A. Proof of Theorem 1

Proof. To prove the guarantee for Algorithm 1, it suffices to show the following claim: that at each time step $t \in [T]$, the computations of $\boldsymbol{X}$ and $\sum_{i \in \Omega'_j} \boldsymbol{M}_{ij} \cdot \boldsymbol{U}_i$, for all $j \in [m]$ satisfy $\left(\alpha, \alpha \frac{k}{2\sigma^2}\right)$-RDP. One can then compose the privacy losses via simple Rényi composition (Mironov, 2017) to obtain the overall RDP-cost to be $\left(\alpha, \alpha \frac{kT}{2\sigma^2}\right)$.

To prove the claim, notice that at each time step $t \in [T]$, there are m computations of $\boldsymbol{X}$ and $\sum_{i \in \Omega'_j} \boldsymbol{M}_{ij} \cdot \boldsymbol{U}_i$. Since each user $i \in [n]$ can affect only k of those computations, by Gaussian mechanism (Dwork et al., 2006a; Mironov, 2017) and the generalization of standard composition property of RDP (Mironov, 2017, Proposition 1) to the joint RDP, we have the required guarantee.

We now translate joint-RDP to join-DP. By the first part of the theorem, Algorithm 1 is $(\alpha, \alpha\rho^2)$-joint RDP with $\rho^2 = \frac{kT}{2\sigma^2}$. Thus by (Mironov, 2017, Proposition 3) it is (ε, δ)-joint DP with $\varepsilon = \alpha\rho^2 + \frac{\ln(1/\delta)}{\alpha - 1}$ for any $\alpha > 1$. The latter expression is minimized when $\alpha = 1 + \frac{\sqrt{\ln(1/\delta)}}{\rho}$, which yields $\varepsilon^{\min}(\rho) = \alpha\rho^2 + \frac{\ln(1/\delta)}{\alpha - 1} = 2\sqrt{\ln(1/\delta)}\rho + \rho^2$. Now fix $\varepsilon > 0, \delta \in (0, 1)$. To guarantee (ε, δ)-joint DP while minimizing the noise (which scales as $1/\rho$ by definition of ρ), it suffices to maximize ρ subject to $\varepsilon^{\min}(\rho) \leq \varepsilon$, but since $\varepsilon^{\min}$ is increasing in ρ, ρ is maximized when $\varepsilon^{\min}(\rho) = \varepsilon$. This is a second-order polynomial in ρ, and it has a positive root at $\rho^+ = \sqrt{\ln(1/\delta) + \varepsilon} - \sqrt{\ln(1/\delta)}$. Therefore, setting

$$\sigma = \frac{\sqrt{kT/2}}{\rho^+} = \frac{\sqrt{kT/2}}{\sqrt{\ln(1/\delta) + \varepsilon} - \sqrt{\ln(1/\delta)}} = \frac{\sqrt{kT/2}(\sqrt{\ln(1/\delta) + \varepsilon} + \sqrt{\ln(1/\delta)})}{\varepsilon} \leq \frac{\sqrt{2kT(\ln(1/\delta) + \varepsilon)}}{\varepsilon}$$

suffices to guarantee (ε, δ)-joint DP. This completes the proof.

□

B. Proofs from Section 4

Recall the problem setting. $\boldsymbol{M} = \boldsymbol{U}^* \boldsymbol{\Sigma}^* (\boldsymbol{V}^*)^\top$ where $(\boldsymbol{U}^*)^\top \boldsymbol{U}^* = \boldsymbol{I}$ and $(\boldsymbol{V}^*)^\top \boldsymbol{V}^* = \boldsymbol{I}$. Also, $\boldsymbol{U}^*$ and $\boldsymbol{V}^*$ are μ-incoherent by assumption. That is, $\|\boldsymbol{U}^*_i\|_2 \leq \mu\sqrt{r}/\sqrt{n}$ and $\|\boldsymbol{V}^*_j\|_2 \leq \mu\sqrt{r}/\sqrt{m}$. The set of observations is $\Omega = \{(i, j) \; s.t. \; \delta_{ij} = 1\}$, where δ_{ij} are i.i.d. Bernoulli random variables with $\mathbf{Pr}[\delta_{ij} = 1] = p$. We sample a new set of observations before every update.

We now present a basic lemma.

Lemma 5. *Let $\boldsymbol{U}^*$ and $\boldsymbol{U}^t$ be μ and μ_1-incoherent, orthonormal matrices where $\mu_1 \geq \mu$ and $n \cdot p \geq \mu\mu_1 r^2$. Then, the following holds for all $j \in [m]$ (w.p. $\geq 1 - m\delta$):*

$$\left\| \frac{1}{p} \sum_{i=1}^{n} \delta_{ij} \boldsymbol{U}^*_i (\boldsymbol{U}^t_i)^\top - (\boldsymbol{U}^*)^\top \boldsymbol{U}^t \right\|_F \leq C \sqrt{\frac{\mu_1^2 r}{n \cdot p}} \cdot \ln \frac{r}{\delta}.$$

Proof. The proof follows from the matrix Bernstein inequality (Tropp, 2015, Theorem 6.1.1) and incoherence of $\boldsymbol{U}^*$, $\boldsymbol{U}^t$. □

Lemma 6. *Let δ_{ij} be i.i.d. Bernoulli random variables with $\mathbf{Pr}[\delta_{ij} = 1] = p$. Then, the following holds (w.p. $\geq 1 - \delta$):*

$$\left\| \frac{1}{p} \mathsf{P}_\Omega(\boldsymbol{M}) - M \right\|_F \leq C \left(\sqrt{\frac{n}{p} \ln \frac{1}{\delta}} + \frac{1}{p} \ln \frac{1}{\delta} \right) \cdot \|M\|_\infty.$$

Proof. The lemma is similar to Theorem 7 of (Recht, 2011) and follows by the matrix Bernstein inequality (Tropp, 2015, Theorem 6.1.1). □

B.1. Rank-1 Case

Simplifying the notation, denote $\boldsymbol{M} = \sigma^* \boldsymbol{u}^* (\boldsymbol{v}^*)^\top$ where $(\boldsymbol{u}^*)^\top \boldsymbol{u}^* = 1$ and $(\boldsymbol{v}^*)^\top \boldsymbol{v}^* = 1$.

Note that as $k = C \cdot p \cdot m \ln n$ w.p. $\geq 1 - T/n^{100}$, we do not throw any tuples in Line 7 of Algorithm 1. Similarly, using incoherence we have: $\|\boldsymbol{M}\|_\infty \leq \Gamma_{\boldsymbol{M}}$. So, we do not clip any sample in Line 1 of Algorithm 1.

Now, we use mathematical induction to show the incoherence of resulting $\boldsymbol{v}^t$ and $\widehat{\boldsymbol{u}}^t$, and to show that the clipping operations do not really apply in our setting with the selected hyper-parameters.

For the base case ($t = 0$), initialization of $\boldsymbol{v}^0$ ensures that $\mathrm{Err}(\boldsymbol{v}^*, \boldsymbol{v}^0) \leq \frac{C}{\ln n}$. Now, using (Jain et al., 2013, Lemma C.2) that uses clipping only in the first step to ensure incoherence, we get that $\boldsymbol{v}^0$ is 16μ-incoherent.

In the induction step, assuming the Lemma holds for $\boldsymbol{v}^t$, we prove the claim for $\widehat{\boldsymbol{u}}^t$ and $\boldsymbol{v}^{t+1}$. Dropping superscripts of Ω_i' for notation simplicity and using $\lambda = 0$, we have: $\widehat{\boldsymbol{u}}^t = \arg\min_{\boldsymbol{u}} \left\| \mathsf{P}_\Omega \left(\boldsymbol{M} - \boldsymbol{u}(\boldsymbol{v}^t)^\top \right) \right\|_F^2$. The update of $\boldsymbol{u}^t = \widehat{\boldsymbol{u}}^t / \|\widehat{\boldsymbol{u}}^t\|_2$. So using (Jain et al., 2013, Lemma 5.5, Lemma 5.7, Theorem 5.1), we get w.p. $\geq 1 - 1/n^{100}$:

$$\begin{aligned} &\boldsymbol{u}^t \text{ is } 16\mu\text{-incoherent}, \\ &\|\widehat{\boldsymbol{u}}^t\|_2 \geq \sigma^*/16, \\ &\mathrm{Err}(\boldsymbol{u}^t, \boldsymbol{u}^*) \leq \frac{1}{4}\mathrm{Err}(\boldsymbol{v}^t, \boldsymbol{v}^*). \end{aligned} \tag{3}$$

To complete the claim, we only need to study the update for $\boldsymbol{v}^{t+1}$, which is a noisy version of the ALS update:

$$\widehat{\boldsymbol{v}}^{t+1} = (\boldsymbol{D} + \boldsymbol{G}^t)^{-1} \left(\mathsf{P}_\Omega(M)\widehat{\boldsymbol{u}}^t + \boldsymbol{g} \right),$$

where $\boldsymbol{D}$ and $\boldsymbol{G}^t$ are diagonal matrices s.t. $\boldsymbol{D}_{jj} = \sum_{(i,j)\in\Omega} (\widehat{\boldsymbol{u}}_i^t)^2$ and $\boldsymbol{G}_{jj}^t \sim \Gamma_{\boldsymbol{u}}^2 \sigma \cdot \mathcal{N}(0,1)$.

We first prove that $\boldsymbol{D} + \boldsymbol{G}^t$ is indeed invertible, and has lower-bounded smallest eigenvalue. Using Lemma 5, and $pn \geq \mu^2 \log n \log(1/\delta)$, we have w.p. $\geq 1 - m\delta$,

$$\frac{1}{p}\boldsymbol{D}_{jj} \geq \|\widehat{\boldsymbol{u}}^t\|_2^2 \left(1 - \sqrt{\frac{1}{\log n}} \right).$$

Also, using maximum of Gaussians, we have w.p. $\geq 1 - m\delta$,

$$\frac{1}{p}\left\|\boldsymbol{G}^t\right\|_2 \leq \frac{\Gamma_{\boldsymbol{u}}^2 \sigma \sqrt{\log(n/\delta)}}{p} \leq \frac{\mu^2 (\sigma^*)^2 \sigma \sqrt{\log(n/\delta)}}{np} \leq \frac{\|\widehat{\boldsymbol{u}}^t\|_2^2}{16 \times 256},$$

where the final inequality follows by the assumption on p.

So,

$$\|(\boldsymbol{D} + \boldsymbol{G}^t)^{-1}\|_2 \leq \frac{2}{p \cdot \|\widehat{\boldsymbol{u}}^t\|_2^2}. \tag{4}$$

We now conduct error analysis for $\widehat{\boldsymbol{v}}^{t+1}$:

$$\widehat{\boldsymbol{v}}^{t+1} = \alpha \cdot \boldsymbol{v}^* - \boldsymbol{E},$$

where $\alpha = \frac{\sigma^* \cdot (\boldsymbol{u}^*)^\top \boldsymbol{u}^t}{\|\widehat{\boldsymbol{u}}^t\|_2}$. Furthermore, for a matrix $\boldsymbol{C}$ with $\boldsymbol{C}_{jj} = \sum_{(i,j)\in\Omega} \widehat{\boldsymbol{u}}_i^t \boldsymbol{u}_i^*$, we have $\boldsymbol{E} = \boldsymbol{E}^1 + \boldsymbol{E}^2$ with

$$\boldsymbol{E}_j^1 = (\boldsymbol{D}_{jj} + \boldsymbol{G}_{jj}^t)^{-1}(\alpha \boldsymbol{D}_{jj} - \sigma^* \boldsymbol{C}_{jj})\boldsymbol{v}_j^*, \;\; \boldsymbol{E}_j^2 = (\boldsymbol{D}_{jj} + \boldsymbol{G}_{jj}^t)^{-1}(\alpha \boldsymbol{G}_{jj}^t \boldsymbol{v}_j^* - \boldsymbol{g}_j).$$

This step follows from the observation that $(\mathsf{P}_\Omega(\boldsymbol{M})\widehat{\boldsymbol{u}}^t)_j = \sigma^* \boldsymbol{C}_{jj} \boldsymbol{v}_j^*$. (We note that $\boldsymbol{E}$ is a vector but we use upper case to be consistent with Section B.2.)

Note that $\mathbf{E}[\alpha \boldsymbol{D}_{jj} - \sigma^* \boldsymbol{C}_{jj}] = 0$. Furthermore, using incoherence of $\boldsymbol{v}^*$, $\left\|\widehat{\boldsymbol{u}}^t\right\|_2 \geq \sigma^*/16$, and the Bernstein's inequality, we have:

$$\|\boldsymbol{E}^1\|_2 \leq \frac{1}{64}\mathrm{Err}(\boldsymbol{u}^t, \boldsymbol{u}^*). \tag{5}$$

Now,

$$\|\boldsymbol{E}^2\|_2 \leq \frac{2\sqrt{\log n}}{p \cdot \left\|\widehat{\boldsymbol{u}}^t\right\|_2^2} \cdot \left(16\Gamma_{\boldsymbol{u}}^2 \sigma + \Gamma_{\boldsymbol{M}} \Gamma_{\boldsymbol{u}} \sigma \sqrt{m} \right) \leq \frac{C\mu^4 \sqrt{\log n}}{np} \cdot \sigma. \tag{6}$$

Using (5) and (6), $\left\|\widehat{\boldsymbol{v}}^{t+1}\right\|_2 \geq 3/4$.

Thus, we get:

$$\mathrm{Err}(\boldsymbol{v}^*, \boldsymbol{v}^{t+1}) \leq \frac{1}{32}\mathrm{Err}(\boldsymbol{u}^*, \boldsymbol{u}^{t+1}) + \frac{C\mu^4\sqrt{\log n}}{np} \cdot \sigma.$$

Similarly, by incoherence of $\boldsymbol{v}^*$ and using bound on $\boldsymbol{E}_j^1$ and $\boldsymbol{E}_j^2$, we get:

$$\|\widehat{\boldsymbol{v}}^{t+1}\|_\infty \leq 3\mu.$$

Therefore

$$\begin{aligned}
&\boldsymbol{v}^{t+1} \text{ is } 16\mu\text{-incoherent,} \\
&\mathrm{Err}(\boldsymbol{v}^{t+1}, \boldsymbol{v}^*) \leq \frac{1}{32}\mathrm{Err}(\boldsymbol{u}^t, \boldsymbol{u}^*) + \frac{C\mu^4\sqrt{\log n}}{np} \cdot \sigma.
\end{aligned} \tag{7}$$

So, the inductive hypothesis holds. Furthermore, we get Theorem 2, by combining the error terms of $\boldsymbol{u}^t$ and $\boldsymbol{v}^{t+1}$.

B.2. Rank-r Case

B.2.1. Proof of Lemma 3

Note that as $k = C \cdot p \cdot m \ln n$, w.p. $\geq 1 - T/n^{100}$, we do not throw any tuples in Line 7 of Algorithm 1. Similarly, using incoherence we have: $\|\boldsymbol{M}\|_\infty \leq \Gamma_{\boldsymbol{M}}$. So, we do not clip any samples in Line 1 of Algorithm 1.

Now, we use mathematical induction to show the incoherence of resulting $\widehat{\boldsymbol{V}}^t$ and $\widehat{\boldsymbol{U}}^t$, and to show that the clipping operations do not really apply in our setting with the selected hyperparamters.

For the base case ($t = 0$), initialization of $\widehat{\boldsymbol{V}}^0$ ensures that $(\widehat{\boldsymbol{V}}^0)^\top \widehat{\boldsymbol{V}}^0 = \boldsymbol{I}$ and $\mathrm{Err}(\boldsymbol{V}^*, \widehat{\boldsymbol{V}}^0) \leq \frac{C}{\kappa^2 r^2 \ln n}$. Now, using (Jain et al., 2013, Lemma C.2) that uses clipping only in the first step to ensure incoherence, we get that $\widehat{\boldsymbol{V}}^0$ is $16\mu\sqrt{r}$-incoherent.

For the induction step, assuming the Lemma holds for $\widehat{\boldsymbol{V}}^t$, we prove the claim for $\widehat{\boldsymbol{U}}^t$ and $\widehat{\boldsymbol{V}}^{t+1}$. Dropping superscripts of Ω_i' for notation simplicity and using $\lambda = 0$, we have: $\widehat{\boldsymbol{U}}^t = \arg\min_{\boldsymbol{U}} \left\|\mathsf{P}_\Omega\left(\boldsymbol{M} - \boldsymbol{U}(\boldsymbol{V}^t)^\top\right)\right\|_F^2$. That is, the update of $\widehat{\boldsymbol{U}}^t = \boldsymbol{U}^t\boldsymbol{R}_U$, with $\boldsymbol{U}^t$ being the Q part of QR-decomposition, is identical to the standard non-noisy ALS. So using (Jain et al., 2013, Lemma 5.5, Lemma 5.7, Theorem 5.1)[1], we get w.p. $\geq 1 - 1/n^{100}$,

$$\begin{aligned}
&\boldsymbol{U}^t \text{ is } 16\kappa\mu\text{-incoherent,} \\
&\|\boldsymbol{\Sigma}^*\boldsymbol{R}_U^{-1}\|_2 \leq 16\kappa, \text{ i.e., } \|\boldsymbol{R}_U^{-1}\|_2 \leq \left\|(\boldsymbol{\Sigma}^*)^{-1}\right\|_2 \|\boldsymbol{\Sigma}^*\boldsymbol{R}_U^{-1}\|_2 \leq 16\|(\boldsymbol{\Sigma}^*)^{-1}\|_2\kappa, \\
&\mathrm{Err}(\boldsymbol{U}^t, \boldsymbol{U}^*) \leq \frac{1}{4}\mathrm{Err}(\boldsymbol{V}^t, \boldsymbol{V}^*).
\end{aligned} \tag{8}$$

That is, now to complete the claim we only need to study the update for $\widehat{\boldsymbol{V}}^{t+1}$, which is a noisy version of the ALS update.

Now consider,

$$\boldsymbol{X}_j^t = (\widehat{\boldsymbol{U}}^t)^\top \left(\sum_{i:(i,j)\in\Omega} \boldsymbol{e}_i\boldsymbol{e}_i^\top\right)\widehat{\boldsymbol{U}}^t + \boldsymbol{G}_j^t = p \cdot \boldsymbol{R}_U\left((\boldsymbol{U}^t)^\top\left(\frac{1}{p}\sum_{i:(i,j)\in\Omega} \boldsymbol{e}_i\boldsymbol{e}_i^\top\right)\boldsymbol{U}^t + \boldsymbol{N}_j^t\right)\boldsymbol{R}_U, \tag{9}$$

where $\boldsymbol{G}^t$ is the noise added in Line 10 of Algorithm 1 at time step t, $\boldsymbol{D}_j^t = (\boldsymbol{U}^t)^\top\left(\frac{1}{p}\sum_{i:(i,j)\in\Omega} \boldsymbol{e}_i\boldsymbol{e}_i^\top\right)\boldsymbol{U}^t$ and $\boldsymbol{N}_j^t = \frac{1}{p}\boldsymbol{R}_U^{-1}\boldsymbol{G}_j^t\boldsymbol{R}_U^{-1}$. Note that using Gaussian eigenvalue bound (Vershynin, 2010) and Weyl's inequality (Bhatia, 2013), we have w.p. $\geq 1 - 1/n^{100}$,

$$\sigma_{\min}(\boldsymbol{D}_j^t + \boldsymbol{N}_j^t) \geq \left(1 - C\sqrt{\frac{\mu^2\kappa^2 r}{n \cdot p}} \cdot \ln n - \frac{2\Gamma_{\boldsymbol{u}}^2\sigma\sqrt{r}}{p \cdot \sigma_{\min}(\boldsymbol{R}_U)^2}\right) \geq \frac{1}{2}, \tag{10}$$

[1] Lemma 5.5 of (Jain et al., 2013) has a redundant $\sqrt{r}$ term in incoherence claim

where the last inequality follows from: $np \geq C\mu^2\kappa^2 r \ln^2 n$ and $n\sqrt{p} \geq C\mu^2\kappa^6 r\sqrt{r} \cdot \frac{\sqrt{m \ln n}\cdot(T \ln(1/\delta))}{\varepsilon}$.

Next, we argue that $\boldsymbol{X}_j^t$ is PSD. Observe that

$$\sigma_{\min}(\boldsymbol{X}_j^t) \geq \frac{1}{2}p \cdot \sigma_{\min}(\boldsymbol{R}_U)^2 \geq C\frac{p \cdot \sigma_{\min}(\boldsymbol{\Sigma}^*)^2}{\kappa^2} > 0, \tag{11}$$

where the last inequality follows from (8).

This shows that X used in update of $\widehat{\boldsymbol{V}}^{t+1}$ is *PSD*, and hence the update for $\widehat{\boldsymbol{V}}^{t+1}$ is given by:

$$\begin{aligned}
&\boldsymbol{R}_U(\widehat{\boldsymbol{V}}^{t+1})_j^\top && (12)\\
=&(\boldsymbol{D}_j + \boldsymbol{N}_j^t)^{-1}\left(\boldsymbol{C}_j\boldsymbol{\Sigma}^*(\boldsymbol{V}^*)_j^\top + \bar{\boldsymbol{g}}_j^t\right) && (13)\\
=&(\boldsymbol{U}^t)^\top\boldsymbol{U}^*\boldsymbol{\Sigma}^*(\boldsymbol{V}^*)_j^\top && (14)\\
&-(\boldsymbol{D}_j + \boldsymbol{N}_j^t)^{-1}(\boldsymbol{D}_j(\boldsymbol{U}^t)^\top\boldsymbol{U}^* - \boldsymbol{C}_j)\boldsymbol{\Sigma}^*(\boldsymbol{V}^*)_j^\top - (\boldsymbol{D}_j + \boldsymbol{N}_j^t)^{-1}(\boldsymbol{N}_j^t(\boldsymbol{U}^t)^\top\boldsymbol{U}^*\boldsymbol{\Sigma}^*(\boldsymbol{V}^*)_j^\top - \bar{\boldsymbol{g}}_j^t), && (15)
\end{aligned}$$

where $\boldsymbol{D}_j = (\boldsymbol{U}^t)^\top\left(\frac{1}{p}\sum_{i:(i,j)\in\Omega}\boldsymbol{e}_i\boldsymbol{e}_i^\top\right)\boldsymbol{U}^t$, $\boldsymbol{C}_j = (\boldsymbol{U}^t)^\top\left(\frac{1}{p}\sum_{i:(i,j)\in\Omega}\boldsymbol{e}_i\boldsymbol{e}_i^\top\right)\boldsymbol{U}^*$, and $\bar{\boldsymbol{g}}_j^t = \frac{1}{p}\boldsymbol{R}_U^{-1}\boldsymbol{g}_j^t$.

That is,

$$\begin{aligned}
&\widehat{\boldsymbol{V}}^{t+1}\boldsymbol{R}_U = \boldsymbol{V}^*\boldsymbol{\Sigma}^*(\boldsymbol{U}^*)^\top\boldsymbol{U}^t - \boldsymbol{E}^\top,\ \ \boldsymbol{E}_j = \boldsymbol{E}_j^1 + \boldsymbol{E}_j^2,\\
&\boldsymbol{E}_j^1 = (\boldsymbol{D}_j + \boldsymbol{N}_j^t)^{-1}(\boldsymbol{D}_j(\boldsymbol{U}^t)^\top\boldsymbol{U}^* - \boldsymbol{C}_j)\boldsymbol{\Sigma}^*(\boldsymbol{V}^*)_j^\top,\ \ \boldsymbol{E}_j^2 = (\boldsymbol{D}_j + \boldsymbol{N}_j^t)^{-1}(\boldsymbol{N}_j^t(\boldsymbol{U}^t)^\top\boldsymbol{U}^*\boldsymbol{\Sigma}^*(\boldsymbol{V}^*)_j^\top - \bar{\boldsymbol{g}}_j^t).
\end{aligned} \tag{16}$$

Let $\widehat{\boldsymbol{V}}^{t+1} = \boldsymbol{V}^{t+1}\boldsymbol{R}_V$. Then,

$$\begin{aligned}
&\boldsymbol{V}^{t+1}\boldsymbol{R}_V\boldsymbol{R}_U = \boldsymbol{V}^*\boldsymbol{\Sigma}^*(\boldsymbol{U}^*)^\top\boldsymbol{U}^t - \boldsymbol{E}^\top,\ \ \boldsymbol{E}_j = \boldsymbol{E}_j^1 + \boldsymbol{E}_j^2,\\
&\boldsymbol{E}_j^1 = (\boldsymbol{D}_j + \boldsymbol{N}_j^t)^{-1}(\boldsymbol{D}_j(\boldsymbol{U}^t)^\top\boldsymbol{U}^* - \boldsymbol{C}_j)\boldsymbol{\Sigma}^*(\boldsymbol{V}^*)_j^\top,\ \ \boldsymbol{E}_j^2 = (\boldsymbol{D}_j + \boldsymbol{N}_j^t)^{-1}(\boldsymbol{N}_j^t(\boldsymbol{U}^t)^\top\boldsymbol{U}^*\boldsymbol{\Sigma}^*(\boldsymbol{V}^*)_j^\top - \bar{\boldsymbol{g}}_j^t).
\end{aligned} \tag{17}$$

Using the technique of (Jain et al., 2013, Lemma 5.6) and the bound on $\sigma_{\min}(\boldsymbol{D}_j^t + \boldsymbol{N}_j^t)$ (see (10)), we get:

$$\left\|(\boldsymbol{\Sigma}^*)^{-1}\boldsymbol{E}^1\right\|_F \leq \frac{C}{\kappa}\mathrm{Err}(\boldsymbol{U}^t, \boldsymbol{U}^*). \tag{18}$$

Similarly, w.p. $\geq 1 - 1/n^{100}$:

$$\begin{aligned}
\|(\boldsymbol{\Sigma}^*)^{-1}\boldsymbol{E}^2\|_F &\leq \frac{2}{\sigma_{\min}(\boldsymbol{\Sigma}^*)}\cdot\left(\frac{\Gamma_{\boldsymbol{u}}^2\sigma}{p\sigma_{\min}(\boldsymbol{R}_U)^2}\frac{\sigma_{\max}(\boldsymbol{\Sigma}^*)\mu\sqrt{r^2 m \ln n}}{\sqrt{m}} + \frac{\Gamma_{\boldsymbol{u}}\Gamma_{\boldsymbol{M}}\sigma}{p\sigma_{\min}(\boldsymbol{R}_U)}\cdot\sqrt{mr\ln n}\right),\\
&\leq \frac{C\sigma\sqrt{\ln n}}{pn}\cdot\left(\kappa^5\cdot\mu^3 r^2 + \mu^3 r^2\kappa^3\right) \leq \frac{C\kappa^5\cdot\mu^3 r^2\sqrt{\ln n}}{\sqrt{p}n}\cdot\frac{\sqrt{m\ln n}\cdot T\ln(1/\delta)}{\varepsilon}.
\end{aligned} \tag{19}$$

Let $\beta = \frac{C\kappa^5\cdot\mu^3 r^2\sqrt{\ln n}}{\sqrt{p}n}\frac{\sqrt{m\ln n}\cdot T\ln(1/\delta)}{\varepsilon}$. Now,

$$\sigma_{\min}(\boldsymbol{R}_V\boldsymbol{R}_U) \geq \sigma_{\min}(\boldsymbol{\Sigma}^*)\left(1 - 2\mathrm{Err}(\boldsymbol{U}^t, \boldsymbol{U}^*) - \kappa\beta\right) \geq \frac{\sigma_{\min}(\boldsymbol{\Sigma}^*)}{2}, \tag{20}$$

where the last inequality holds because:

$$\sqrt{p}n \geq C\kappa^6\mu^3 r^2\sqrt{m}\frac{T\ln n\ln(1/\delta)}{\varepsilon}.$$

Using (17), we have:

$$\max_j\left\|(\boldsymbol{V}^{t+1})_j^\top\right\|_2 \leq \frac{2\mu\kappa\sqrt{r}}{\sqrt{m}} + \frac{4\mu\kappa\sqrt{r}}{\sqrt{m}} + \frac{2\mu\kappa r\Gamma_{\boldsymbol{u}}^2\sigma}{\sqrt{m}p\sigma_{\min}(\boldsymbol{R}_U)^2} + \frac{2\Gamma_{\boldsymbol{u}}\Gamma_{\boldsymbol{M}}\sigma\sqrt{r}}{p\sigma_{\min}(\boldsymbol{\Sigma}^*)\sigma_{\min}(\boldsymbol{R}_U)} \leq \frac{16\mu\kappa\sqrt{r}}{\sqrt{m}}, \tag{21}$$

where the last inequality follows from the assumption that $\sqrt{p}n \geq C\kappa^6\mu^3 r^2\sqrt{m}\frac{T\ln n\ln(1/\delta)}{\varepsilon}$. This concludes the proof.

B.2.2. PROOF OF LEMMA 4

The proof for this key Lemma follows technique similar to the above proof. That is, using previous lemma, the clipping operations do not have any effect, and hence we get noisy ALS updates. Now, using (18), (19), (20), and Lemma 7, we have:

$$\mathrm{Err}(\boldsymbol{V}^*, \boldsymbol{V}^{t+1}) \leq \frac{1}{4}\mathrm{Err}(\boldsymbol{U}^*, \boldsymbol{U}^t) + 4\kappa\beta. \tag{22}$$

This proves the lemma.

B.3. Proof of Theorem 2

Using Lemma 7, we have:

$$\mathrm{Err}(\boldsymbol{V}^*, \boldsymbol{V}^t) \leq \frac{1}{4} + \mathrm{Err}(\boldsymbol{V}^*, \boldsymbol{V}^{t+1}) + \alpha,$$

where $\alpha \leq \frac{C\kappa^6 \cdot \mu^3 r^2 \sqrt{\ln n}}{\sqrt{p}n} \frac{\sqrt{m \ln n} \cdot T \ln 1/\delta}{\varepsilon}$. So, after $T = \ln \frac{\mathrm{Err}(\boldsymbol{V}^*, \boldsymbol{V}^0)}{\alpha}$ iterations, $\mathrm{Err}(\boldsymbol{V}^*, \boldsymbol{V}^T) \leq 2\alpha$.

As $\widehat{\boldsymbol{U}}^T = \arg\min_{\widehat{U}} \|\boldsymbol{M} - \widehat{\boldsymbol{U}}^T(\boldsymbol{V}^T)^\top\|_F$, we have:

$$\|\boldsymbol{M} - \widehat{\boldsymbol{U}}^T(\boldsymbol{V}^T)^\top\|_F \leq \|\boldsymbol{M} - \boldsymbol{U}^*\boldsymbol{\Sigma}^*(\boldsymbol{V}^T)^\top\|_F \leq \|\boldsymbol{M}\|_F\|\boldsymbol{V}^* - \boldsymbol{V}^T\|_2 \leq 2\alpha\|\boldsymbol{M}\|_F, \tag{23}$$

where last inequality follows from the fact that $\|\boldsymbol{V}^* - \boldsymbol{V}^T\|_2 \leq 2\mathrm{Err}(\boldsymbol{V}^*, \boldsymbol{V})$.

This shows the second claim of the theorem. The third claim follows similarly while using incoherence of $\boldsymbol{V}^T$.

Lemma 7. *Let $\widehat{\boldsymbol{U}} = \boldsymbol{U}^*\boldsymbol{\Sigma}^*\boldsymbol{W} + \boldsymbol{E}$ and $\boldsymbol{U} = \widehat{\boldsymbol{U}}\boldsymbol{R}^{-1}$ where $\boldsymbol{\Sigma}^*$ is a diagonal matrix, $\boldsymbol{W} \in \mathbb{R}^{r\times r}$, and $\boldsymbol{R}^2 = \widehat{\boldsymbol{U}}^\top\widehat{\boldsymbol{U}}$. Then, assuming $\sigma_{\min}(\boldsymbol{\Sigma}^*)\sigma_{\min}(\boldsymbol{W}) > \|\boldsymbol{\Sigma}^*\|_2\|\boldsymbol{E}(\boldsymbol{\Sigma}^*)^{-1}\|_2$, the following holds:*

$$\|(\boldsymbol{I} - \boldsymbol{U}^*(\boldsymbol{U}^*)^\top)\boldsymbol{U}\|_2 \leq \frac{\|\boldsymbol{E}\cdot(\boldsymbol{\Sigma}^*)^{-1}\|_2}{\frac{\sigma_{\min}(\boldsymbol{\Sigma}^*)}{\|\boldsymbol{\Sigma}^*\|_2}\sigma_{\min}(\boldsymbol{W}) - \|\boldsymbol{E}(\boldsymbol{\Sigma}^*)^{-1}\|_2}.$$

That is,

$$\|(\boldsymbol{I} - \boldsymbol{U}^*(\boldsymbol{U}^*)^\top)\boldsymbol{U}\|_2 \leq \frac{\kappa\|\boldsymbol{E}\|_2}{\sigma_{\min}(\boldsymbol{\Sigma}^*)\sigma_{\min}(\boldsymbol{W}) - \kappa\|\boldsymbol{E}\|_2}.$$

Proof.

$$\|(\boldsymbol{I} - \boldsymbol{U}^*(\boldsymbol{U}^*)^\top)\boldsymbol{U}\|_2 \leq \|\boldsymbol{E}\cdot\boldsymbol{R}^{-1}\|_2 \leq \|\boldsymbol{E}(\boldsymbol{\Sigma}^*)^{-1}\|_2\|\boldsymbol{\Sigma}^*\boldsymbol{R}^{-1}\|_2.$$

Furthermore, $\|\boldsymbol{\Sigma}^*\boldsymbol{R}^{-1}\| \leq \|\boldsymbol{\Sigma}^*\|_2\|\boldsymbol{R}^{-1}\|_2$. Now,

$$\frac{1}{\|\boldsymbol{R}^{-1}\|_2} = \sigma_{\min}(\boldsymbol{R}) \geq \sigma_{\min}(\boldsymbol{\Sigma}^*)\sigma_{\min}(\boldsymbol{W}) - \|\boldsymbol{\Sigma}^*\|_2\|\boldsymbol{E}(\boldsymbol{\Sigma}^*)^{-1}\|_2.$$

That is,

$$\|(\boldsymbol{I} - \boldsymbol{U}^*(\boldsymbol{U}^*)^\top)\boldsymbol{U}\|_2 \leq \frac{\|\boldsymbol{E}\cdot(\boldsymbol{\Sigma}^*)^{-1}\|_2}{\frac{\sigma_{\min}(\boldsymbol{\Sigma}^*)}{\|\boldsymbol{\Sigma}^*\|_2}\sigma_{\min}(\boldsymbol{W}) - \|\boldsymbol{E}\cdot(\boldsymbol{\Sigma}^*)^{-1}\|_2}.$$

□

B.4. Initialization

In this subsection, we describe the initialization routine used by our method. At a high level, similar to (Jain et al., 2013), we use the top eigen-vectors of $\boldsymbol{A} = \mathsf{P}_\Omega(\boldsymbol{M})^\top\mathsf{P}_\Omega(\boldsymbol{M})$ to obtain $\widehat{\boldsymbol{V}}^0$. However, to ensure privacy, we need to add noise to $\boldsymbol{A}$. That is, we compute top-r eigenvectors of $\boldsymbol{W} = \boldsymbol{A} + \boldsymbol{G}$ where $\boldsymbol{G}$ is a symmetric Gaussian matrix with standard deviation $\sigma\Gamma^2_{\boldsymbol{M}}$.

Now using Theorem 2 of (Dwork et al., 2014) which is similar to applying Davis-Kahan theorem to $\boldsymbol{W}$, we get:

$$\mathrm{Err}(\bar{\boldsymbol{V}}, \boldsymbol{V}^{(0)}) \leq \frac{2\|\boldsymbol{G}\|_2}{\lambda_r(\boldsymbol{A}) - \lambda_{r+1}(\boldsymbol{A})} \leq \frac{2\sqrt{m}\sigma\Gamma_{\boldsymbol{M}}^2}{\lambda_r(\boldsymbol{A}) - \lambda_{r+1}(\boldsymbol{A})}.$$

Furthermore, $\|\frac{1}{p}\mathsf{P}_\Omega(\boldsymbol{M}) - \boldsymbol{M}\|_2 \leq \frac{\|\boldsymbol{M}\|_2\mu^2 r}{\sqrt{pm}}$. That is, $\mathsf{P}_\Omega(\boldsymbol{M}) = p\boldsymbol{M} + p\boldsymbol{E}$ where $\|\boldsymbol{E}\|_2 \leq \frac{\|\boldsymbol{M}\|_2\mu^2 r}{\sqrt{pm}}$. This implies, $\boldsymbol{A} = p^2\boldsymbol{M}^\top\boldsymbol{M} + \bar{\boldsymbol{E}}$ where $\|\bar{\boldsymbol{E}}\|_2 \leq p^2\|\boldsymbol{M}\|_2\|\boldsymbol{E}\|_2 \leq p^2\frac{\|\boldsymbol{M}\|_2^2\mu^2 r}{\sqrt{pm}}$.

Using Weyl's inequality: $\lambda_r(\boldsymbol{A}) - \lambda_{r+1}(\boldsymbol{A}) \geq p^2(\sigma_r^*)^2 - 2\|\bar{\boldsymbol{E}}\|_2 \geq (1 - \frac{1}{\log n})p^2(\sigma_r^*)^2$ due to assumption on p. So, using the bound above, we get:

$$\mathrm{Err}(\bar{\boldsymbol{V}}, \boldsymbol{V}^{(0)}) \leq \frac{2\sqrt{m}\sigma\Gamma_{\boldsymbol{M}}^2}{p^2(\sigma_r^*)^2} \leq \frac{2\sqrt{m}\sigma\|\boldsymbol{M}\|_2^2}{p^2 mn}.$$

Similarly using Davis-Kahan on $\boldsymbol{A} = p^2\boldsymbol{M}^\top\boldsymbol{M} + \bar{\boldsymbol{E}}$, we get:

$$\mathrm{Err}(\boldsymbol{V}^*, \bar{\boldsymbol{V}}) \leq \frac{2\|\bar{\boldsymbol{E}}\|_2}{p^2(\sigma_r^*)^2} \leq 2\kappa^2\frac{\mu^2 r}{\sqrt{pm}}.$$

That is,

$$\mathrm{Err}(\boldsymbol{V}^*, \boldsymbol{V}^{(0)}) \leq \frac{2\sqrt{m}\sigma\|\boldsymbol{M}\|_2^2}{p^2 mn} + 2\kappa^2\frac{\mu^2 r}{\sqrt{pm}}.$$

The initialization condition is now matched by the assumption on p (Theorem 2) combined with the assumption that $n \geq \widetilde{\Omega}(m\sqrt{m\log 1/\delta}/\varepsilon)$.

C. Additional Details on Heuristic Improvements

Algorithm 2 summarizes the data pre-processing and sampling heuristics described in Section 5.

Algorithm 2 Data pre-processing heuristics.

Required: $\mathsf{P}_\Omega(\boldsymbol{M})$: Observed ratings, $\Gamma_{\boldsymbol{M}}$: entry clipping parameter, k: maximum number of ratings per user, σ_p: standard deviation of the pre-processing noise, β: fraction of movies to train on.

1 Clip entries in $\mathsf{P}_\Omega(M)$ so that $\|\mathsf{P}_\Omega(M)\|_\infty \le \Gamma_{\boldsymbol{M}}$

2 Uniformly sample Ω':
for $1 \le i \le n$ **do**
  $\Omega'_i \leftarrow$ sample k items from Ω_i uniformly.
end

3 Compute movie counts $\widetilde{\boldsymbol{c}} \leftarrow \text{Counts}(\Omega')$.

4 Partition movies:
Let Frequent be the $\lceil \beta m \rceil$ movies with the largest $\widetilde{\boldsymbol{c}}$, and let Infrequent be the rest.

5 Adaptively sample Ω'':
for $1 \le i \le n$ **do**
  $\Omega''_i \leftarrow$ the k items in $(\Omega_i \cap \text{Frequent})$ with the lowest count $\widetilde{\boldsymbol{c}}$.
end

6 Recompute movie counts $\widetilde{\boldsymbol{c}} \leftarrow \text{Counts}(\Omega'')$

7 Center the data $\mathsf{P}_{\Omega''}(M) \leftarrow \mathsf{P}_{\Omega''}(M) - \widetilde{m}$, where $\widetilde{m} = \frac{\sum_{(i,j)\in\Omega''} M_{ij} + \mathcal{N}(0, k\Gamma_{\boldsymbol{M}}^2\sigma_p^2)}{|\Omega''| + \mathcal{N}(0, k\sigma_p^2)}$

return $\mathsf{P}_{\Omega''}(\boldsymbol{M}), \widetilde{\boldsymbol{c}}$

Procedure *Counts*(Ω)
  for $1 \le j \le m$ **do**
    $\widetilde{\boldsymbol{c}}_j \leftarrow |\Omega_j| + \mathcal{N}(0, \sigma_p^2)$
    return $\widetilde{\boldsymbol{c}}$
  end

First, we compute differentially private movie counts (Line 3) using a uniform sample Ω', and use it to partition the movies (Line 4) and to perform adaptive sampling (Line 5). The final subset used for training is Ω'', which consists only of Frequent movies. Finally, to have a more accurate estimate of the counts, we recompute $\widetilde{\boldsymbol{c}}$ on Ω'' (Line 6). We redo this computation as the counts are also used during optimization, as described in the next section. Note that in both computations of $\widetilde{\boldsymbol{c}}$, we use a subset of Ω that contains at most k movies per user, in order to guarantee user-level differential privacy.

Privacy accounting. As we saw in Theorem 1, Algorithm 1 with random initialization satisfies $\left(\alpha, \frac{\alpha(kT)}{2\sigma^2}\right)$-joint RDP. The data processing heuristics in Algorithm 2 satisfy $\left(\alpha, \frac{\alpha(2k+2)}{2\sigma_p^2}\right)$-RDP. So, by standard composition of RDP, we have the total privacy cost at any order $\alpha > 1$ to be: $\left(\alpha, \alpha \cdot \left(\frac{kT}{2\sigma^2} + \frac{k+1}{\sigma_p^2}\right)\right)$. We can obtain the final (ε, δ)-joint differential privacy guarantee by optimizing for α, similarly to Appendix A.

Loss function. We minimize the following loss in practice.

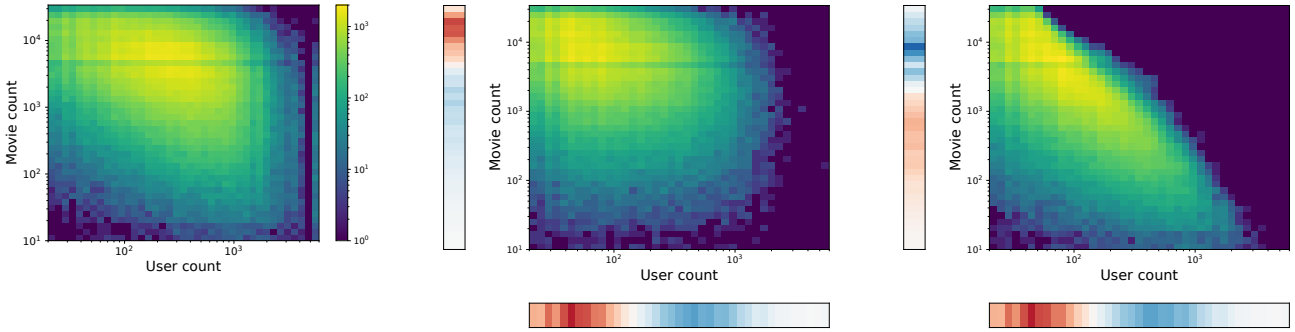
$$f(\widehat{\boldsymbol{U}}, \widehat{\boldsymbol{V}}) = \|\mathsf{P}_\Omega\big(\boldsymbol{M} - \widehat{\boldsymbol{U}}\widehat{\boldsymbol{V}}^\top\big)\|_F^2 + \lambda_0\|\widehat{\boldsymbol{U}}\widehat{\boldsymbol{V}}^\top\|_F^2 + \lambda\sum_{i=1}^{n}\frac{\boldsymbol{c}_i^\nu}{Z}\|\widehat{\boldsymbol{U}}_i\|^2 + \lambda\sum_{j=1}^{m}\frac{\widetilde{\boldsymbol{c}}_j^\mu}{Z'}\|\widehat{\boldsymbol{V}}_j\|^2, \tag{24}$$

where λ_0, λ, μ, and ν are hyper-parameters. The loss function used in the description of Algorithm 1 is a special case of (24) where $\lambda_0 = \mu = \nu = 0$. The additional terms in (24) do not change the essence of the algorithm, but we find that they make a significant difference in practice.

First, the term $\lambda_0\|\widehat{\boldsymbol{U}}\widehat{\boldsymbol{V}}^\top\|_F^2$ is often used in problems with implicit feedback, as in (Hu et al., 2008). In such problems, the observed entries are often binary, and minimizing the objective $\|\mathsf{P}_\Omega\big(\boldsymbol{M} - \widehat{\boldsymbol{U}}\widehat{\boldsymbol{V}}^\top\big)\|_F^2$ can yield a trivial solution – the matrix of all ones. The addition of the second term penalizes non-zero predictions outside of Ω, leading to better generalization. One of the benchmarks we use is an implicit feedback task, in which the use of the second term is necessary. As described in Section 5.2, this results in an additional term $\boldsymbol{K}$ in Line 10 of $\mathcal{A}_{\text{item}}$, and care is needed when adding privacy protection

to this term, since it involves a sum over all user embeddings. The key observation is that this term is constant for all items, so we only need to compute a noisy version of $\boldsymbol{K}$ once and use it for all items, thus limiting the privacy loss it incurs.

Second, we use a weighted ℓ_2 regularization, where the weights are defined as follows. The weight of movie j is $\widetilde{\boldsymbol{c}}_j^{\mu}/Z'$, where $\widetilde{\boldsymbol{c}}$ is the vector of approximate counts (computed in Algorithm 2), μ is a non-negative hyper-parameter and Z' is the normalizing constant $Z' = \frac{1}{m}\sum_{j=1}^{m} \widetilde{\boldsymbol{c}}_j^{\mu}$. When μ is positive, this corresponds to applying heavier regularization to more frequent items, and we found in our experiments that this can significantly help generalization. The weights for the users are defined similarly, with one main difference: instead of using approximate counts $\widetilde{\boldsymbol{c}}$, we use the exact counts $\boldsymbol{c}$, as this term only affects the solution in $\mathcal{A}_{\text{user}}$, which is a privileged computation as illustrated in Figure 1.



(a) ML-10M (unsampled) (b) Uniform sampling (k = 50) (c) Adaptive sampling (k = 50)

Figure 6. Histogram of user and movie counts in ML-10M, in the original data, and under uniform and adaptive sampling. The color bars in Figures 6b and 6c show the difference in marginal probability compared to the original data in 6a. Red indicates an increase in marginal probability, while blue indicates a decrease. Note that the probability of frequent movies increases under uniform sampling, and decreases under adaptive sampling.

Effect of uniform and adaptive sampling. As observed in Figure 2, the movie count distribution of the MovieLens data set is heavily skewed. We also observed that, perhaps surprisingly, uniformly sampling k items per user tangibly increases the skew. This can be explained by a negative correlation between user counts and movie counts; we computed a correlation coefficient of -0.243. This is also visible in Figure 6a, which shows the joint histogram of $\{(c_i, c_j), (i,j) \in \Omega\}$, where $c_i = |\Omega_i|$ is the user count (the number of ratings this user produced) and $c_j = |\Omega_j|$ is the movie count (the number of ratings the movie received). The figure illustrates that infrequent users are more likely to rate frequent movies than the average user. By uniformly sampling a constant number of movies per user (Figure 6b), we are, by definition, increasing the probability of infrequent users, hence increasing the probability of frequent movies (due to the negative correlation). This is made clear by the color bar left of Figure 6b, which shows the change in movie count probability with respect to the original data set. This increase in the probability of frequent movies aggravates the skew of the movie distribution, as seen in Figure 2.

Adaptive sampling has the opposite effect: Figure 6c shows that the probability of frequent movies *decreases* under adaptive sampling, while that of infrequent movies increases. This leads to a decrease in bias toward frequent movies, as shown in Figure 2, and results in a significant improvement in the privacy/utility trade-off as discussed in Section 6.3.

D. Additional Details on Experiments

D.1. Details on the Experimental Setup

Table 2 shows the statistics of the MovieLens data sets.

Table 2. Statistics of the experiment data sets.

	ML-10M-top400	ML-10M	ML-20M
n (number of users)	69,692	69,878	136,677
m (number of items)	400	10,677	20,108
$\lvert\Omega\rvert$ (number of observations)	4.49M	10M	9.99M

For each data set, we partition the set of observations Ω into $\Omega = \Omega^{\mathsf{train}} \sqcup \Omega^{\mathsf{valid}} \sqcup \Omega^{\mathsf{test}}$. Hyper-parameter tuning is performed on Ω^{valid}, and the final results are reported on Ω^{test}. The pre-processing described in Algorithm 2 is only applied to Ω^{train}.

In the ML-10M benchmark, we follow the setup of (Lee et al., 2013) and use a 80-10-10 split (random uniform over Ω). In the ML-10M-top400 benchmark, we follow the setup of (Jain et al., 2018) and use a 98-1-1 split (random uniform over Ω). In the ML-20M benchmark, we follow the setup of (Liang et al., 2018) and partition the set by *users*, that is, a set of 20K random users are held-out, half of which are used for validation, and the other half for testing. Note that since held-out users are never seen in training, the protocol is to further split each user's observations Ω_i^{test} (uniformly at random) into $\Omega_i^{\mathsf{test\ query}} \sqcup \Omega_i^{\mathsf{test\ target}}$. At test time, the model is allowed access to $\Omega_i^{\mathsf{test\ query}}$ to compute a user embedding and make a prediction for the user, and $\Omega_i^{\mathsf{test\ target}}$ is used as the ground truth target. The user embedding $\widehat{\boldsymbol{U}}_i$ is computed at test time simply by minimizing the loss in Eq. (24) given the learned movie embeddings $\widehat{\boldsymbol{V}}$, that is,

$$\widehat{\boldsymbol{U}}_i = \operatorname*{arg\,min}_{u \in \mathbb{R}^r} \|\mathsf{P}_{\Omega_i^{\mathsf{test\ query}}}\big(\boldsymbol{M}_i - u\widehat{\boldsymbol{V}}^\top\big)\|_F^2 + \lambda_0 \|u\widehat{\boldsymbol{V}}^\top\|_F^2 + \lambda \frac{\boldsymbol{c}_i^\nu}{Z'}\|u\|^2.$$

The resulting $\widehat{\boldsymbol{U}}_i$ is used to generate predictions for user i. Note that this procedure is consistent with the Joint-DP setting: the computation of $\widehat{\boldsymbol{U}}_i$ corresponds to one step of $\mathcal{A}_{\mathsf{user}}$ in Algorithm 1, and is considered privileged (see Figure 1). Besides, since the resulting embedding is not further used for training, it is unnecessary to clip the embedding norm. Avoiding norm clipping at test time could result in better predictions.

Finally the recall for user i is computed as follows. Let $\Omega_i^{\mathsf{prediction}}$ be the top k items that are not in $\Omega_i^{\mathsf{test\ query}}$. Then $\text{Recall@}k = \frac{|\Omega_i^{\mathsf{prediction}} \cap \Omega_i^{\mathsf{test\ target}}|}{\min(k, |\Omega_i^{\mathsf{test\ target}}|)}$.

D.2. Hyper-Parameter Description and Ranges

Table 3 summarizes the complete list of hyper-parameters used in Algorithm 1, Algorithm 2, and in the loss function (24), and specifies the ranges used in our experiments. We make several remarks about hyper-parameters:

Table 3. Hyper-parameter description and ranges.

Symbol	Description	Range
Model and training parameters		
r	rank	[2, 128]
λ	ℓ_2 regularization coefficient	[0.1, 100]
λ_0	coefficient of the global penalty term	[0.1, 5]
μ	item regularization exponent	$\{0, 0.5, 1\}$
ν	user regularization exponent	$\{0, 0.5, 1\}$
T	number of steps	[1, 5]
Privacy parameters		
$\Gamma_{\boldsymbol{u}}$	row clipping parameter	1
$\Gamma_{\boldsymbol{M}}$	entry clipping parameter	$\{1, 5\}$
k	maximum number of ratings per user	[20, 150]
σ	noise standard deviation	see remark below
Pre-processing parameters		
β	fraction of items to train on	[0, 1]
σ_p	standard deviation of pre-processing noise	[10, 200]

- In the non-private baselines, only the model and training parameters are tuned.
- Pre-processing (Algorithm 2) is not used in synthetic experiments. Indeed, these heuristics are designed to deal with the non-uniform distribution of observations in practice. In synthetic experiments, the distribution is uniform by design.
- In the MovieLens experiments, the maximum value in $\boldsymbol{M}$ is known by definition of the task: In ML-10M, entries represent ratings in the interval $[0.5, 5]$, and in ML-20M the entries are binary. Thus, we simply set $\Gamma_{\boldsymbol{M}}$ to this value without tuning.

– We find that carefully tuning the model parameters, including the regularization coefficients λ, λ_0 and the exponents μ, ν is important and can have a significant effect.

– For the rating prediction tasks (ML-10M and ML-10M-top400), we find that setting λ_0 to a positive number is detrimental, so we always use 0. For the item recommendation task (ML-20M), using a non-zero λ_0 is important.

– The partitioning of the movies into Frequent and Infrequent is important for the private models, especially at lower values of ε (see Figure 10), but does not help for the non-private baselines.

– To set the standard deviation σ, we use the simple observation that when all hyper-parameters except σ are fixed, ε is a decreasing function of σ that can be computed in closed form. Therefore, in each experiment, we set a target value of ε and do a binary search over σ to select the smallest value that achieves the target ε.

– Finally, note that in Algorithm 1, the parameter σ determines the standard deviation of two noise terms: $\boldsymbol{G}$ in Line 8 and $\boldsymbol{g}$ in Line 9. While this is sufficient for the analysis, we find in practice that the model is often more sensitive to $\boldsymbol{g}$, thus it can be advantageous to use different scales of noise. We will use the symbols σ_G, σ_g to specify the scales of each term.

The optimal hyper-parameter values for each experiment and each value of ε are given in Table 4. These values are obtained through cross-validation. We do not include the privacy loss of hyper-parameter search because our main objective is to give insights into the choice of hyper-parameters at different privacy budgets. In practice, this can be accounted for, for example by the method in (Liu & Talwar, 2019).

Table 4. Optimal hyper-parameter values for the experiments in Figure 3. The clipping parameter $\Gamma_{\boldsymbol{u}}$ is set to 1 in all experiments.

	ML-10M-top400					ML-10M					ML-20M				
	DPALS				ALS	DPALS				ALS	DPALS				ALS
ε	0.8	4	8	16	-	1	5	10	20	-	1	5	10	20	-
r	50	50	50	50	50	32	128	128	128	128	32	32	32	128	128
λ	90	90	80	80	70	120	80	70	60	70	0.5	0.5	0.1	50	30
λ_0	0	0	0	0	0	0	0	0	0	0	2	0.6	0.4	0.4	0.1
μ	0.5	0.5	0.5	0.5	1	0.5	0.5	0.5	0.5	1	-	-	-	-	-
ν	1	1	1	1	1	1	1	1	1	1	-	-	-	-	-
T	2	2	2	2	15	2	2	2	2	15	1	3	3	1	15
k	40	50	50	50	-	50	50	50	50	-	60	60	100	60	-
σ_G	126.9	29.0	11.3	5.86	-	125.9	27.8	15.5	7.5	-	64.0	20.2	14.0	3.5	-
σ_g	63.4	14.5	11.3	5.86	-	63.0	13.9	7.7	3.8	-	64.0	20.2	14.0	3.5	-
β	1	1	1	1	-	0.05	0.4	0.5	0.6	-	0.05	0.1	0.05	0.05	-
σ_p	200	200	20	20	-	100	20	10	10	-	100	100	100	100	-

D.3. Standard Deviation

Finally, Table 5 reports the standard deviation of the DPALS metrics in Figure 3. For each data point, we repeat the experiment 20 times, using the same set of hyper-parameters selected on the validation set, and report the mean and standard deviation of the metric measured on the test set. In all cases, the standard deviation is less than 0.5% of the mean.

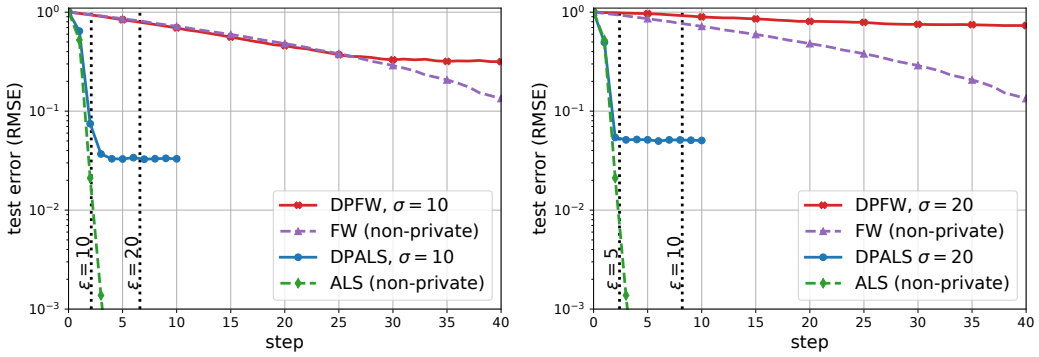
Table 5. Mean and standard deviation of the DPALS metrics in Figure 3.

	ML-10M-top400 (test RMSE)				ML-10M (test RMSE)				ML-20M (test Recall@20)			
ε	0.8	4	8	16	1	5	10	20	1	5	10	20
mean	0.8855	0.8321	0.8201	0.8147	0.9398	0.8725	0.8530	0.8373	0.3120	0.3330	0.3368	0.3444
stddev	0.0025	0.0009	0.0011	0.0008	0.0009	0.0006	0.0004	0.0005	0.0016	0.0010	0.0012	0.0013

D.4. Additional Experiments

Convergence plots for DPALS and DPFW. This experiment illustrates the fact that ALS converges faster than FW, both in its exact and private variants, making it more suitable for training private models. Figure 7 shows the test error (RMSE) against number of iterations, on the synthetic data set with $n = 20K$ users. We use the vanilla version of DPALS without the heuristics introduced in Section 5. The hyper-parameters of both methods are tuned on the validation set.

For the non-private baselines, ALS converges significantly faster than FW. For example, the error of ALS after 2 iterations is lower than the error of FW after 40 iterations. For the private models, we compare the two methods with the same sampling rate ($k = 150$) and same noise level ($\sigma = 10$ in Figure 7a and $\sigma = 20$ in Figure 7b), and tune other parameters. Since the sampling rate and noise level are fixed, the ε level is directly determined by the number of steps, and the vertical lines show different levels of ε. We can make the following observations. For both methods, in the presence of noise, the error decreases for a few iterations at a rate similar to their exact variants, then plateaus at a fixed error. The fixed error for DPALS the is an order of magnitude lower than DPFW. Furthermore, the error reached by DPALS in 2 iterations is lower than the error of DPFW after 40 iterations. The faster convergence of DPALS, even in the presence of noise, directly translates to a better privacy/utility trade-off as demonstrated in Section 6.



(a) $\sigma = 10$ (b) $\sigma = 20$

Figure 7. RMSE against steps on the synthetic data set with $n = 20K$. Dashed lines correspond to the non-private baselines (without noise) and solid lines correspond to the private methods with a fixed noise level (left: $\sigma = 10$, right: $\sigma = 20$).

Varying the number of users. This experiment further illustrates the effect of increasing the number of users. We train the DPALS on a subset of the ML-10M-top400 data set, obtained by randomly sampling a subset of n users. Figure 8 shows the results for different values of n, and confirms that increasing the number of users (while keeping the number of movies constant) improves the privacy/utility trade-off. The figure also compares to the DPFW baseline trained on the full data ($n = 69692$). Note that DPALS significantly outperforms DPFW even when trained on a small fraction of the users ($n = 16000$, or 26.4% of the total users).

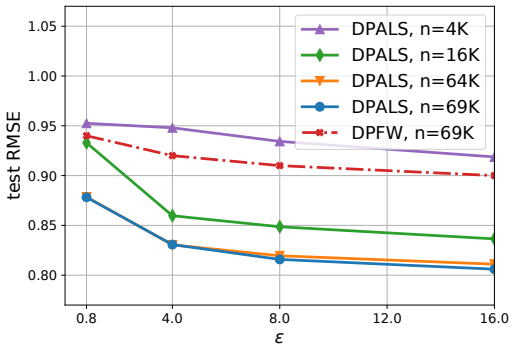
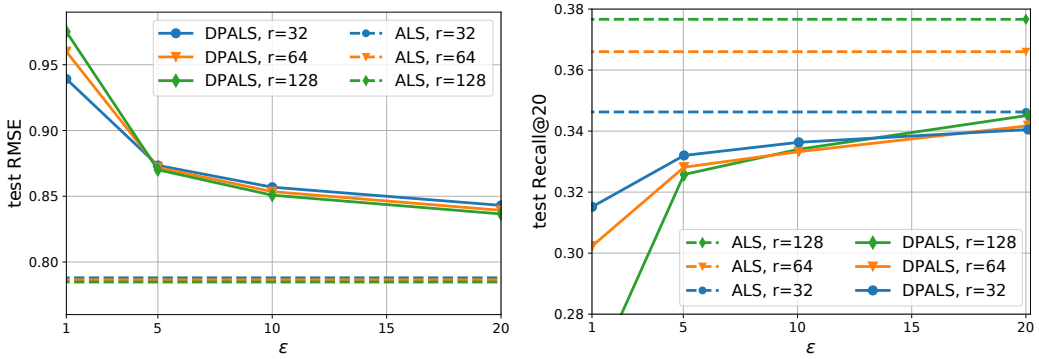


Figure 8. DPALS on ML-10M-top400 with a varying number of users, n.

Effect of the rank. This experiment explores the effect of the rank on the privacy/utility tradeoff. Figure 9 shows the trade-offs for models of different ranks r on ML-10M and ML-20M. We observe that for non-private ALS, models of higher rank consistently achieve better performance in the range of ranks that we have tried. This is not always the case for the private models. For the ML-10M task, the higher rank model ($r = 128$) performs well for larger values of ε, but not for $\varepsilon = 1$. On the ML-20M task, the private model with $r = 128$ gives the best recall for $\varepsilon \approx 20$ while $r = 32$ performs the best for smaller ε. Therefore, unlike in the non-private ALS algorithm where a higher rank is often more desirable given enough computational and storage resources, when training a private model, one needs to carefully choose the rank to balance model capacity and utility degradation due to privacy.



(a) RMSE on ML-10M (lower is better)

(b) Recall@20 on ML-20M (higher is better)

Figure 9. Privacy/utility trade-off for models of different ranks. For lower values of ε, a lower rank achieves a better privacy/utility.

Training on Frequent **movies.** This experiment illustrates the effect of partitioning the set movies into (Frequent $\sqcup$ Infrequent) and training only on Frequent movies. Figure 10 shows the test RMSE vs movie fraction, at different levels of ε. The rank of the model is fixed to $r = 32$, the sample size is fixed to $k = 50$, and other hyper-parameters are re-tuned. The results show that as ε decreases, the optimal fraction of movies decreases. In particular, for $\varepsilon = 1$, the optimal fraction is 5%; note however that this still corresponds to more than 50% of the ratings, as shown on the right sub-figure.

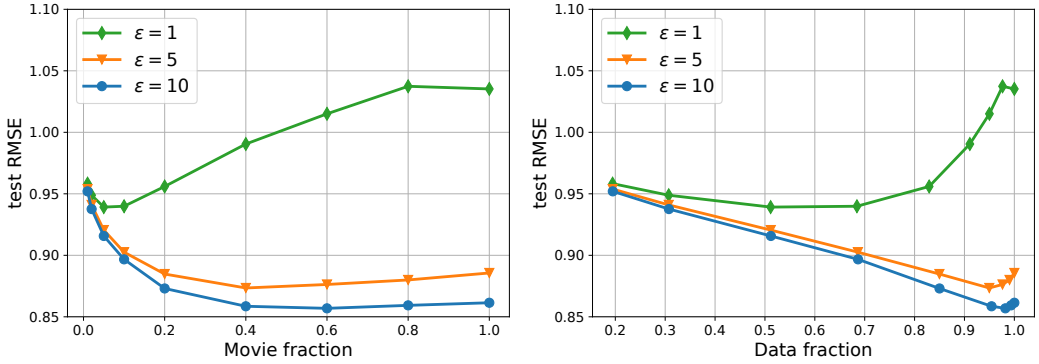


Figure 10. RMSE vs movie fraction, for a rank 32 model on ML-10M, at different privacy levels ε. Both figures show the same data, but with a different x axis. The movie fraction (left figure) is defined as $|\text{Frequent}|/m$. The data fraction (right figure) is defined as $|\{(i,j) \in \Omega : j \in \text{Frequent}\}|/|\Omega|$. The right figure emphasizes the long-tail distribution of movie counts – a small fraction of Frequent movies corresponds to a large fraction of data.

Figure 11 shows a similar result for ML-20M. The optimal movie fraction in this example is between 5% and 10% depending on the rank.

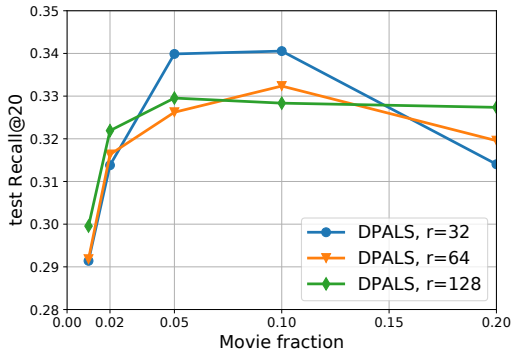


Figure 11. Recall@20 vs movie fraction on ML-20M, for $\varepsilon = 5$.

Effect of the regularization exponents. This experiment illustrates the effect of the regularization exponents (ν, μ) in the loss function (24). We vary (ν, μ) for a rank 128 model with $\varepsilon = 10$ on ML-10M (and re-tune other parameters). The results are reported in Figure 12. This example indicates that a careful tuning of the ℓ_2 regularization can have a significant impact on utility, and can also make the private models more robust to noise: Notice that with the optimal setting of (ν, μ) the model can be trained on a much larger fraction of movies, with only a slight degradation in utility.

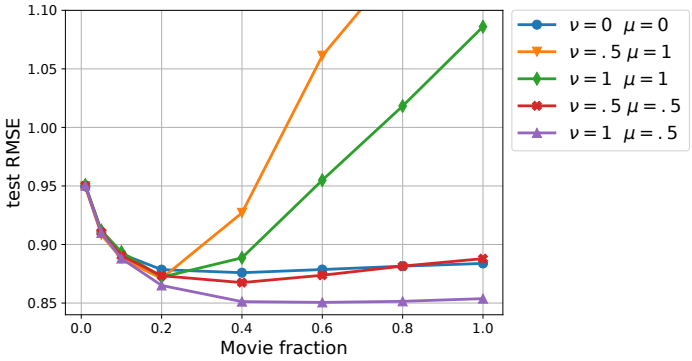


Figure 12. RMSE vs movie fraction on ML-10M, for $\varepsilon = 10$ and $r = 128$, and for different values of regularization exponents (ν, μ).