Recent Progress on the Matrix Factorization Approach for Differentially Private Learning

Christoph Lampert (ISTA)





Workshop on Privacy-preserving Machine Learning and Privacy in Distributed Settings,
Aarhus, 2026/11/03

Differential Privacy Recap

(ϵ, δ)-Differential Privacy [Dwork *et al.*, EUROCRYPT 2006]

A randomized mechanism $\mathcal{M}$ is (ϵ, δ)-DP, if for all *adjacent* datasets $D \simeq D'$ and all measurable sets S of outputs:

$$\Pr\{\mathcal{M}(D) \in S\} \leq e^{\epsilon} \Pr\{\mathcal{M}(D') \in S\} + \delta$$

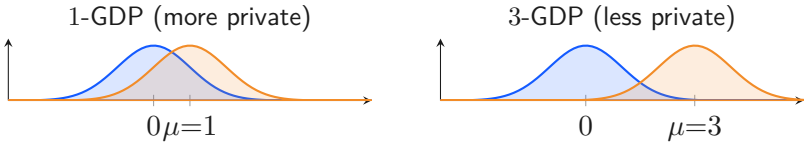
(ϵ, δ)-Differential Privacy [Dwork *et al.*, EUROCRYPT 2006]

A randomized mechanism $\mathcal{M}$ is (ϵ, δ)-DP, if for all *adjacent* datasets $D \simeq D'$ and all measurable sets S of outputs:

$$\Pr\{\mathcal{M}(D) \in S\} \leq e^{\epsilon} \Pr\{\mathcal{M}(D') \in S\} + \delta$$

μ-Gaussian Differential Privacy (GDP) [Dong *et al.*, Journal of the RSS, 2022]

Distinguishing $\mathcal{M}(D)$ from $\mathcal{M}(D')$ is no easier than distinguishing $\mathcal{N}(0, 1)$ from $\mathcal{N}(\mu, 1)$ from a single sample.



GDP is convenient for Gaussian noise; tight conversions to (ϵ, δ)-DP or zCDP exist.

The Gaussian mechanism

- ℓ_2-sensitivity of a function f: $\quad \mathrm{sens}(f) = \max_{D \simeq D'} \|f(D) - f(D')\|_2$
- neighborhood relation $D \simeq D'$: typically zero-out, but others are possible
- Gaussian mechanism with noise multiplier σ:

$$\mathcal{M}(D) = f(D) + z, \qquad z \sim \mathcal{N}\big(0,\ \nu^2\,\mathsf{Id}\big), \qquad \nu = \sigma \cdot \mathrm{sens}(f)$$

  $\to \mathcal{M}$ is $\frac{1}{\sigma}$-GDP $\quad$ (works identically for matrix-valued f: flatten, use $\|\cdot\|_F$)
- Post-processing: if $\mathcal{M}$ is μ-GDP, then so is $g \circ \mathcal{M}$ for any g that does not touch the data

From SGD to Prefix Sums

SGD

input data D, steps n, learning rate η

1: init θ_0
2: **for** $t = 1, \ldots, n$ **do**
3: receive $x_t \in D$
4: $g_t \leftarrow \nabla_\theta \ell(\theta_{t-1}, x_t)$
5: $\theta_t \leftarrow \theta_{t-1} - \eta\, g_t$
6: **end for**

SGD

input data D, steps n, learning rate η

1: init θ_0
2: **for** $t = 1, \dots, n$ **do**
3: receive $x_t \in D$
4: $g_t \leftarrow \nabla_\theta \ell(\theta_{t-1}, x_t)$
5: $\theta_t \leftarrow \theta_{t-1} - \eta\, g_t$
6: **end for**

- clipping enforces bounded sensitivity
- batch size 1 for simpler notation; mini-batches work analogously

DP-SGD [Song *et al.* 2013, Abadi *et al.* 2016]

input D, n, η, clip norm ζ, noise multiplier σ

1: init θ_0
2: **for** $t = 1, \dots, n$ **do**
3: receive $x_t \in D$
4: $g_t \leftarrow \mathsf{clip}_\zeta\big(\nabla_\theta \ell(\theta_{t-1}, x_t)\big)$
5: $\widehat{g}_t \leftarrow g_t + \mathcal{N}(0, \nu^2 \mathsf{Id})$ with $\nu = \zeta\sigma$
6: $\theta_t \leftarrow \theta_{t-1} - \eta\, \widehat{g}_t$
7: **end for**

$$\mathsf{clip}_\zeta(v) := \begin{cases} v & \text{for } \|v\| \le \zeta, \\ \frac{\zeta}{\|v\|_2} v & \text{otherwise.} \end{cases}$$

Unrolling SGD

SGD recursion: $\theta_t = \theta_{t-1} - \eta g_t$. Unrolling (with convention $\theta_0 = 0$)

$$\theta_t = -\eta\,(g_1 + g_2 + \cdots + g_t) = -\eta\, s_t, \qquad s_t := \sum_{\tau=1}^{t} g_\tau \quad \text{(prefix sums)}$$

Stack gradients and prefix sums row-wise into matrices:

$$G = \begin{pmatrix} g_1^\top \\ g_2^\top \\ \vdots \\ g_n^\top \end{pmatrix} \in \mathbb{R}^{n\times m}, \quad S = \begin{pmatrix} s_1^\top \\ s_2^\top \\ \vdots \\ s_n^\top \end{pmatrix} = A_{\text{pre}}\, G, \quad A_{\text{pre}} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 1 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{pmatrix}$$

"workload matrix"

(n: number of steps, m: model dimension; typically $m \gg n$)

private training $\Leftrightarrow$ privately estimating $S = A_{\text{pre}} G$

Other workloads I: SGD with weight decay

SGD with weight decay (factor $\alpha \in (0, 1]$):

$$\theta_t = \alpha\, \theta_{t-1} - \eta\, g_t.$$

SGD with weight decay (factor $\alpha \in (0, 1]$):

$$\theta_t = \alpha\, \theta_{t-1} - \eta\, g_t.$$

Unrolling:

$$\theta_t = -\eta \sum_{\tau=1}^{t} \alpha^{t-\tau}\, g_\tau \quad \Longrightarrow \quad \Theta = -\eta\, A_\alpha\, G, \qquad A_\alpha = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ \alpha & 1 & 0 & \cdots & 0 \\ \alpha^2 & \alpha & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \alpha^{n-1} & \alpha^{n-2} & \alpha^{n-3} & \cdots & 1 \end{pmatrix}$$

- lower-triangular Toeplitz matrix with first column $(1, \alpha, \alpha^2, \dots)$
  - $\alpha = 1$ recovers A_{pre}
  - $\alpha < 1$: exponentially decaying influence of old gradients

Other workloads II: SGD with momentum

SGD with momentum (factor $\beta \in [0, 1)$): $m_t = \beta\, m_{t-1} + g_t, \qquad \theta_t = \theta_{t-1} - \eta\, m_t.$

Other workloads II: SGD with momentum

SGD with momentum (factor $\beta \in [0,1)$): $\quad m_t = \beta\, m_{t-1} + g_t, \qquad \theta_t = \theta_{t-1} - \eta\, m_t.$

Unrolling gives $m_t = \sum_{\tau=1}^{t} \beta^{t-\tau} g_\tau$ and $\theta_t = -\eta \sum_{\tau=1}^{t} \underbrace{\frac{1-\beta^{t-\tau+1}}{1-\beta}}_{=:\, a_{t-\tau}} g_\tau$

$$\Theta = -\eta\, A_\beta\, G \quad \text{with} \quad A_\beta = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1+\beta & 1 & 0 & \cdots & 0 \\ 1+\beta+\beta^2 & 1+\beta & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ a_{n-1} & a_{n-2} & a_{n-3} & \cdots & 1 \end{pmatrix}$$

- lower-triangular Toeplitz matrix with first column $(1, a_1, a_2, \dots)$
  - $\beta = 0$ recovers A_{pre},
  - $0 < \beta < 1$: influence of old gradients approaches $\frac{1}{1-\beta}$.

Other workloads III: SGD with learning rate schedule

SGD with time-varying learning rate $\eta_1, \eta_2, \ldots, \eta_n$ (warm-up, cosine decay, ...):

$$\theta_t = \theta_{t-1} - \eta_t \, g_t.$$

Other workloads III: SGD with learning rate schedule

SGD with time-varying learning rate $\eta_1, \eta_2, \ldots, \eta_n$ (warm-up, cosine decay, ...):

$$\theta_t = \theta_{t-1} - \eta_t \, g_t.$$

$$\theta_t = -\sum_{\tau=1}^{t} \eta_\tau \, g_\tau \quad \Longrightarrow \quad A = \begin{pmatrix} \eta_1 & 0 & \cdots & 0 \\ \eta_1 & \eta_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \eta_1 & \eta_2 & \cdots & \eta_n \end{pmatrix}$$

SGD with time-varying learning rate $\eta_1, \eta_2, \ldots, \eta_n$ (warm-up, cosine decay, ...):

$$\theta_t = \theta_{t-1} - \eta_t \, g_t.$$

$$\theta_t = -\sum_{\tau=1}^{t} \eta_\tau \, g_\tau \quad \Longrightarrow \quad A = \begin{pmatrix} \eta_1 & 0 & \cdots & 0 \\ \eta_1 & \eta_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \eta_1 & \eta_2 & \cdots & \eta_n \end{pmatrix}$$

- lower-triangular, but no longer Toeplitz
  - $\eta_t = \eta$ (constant) recovers ηA_{pre}

General problem: private weighted prefix sums

Problem

Given a lower-triangular workload matrix $A \in \mathbb{R}^{n \times n}$ and a stream of inputs $g_1, \ldots, g_n \in \mathbb{R}^m$ with $\|g_t\|_2 \leq 1$, release (with DP) the weighted partial sums

$$s_t = \sum_{\tau=1}^{t} A[t, \tau]\, g_\tau \quad \text{for all } t, \qquad \text{i.e.} \quad S = AG,$$

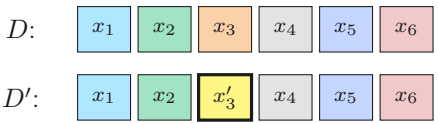
where s_t must be released at step t (it may not depend on the future).

Examples from ML/Stats:

- SGD with weight-decay, momentum, learning-rate schedule (including combinations), ...
- mean and covariance estimation, exponentially moving averages, sliding windows, ...

Caveat: adjacency of gradient streams

Streaming setting: each example is used in exactly one update step.



- $D \simeq D'$ differ in a single element

Caveat: adjacency of gradient streams

Streaming setting: each example is used in exactly one update step.



- $D \simeq D'$ differ in a single element
- But: changing x_t changes g_t, which changes $\theta_t, \theta_{t+1}, \ldots$ and thus *all later* gradients!

Caveat: adjacency of gradient streams

Streaming setting: each example is used in exactly one update step.



- $D \simeq D'$ differ in a single element
- But: changing x_t changes g_t, which changes $\theta_t, \theta_{t+1}, \ldots$ and thus *all later* gradients!

Theorem (adaptive = non-adaptive for Gaussian noise) [Denisov *et al.*, NeurIPS 2022]

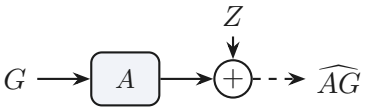
For Gaussian additive noise, the privacy of the *adaptive* process (each θ_t depends on past private outputs) is the same as that of the *non-adaptive* process with fixed sequence $\theta_1, \ldots, \theta_n$.

$\rightarrow$ for the privacy analysis we can treat G as fixed; neighboring $G \simeq G'$ differ in a single row

Enforcing Privacy

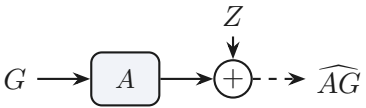
Two ways to privatize AG

Output perturbation $\widehat{AG} = AG + Z$



Two ways to privatize AG

Output perturbation $\widehat{AG} = AG + Z$



Gaussian mechanism:

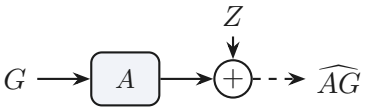
- $Z \sim \mathcal{N}(0; \nu^2 \mathsf{Id})$ with $\nu = \mathrm{sens}(AG)\sigma$
- $\mathrm{sens} = \sup_{G \simeq G'} \|A(G - G')\|_F = \|A\|_{\mathrm{col}}$,
  so $\nu := \sigma \|A\|_{\mathrm{col}}$.

Expected Error:

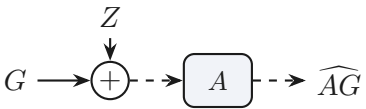
$$\begin{aligned}\mathbb{E}\,\|\widehat{AG} - AG\|_F^2 &= \mathbb{E}\,\|Z\|_F^2 \\ &= nm\,\nu^2 = nm\,\sigma^2 \|A\|_{\mathrm{col}}^2\end{aligned}$$

Two ways to privatize AG

Output perturbation $\widehat{AG} = AG + Z$



Input perturbation $\widehat{AG} = A(G + Z)$



Gaussian mechanism:

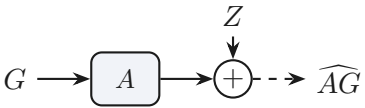
- $Z \sim \mathcal{N}(0; \nu^2 \mathsf{Id})$ with $\nu = \mathrm{sens}(AG)\sigma$
- $\mathrm{sens} = \sup_{G \simeq G'} \|A(G - G')\|_F = \|A\|_{\mathrm{col}}$,
  so $\nu := \sigma \|A\|_{\mathrm{col}}$.

Expected Error:

$$\begin{aligned}\mathbb{E}\,\|\widehat{AG} - AG\|_F^2 &= \mathbb{E}\,\|Z\|_F^2 \\ &= nm\,\nu^2 = nm\,\sigma^2 \|A\|_{\mathrm{col}}^2\end{aligned}$$

Two ways to privatize AG

Output perturbation $\widehat{AG} = AG + Z$



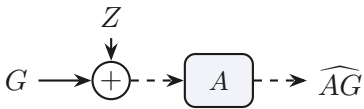
Gaussian mechanism:

- $Z \sim \mathcal{N}(0; \nu^2 \mathsf{Id})$ with $\nu = \mathrm{sens}(AG)\sigma$
- $\mathrm{sens} = \sup_{G \simeq G'} \|A(G - G')\|_F = \|A\|_{\mathrm{col}}$, so $\nu := \sigma\|A\|_{\mathrm{col}}$.

Expected Error:

$$\mathbb{E}\,\|\widehat{AG} - AG\|_F^2 = \mathbb{E}\,\|Z\|_F^2 = nm\,\nu^2 = nm\,\sigma^2\|A\|_{\mathrm{col}}^2$$

Input perturbation $\widehat{AG} = A(G + Z)$



Gaussian mechanism:

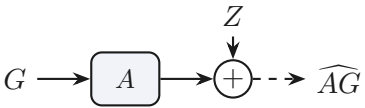
- $Z \sim \mathcal{N}(0; \nu^2 \mathsf{Id})$ with $\nu = \mathrm{sens}(G)\sigma$
- $\mathrm{sens} = \sup_{G \simeq G'} \|G - G'\|_F \leq 1$, so $\nu := \sigma$.

Expected Error:

$$\mathbb{E}\,\|\widehat{AG} - AG\|_F^2 = \mathbb{E}\,\|AZ\|_F^2 = m\nu^2\|A\|_{\mathrm{F}}^2 = m\sigma^2\|A\|_{\mathrm{F}}^2$$

Two ways to privatize AG

Output perturbation $\widehat{AG} = AG + Z$



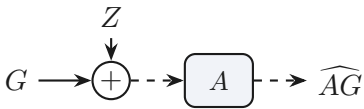
Gaussian mechanism:

- $Z \sim \mathcal{N}(0; \nu^2 \mathsf{Id})$ with $\nu = \mathrm{sens}(AG)\sigma$
- $\mathrm{sens} = \sup_{G \simeq G'} \|A(G - G')\|_F = \|A\|_{\mathrm{col}}$, so $\nu := \sigma\|A\|_{\mathrm{col}}$.

Expected Error:

$$\mathbb{E}\,\|\widehat{AG} - AG\|_F^2 = \mathbb{E}\,\|Z\|_F^2 = nm\,\nu^2 = nm\,\sigma^2\|A\|_{\mathrm{col}}^2$$

Input perturbation $\widehat{AG} = A(G + Z)$



Gaussian mechanism:

- $Z \sim \mathcal{N}(0; \nu^2 \mathsf{Id})$ with $\nu = \mathrm{sens}(G)\sigma$
- $\mathrm{sens} = \sup_{G \simeq G'} \|G - G'\|_F \leq 1$, so $\nu := \sigma$.

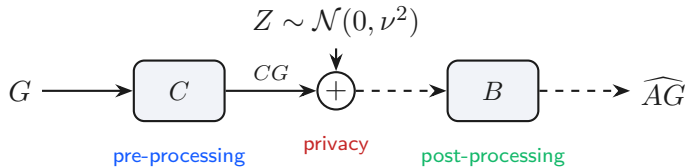
Expected Error:

$$\mathbb{E}\,\|\widehat{AG} - AG\|_F^2 = \mathbb{E}\,\|AZ\|_F^2 = m\nu^2\|A\|_{\mathrm{F}}^2 = m\sigma^2\|A\|_{\mathrm{F}}^2$$

For prefix sum: $\|A_{\mathrm{pre}}\|_{\mathrm{col}}^2 = n$, $\|A_{\mathrm{pre}}\|_{\mathrm{F}}^2 = \frac{n(n+1)}{2}$ $\rightarrow$ input perturbation better, but same rate

Matrix factorization (MF) mechanism [Li, Miklau, Hay, McGregor, Rastogi. VLDB 2015]

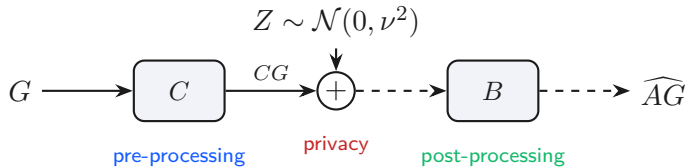
Pick any factorization $A = BC$ with lower-triangular B and C.



$$\widehat{AG} = B\,(CG + Z) \;=\; AG + BZ \;=\; A\Big(G + \overbrace{C^{-1}Z}^{\text{correlated noise}}\Big)$$

Matrix factorization (MF) mechanism [Li, Miklau, Hay, McGregor, Rastogi. VLDB 2015]

Pick any factorization $A = BC$ with lower-triangular B and C.



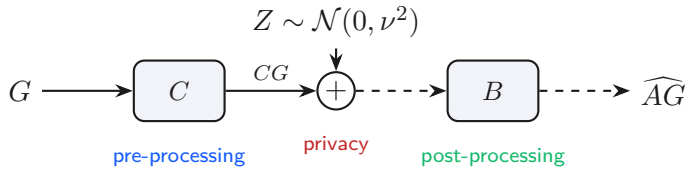
$$\widehat{AG} = B\,(CG + Z) \;=\; AG + BZ \;=\; A\Big(G + \overbrace{C^{-1}Z}^{\text{correlated noise}}\Big)$$

- privacy: Gaussian mechanism $CG + Z$, noise scale determined by $\mathrm{sens}(C)$
- error is BZ: depends on B *and* on the noise scale ν (which depends on C)
- (privacy-normalized) root mean squared error (RMSE)

$$\mathcal{E}(B, C) := \sqrt{\frac{1}{mn\sigma^2}\,\mathbb{E}_Z\,\|\widehat{A}G - AG\|^2} = \frac{1}{\sqrt{n}}\|B\|_{\mathrm{F}}\|C\|_{\mathrm{col}}$$

Matrix factorization (MF) mechanism [Li, Miklau, Hay, McGregor, Rastogi. VLDB 2015]

Pick any factorization $A = BC$ with lower-triangular B and C.



$$\widehat{AG} = B\,(CG + Z) \;=\; AG + BZ \;=\; A\Big(G + \overbrace{C^{-1}Z}^{\text{correlated noise}}\Big)$$

- privacy: Gaussian mechanism $CG + Z$, noise scale determined by $\mathrm{sens}(C)$
- error is BZ: depends on B *and* on the noise scale ν (which depends on C)
- (privacy-normalized) root mean squared error (RMSE)

$$\mathcal{E}(B, C) := \sqrt{\frac{1}{mn\sigma^2}\,\mathbb{E}_Z\,\|\widehat{A}G - AG\|^2} = \frac{1}{\sqrt{n}}\|B\|_{\mathrm{F}}\|C\|_{\mathrm{col}}$$

- input perturbation: $B = A$, $C = \mathsf{Id}$. for A_{pre}: $\mathcal{E}(A_{\mathrm{pre}}, \mathsf{Id}) = \Theta(\sqrt{n})$
- output perturbation: $B = \mathsf{Id}$, $C = A$. for A_{pre}: $\mathcal{E}(\mathsf{Id}, A_{\mathrm{pre}}) = \Theta(\sqrt{n})$

Can MF do better than input perturbation and output perturbation?

Example for A_{pre}: 1-parameter family of factorizations, for $0 \le \lambda < 1$ [Pillutla *et al.* arXiv:2506.08201]:

$$C_\lambda = \begin{pmatrix} 1 & & & & \\ \lambda & 1 & & & \\ \lambda^2 & \lambda & 1 & & \\ \vdots & \ddots & \ddots & \ddots & \end{pmatrix}, \quad C_\lambda^{-1} = \begin{pmatrix} 1 & & & \\ -\lambda & 1 & & \\ & -\lambda & 1 & \\ & & \ddots & \ddots \end{pmatrix}, \quad B_\lambda = A_{\mathrm{pre}} C_\lambda^{-1} = \begin{pmatrix} 1 & & & \\ 1-\lambda & 1 & & \\ 1-\lambda & 1-\lambda & 1 & \\ \vdots & & \ddots & \ddots \end{pmatrix}$$

$\lambda = 0$: input perturbation, $\lambda \to 1$: output perturbation

Can MF do better than input perturbation and output perturbation?

Example for A_{pre}: 1-parameter family of factorizations, for $0 \le \lambda < 1$ [Pillutla *et al.* arXiv:2506.08201]:

$$C_\lambda = \begin{pmatrix} 1 & & & & \\ \lambda & 1 & & & \\ \lambda^2 & \lambda & 1 & & \\ \vdots & \ddots & \ddots & \ddots & \end{pmatrix}, \quad C_\lambda^{-1} = \begin{pmatrix} 1 & & & \\ -\lambda & 1 & & \\ & -\lambda & 1 & \\ & & \ddots & \ddots \end{pmatrix}, \quad B_\lambda = A_{\mathrm{pre}} C_\lambda^{-1} = \begin{pmatrix} 1 & & & \\ 1-\lambda & 1 & & \\ 1-\lambda & 1-\lambda & 1 & \\ \vdots & & \ddots & \ddots \end{pmatrix}$$

$\lambda = 0$: input perturbation, $\lambda \to 1$: output perturbation

$$\|C_\lambda\|_{\mathrm{col}}^2 = \sum_{j=0}^{n-1} \lambda^{2j} = \frac{1-\lambda^{2n}}{1-\lambda^2}, \qquad \|B_\lambda\|_{\mathrm{F}}^2 = n + \frac{n(n-1)}{2}(1-\lambda)^2$$

$$\mathcal{E}(B_\lambda, C_\lambda) = \sqrt{\frac{1}{n}\|C_\lambda\|_{\mathrm{col}}^2 \|B_\lambda\|_{\mathrm{F}}^2}$$

Can MF do better than input perturbation and output perturbation?

Example for A_{pre}: 1-parameter family of factorizations, for $0 \le \lambda < 1$ [Pillutla *et al.* arXiv:2506.08201]:

$$C_\lambda = \begin{pmatrix} 1 & & & & \\ \lambda & 1 & & & \\ \lambda^2 & \lambda & 1 & & \\ \vdots & \ddots & \ddots & \ddots & \end{pmatrix}, \quad C_\lambda^{-1} = \begin{pmatrix} 1 & & & \\ -\lambda & 1 & & \\ & -\lambda & 1 & \\ & & \ddots & \ddots \end{pmatrix}, \quad B_\lambda = A_{\mathrm{pre}} C_\lambda^{-1} = \begin{pmatrix} 1 & & & \\ 1-\lambda & 1 & & \\ 1-\lambda & 1-\lambda & 1 & \\ \vdots & & \ddots & \ddots \end{pmatrix}$$

$\lambda = 0$: input perturbation, $\lambda \to 1$: output perturbation

$$\|C_\lambda\|_{\mathrm{col}}^2 = \sum_{j=0}^{n-1} \lambda^{2j} = \frac{1-\lambda^{2n}}{1-\lambda^2}, \qquad \|B_\lambda\|_{\mathrm{F}}^2 = n + \frac{n(n-1)}{2}(1-\lambda)^2$$

$$\mathcal{E}(B_\lambda, C_\lambda) = \sqrt{\frac{1}{n}\|C_\lambda\|_{\mathrm{col}}^2 \|B_\lambda\|_{\mathrm{F}}^2}$$

E.g. for $\lambda^* = 1 - \frac{2}{\sqrt{n}}$: $\quad \Rightarrow \quad \mathcal{E}(B_{\lambda^*}, C_{\lambda^*}) = \Theta(n^{1/4})$

$\to$ improvement over $\Theta(n^{1/2})$ of input/output perturbation

Can we do even better?

Idea: split the workload evenly: $A = C \cdot C$, i.e. $B = C = A^{1/2}$ (lower-triangular square root)

Can we do even better?

Idea: split the workload evenly: $A = C \cdot C$, i.e. $B = C = A^{1/2}$ (lower-triangular square root)

E.g., SGD workload with *weight decay* α and *momentum* $\beta < \alpha$ (prefix sum: $\alpha = 1, \beta = 0$).

$$A_{\alpha,\beta} = \begin{pmatrix} a_0 & & & \\ a_1 & a_0 & & \\ a_2 & a_1 & a_0 & \\ \vdots & \ddots & \ddots & \ddots \end{pmatrix}, \quad \text{with} \quad a_j = \frac{\alpha^{j+1} - \beta^{j+1}}{\alpha - \beta}$$

has a closed-form Toeplitz square root:

$$C = A^{1/2}_{\alpha,\beta} = \begin{pmatrix} c_0 & & & \\ c_1 & c_0 & & \\ c_2 & c_1 & c_0 & \\ \vdots & \ddots & \ddots & \ddots \end{pmatrix}, \quad \text{with} \quad c_j = \sum_{i=0}^{j} \alpha^{j-i} r_{j-i} r_i \beta^i \quad \text{for} \quad r_i = \left| \binom{-1/2}{i} \right|.$$

Can we do even better?

Idea: split the workload evenly: $A = C \cdot C$, i.e. $B = C = A^{1/2}$ (lower-triangular square root)

Theorem – Square Root Factorization [Henzinger, Upadhyay, Upadhyay, SODA 2024], [Kalinin, L, NeurIPS 2024]

For SGD with momentum β and weight decay $\alpha > \beta$, let $C = A^{1/2}_{\alpha,\beta}$. Then

$$\begin{aligned} \mathbf{max}\{1, \tfrac{\log(n+1)-1}{4}\} \;\leq\; \mathcal{E}(C,C) \;\leq\; \tfrac{1+\log n}{(1-\beta)^2} \qquad &\text{for } \alpha = 1 \\ 1 \;\leq\; \mathcal{E}(C,C) \;\leq\; \tfrac{1}{(\alpha-\beta)^2} \log \tfrac{1}{1-\alpha^2} \qquad &\text{for } \alpha < 1 \end{aligned}$$

Can we do even better?

Idea: split the workload evenly: $A = C \cdot C$, i.e. $B = C = A^{1/2}$ (lower-triangular square root)

Theorem – Square Root Factorization [Henzinger, Upadhyay, Upadhyay, SODA 2024], [Kalinin, L, NeurIPS 2024]

For SGD with momentum β and weight decay $\alpha > \beta$, let $C = A^{1/2}_{\alpha,\beta}$. Then

$$\begin{aligned} \max\{1, \tfrac{\log(n+1)-1}{4}\} \;\le\; \mathcal{E}(C,C) \;&\le\; \tfrac{1+\log n}{(1-\beta)^2} \qquad \text{for } \alpha = 1 \\ 1 \;\le\; \mathcal{E}(C,C) \;&\le\; \tfrac{1}{(\alpha-\beta)^2} \log \tfrac{1}{1-\alpha^2} \qquad \text{for } \alpha < 1 \end{aligned}$$

- for prefix sums ($\alpha = 1, \beta = 0$): RMSE $= \Theta(\log n)$,
  - input/output perturbation: $\Theta(n^{1/2})$, 1-parameter family: $\Theta(n^{1/4})$
- general lower bound for *any* factorization: $\Omega(\log n)$ for $\alpha = 1$, $\Omega(1)$ for $\alpha < 1$
  $\rightarrow$ square root is rate-optimal

Matrix Factorization for Learning

DP-SGD with Correlated Noise (aka DP-FTRL, MF-DP-SGD)

input data D, steps n, learning rate η, clip norm ζ, noise mult. σ, strategy matrix C

1: init θ_0
2: **for** $t = 1, \dots, n$ **do**
3: receive $x_t \in D$
4: $g_t \leftarrow \mathsf{clip}_\zeta(\nabla_\theta \ell(\theta_{t-1}, x_t))$
5: sample new i.i.d. noise $z_t \sim \mathcal{N}(0, \nu^2 \, \mathsf{Id}_m)$ with $\nu = \|C\|_{\mathrm{col}} \zeta \sigma$
6: correlated noise $\tilde{z}_t \leftarrow \sum_{\tau=1}^{t} C^{-1}[t, \tau] \, z_\tau$
7: $\widehat{g}_t \leftarrow g_t + \tilde{z}_t$
8: $\theta_t \leftarrow \theta_{t-1} - \eta \, \widehat{g}_t$ // or any other optimizer step
9: **end for**

- C is needed to determine noise strength ν (line 5)
- C^{-1} is needed to compute correlated noise $\tilde{z}_t$ (line 6)

Intuition: noise cancellation

For example, for $0 \le \lambda < 1$, with $C_\lambda = \begin{pmatrix} 1 & & & \\ \lambda & 1 & & \\ \lambda^2 & \lambda & 1 & \\ \vdots & \ddots & \ddots & \ddots \end{pmatrix}$, $\quad C_\lambda^{-1} = \begin{pmatrix} 1 & & & \\ -\lambda & 1 & & \\ & -\lambda & 1 & \\ & & \ddots & \ddots \end{pmatrix}$

Noisy gradients are

$$\widehat{g}_t = g_t + z_t - \lambda\, z_{t-1}$$

- each step cancels a λ-fraction of the previously added noise
- trade-off: larger $\lambda \Rightarrow$ larger $\|C_\lambda\|_{\mathrm{col}} \Rightarrow$ larger noise ν needed for same privacy

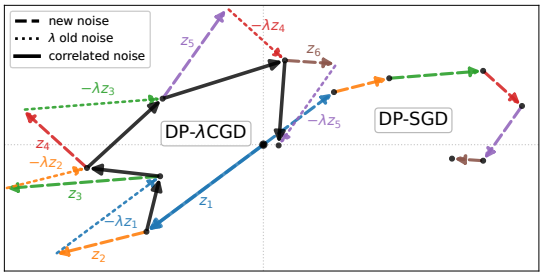
Intuition: noise cancellation

For example, for $0 \le \lambda < 1$, with $C_\lambda = \begin{pmatrix} 1 & & & \\ \lambda & 1 & & \\ \lambda^2 & \lambda & 1 & \\ \vdots & \ddots & \ddots & \ddots \end{pmatrix}$, $C_\lambda^{-1} = \begin{pmatrix} 1 & & & \\ -\lambda & 1 & & \\ & -\lambda & 1 & \\ & & \ddots & \ddots \end{pmatrix}$

Noisy gradients are

$$\widehat{g}_t = g_t + z_t - \lambda\, z_{t-1}$$

- each step cancels a λ-fraction of the previously added noise
- trade-off: larger $\lambda \Rightarrow$ larger $\|C_\lambda\|_{\text{col}} \Rightarrow$ larger noise ν needed for same privacy



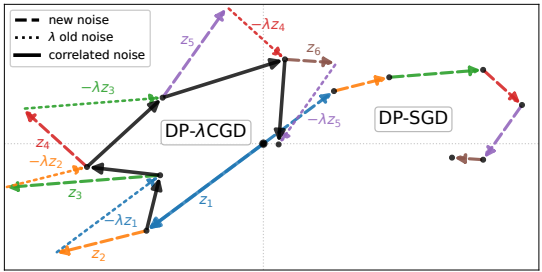
Intuition: noise cancellation

For example, for $0 \le \lambda < 1$, with $C_\lambda = \begin{pmatrix} 1 & & & \\ \lambda & 1 & & \\ \lambda^2 & \lambda & 1 & \\ \vdots & \ddots & \ddots & \ddots \end{pmatrix}$, $C_\lambda^{-1} = \begin{pmatrix} 1 & & & \\ -\lambda & 1 & & \\ & -\lambda & 1 & \\ & & \ddots & \ddots \end{pmatrix}$

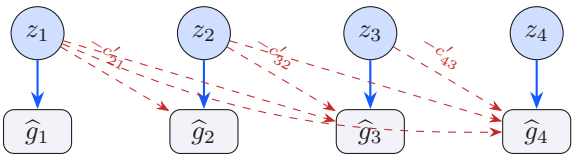
Noisy gradients are

$$\widehat{g}_t = g_t + z_t - \lambda\, z_{t-1}$$

- each step cancels a λ-fraction of the previously added noise
- trade-off: larger $\lambda \Rightarrow$ larger $\|C_\lambda\|_{\text{col}} \Rightarrow$ larger noise ν needed for same privacy



General C^{-1}: each step t cancels a mixture of *all* previous noise vectors.



fresh noise
anti-correlated correction

Correlated Noise for Actual Learning

Correlated Noise for Actual Learning

For practical tasks, a good factorization needs more than small $\mathcal{E}(B, C)$:

- handling repeated data participation
- compute efficiency
- real-world accuracy

Consideration I: Multiple Participations

Practical training often needs to allow for multiple participations:

- item-level privacy: training for multiple epochs
- user-level privacy: data from same user contributes at different times

Consideration I: Multiple Participations

Practical training often needs to allow for multiple participations:

- item-level privacy: training for multiple epochs
- user-level privacy: data from same user contributes at different times

For non-trivial guarantees, we need some control over participation patterns:

Definition: (k, b)-min-separated participation [Choquette-Choo *et al.*, NeurIPS 2023]

- each item/user contributes at most k times
- two participations by the same item/user lie at least b steps apart

- $k = 1$ or $b \geq n$: single participation
- $k \geq n$ and $b = 1$: no restriction on participation

Lemma: Sensitivity under multiple participations

Let Π be the set of possible participation patterns, i.e. each $\Pi \ni \pi \subset \{1, \dots, n\}$ is a possible set of steps to which an item contributes. Then

$$\mathrm{sens}_\Pi(C) = \max_{\pi \in \Pi} \max_{\|u_t\| \le 1} \Big\| \sum_{t \in \pi} C[:, t]\, u_t^\top \Big\|_F \overset{\text{if } [C^\top C]_{s,t} \ge 0}{=} \max_{\pi \in \Pi} \sqrt{\sum_{s,t \in \pi} (C^\top C)[s,t]}$$

- exact sens_Π can be hard to compute in general; efficient in structured cases
- columns of C belonging to the same item add up $\rightarrow$ column correlations matter
- optimal factorizations for single-participation can be bad for multi-epoch training

Lemma: Sensitivity under multiple participations

Let Π be the set of possible participation patterns, i.e. each $\Pi \ni \pi \subset \{1, \dots, n\}$ is a possible set of steps to which an item contributes. Then

$$\mathrm{sens}_\Pi(C) = \max_{\pi \in \Pi} \max_{\|u_t\| \le 1} \Big\| \sum_{t \in \pi} C[:, t]\, u_t^\top \Big\|_F \overset{\text{if } [C^\top C]_{s,t} \ge 0}{=} \max_{\pi \in \Pi} \sqrt{\sum_{s,t \in \pi} (C^\top C)[s,t]}$$

- exact sens_Π can be hard to compute in general; efficient in structured cases
- columns of C belonging to the same item add up $\rightarrow$ column correlations matter
- optimal factorizations for single-participation can be bad for multi-epoch training

Natural idea: choose C such that columns separated by b steps are orthogonal

Banded mechanisms [Choquette-Choo *et al.*, NeurIPS 2023]

Idea: make C p-banded: $C[t,\tau] = 0$ for $t - \tau \geq p$.

- $p = 1$: independent noise; $p = n$: dense mechanism
- memory: $O(pn)$ instead of $O(n^2)$

Under (k, b)-min-separation:

- $p \leq b$ makes b-min separated columns orthogonal

banded C $(p = 3)$

$$\begin{pmatrix} c_{1,1} & & & & & \\ c_{2,1} & c_{2,2} & & & & \\ c_{3,1} & c_{3,2} & c_{3,3} & & & \\ 0 & c_{4,2} & c_{4,3} & c_{4,4} & & \\ 0 & 0 & c_{5,3} & c_{5,4} & c_{5,5} & \\ \vdots & & \ddots & \ddots & \ddots & \ddots \end{pmatrix}$$

Banded mechanisms [Choquette-Choo *et al.*, NeurIPS 2023]

Idea: make C p-banded: $C[t, \tau] = 0$ for $t - \tau \geq p$.

- $p = 1$: independent noise; $p = n$: dense mechanism
- memory: $O(pn)$ instead of $O(n^2)$

Under (k, b)-min-separation:

- $p \leq b$ makes b-min separated columns orthogonal

banded C $(p = 3)$

$$\begin{pmatrix} c_{1,1} & & & & & \\ c_{2,1} & c_{2,2} & & & & \\ c_{3,1} & c_{3,2} & c_{3,3} & & & \\ 0 & c_{4,2} & c_{4,3} & c_{4,4} & & \\ 0 & 0 & c_{5,3} & c_{5,4} & c_{5,5} & \\ \vdots & & \ddots & \ddots & \ddots & \ddots \end{pmatrix}$$

p-Banded Square Root (p-BSR)

For workload A, set $C^{|p|}$ as identical to $A^{1/2}$ for the central p bands and 0 elsewhere.

- E.g., SGD with momentum β and weight decay α: $A_{\alpha,\beta}$
  $\rightarrow C^{|p|}$ is Toeplitz with closed-form expressions for entries

Theorem (RMSE of BSR)

Let $C^{|p|}$ be the p-BSR of $A_{\alpha,\beta}$ with $\alpha > \beta$. Then, under (k, b)-min-separation with $2 \leq p \leq b$:

$$\mathcal{E}_{\Pi}(B^{|p|}, C^{|p|}) = \begin{cases} O_{\beta}\left(\sqrt{\frac{nk \log p}{p}}\right) + O_{\beta}\left(\sqrt{k} \log p\right) & \alpha = 1, \\ O_{\beta,\alpha}\left(\sqrt{k}\right) & \alpha < 1. \end{cases}$$

Theorem (RMSE of BSR)

Let $C^{|p|}$ be the p-BSR of $A_{\alpha,\beta}$ with $\alpha > \beta$. Then, under (k, b)-min-separation with $2 \le p \le b$:

$$\mathcal{E}_{\Pi}(B^{|p|}, C^{|p|}) = \begin{cases} O_\beta\left(\sqrt{\frac{nk\log p}{p}}\right) + O_\beta\left(\sqrt{k}\log p\right) & \alpha = 1, \\ O_{\beta,\alpha}\left(\sqrt{k}\right) & \alpha < 1. \end{cases}$$

For comparison (with $p = b$ and $k = n/b$):

	$\alpha = 1$	$\alpha < 1$
lower bound, any factorization	$\ge \frac{1}{\pi}\sqrt{k}\log(n{+}1)$	$\ge \sqrt{k}$
square root (non-banded)	$\Theta_\beta\left(k\sqrt{\log n} + \sqrt{k}\log n\right)$	$\Theta_{\alpha,\beta}(\sqrt{k})$
input perturbation	$\ge \sqrt{nk/2}$	$\ge \sqrt{k}$
output perturbation	$\ge k\sqrt{n/3}$	$\ge \sqrt{k}$

Theorem (RMSE of BSR)

Let $C^{|p|}$ be the p-BSR of $A_{\alpha,\beta}$ with $\alpha > \beta$. Then, under (k, b)-min-separation with $2 \le p \le b$:

$$\mathcal{E}_{\Pi}(B^{|p|}, C^{|p|}) = \begin{cases} O_\beta\Big(\sqrt{\frac{nk\log p}{p}}\Big) + O_\beta\big(\sqrt{k}\log p\big) & \alpha = 1, \\ O_{\beta,\alpha}\big(\sqrt{k}\big) & \alpha < 1. \end{cases}$$

For comparison (with $p = b$ and $k = n/b$):

	$\alpha = 1$	$\alpha < 1$
lower bound, any factorization	$\ge \frac{1}{\pi}\sqrt{k}\log(n{+}1)$	$\ge \sqrt{k}$
square root (non-banded)	$\Theta_\beta\big(k\sqrt{\log n} + \sqrt{k}\log n\big)$	$\Theta_{\alpha,\beta}\big(\sqrt{k}\big)$
input perturbation	$\ge \sqrt{nk/2}$	$\ge \sqrt{k}$
output perturbation	$\ge k\sqrt{n/3}$	$\ge \sqrt{k}$

For $\alpha = 1$: BSR better than baselines, including square-root;

$$O(k\sqrt{\log b} + \sqrt{k}\log b) \quad \text{vs.} \quad O(k\sqrt{\log n} + \sqrt{k}\log n)$$

MF-DP-SGD Main Loop

for $t = 1, \ldots, n$ **do**
  receive $x_t \in D$
  $g_t \leftarrow \mathsf{clip}_\zeta(\nabla_\theta \ell(\theta_{t-1}, x_t))$
  sample new noise $z_t \sim \mathcal{N}(0, \nu^2\, \mathsf{Id}_m)$
  correlate noise $\tilde{z}_t \leftarrow \sum_{\tau=1}^{t} C^{-1}[t, \tau]\, z_\tau$
  $\widehat{g}_t \leftarrow g_t + \tilde{z}_t$
  $\theta_t \leftarrow \theta_{t-1} - \eta\, \widehat{g}_t$
end for

Consideration II: Computational Efficiency

MF-DP-SGD Main Loop

1: **for** $t = 1, \dots, n$ **do**
2: receive $x_t \in D$
3: $g_t \leftarrow \mathsf{clip}_\zeta(\nabla_\theta \ell(\theta_{t-1}, x_t))$
4: sample new noise $z_t \sim \mathcal{N}(0, \nu^2\, \mathsf{Id}_m)$
5: correlate noise $\tilde{z}_t \leftarrow \sum_{\tau=1}^{t} C^{-1}[t, \tau]\, z_\tau$
6: $\widehat{g}_t \leftarrow g_t + \tilde{z}_t$
7: $\theta_t \leftarrow \theta_{t-1} - \eta\, \widehat{g}_t$
8: **end for**

Per-step complexity:

- $O(m)$ memory and time to compute z_t
- $O(m)$ memory and time to compute θ_t
- $O(nm)$ time per step to compute $\tilde{z}_t$

One-time complexity:

- $O(n^2)$ memory to store C^{-1}
- $O(nm)$ memory to store $z_1, \dots, z_n$

Typical ML regime:

- model dimension m: millions to billions, number of steps n: (tens of) thousands
- neither $O(n^2)$ nor $O(nm)$ overhead are acceptable

Back to Square Roots: banding the *inverse* [Kalinin, McKenna, Upadhyay, L. ICLR 2026]

$$\text{correlate noise } \tilde{z}_t \leftarrow \sum_{\tau=1}^{t} C^{-1}[t,\tau]\, z_\tau \qquad (*)$$

Design choices to increase efficiency:

- if A and C are Toeplitz, then C^{-1} is also Toeplitz $\rightarrow$ only $O(n)$ storage
- noise z_τ can be generated on-the-fly from *seeds*
  $\rightarrow$ only $O(m)$ memory, but $O(nm)$ compute per step
- if C is also p-banded, C^{-1} would still be dense,
  but one can solve $(*)$ in $O(mp)$ per step by banded elimination

Back to Square Roots: banding the *inverse* [Kalinin, McKenna, Upadhyay, L. ICLR 2026]

$$\text{correlate noise } \tilde{z}_t \leftarrow \sum_{\tau=1}^{t} C^{-1}[t,\tau]\, z_\tau \qquad (*)$$

Design choices to increase efficiency:

- if A and C are Toeplitz, then C^{-1} is also Toeplitz $\rightarrow$ only $O(n)$ storage
- noise z_τ can be generated on-the-fly from *seeds*
  $\rightarrow$ only $O(m)$ memory, but $O(nm)$ compute per step
- if C is also p-banded, C^{-1} would still be dense,
  but one can solve $(*)$ in $O(mp)$ per step by banded elimination

Simpler alternative: instead of C, make C^{-1} p-banded!

- compute complexity: $O(pm)$,
- memory: $O(pm)$, or $O(m)$ with on-the-fly noise regeneration

p-Banded Inverse Square Root (p-BISR)

$$\tilde{C} := A^{1/2} \;\to\; \tilde{C}^{-1} \;\to\; \text{keep } p \text{ bands} \;\to\; \text{invert back} =: C, \qquad B := AC^{-1}$$

Example: $A = A_{\text{pre}} \in \mathbb{R}^{5\times 5}$, $p = 3$.

$$A = \begin{pmatrix} 1 & & & & \\ 1 & 1 & & & \\ 1 & 1 & 1 & & \\ 1 & 1 & 1 & 1 & \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix} \xrightarrow{\text{sqrt}} \tilde{C} = \begin{pmatrix} 1 & & & & \\ \frac{1}{2} & 1 & & & \\ \frac{3}{8} & \frac{1}{2} & 1 & & \\ \frac{5}{16} & \frac{3}{8} & \frac{1}{2} & 1 & \\ \frac{35}{128} & \frac{5}{16} & \frac{3}{8} & \frac{1}{2} & 1 \end{pmatrix} \xrightarrow{\text{invert}} \tilde{C}^{-1} = \begin{pmatrix} 1 & & & & \\ -\frac{1}{2} & 1 & & & \\ -\frac{1}{8} & -\frac{1}{2} & 1 & & \\ -\frac{1}{16} & -\frac{1}{8} & -\frac{1}{2} & 1 & \\ -\frac{5}{128} & -\frac{1}{16} & -\frac{1}{8} & -\frac{1}{2} & 1 \end{pmatrix}$$

keep $p = 3$ bands ↓

$$C = \begin{pmatrix} 1 & & & & \\ \frac{1}{2} & 1 & & & \\ \frac{3}{8} & \frac{1}{2} & 1 & & \\ \frac{1}{4} & \frac{3}{8} & \frac{1}{2} & 1 & \\ \frac{11}{64} & \frac{1}{4} & \frac{3}{8} & \frac{1}{2} & 1 \end{pmatrix} \xleftarrow{\text{invert}} (\tilde{C}^{-1})^{|p|} = \begin{pmatrix} 1 & & & & \\ -\frac{1}{2} & 1 & & & \\ -\frac{1}{8} & -\frac{1}{2} & 1 & & \\ 0 & -\frac{1}{8} & -\frac{1}{2} & 1 & \\ 0 & 0 & -\frac{1}{8} & -\frac{1}{2} & 1 \end{pmatrix}$$

- C itself is *not* banded, but its inverse is.

Back to Square Roots: banding the *inverse* [Kalinin, McKenna, Upadhyay, L. ICLR 2026]

[N. P. Kalinin, R. McKenna, J. Upadhyay, C. H. Lampert, "Back to Square Roots: An Optimal Bound on the Matrix Factorization Error for Multi-Epoch DP-SGD", ICLR 2026]

Theorem (RMSE of BISR)

Let C be the p-BISR of $A_{\alpha,\beta}$. Then, under (k, b)-min-separation with $1 \le b \le n$,

$$\mathcal{E}_{\Pi}(B, C) = O_{\beta}\Big(\sqrt{k}\log p + \sqrt{\frac{nk}{b}} + \sqrt{\frac{nk\log p}{p}} + \sqrt{\frac{kp\log p}{b}}\Big) \qquad \text{for } \alpha = 1.$$

Specifically, with $p^* = O(b\log b)$, for $k = \lceil n/b \rceil$: $\quad \mathcal{E}_{\Pi}(B^*, C^*) = O_{\beta}(\sqrt{k}\log n + k)$.

Back to Square Roots: banding the *inverse* [Kalinin, McKenna, Upadhyay, L. ICLR 2026]

[N. P. Kalinin, R. McKenna, J. Upadhyay, C. H. Lampert, "Back to Square Roots: An Optimal Bound on the Matrix Factorization Error for Multi-Epoch DP-SGD", ICLR 2026]

Theorem (RMSE of BISR)

Let C be the p-BISR of $A_{\alpha,\beta}$. Then, under (k,b)-min-separation with $1 \le b \le n$,

$$\mathcal{E}_\Pi(B,C) = O_\beta\Big(\sqrt{k}\log p + \sqrt{\frac{nk}{b}} + \sqrt{\frac{nk\log p}{p}} + \sqrt{\frac{kp\log p}{b}}\Big) \qquad \text{for } \alpha = 1.$$

Specifically, with $p^* = O(b\log b)$, for $k = \lceil n/b \rceil$: $\quad \mathcal{E}_\Pi(B^*,C^*) = O_\beta(\sqrt{k}\log n + k)$.

Theorem (Lower bounds)

For any factorization $A_{\alpha,\beta} = BC$ it holds under (k,b)-min-separation with $k = \lceil n/b \rceil$:

$$\mathcal{E}_\Pi(B,C) = \Omega(\sqrt{k}\log n + k) \quad \text{for } \alpha = 1.$$

$\rightarrow$ p^*-BISR is asymptotically rate-optimal.

Consideration III: Real-World Accuracy

Observation:

- for privacy, DP-SGD is equivalent to prefix-sum of fixed vectors.
- for accuracy, it is not! DP-SGD can perform some "implicit denoising".

Consideration III: Real-World Accuracy

Observation:

- for privacy, DP-SGD is equivalent to prefix-sum of fixed vectors.
- for accuracy, it is not! DP-SGD can perform some "implicit denoising".

Illustrative example: GD on $\ell(\theta) = \frac{1}{2}\|\theta - \theta^*\|^2$

Non-Private

$$g_t = \nabla \ell(\theta_{t-1}, x_t) = \theta_{t-1} - \theta^*$$

$$\theta_t = \theta_{t-1} - \eta g_t = -\eta \sum_{\tau=1}^{t} g_t$$

Consideration III: Real-World Accuracy

Observation:

- for privacy, DP-SGD is equivalent to prefix-sum of fixed vectors.
- for accuracy, it is not! DP-SGD can perform some "implicit denoising".

Illustrative example: GD on $\ell(\theta) = \frac{1}{2}\|\theta - \theta^*\|^2$

Non-Private

$$g_t = \nabla \ell(\theta_{t-1}, x_t) = \theta_{t-1} - \theta^*$$

$$\theta_t = \theta_{t-1} - \eta g_t = -\eta \sum_{\tau=1}^{t} g_t$$

$$= (1-\eta)\theta_{t-1} + \eta\theta^*$$

$$= \theta^* + (1-\eta)^t(\theta_0 - \theta^*)$$

Consideration III: Real-World Accuracy

Observation:

- for privacy, DP-SGD is equivalent to prefix-sum of fixed vectors.
- for accuracy, it is not! DP-SGD can perform some "implicit denoising".

Illustrative example: GD on $\ell(\theta) = \frac{1}{2}\|\theta - \theta^*\|^2$

Non-Private	"Prefix-Sum Private"
$g_t = \nabla \ell(\theta_{t-1}, x_t) = \theta_{t-1} - \theta^*$	$\tilde{g}_t = g_t + z_t$
$\theta_t = \theta_{t-1} - \eta g_t = -\eta \sum_{\tau=1}^{t} g_t$	$\tilde{\theta}_t = \tilde{\theta}_{t-1} - \eta \tilde{g}_t$
$= (1-\eta)\theta_{t-1} + \eta\theta^*$	$= -\eta \sum_{\tau=1}^{t} \tilde{g}_t$
$= \theta^* + (1-\eta)^t(\theta_0 - \theta^*)$	$= \theta_t - \eta \sum_{\tau=1}^{t} z_t$

Consideration III: Real-World Accuracy

Observation:

- for privacy, DP-SGD is equivalent to prefix-sum of fixed vectors.
- for accuracy, it is not! DP-SGD can perform some "implicit denoising".

Illustrative example: GD on $\ell(\theta) = \frac{1}{2}\|\theta - \theta^*\|^2$

Non-Private

$$g_t = \nabla \ell(\theta_{t-1}, x_t) = \theta_{t-1} - \theta^*$$

$$\theta_t = \theta_{t-1} - \eta g_t = -\eta \sum_{\tau=1}^{t} g_t$$

$$= (1-\eta)\theta_{t-1} + \eta\theta^*$$

$$= \theta^* + (1-\eta)^t(\theta_0 - \theta^*)$$

"Prefix-Sum Private"

$$\tilde{g}_t = g_t + z_t$$

$$\tilde{\theta}_t = \tilde{\theta}_{t-1} - \eta\tilde{g}_t$$

$$= -\eta \sum_{\tau=1}^{t} \tilde{g}_t$$

$$= \theta_t - \eta \sum_{\tau=1}^{t} z_t$$

Actual DP-GD

$$\widehat{g}_t = \nabla \ell(\widehat{\theta}_{t-1}, x_t) + z_t = \widehat{\theta}_{t-1} - \theta^* + z_t$$

$$\widehat{\theta}_t = \widehat{\theta}_{t-1} - \eta\widehat{g}_t = -\eta \sum_{\tau=1}^{t} \widehat{g}_t$$

Consideration III: Real-World Accuracy

Observation:

- for privacy, DP-SGD is equivalent to prefix-sum of fixed vectors.
- for accuracy, it is not! DP-SGD can perform some "implicit denoising".

Illustrative example: GD on $\ell(\theta) = \frac{1}{2}\|\theta - \theta^*\|^2$

Non-Private

$$g_t = \nabla \ell(\theta_{t-1}, x_t) = \theta_{t-1} - \theta^*$$

$$\theta_t = \theta_{t-1} - \eta g_t = -\eta \sum_{\tau=1}^{t} g_t$$

$$= (1-\eta)\theta_{t-1} + \eta\theta^*$$

$$= \theta^* + (1-\eta)^t(\theta_0 - \theta^*)$$

"Prefix-Sum Private"

$$\tilde{g}_t = g_t + z_t$$

$$\tilde{\theta}_t = \tilde{\theta}_{t-1} - \eta\tilde{g}_t$$

$$= -\eta \sum_{\tau=1}^{t} \tilde{g}_t$$

$$= \theta_t - \eta \sum_{\tau=1}^{t} z_t$$

Actual DP-GD

$$\widehat{g}_t = \nabla \ell(\widehat{\theta}_{t-1}, x_t) + z_t = \widehat{\theta}_{t-1} - \theta^* + z_t$$

$$\widehat{\theta}_t = \widehat{\theta}_{t-1} - \eta\widehat{g}_t = -\eta \sum_{\tau=1}^{t} \widehat{g}_t$$

$$= (1-\eta)\widehat{\theta}_{t-1} + \eta\theta^* - \eta z_t$$

$$= \theta_t - \eta \sum_{\tau=1}^{t} (1-\eta)^{t-\tau} \; z_\tau$$

Consideration III: Real-World Accuracy

Observation:

- for privacy, DP-SGD is equivalent to prefix-sum of fixed vectors.
- for accuracy, it is not! DP-SGD can perform some "implicit denoising".

Illustrative example: GD on $\ell(\theta) = \frac{1}{2}\|\theta - \theta^*\|^2$

Non-Private

$$g_t = \nabla \ell(\theta_{t-1}, x_t) = \theta_{t-1} - \theta^*$$

$$\theta_t = \theta_{t-1} - \eta g_t = -\eta \sum_{\tau=1}^{t} g_t$$
$$= (1-\eta)\theta_{t-1} + \eta\theta^*$$
$$= \theta^* + (1-\eta)^t(\theta_0 - \theta^*)$$

"Prefix-Sum Private"

$$\tilde{g}_t = g_t + z_t$$
$$\tilde{\theta}_t = \tilde{\theta}_{t-1} - \eta\tilde{g}_t$$
$$= -\eta \sum_{\tau=1}^{t} \tilde{g}_t$$
$$= \theta_t - \eta \underbrace{\sum_{\tau=1}^{t} z_t}_{\text{unbounded variance}}$$

Actual DP-GD

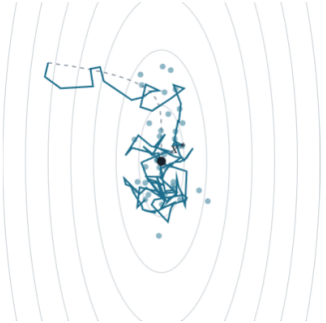
$$\widehat{g}_t = \nabla \ell(\widehat{\theta}_{t-1}, x_t) + z_t = \widehat{\theta}_{t-1} - \theta^* + z_t$$

$$\widehat{\theta}_t = \widehat{\theta}_{t-1} - \eta\widehat{g}_t = -\eta \sum_{\tau=1}^{t} \widehat{g}_t$$
$$= (1-\eta)\widehat{\theta}_{t-1} + \eta\theta^* - \eta z_t$$
$$= \theta_t - \eta \underbrace{\sum_{\tau=1}^{t} \overbrace{(1-\eta)^{t-\tau}}^{\text{exponential decay}} z_\tau}_{\text{bounded variance}}$$

Consideration III: Real-World Accuracy

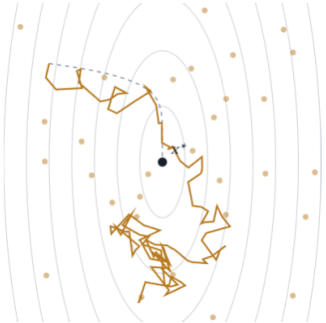
Illustrative Simulation:

DP-SGD. Step toward x*, then add noise.



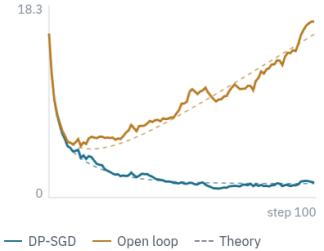
Mean squared error now 1.43 (stationary limit 1.35)

Fixed vectors plus noise. Same noise, no feedback.



Mean squared error now 16.90 (grows by 0.157 per step)

Mean squared distance to x* over 40 runs



DP-λCGD: accuracy for different λ

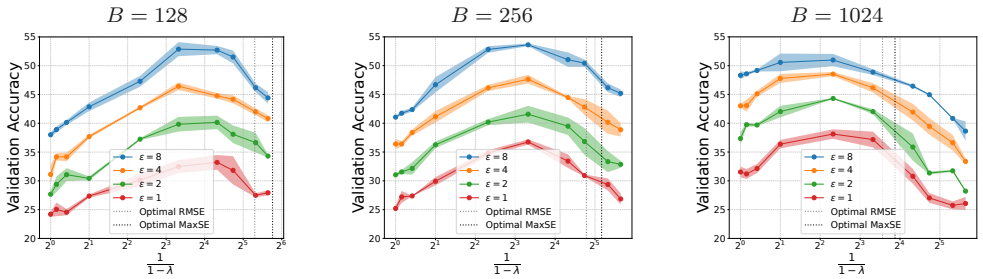
Reminder: 1-parameter family $C_\lambda = \begin{pmatrix} 1 & & & \\ \lambda & 1 & & \\ \lambda^2 & \lambda & 1 & \\ \vdots & \ddots & \ddots & \ddots \end{pmatrix}$, $C_\lambda^{-1} = \begin{pmatrix} 1 & & & \\ -\lambda & 1 & & \\ & -\lambda & 1 & \\ & & \ddots & \ddots \end{pmatrix}$.

What's the best value for λ in practice?

DP-λCGD: accuracy for different λ

Reminder: 1-parameter family $C_\lambda = \begin{pmatrix} 1 & & & \\ \lambda & 1 & & \\ \lambda^2 & \lambda & 1 & \\ \vdots & \ddots & \ddots & \ddots \end{pmatrix}$, $C_\lambda^{-1} = \begin{pmatrix} 1 & & & \\ -\lambda & 1 & & \\ & -\lambda & 1 & \\ & & \ddots & \ddots \end{pmatrix}$.

What's the best value for λ in practice?



ConvNet on CIFAR-10: validation accuracy of DP-λCGD for $k = 10$ epochs, batch size B, privacy budgets ϵ, $\delta = 10^{-5}$

Optimal λ usually smaller than RMSE suggests.

Reminder:

p-Banded Inverse Square Root (p-BISR): asymptotically rate-optimal RMSE

$$\tilde{C} := A^{1/2} \;\to\; \tilde{C}^{-1} \;\to\; \text{keep } p \text{ bands} \;\to\; \text{invert back} =: C, \qquad B := AC^{-1}$$

Banded Inverse Fractional Roots: γ-BIFR [Kalinin, Rehn, Andersson, Honkela, L. arXiv:2605.18379]

Reminder:

p-Banded Inverse Square Root (p-BISR): asymptotically rate-optimal RMSE

$$\tilde{C} := A^{1/2} \;\to\; \tilde{C}^{-1} \;\to\; \text{keep } p \text{ bands} \;\to\; \text{invert back} =: C, \qquad B := AC^{-1}$$

Replace the square root by a tunable fractional power $\gamma \in [0, 1]$:

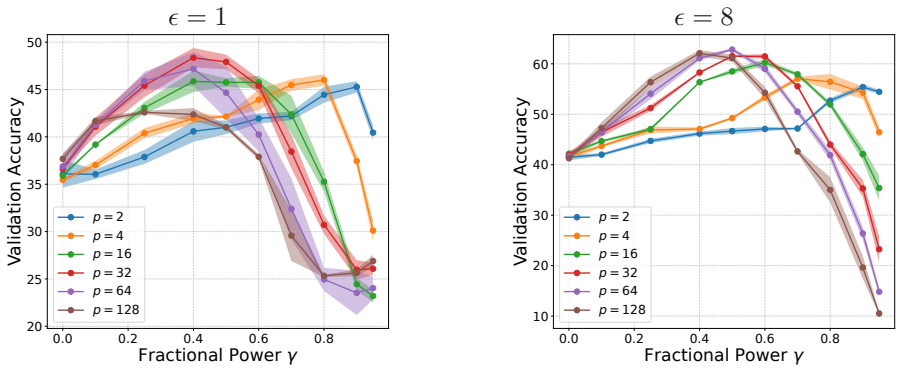
Banded Inverse Fractional Root (γ-BIFR)

$$\tilde{C} := A^{\gamma} \;\to\; \tilde{C}^{-1} \;\to\; \text{keep } p \text{ bands} \;\to\; \text{invert back} =: C, \qquad B := AC^{-1}$$

Flexible generalization of prior methods:

- recover BISR by $\gamma = \frac{1}{2}$; recover DP-λCGD for $p = 2$
- banded inverse $\to$ efficient
- allows for *amplification* in batch setting

What's the best value for γ in practice?



ConvNet on CIFAR-10, $k = 10$, validation accuracy over the fractional power γ for bandwidths p, batchsize $B = 128$.

Smaller bandwidth $p \rightarrow$ larger optimal γ.

Bonus: Beyond Toeplitz Workloads

Learning-rate-aware Factorizations [Kalinin, Andersson. FORC 2026]

Practical model training uses learning rate schedules: warm-up, cosine, exponential decay, ...

$$\theta_t = -\sum_{\tau=1}^{t} \eta_\tau \, g_\tau \quad \Rightarrow \quad A_\eta = \begin{pmatrix} \eta_1 & & & & \\ \eta_1 & \eta_2 & & & \\ \eta_1 & \eta_2 & \eta_3 & & \\ \eta_1 & \eta_2 & \eta_3 & \eta_4 & \\ \eta_1 & \eta_2 & \eta_3 & \eta_4 & \eta_5 \end{pmatrix}$$

- Toeplitz-structured factorization suggested
- theoretical guarantees for exponential schedule , $\eta_\tau \propto \bar{\eta}^{-\tau}$

Factorizations for Continual Mean Estimation [Kalinin, Najar, Roth, L. arXiv:2601.22320]

Some tasks need *averaging* rather than summing vectors: $\theta_t = \frac{1}{t}\sum_{\tau=1}^{t} g_\tau$.

$$A_{\mathbf{avg}} = \begin{pmatrix} 1 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & & \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \\ \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \end{pmatrix}$$

- (Banded) Toeplitz-structured factorization suggested
- better RMSE than e.g. square root

Summary – Matrix Factorization for Private Learning [Tutorial: Pillutla *et al.*, https://arxiv.org/abs/2506.08201]

Private training is private weighted prefix-sum estimation

- SGD iterates are (weighted) prefix sums of gradients: $\Theta = -\eta AG$
- Caveat: gradients depend on previous model parameters

Factorize the workload matrix: $A = BC$

- privatize CG with sensitivity $\|C\|_{\text{col}}$, decode with B $\Leftrightarrow$ DP-SGD with noise $C^{-1}Z$
- anti-correlated noise cancels earlier noise: RMSE $\Theta(\sqrt{n})$ $\rightarrow$ $\Theta(\log n)$

Structure makes it practical

- Toeplitz (prefix-sum, square root, etc.) $\rightarrow$ memory-efficient
- multi-epoch: banded / banded-inverse matrices
  $\rightarrow$ efficient, asymptotically rate-optimal

slides