)$ in $\mathbb{R}^{n\times d}$ contains the singular value of $\mathbf{S}$. We consider that there exists a true low-rank matrix $\mathbf{X}$ in $\mathbb{R}^{n\times d}$ and a white noise $\mathbf{E}$ in $\mathbb{R}^{n\times d}$ s.t. $\mathbf{S} = \mathbf{X} + \mathbf{E}$, which implies $r_* = \mathrm{rank}\,(\mathbf{X}) \leq \mathrm{rank}\,(\mathbf{S}) \leq \min(n, d)$. The r_* first eigenvalues of $\mathbf{S}$ correspond to the signal (potentially noised if $\mathbf{E} \neq 0$) and the $n \wedge d - r_*$ last correspond to the white noise from $\mathbf{E}$.

Remark 1. *In distributed settings, the errors $\epsilon_{\min}$ and ϵ are determined by the spectrum of $\mathbf{S}$, and not by the spectrum of clients' matrices $(\mathbf{S}_i)_{i=1}^N$.*

Our work progresses in two directions: (1) offering new theoretical insights on matrix factorization and (2) developing a robust algorithm suited for federated environments, the challenge being to design a distributed algorithm that computes a low-rank matrix factorization of $\mathbf{S}$ with minimal communication/computation overhead, while guaranteeing a linear convergence towards the reconstruction error $\epsilon_{\min} + \epsilon$. To tackle this challenge, we combine a global distributed randomized power iteration (Halko, Martinsson, and Tropp 2011) with local gradient descents on each client, bridging these two lines of research.

The randomized power iteration method (Halko, Martinsson, and Tropp 2011; Hardt and Price 2014; Li et al. 2021a; Saha, Srivastava, and Pilanci 2023) is one of the main approaches used in practice[1] to compute low-rank matrix factorizations, therefore. Theoretically, the authors have provided in a *centralized* setting, solely *if taking $r \leq r_* + 2$*, a bound on the 2-*norm for α in $\mathbb{N}$ proportional to r*. In this work, we are interested in giving *new results in the **federated setting** on the **Frobenius-norm** and r **in** $\{1, \ldots, n_i \wedge d\}$*. Indeed, using the existing results on the 2-norm to derive results on the Frobenius norm would result in an undesirable $\sqrt{d}$-factor to both $\epsilon_{\min}$ and ϵ. Thus, the results on the two different norms are not directly comparable and ask for a new analysis.

On the other hand, literature based on gradient descent (Jain et al. 2017; Zhu et al. 2019; Ye and Du 2021; Jain, Netrapalli, and Sanghavi 2013; Ward and Kolda 2023; Gu et al. 2023) usually provides results on the Frobenius norm with linear convergence rates depending on $\kappa^2(\mathbf{S})$ (Ward and Kolda 2023), which might be arbitrarily large, thus hindering the convergence. The primary challenge with this approach lies in the non-convexity of F, rendering the solution space for Equation (MF) infinite and impeding its theoretical analysis. Numerous research endeavors are directed towards these algorithms with the aims of (1) enhancing the theoretical convergence rate, (2) writing elegant and concise proofs, and (3) understand the underlying mechanism that allows for convergence. More generally, advancements in understanding Equation (MF) can significantly enrich our comprehension of other non-convex problems such as matrix sensing or deep learning optimization (Du, Hu, and Lee 2018).

[1]For instance, the `TruncatedSVD` class of Scikit-learn (Pedregosa et al. 2011) or the `svd_lowrank` function of PyTorch (Paszke et al. 2019) are based on the work of Halko, Martinsson, and Tropp (2011).

In Section 3, we conduct a short overview of the related work. In Section 4, we propose and analyze a parallel algorithm derived from Halko, Martinsson, and Tropp (2011), we explain the associated computational/communicational trade-offs, exhibit a linear rate of convergence, and provide an upper bound in probability of both the condition number $\kappa(\mathbf{S})$ and the Frobenius-norm error. In Section 5 we illustrate our theoretical findings with experiments on both synthetic and real data, and finally, in Section 6, we conclude by highlighting the main takeaways of this article and by listing a few open directions.

Contributions. We make the following contributions.

- We solve the low-rank matrix factorization problem with a local personalized variable $\mathbf{U}^i$ in $\mathbb{R}^{n_i\times r}$ and a global shared $\mathbf{V}$ in $\mathbb{R}^{r\times r}$ s.t. $\mathbf{S}^i \approx \mathbf{U}^i\mathbf{V}^\top$ for any client i in $\{1, \ldots, N\}$. Following Halko, Martinsson, and Tropp (2011), we use a power method to initialize $\mathbf{V}$, then we run a gradient descent on each client to compute $\mathbf{U}^i$.
- We provide a novel result regarding the Frobenius-norm of $\mathbf{S} - \mathbf{U}\mathbf{V}^\top$. Our result is non-trivial as it was lacking in Halko, Martinsson, and Tropp (2011, see Remark 10.1). This result provides new insight on Equation (MF) and can not be compared for $\alpha \geq 1$ to Halko, Martinsson, and Tropp (2011); Hardt and Price (2014) or Li et al. (2021a) which analyze different quantities.
- In the distributed setting, under low noise conditions, unlike Li et al. (2021a), we can achieve a finite number of communications in $\Omega\left(\frac{\log(\sigma_{\max} d r_*^2 \epsilon^{-1})}{\log(\sigma_{r_*}) - \log(\sigma_{r_*+1})}\right)$. And potentially a single communication stage is enough, i.e., our algorithm is possibly *embarrassingly parallel*.
- Algorithmically, our method involves sampling different random Gaussian matrices Φ to obtain better condition numbers, this sampling allows rapid gradient descent, independently of $\sigma_{\min}$. It increases the average number of exchanges by a factor m, yet it ensures convergence almost surely. We are the first to propose such a result for (MF) problems.
- Compared to existing literature on (distributed) gradient descent (e.g. Zhu et al. 2019; Ward and Kolda 2023) our approach surpasses them both in terms of communicational and computational complexity. Our guarantee of convergence are stronger and our analysis bridges the gap between works on distributed gradient descent and on the power method. Besides, our proof of convergence is simpler as it simply requires to show that the problem is smooth and strongly-convex.
- All the theory from strongly-convex gradient descent apply to our problem. Thereby, we easily introduce acceleration – being the first to do so – outperforming the convergence rates for simple gradient descent.

3 Related Works

In this Section, we describe the distributed randomized power iteration and review related works on gradient descent.

Algorithm 1: Distributed Randomized Power Iteration

Input: Number of iteration α in $\mathbb{N}$.
Output: Global variable $\mathbf{V}$ in $\mathbb{R}^{d\times r}$
for *each client* i *in* $\{1,\dots,N\}$ **do**
  Generate a Gaussian matrix Φ^i in $\mathbb{R}^{n_i\times r}$.
  Compute $\mathbf{V}^i = (\mathbf{S}^i)^\top \Phi^i$.
  Share $\mathbf{V}^i$.
Compute $\mathbf{V} = \sum_{i=1}^N \mathbf{V}^i$ using a secure aggregation protocol.
Share $\mathbf{V}$ with all clients.
for $a \in \{1,\dots,\alpha\}$ **do**
  for *each client* i *in* $\{1,\dots,N\}$ **do**
    Compute $\mathbf{V}^i = (\mathbf{S}^i)^\top \mathbf{S}^i \mathbf{V}$.
    Share $\mathbf{V}^i$.
  Compute $\mathbf{V} = \sum_{i=1}^N \mathbf{V}^i$ using a secure aggregation protocol.
  Share $\mathbf{V}$ with all clients.

The power method (Mises and Pollaczek-Geiringer 1929) is a simple iterative algorithm used to approximate the dominant eigenvalue and the corresponding eigenvector of a square matrix by repeatedly multiplying the matrix by a vector and normalizing the result. To improve scalability to large problems, Halko, Martinsson, and Tropp (2011) have proposed *Randomized SVD* a technique that sample a random Gaussian matrix Φ in $\mathbb{R}^{n\times r}$, multiply it by $(\mathbf{S}^\top\mathbf{S})^\alpha\mathbf{S}^\top$ (α in $\mathbb{N}$) to obtain a matrix $\mathbf{V}$. While authors never mention the distributed setting, the adaptation is straightforward and we give the pseudo-code to compute $\mathbf{V}$ in Algorithm 1. Note that to avoid storing a $d\times d$ matrix, we compute the product $(\mathbf{S}^\top\mathbf{S})^\alpha\mathbf{S}^\top\Phi$ from right to left, resulting in the computation of a $d\times r$ matrix. The total number of computational operations is $(2\alpha+1)ndr + (\alpha+1)dr$. Next, the authors either construct a QR factorization of $\mathbf{V}$ to obtain an orthonormal matrix factorization of $\mathbf{S}$, or compute its SVD decomposition to get the singular values.

Remark 2 (Secure aggregation)**.** *The distributed power iteration (Algorithm 1) requires a federated setting with a central server that can perform a* secure aggregation *(Bonawitz et al. 2017) of the local* $(\mathbf{V}^i)_{\{1,\dots,N\}}$*. Extending to the decentralized setting is an interesting but out of scope direction.*

The idea of using gradient descent to solve low-rank matrix factorization can be traced back to Burer and Monteiro (2003, 2005). Several theoretical results and convergence rate have been provided in the recent years (Zhao, Wang, and Liu 2015; Tu et al. 2016; Zheng and Lafferty 2016; Jain et al. 2017; Chen et al. 2019; Du, Hu, and Lee 2018; Ye and Du 2021; Jiang, Chen, and Ding 2023). Ward and Kolda (2023) has a rate depending on $\kappa^{-2}(\mathbf{S})$ using a power initialization with $\alpha = 0$ but only in the case of a low-rank matrix, for which it is known that we can have in fact a zero error (Proposition 1). In contrast, our analysis in a strongly-convex setting allows, first, to plugging in any faster algorithms than simple gradient descent and thus to obtain a faster rate depending on κ^{-1}, and second, holds in the more general setting of a full-rank matrix.

Note that Ward and Kolda (2023) do not mention the FL setting, however, their work naturally adapts itself to this setting as we have $\frac{1}{2}\sum_{i=1}^N \left\|\mathbf{S}^i - \mathbf{U}^i\mathbf{V}^\top\right\|_{\mathrm{F}}^2 = \frac{1}{2}\|\mathbf{S}-\mathbf{U}\mathbf{V}^\top\|_{\mathrm{F}}^2$. On the other hand, it results in a high communication cost, given that it requires sharing at each iteration k in $\mathbb{N}$ the matrix $\mathbf{V}_k$ to compute the gradient (but does not require sharing $\mathbf{U}_k$ which remains local), resulting in a communication cost of $O(Krd)$, where K is the total number of iterations/communications. Besides, K depends on the condition number and thereby might be very large. On the contrary, our work and the one of Halko, Martinsson, and Tropp (2011) requires small (and potentially a single) communication steps.

Most of the articles extending gradient descent to the distributed setting (Hegedűs et al. 2016; Zhu et al. 2019; Hegedűs, Danner, and Jelasity 2019; Li et al. 2020, 2021b; Wang and Chang 2022) consider an approach minimizing the problem over a global variable $\mathbf{V}$ and personalized ones $(\mathbf{U}_i)_{i=1}^N$. This setup creates a global-local matrix factorization: $\mathbf{V}$ contains information on features shared by all clients (item embedding), while $(\hat{\mathbf{U}}^i)_{i=1}^N$ captures the unique characteristics of client i in $\{1,\dots,N\}$ (user embeddings). We build on this approach in the next Section.

4 Practical Algorithm and Theoretical Analysis

In the next Subsections, we propose and analyze a parallel algorithm derived from Halko, Martinsson, and Tropp (2011) that combines power method and gradient descent.

4.1 Combining Power Method with Parallel Gradient Descent

In order to solve (Dist. MF), a natural idea is to fix the matrix $\mathbf{V}$ shared by all clients, and then compute locally the exact solution of the least-squares problem or run a gradient descent *with respect to the variable* $\mathbf{U}$. All computation are thus performed solely on clients once $\mathbf{V}$ is initialized. This requires a number of communications equal to $\alpha+1$ as the only communication steps occur when initializing $\mathbf{V}$. In the low-noise or the low-rank regimes, taking $\alpha = 0$ allows to obtain $\epsilon = 0$ (Corollary 1) and results to an *embarrassingly parallel* algorithm. Besides, parallelizing the computation allows to go from a computational cost in $O(dr\sum_{i=1}^N n_i)$ to $O(drn_i)$ which is potentially much smaller. Below, we give the optimal solution of (Dist. MF) for a fixed $\mathbf{V}$.

Proposition 2 (Optimal solution for a fixed matrix $\mathbf{V}$)**.** *Let* $\mathbf{V} \in \mathbb{R}^{d\times r}$ *be a* fixed *matrix, then Equation* (Dist. MF) *is minimized for* $\hat{\mathbf{U}}^i = \mathbf{S}^i\mathbf{V}(\mathbf{V}^\top\mathbf{V})^\dagger$.

The main challenge in computing $(\hat{\mathbf{U}}^i)_{i=1}^N$ is to (pseudo-)inverse $\mathbf{V}^\top\mathbf{V}$, which is known to be unstable under small numerical errors and potentially slow. We propose instead to do a local gradient descent on each client i in $\{1,\dots,N\}$ in

Algorithm 2: GD w.r.t. $\mathbf{U}$ with a power init.

Input: Number of iteration α in $\mathbb{N}$, step-size γ.
Output: $(U^i)_{i=1}^N$.
Run Algorithm 1 to compute $\mathbf{V} = (\mathbf{S}^\top\mathbf{S})^\alpha\mathbf{S}^\top\Phi$.
for *each client i in $\{1,\dots,N\}$ without any communication* **do**
  Sample a random matrix $\mathbf{U}_0^i$ in $\mathbb{R}^{n_i\times r}$.
  for $t \in \{1,\dots,T\}$ **do**
    Compute $\nabla_{\mathbf{U}}F(\mathbf{U}_{t-1}^i,\mathbf{V}) = (\mathbf{U}_{t-1}^i\mathbf{V}^\top - \mathbf{S}^i)\mathbf{V}$.
    $\mathbf{U}_t^i = \mathbf{U}_{t-1}^i - \gamma\nabla_{\mathbf{U}}F(\mathbf{U}_{t-1}^i,\mathbf{V})$.

order to approximate the optimal $\hat{\mathbf{U}}^i$ minimizing $F^i(\cdot,\mathbf{V})$. It results to a parallel algorithm that does not require any communication after the initialization of $\mathbf{V}$. We give in Algorithm 2 the pseudo-code: the server requires to do exactly $N(\alpha+1)dr + dr^2$ computational operations and the client $i \in \{1,\dots,N\}$ carries out $(4T+2\alpha+2)n_idr$ operations.

This approach offers much more numerical stability, allows for any kind of regularization, ensures strong guarantees of convergence, and explicit the exact number of epochs required to reach a given accuracy. A simple analysis of gradient descent in a smooth strongly-convex setting leads to a linear convergence toward a global minimum of the function $F^i(\cdot,\mathbf{V})$ with a convergence rate equal to $1-\kappa^{-2}(\mathbf{V})$, or even to $1-\kappa^{-1}(\mathbf{V})$ if a momentum is added.

Remark 3 ($L_*/L_1/L_2$-regularization.)**.** *We can use regularization on $\mathbf{U}$ with various norms: nuclear norm which yields a low-rank $\mathbf{U}$, L_1-norm resulting in a sparse $\mathbf{U}$, and L_2-norm leading to small values. The nuclear regularization requires to compute the SVD of $\mathbf{U}$ to compute the gradient (Avron et al. 2012), which generates an additional $O(nr\log r)$ complexity.*

4.2 Rate of Convergence of Algorithm 2

The cornerstone of the analysis relies on using power initialization (Algorithm 1), which forces having $\mathbf{V}$ in the column span of $\mathbf{S}$. Besides, once $\mathbf{V}$ is set by Algorithm 1, for any client i in $\{1,\dots,N\}$, $F^i(\cdot,\mathbf{V})$ is L-smooth and μ-strongly-convex as proved in the following properties.

Property 1 (Smoothness)**.** *Let $\mathbf{V}$ in $\mathbb{R}^{d\times r}$ initialized by Algorithm 1, then all $\left(F^i(\cdot,\mathbf{V})\right)_{i=1}^N$ are L-smooth, i.e., for any i in $\{1,\dots,N\}$, for any $\mathbf{U},\mathbf{U}'$ in $\mathbb{R}^{n_i\times d}$, we have $\|\nabla F^i(\mathbf{U},\mathbf{V}) - \nabla F^i(\mathbf{U}',\mathbf{V})\|_{\mathrm{F}} \leq L\|\mathbf{U}-\mathbf{U}'\|_{\mathrm{F}}$, with $L = \sigma_{\max}^2(\mathbf{V})$.*

Proof. Let i in $\{1,\dots,N\}$ and $\mathbf{U},\mathbf{U}'$ in $\mathbb{R}^{n_i\times r}$, we have that:

$$\begin{aligned}\|\nabla F(\mathbf{U},\mathbf{V}) - \nabla F(\mathbf{U}',\mathbf{V})\|_{\mathrm{F}} &= \|(\mathbf{U}-\mathbf{U}')\mathbf{V}^\top\mathbf{V}\|_{\mathrm{F}} \\ &\overset{\text{Prop. S2}}{\leq} \sigma_{\max}(\mathbf{V}^\top\mathbf{V})\|\mathbf{U}-\mathbf{U}'\|_{\mathrm{F}} \\ &= \sigma_{\max}^2(\mathbf{V})\|\mathbf{U}-\mathbf{U}'\|_{\mathrm{F}}\,.\,\square\end{aligned}$$

Property 2 (Strongly-convex)**.** *Let $\mathbf{V}$ in $\mathbb{R}^{d\times r}$ initialized by Algorithm 1, then all $\left(F^i(\cdot,\mathbf{V})\right)_{i=1}^N$ are μ-strongly-convex, i.e., for any i in $\{1,\dots,N\}$, for any $\mathbf{U},\mathbf{U}'$ in $\mathbb{R}^{n_i\times d}$, we have $\langle\nabla F^i(\mathbf{U},\mathbf{V}) - \nabla F^i(\mathbf{U}',\mathbf{V}), \mathbf{U}-\mathbf{U}'\rangle \geq \mu\|\mathbf{U}-\mathbf{U}'\|_{\mathrm{F}}^2$, with $\mu = \sigma_{\min}^2(\mathbf{V})$.*

Proof. Let i in $\{1,\dots,N\}$ and $\mathbf{U},\mathbf{U}'$ in $\mathbb{R}^{n_i\times r}$, we have that:

$$\begin{aligned}&\langle\nabla F^i(\mathbf{U},\mathbf{V}) - \nabla F^i(\mathbf{U}',\mathbf{V}), \mathbf{U}-\mathbf{U}'\rangle \\ &\quad= \mathrm{Tr}\left((\mathbf{U}-\mathbf{U}')^\top(\mathbf{U}-\mathbf{U}')\mathbf{V}^\top\mathbf{V}\right) \\ &\quad= \|(\mathbf{U}-\mathbf{U}')\mathbf{V}^\top\|_{\mathrm{F}}^2 \overset{\text{Prop. S2}}{\geq} \sigma_{\min}^2(\mathbf{V})\|\mathbf{U}-\mathbf{U}'\|_{\mathrm{F}}^2\,.\,\square\end{aligned}$$

This proves that the surrogate objective functions $(F^i(\cdot,\mathbf{V}))_{i=1}^N$ are μ-strongly-convex, with $\mu \geq 0$. As discussed in Theorem 2, as long as $r < \mathrm{rank}\,(\mathbf{S})$, we have with high probability $\mu > 0$. This is always true in the noisy setting (full rank); otherwise, we simply need to decrease r.

Properties 1 and 2 allows to use classical results in optimization and draws a linear rate of convergence depending on μ/L, or $\sqrt{\mu/L}$ if we use acceleration.

Theorem 1. *Under the distributed power initialization (Algorithm 1), considering Properties 1 and 2, let T in $\mathbb{N}^*$, $\gamma = 1/L$, then after running Algorithm 2 for T iterations, the excess loss function is upper bounded: $F^i(\mathbf{U}_T^i) - F^i(\hat{\mathbf{U}}^i) \leq (1-\mu/L)^T\left(F^i(\mathbf{U}_0^i) - F^i(\hat{\mathbf{U}}^i)\right)$, where $\mu/L = \kappa^{-2}(\mathbf{V})$. Using Nesterov momentum, this rate is accelerated to: $F^i(\mathbf{U}_T^i) - F^i(\hat{\mathbf{U}}^i) \leq 2(1-\sqrt{\mu/L})^T(F^i(\mathbf{U}_0^i) - F^i(\hat{\mathbf{U}}^i))$.*

Remark 4 (Convergence in a single iteration)**.** *If we orthogonalize $\mathbf{V}$, then we can converge in one iteration as gradient descent reduces to Newton method. Our algorithm would be equivalent to the one proposed by Halko, Martinsson, and Tropp (2011). However, orthogonalizing $\mathbf{V}$ requires to compute the SVD or to run a gradient descent on $\mathbf{V}$.*

Theorem 1 establishes that $(F^i(\mathbf{U}_t^i,\mathbf{V}))_{t\in\mathbb{N}^*}$ converges to $F^i(\hat{\mathbf{U}}^i,\mathbf{V})$ at a linear rate dominated by $\exp(-\kappa^{-1}(\mathbf{V}))$ or $\exp(-\kappa^{-2}(\mathbf{V}))$. A question then emerges, *can we control $\kappa(\mathbf{V})$?* This parameter plays a pivotal role in determining the convergence rate and is affected by the sampled matrix Φ. Consequently, an ill-conditioned matrix $\mathbf{V}$ may arise, significantly hindering convergence. The below corollary gives a bound in probability on $\kappa^2(\mathbf{V})$ that depends on the spectrum of $\mathbf{S}$ while being independent of the sampled $(\Phi^i)_{i=1}^N$.

Theorem 2. *Under the distributed power initialization (Algorithm 1), considering Properties 1 and 2, for any p in $]0,1[$, with probability at least $1-3\mathrm{p}$, we have $\kappa(\mathbf{V})^2 < \kappa_{\mathrm{p}}^2$, with:*

$$\kappa_{\mathrm{p}}^2 := \frac{1}{\mathrm{p}^2}\left(9r^2\frac{\sigma_{\max}^{2(2\alpha+1)}}{\sigma_r^{2(2\alpha+1)}} + 4r\left(d+\log(2\mathrm{p}^{-1})\right)\frac{\sigma_{r+1}^{2\alpha}}{\sigma_r^{2\alpha}}\right).$$

With probability P, if we sample $m = \lfloor -\log_2(1-\mathrm{P})\rfloor$ independent matrices $(\Phi_j)_{j=1}^m$ to form $\mathbf{V}_j = \mathbf{S}^\alpha\Phi_j$ and run Algorithm 2, at least one initialization results to a convergence rate upper bounded by $1-\kappa_{1/6}^{-2}$.

We can make the following remarks.

- **Impact of α.** Increasing α enworse the rate of convergence as $\sigma_{\max}^2/\sigma_r^2 \geq 1$. In the regime where $\sigma_{\max}^2$ is very large and $\sigma_r^2 = \Omega(1)$, having $\alpha > 0$ drastically *hinders* the gradient descent convergence. However, in this case, it is possible to compute the exact solution of (Dist. MF), further, as emphasized in Corollary 1 which showcases the interest of having $\alpha \neq 0$, increasing α allows reducing ϵ, i.e., the gap between the approximation error (induced by taking $r \leq \operatorname{rank}(\mathbf{S})$) and the minimal reconstruction error $\epsilon_{\min}$. Therefore, there is a trade-off associated with the choice of α; we illustrate it in the experiments on three real datasets: mnist, w8a and celeba-200k.
- **Asymptotic values of $\kappa(\mathbf{V})$.** In the regime $\sigma_{\max}, \sigma_r = O(1)$ and $\sigma_{r+1} \ll 1$, we have $1 - \kappa_{\mathrm{p}}^{-2} = O(1 - p^2/\mathrm{r}^2)$. Further, by employing acceleration techniques, we have an improved rate depending on r^{-1} and not r^{-2}.
- **Ill-conditioned matrix**. Even if the matrix $\mathbf{S}$ is ill-conditioned, the rate of convergence does not suffer from $\sigma_{\min}$.
- **Result in probability.** The bound on $\kappa^2(\mathbf{V})$ is given with high probability $1 - 3\mathrm{p}$, which flows from the concentration inequalities proposed by Davidson and Szarek (2001) and Vershynin (2012) on the largest/smallest eigenvalues of a random Gaussian matrix.
- **Rotated matrix.** Note that the probability p is taken not on Φ but on the rotated matrix $\hat{\Phi} = \mathbf{U}_*^\top \Phi$.
- **Almost sure convergence.** We propose to sample several matrix Φ until the condition number of $(\mathbf{S}^\top\mathbf{S})^\alpha \mathbf{S}^\top\Phi$ is good enough. By leveraging theory on random Gaussian matrix, we can compute the number of sampling m to achieve this bound on κ_{p} with probability p. Alternatively, we can repeatedly sample Φ until the condition number of $\mathbf{V}$ falls below κ_{p}. Since we know that this is achievable within a finite number of samples, we can obtain a convergence almost surely, with a rate dominated by $\exp(-\kappa_{\mathrm{p}}^{-2})$.

Theorem 2 states that $\kappa(\mathbf{V})$, which determines the convergence rate of Algorithm 2, can be upper-bounded with high probability. Next question is: *how far is $F^i(\hat{\mathbf{U}}, \mathbf{V})$ from the minimal possible error of reconstruction $\epsilon_{\min}$?* With a lower-bound established (Proposition 1), the subsequent section endeavors to upper-bound the Frobenius-norm error when approximating $\mathbf{S}$ with a rank-r matrix.

4.3 Bound on the Frobenius-norm of the Error of Reconstruction

In the case of a low-rank matrix $\mathbf{S}$ with $\operatorname{rank}(\mathbf{S}) = r$ (i.e., $\mathbf{E} = 0$), the couple $(\hat{\mathbf{U}}^i, \mathbf{V})_{i=1}^N$ allows to reconstruct $(\mathbf{S}^i)_{i=1}^N$ without error. This can be proved using Theorem 9.1 from Halko, Martinsson, and Tropp (2011) and by underlining that for any client $i \in \{1, \ldots, N\}$, we have $\hat{\mathbf{U}}^i\mathbf{V}^\top = \mathbf{S}^i\mathbf{V}(\mathbf{V}^\top\mathbf{V})^{-1}\mathbf{V}^\top = \mathbf{S}^i\mathbf{P}$, where $\mathbf{P}$ is a projector on the subspace spanned by the columns of $\mathbf{V}$.

Proposition 3. *Let $r \in \{r_*, \ldots, d \wedge n\}$, in the low-rank matrix regime where we have $\mathbf{E} = 0$, using the power initialization (Algorithm 1), we achieve $\min_{\mathbf{U}^i \in \mathbb{R}^{n_i \times r}} \left\|\mathbf{S}^i - \mathbf{U}^i\mathbf{V}^\top\right\|_{\mathrm{F}}^2 = 0$.*

Second, we are interested in the full-rank matrix $\mathbf{S}$ scenario and give below a theorem upper-bounding the Frobenius-norm error when approximating $\mathbf{S}$ with a rank-r matrix. The proof requires (1) to show the link between the error of reconstruction and the diagonal elements of the projector on the subspace spanned by the columns of $\mathbf{V}_*^\top\mathbf{V}$, (2) upper bound it by the norm of a Gaussian vector and the smallest singular value of a Whishart matrix, and finally (3) apply concentration inequalities.

Theorem 3. *Let $r \leq d \wedge n$ in $\mathbb{N}^*$, using the power initialization (Algorithm 1), for $\mathrm{p} \in]0, 1[$, with probability at least $1 - 2\mathrm{p}$, we have:*

$$\min_{\mathbf{U} \in \mathbb{R}^{n \times r}} \left\|\mathbf{S} - \mathbf{U}\mathbf{V}^\top\right\|_{\mathrm{F}}^2 < \sum_{i>r} \sigma_i^2 \times \left(1 + 2r\mathrm{p}^{-1}\left(\ln(\mathrm{p}^{-2}) + \ln(2)r\right)\frac{(\sigma_{\max}^2 - \sigma_i^2)}{\sigma_r^2}\frac{\sigma_i^{4\alpha}}{\sigma_r^{4\alpha}}\right).$$

The main takeaway is that our algorithm can run in a finite number of communication rounds. We can make the following remarks.

- **Frobenius-norm.** This result *on the Frobenius-norm* for $\alpha > 0$ is new and was missing in Halko, Martinsson, and Tropp (2011). The rate controlling ϵ depends on r^2.
- **Comparison with Halko, Martinsson, and Tropp (2011) in the case $\alpha = 0$.** In contrast, they obtain a dependency on r for both the 2-norm and the Frobenius norm when $\alpha = 0$. Another difference: our bound depends on the ratio $\sigma_{\max}^2/\sigma_r^2$ unlike theirs (Theorem 10.7). But these two drawbacks are annihilated as soon as $\alpha > 0$, we provide more details after Corollary 1.
- **Multiplicative noise.** Following Halko, Martinsson, and Tropp (2011), we have a multiplicative noise. w.r.t. to the minimal possible error $\epsilon_{\min}$.
- **Range of r.** Contrary to Halko, Martinsson, and Tropp (2011); Hardt and Price (2014); Li et al. (2021a), this theorem holds for any value r. However, for $r < r_*$ (corresponds to not taking the whole signal into account), the bound is very large as $\sigma_i = \Omega(1)$, this is why we give a corollary with $r \geq r_*$ in Corollary 1.

The upper bound given in Theorem 3 is minimized if $\sigma_{\max}^2 = \sigma_r^2 + o(1)$ and if for $i > r$ we have $\sigma_i^2 \ll \sigma_r^2$, which we assume to be the case for $r = r_*$. Therefore, taking $r = r_*$ in Theorem 3 minimizes the provided bound. However, the error of reconstruction can only be reduced if taking more than r_* components. Indeed, it mathematically corresponds to having two projectors $\mathbf{P}, \mathbf{P}'$ on the subspaces spanned by the r_* or r components; therefore we have $\operatorname{Im}(\mathbf{P}') \subset \operatorname{Im}(\mathbf{P})$. In particular, it means that if $r > r_*$, the error ϵ will be lower than in the case $r = r_*$. This results in a tighter bound for any $r_* \leq r \leq n \wedge d$. Additionally, we consider $p = 1/4$ in Theorem 3 in order to obtain a bound on the Frobenius-norm with probability at least $1/2$ and derive a number of sampling m s.t. the bound is verified for at least one sampled matrix Φ with probability $\mathrm{P} \in]0, 1[$.

Corollary 1. *For any $r_* \leq r \leq n \wedge d$, for $\alpha \in \mathbb{N}^*$, with probability $\mathrm{P} \in]0,1[$, if we sample $m = \lfloor -\log_2(1-\mathrm{P}) \rfloor$ independent matrices $(\Phi_j)_{j=1}^m$ to form $\mathbf{V}_j = \mathbf{S}^\alpha \Phi$ and $\mathbf{U}_j = \mathbf{S}\mathbf{V}_j(\mathbf{V}_j\mathbf{V}_j)^{-1}\mathbf{V}_j$, at least one of the couple $(\mathbf{U}_j, \mathbf{V}_j)$ results in verifying $\|\mathbf{S} - \mathbf{U}_j\mathbf{V}_j\|_{\mathrm{F}}^2 < \epsilon + \sum_{i>r_*} \sigma_i^2$, with:*

$$\epsilon = \sum_{i>r_*} \sigma_i^2 \left(32 \ln(4) r_*(r_*+1) \frac{(\sigma_{\max}^2 - \sigma_i^2)}{\sigma_{r_*}^2} \frac{\sigma_i^{4\alpha}}{\sigma_{r_*}^{4\alpha}} \right). \tag{1}$$

We can make the following remarks.

- **Dominant term.** The dominant term for ϵ is proportional to r_*^2 while it is proportional to r_* for Halko, Martinsson, and Tropp (2011) in the scenario $\alpha = 0$. Nonetheless, the exacerbated r_*^2 rate is mitigated by its appearance within a logarithmic as we have $\alpha = \Omega\left(\frac{\log(\sigma_{\max} d r_*^2 \epsilon^{-1})}{\log(\sigma_{r_*}) - \log(\sigma_{r_*+1})}\right)$. In other words, doubling α yields an equivalent rate.
- **Impact of α.** Given that for any i in $\{r_*+1, \ldots, n \wedge d\}$, we have $\sigma_i \leq \sigma_{r_*}$, increasing α reduces ϵ by a factor $\sigma_{r_*+1}^{4\alpha}/\sigma_{r_*}^{4\alpha}$ and improves the convergence of the algorithm towards the minimal error $\epsilon_{\min}$. This has a major impact in the particular regime underlined by Corollary 2, *where we get a finite number of communication rounds!*
- **"Comparison" with related works.** Halko, Martinsson, and Tropp (2011) provide a result solely on the 2-norm which is not comparable to our Frobenius-norm: they obtain $\alpha = \Omega(\frac{\sigma_{r_*+1}\log(d)}{\epsilon})$. Our asymptotic rate on α would be better if the quantities were comparable. Li et al. (2021a) provide a result solely on the 2-norm distance between eigenspaces which is again not comparable: they obtain $\alpha = \Omega\left(\frac{\sigma_{r_*}\log(d\epsilon^{-1})}{\sigma_{r_*} - \sigma_{r_*+1}}\right)$. In the regime where $\sigma_{r_*+1} \ll 1$, it can not be equal to zero, unlike us. Note that however in the regime $\sigma_{r_*} \approx \sigma_{r+1}$, the two bounds would be equivalent.
- **Value of m.** Sampling $m = 10$ random matrices is enough to have at least one of them resulting to verify Equation (1) with probability $1 - 10^{-3}$.

The next corollary emphasizes the special regime of a full-rank matrix $\mathbf{S}$ s.t. $\sigma_{\max}, \ldots, \sigma_{r_*} = o(1)$ and $\sigma_{r_*+1} \ll 1$, which is of great interest as raised by Corollary 1. In this regime, increasing α and using gradient descent is particularly efficient in terms of communication cost and precision ϵ.

Corollary 2 (Full-rank scenario with small $\sigma_{\max}/\sigma_{r_*}$)**.** *Let $\lambda, \xi > 0$. Suppose the first r_* (resp. the last $n \wedge d - r_*$) singular values of $\mathbf{S}$ are equal to a large λ (resp. to a small ξ), then Algorithm 2 has a linear rate of convergence determined by $\kappa^2 \leq O(r_*^2 + rd\xi/\lambda) = O(r_*^2)$, and we have an error $\epsilon = O(dr_*^2\xi^{4\alpha+2}/\lambda^{4\alpha})$.*

In the next section, we illustrate the insights highlighted by our theorems. on both synthetic and real data. In particular, on Figures 2b and 3a, we illustrate the setting of Corollary 2 with $\lambda = 1$, $\xi = 10^{-6}$, $d = 200$, and $r_* = 5$ resulting to $\kappa^2 \approx 25$ and $\epsilon_{\alpha=0} \approx 5^{-7}$ for $\alpha = 0$; in contrast, for $\alpha = 1$, we have $\epsilon_{\alpha=1} \approx 5^{-37}$ after only two communications!

Settings	mnist	celeba	w8a
dimension d	784	96	300
latent dimension r	20	20	20
training dataset size	6000	557	49,749
number of clients N	10	25	25

Table 1: Settings of the experiments.

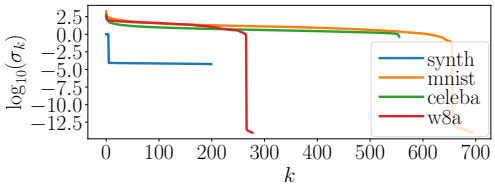


Figure 1: Singular values of the four used datasets.

5 Experiments

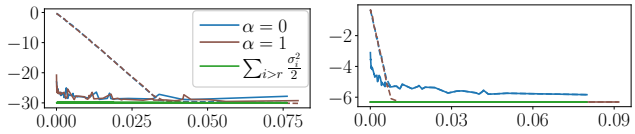
Our code is provided on Github. Experiments have been run on a 13th Gen Intel Core i7 processor with 14 cores. Our benchmark is SVD in the centralized setting (SVD of concatenated matrices) using `scipy.linalg.svd`, this corresponds to the green line in all experiments.

Synthetic dataset generation. We consider synthetic datasets with $N = 25$ clients. For each client i in $\{1, \ldots, N\}$, we have $n_i = 200$ and $d = 200$. We build a global matrix $\mathbf{S} = \mathbf{X} + \mathbf{E}$ and then split it across clients. We set $\mathrm{rank}\,(\mathbf{X}) = 5$ with $\mathbf{X} = \mathbf{U}_\mathbf{X}\mathbf{\Sigma}_\mathbf{X}\mathbf{V}_\mathbf{X}$, where $\mathbf{U}_\mathbf{X}$ (resp. $\mathbf{V}_\mathbf{X}$) are in $\mathcal{O}_n(\mathbb{R})$ (resp. $\mathcal{O}_d(\mathbb{R})$). $\mathbf{\Sigma}_\mathbf{X}$ is a diagonal matrix in $\mathbb{R}^{n\times d}$ with the 5 first values equal to 1, and the other to 0. $\mathbf{E}$ is the noise matrix which elements are independently sampled from a zero-centered Gaussian distribution.

Real datasets. We consider three real datasets: mnist (LeCun, Cortes, and Burges 2010), celeba-200k (Liu et al. 2015) and w8a (Chang and Lin 2011). We do not use the whole datasets for mnist and celeba-200k. For mnist (resp. celeba-200k), clients receive images from a single digit (resp from a single celebrity). For w8a, the dataset is split randomly across clients. This results in two image datasets with low (mnist) or high (celeba-200k) complexity with heterogeneous clients, and a tabular dataset with homogeneous ones.

We plot the SVD decomposition of each dataset on Figure 1. On Figure 2, we run experiments on the synthetic dataset with $r = \mathrm{rank}\,(\mathbf{X}) = 5$ for 50 different samples of Φ. We plot $\sigma_{r_*}^2(\mathbf{V})/\sigma_{\max}^2(\mathbf{V})$ on the X-axis, and the logarithm of the loss F after 1000 local iterations on the Y-axis. The goal is therefore to illustrate the impact of the sampled Φ on the convergence rate.

On Figure 3, we run experiments on the four different datasets; we plot the iteration index on the X-axis, and the logarithm of the loss F w.r.t. iterations on the Y-axis. We run a single gradient descent after having sampled $m = 20$ random matrices Φ to take the one resulting in the best condition number $\kappa(\mathbf{V})$. We run experiments w./w.o. a momentum $\beta_k = k/(k+3)$, with k the iteration index. The goal is here to illustrate on real-life datasets how the algorithm behaves in practice. In Table 2, we contrast our approach with



(a) $\mathbf{E}^i_{kl} = 0$ (b) $\mathbf{E}^i_{kl} \sim \mathcal{N}\big(0, (10^{-6})^2\big)$

Figure 2: Matrix factorization of a low-rank matrix (left) and full-rank (right). We sample $m = 50$ different Φ. X-axis: $\kappa^{-2}(\mathbf{V})$. Y-axis: logarithm error $\log_{10}(\|\mathbf{S}-\mathbf{U}\mathbf{V}^\top\|_{\mathrm{F}}^2)$. Plain line: exact solution. Dashed line: gradient descent after $K = 1000$ iterations for each sampled Φ.

Algorithms	synth	w8a	mnist	celeba
$\alpha = 0$	1	1	1	1
WK2023	26	6	35	58
YD2021	$\geq 10^{13}$	$\geq 10^{20}$	$\geq 10^{20}$	$\geq 10^{21}$
Reached error	-5.5	5.5	4.5	5

Table 2: Number of communications to reach error $\epsilon + \epsilon_{\min}$.

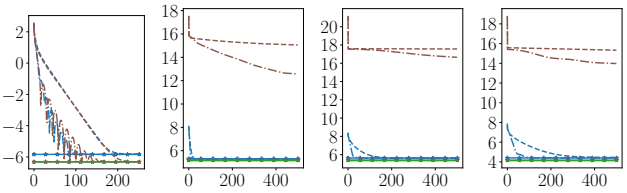
existing algorithms using gradient descent (Ward and Kolda 2023; Ye and Du 2021) by giving the number of communication to reach a given error $\epsilon+\epsilon_{\min}$, showing its superiority in terms of communication.

Observations.

- Figure 2 shows the validity of Theorems 1 and 2 in the scenario without momentum as we obtain a linear convergence rate determined by $\sigma^2_{r_*}(\mathbf{V})/\sigma^2_{\max}(\mathbf{V})$.
- In the low-rank regime (Figure 2a), we recover $\epsilon = 0$ as stated by Proposition 3 (up to machine precision).
- In the regime of a full-rank scenario with a small $\sigma_{\max}/\sigma_{r_*}$ (Figures 2b and 3a, scenario highlighted by Corollary 2), gradient descent is fast (Theorem 2) and recovers the exact solutions $(\hat{\mathbf{U}}^i, \mathbf{V})_{i=1}^N$ of (Dist. MF). Further, in this regime having $\alpha \geq 1$ reduces ϵ (Theorem 3 and Corollary 1) which is observed in practice. For such a setting, gradient descent with $\alpha \geq 1$ is the best strategy.
- In the regime of a full-rank scenario with a large $\sigma_{\max}/\sigma_r$ (Figures 3b to 3d), using $\alpha \geq 1$ might lead to a very slow convergence rate. For illustration, on w8a (resp. mnist and celeba-200k) $\kappa(\mathbf{V})$ is equal to 4 (resp to 10 and 20) for $\alpha = 0$, and equal to 120 (resp. to 5000 and 20000) for $\alpha = 1$. In practice, the slow convergence rate is observed. However, as taking $\alpha \geq 1$ reduces ϵ (Theorem 3 and Corollary 1, and observed on Figures 3b to 3d), it might be preferable in this regime to compute the exact solution of (Dist. MF) with a pseudo-inverse rather than running a gradient descent. If it is not possible to compute the pseudo-inverse or if regularization is used, mandating the use of gradient descent, then it is preferable to take $\alpha = 0$.

6 Conclusion

In this article, we propose an in-depth analysis of low-rank matrix factorisation algorithm within a federated setting.



(a) synthetic (b) w8a (c) mnist (d) celeba

Figure 3: Convergence plot on four datasets. X-axis: number of iterations. Y-axis: logarithm error $\log_{10}(\|\mathbf{S} - \mathbf{U}\mathbf{V}^\top\|_{\mathrm{F}}^2)$. Plain line: exact solution. Dashed line: gradient descent. Dashed-dotted line: gradient descent with momentum $\beta_k = k/(k+3)$, with k the iteration index.

We propose a variant of the power-method that combines a global power-initialization and a local gradient descent, thus, resulting to a smooth and strongly-convex problem. This setup allows for a finite number of communications, potentially even just a single one. We emphasize and experimentally illustrate the regime of high interest raised by our theory (Corollary 2). Finally, drawing from Theorems 1 to 3 and Corollary 1, we highlight the following key insights from our analysis.

Take-away 1. *Increasing the number of communication α leads to reduce the error ϵ by a factor $\sigma^{4\alpha}_{r_*+1}/\sigma^{4\alpha}_{r_*}$, therefore, getting closer to the minimal Frobenius-norm error ϵ.*

Take-away 2. *Using a gradient descent instead of an SVD to approximate the exact solution of the strongly-convex problem allows us to bridge two parallel lines of research. Further, we obtain a simple and elegant proof of convergence and all the theory from optimization can be plugged in.*

Take-away 3. *By sampling several Gaussian matrices Φ, we improve the convergence rate of the gradient descent. Further, based on random Gaussian matrix theory, it results in a almost surely convergence if we sample Φ until $\mathbf{V}$ is well conditioned.*

Drawback 1. *If gradient descent (Algorithm 2) is used to compute $(\hat{\mathbf{U}}^i)_{i\{1,\dots,N\}}$, and if we are in the regime where $\sigma_{\max}/\sigma_r \gg 1$, then increasing α results in increasing the condition number and therefore the number of local iterations. Furthermore, in the scenario $\alpha = 0$, the upper bound on ϵ given by Halko, Martinsson, and Tropp (2011) is better than ours.*

Three open directions to this work can be considered. First, it would be interesting to consider the case of decentralized clients where it is not possible to compute a global V across the whole network. Second, instead of computing an exact $\mathbf{V}$, one could compute an approximation of $\mathbf{V}$ to reduce further the communication cost, this would lead to stochastic-like gradient descent. Third, the extension of our approach to matrix completion is non-trivial (as we require $\mathbf{V}$ to be in the span of $\mathbf{S}$) and could have a lot of interesting applications.

Acknowledgments

This work was supported by the French government managed by the Agence Nationale de la Recherche (ANR) through France 2030 program with the reference ANR-23-PEIA-005 (REDEEM project). It was also funded in part by the Groupe La Poste, sponsor of the Inria Foundation, in the framework of the FedMalin Inria Challenge. Laurent Massoulié was supported by the French government under management of Agence Nationale de la Recherche as part of the "Investissements d'avenir" program, reference ANR19-P3IA-0001 (PRAIRIE 3IA Institute).

References

Achlioptas, D.; and McSherry, F. 2007. Fast computation of low-rank matrix approximations. *Journal of the ACM (JACM)*, 54(2): 9–es.

Avron, H.; Kale, S.; Kasiviswanathan, S.; and Sindhwani, V. 2012. Efficient and practical stochastic subgradient descent for nuclear norm regularization. *arXiv preprint arXiv:1206.6384*.

Bisot, V.; Serizel, R.; Essid, S.; and Richard, G. 2017. Feature learning with matrix factorization applied to acoustic scene classification. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(6): 1216–1229.

Bonawitz, K.; Ivanov, V.; Kreuter, B.; Marcedone, A.; McMahan, H. B.; Patel, S.; Ramage, D.; Segal, A.; and Seth, K. 2017. Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 1175–1191.

Burer, S.; and Monteiro, R. D. 2003. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical programming*, 95(2): 329–357.

Burer, S.; and Monteiro, R. D. 2005. Local minima and convergence in low-rank semidefinite programming. *Mathematical programming*, 103(3): 427–444.

Chang, C.-C.; and Lin, C.-J. 2011. LIBSVM: A library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3): 1–27.

Chen, Y.; Chi, Y.; Fan, J.; and Ma, C. 2019. Gradient descent with random initialization: Fast global convergence for nonconvex phase retrieval. *Mathematical Programming*, 176: 5–37.

Davidson, K. R.; and Szarek, S. J. 2001. Local operator theory, random matrices and Banach spaces. *Handbook of the geometry of Banach spaces*, 1(317-366): 131.

Deshpande, A.; and Vempala, S. 2006. Adaptive sampling and fast low-rank matrix approximation. In *International Workshop on Approximation Algorithms for Combinatorial Optimization*, 292–303. Springer.

Du, S. S.; Hu, W.; and Lee, J. D. 2018. Algorithmic regularization in learning deep homogeneous models: Layers are automatically balanced. *Advances in neural information processing systems*, 31.

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

Gu, Y.; Song, Z.; Yin, J.; and Zhang, L. 2023. Low rank matrix completion via robust alternating minimization in nearly linear time. *arXiv preprint arXiv:2302.11068*.

Halko, N.; Martinsson, P.-G.; and Tropp, J. A. 2011. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2): 217–288.

Hardt, M.; and Price, E. 2014. The noisy power method: A meta algorithm with applications. *Advances in neural information processing systems*, 27.

Hegedűs, I.; Berta, Á.; Kocsis, L.; Benczúr, A. A.; and Jelasity, M. 2016. Robust decentralized low-rank matrix decomposition. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 7(4): 1–24.

Hegedűs, I.; Danner, G.; and Jelasity, M. 2019. Decentralized recommendation based on matrix factorization: A comparison of gossip and federated learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 317–332. Springer.

Jain, P.; Jin, C.; Kakade, S.; and Netrapalli, P. 2017. Global convergence of non-convex gradient descent for computing matrix squareroot. In *Artificial Intelligence and Statistics*, 479–488. PMLR.

Jain, P.; Netrapalli, P.; and Sanghavi, S. 2013. Low-rank matrix completion using alternating minimization. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, 665–674.

Jiang, L.; Chen, Y.; and Ding, L. 2023. Algorithmic regularization in model-free overparametrized asymmetric matrix factorization. *SIAM Journal on Mathematics of Data Science*, 5(3): 723–744.

Konečný, J.; McMahan, H. B.; Yu, F. X.; Richtarik, P.; Suresh, A. T.; and Bacon, D. 2016. Federated Learning: Strategies for Improving Communication Efficiency. In *NIPS Workshop on Private Multi-Party Machine Learning*.

LeCun, Y.; Cortes, C.; and Burges, C. 2010. MNIST handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2.

Li, S.; Li, Q.; Zhu, Z.; Tang, G.; and Wakin, M. B. 2020. The global geometry of centralized and distributed low-rank matrix recovery without regularization. *IEEE Signal Processing Letters*, 27: 1400–1404.

Li, T.; and Ding, C. 2006. The relationships among various nonnegative matrix factorization methods for clustering. In *Sixth International Conference on Data Mining (ICDM'06)*, 362–371. IEEE.

Li, X.; Wang, S.; Chen, K.; and Zhang, Z. 2021a. Communication-efficient distributed SVD via local power iterations. In *International Conference on Machine Learning*, 6504–6514. PMLR.

Li, Z.; Ding, B.; Zhang, C.; Li, N.; and Zhou, J. 2021b. Federated matrix factorization with privacy guarantee. *Proceedings of the VLDB Endowment*, 15(4).

Liberty, E.; Woolfe, F.; Martinsson, P.-G.; Rokhlin, V.; and Tygert, M. 2007. Randomized algorithms for the low-rank approximation of matrices. *Proceedings of the National Academy of Sciences*, 104(51): 20167–20172.

Liu, Z.; Luo, P.; Wang, X.; and Tang, X. 2015. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, 3730–3738.

McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; and Arcas, B. A. y. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Artificial Intelligence and Statistics*, 1273–1282. PMLR. ISSN: 2640-3498.

Mensch, A.; Mairal, J.; Thirion, B.; and Varoquaux, G. 2016. Dictionary learning for massive matrix factorization. In *International Conference on Machine Learning*, 1737–1746. PMLR.

Mirsky, L. 1960. Symmetric gauge functions and unitarily invariant norms. *The quarterly journal of mathematics*, 11(1): 50–59.

Mises, R.; and Pollaczek-Geiringer, H. 1929. Praktische Verfahren der Gleichungsauflösung. *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik*, 9(1): 58–77.

Nguyen, N. H.; Do, T. T.; and Tran, T. D. 2009. A fast and efficient algorithm for low-rank approximation of a matrix. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, 215–224.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, 8024–8035. Curran Associates, Inc.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; and Duchesnay, E. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12: 2825–2830.

Rokhlin, V.; Szlam, A.; and Tygert, M. 2010. A randomized algorithm for principal component analysis. *SIAM Journal on Matrix Analysis and Applications*, 31(3): 1100–1124.

Saha, R.; Srivastava, V.; and Pilanci, M. 2023. Matrix compression via randomized low rank and low precision factorization. *Advances in Neural Information Processing Systems*, 36.

Tong, H.; and Lin, C.-Y. 2011. Non-negative residual matrix factorization with application to graph anomaly detection. In *Proceedings of the 2011 SIAM International Conference on Data Mining*, 143–153. SIAM.

Tropp, J. A.; and Webber, R. J. 2023. Randomized algorithms for low-rank matrix approximation: Design, analysis, and applications. *arXiv preprint arXiv:2306.12418*.

Tropp, J. A.; Yurtsever, A.; Udell, M.; and Cevher, V. 2019. Streaming low-rank matrix approximation with an application to scientific simulation. *SIAM Journal on Scientific Computing*, 41(4): A2430–A2463.

Tu, S.; Boczar, R.; Simchowitz, M.; Soltanolkotabi, M.; and Recht, B. 2016. Low-rank solutions of linear matrix equations via procrustes flow. In *International Conference on Machine Learning*, 964–973. PMLR.

Vershynin, R. 2012. *Introduction to the non-asymptotic analysis of random matrices*, 210–268. Cambridge University Press.

Wang, S.; and Chang, T.-H. 2022. Federated matrix factorization: Algorithm design and application to data clustering. *IEEE Transactions on Signal Processing*, 70: 1625–1640.

Ward, R.; and Kolda, T. 2023. Convergence of alternating gradient descent for matrix factorization. *Advances in Neural Information Processing Systems*, 36: 22369–22382.

Wilson, K. W.; Raj, B.; Smaragdis, P.; and Divakaran, A. 2008. Speech denoising using nonnegative matrix factorization with priors. In *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, 4029–4032. IEEE.

Witten, R.; and Candes, E. 2015. Randomized algorithms for low-rank matrix factorizations: sharp performance bounds. *Algorithmica*, 72: 264–281.

Ye, T.; and Du, S. S. 2021. Global convergence of gradient descent for asymmetric low-rank matrix factorization. *Advances in Neural Information Processing Systems*, 34: 1429–1439.

Zhao, T.; Wang, Z.; and Liu, H. 2015. Nonconvex low rank matrix factorization via inexact first order oracle. *Advances in Neural Information Processing Systems*, 458: 461–462.

Zheng, Q.; and Lafferty, J. 2016. Convergence analysis for rectangular matrix completion using Burer-Monteiro factorization and gradient descent. *arXiv preprint arXiv:1605.07051*.

Zhu, Z.; Li, Q.; Yang, X.; Tang, G.; and Wakin, M. B. 2019. Distributed low-rank matrix factorization with exact consensus. *Advances in Neural Information Processing Systems*, 32.