Deep Learning with Fractional-Order Differential Privacy

Presented by: Mohammad Partohaghighi

Advisor: Dr. YangQuan Chen | Co-Advisor: Dr. Roummel Marcia

Affiliation: MESA Lab, Department of Electrical Engineering and Computer Science, University of California, Merced, USA

Outline

1. Differential Privacy and Motivation
2. Stochastic Gradient Descent (SGD) and Privacy Leakage
3. Differentially Private Stochastic Gradient Descent (DP-SGD)
4. Proposed Fractional-Order Differentially Private Stochastic Gradient Descent (FO-DP-SGD)
5. Theory and Privacy Accounting
6. Experimental Results
7. Conclusion and Future Work

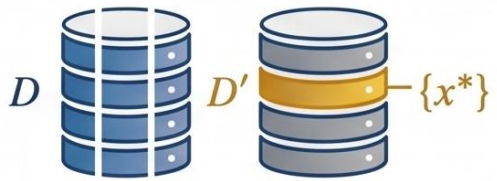
Differential Privacy: Motivation and Formal Guarantee

Threat / Motivation: If a model or mechanism depends too strongly on any single training example, it may increase the risk of privacy leakage through memorization, membership inference, or extraction attacks.

Formal Defense: Differential privacy ensures that the output distribution of a randomized mechanism does not change substantially when any single individual record is added or removed.

Adjacent Datasets ($D \sim D'$)

Datasets differ by exactly one individual record $\{x^*\}$.



DP formalizes protection against dependence on any one individual.

This formal stability is what underlies privacy protection.

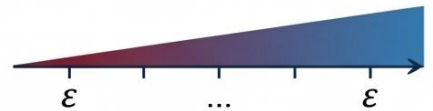
privacy budget (e-epsilon)

$$\Pr[M(D) \in S] \leq e^{\varepsilon} \cdot \Pr[M(D') \in S] + \delta$$

- $\Pr[M(D) \in S]$: output probability for D
- M: mechanism output distribution
- S: measurable event / subset
- $\Pr[M(D') \in S]$: output probability for D'
- δ: failure probability delta

Privacy budget ε

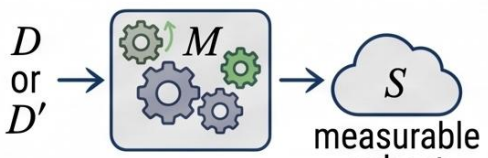
Lower ε means stronger privacy.



Lower ε (Stricter Privacy) — Higher ε (Weaker Privacy)

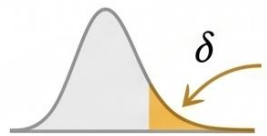
Mechanism M

Randomized mechanism M: Maps dataset to output event subset S.



Failure Probability δ

Small probability the bound is violated.



Tiny probability the bound is violated.

Interpretation: The presence or absence of one individual should not substantially affect the probability of any observable output event.

Privacy Vulnerability of Standard Stochastic Gradient Descent

Standard SGD Optimization and Parameter Updates

1. Per-Example Gradient:

$$g_t(x_i) = \nabla_\theta \ell(\theta_t; x_i)$$

Where ℓ is the loss function, θ_t are parameters, and x_i is a training example.

2. Parameter Update Rule:

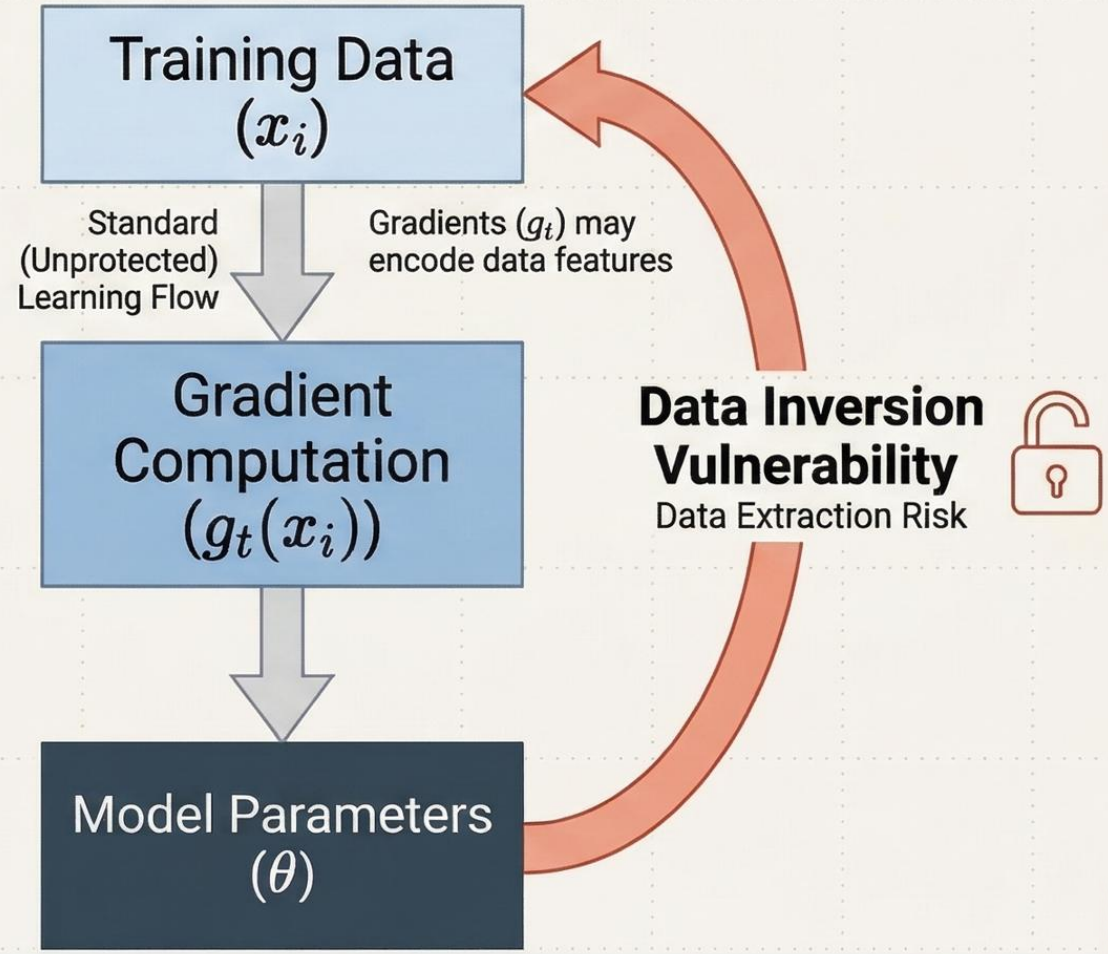
$$\theta_{t+1} = \theta_t - \eta g_t$$

Where η is the learning rate and g_t is the average gradient.

Core Insight

In standard SGD, unprotected per-example gradients can carry highly detailed **information about individual data points**. These features are propagated to the **model parameters**, creating a direct, high-fidelity signal for **data inversion and extraction** by an adversary.

Conceptual Privacy Leakage Pathway



Standard SGD relies on **direct data-dependent gradients**, exposing data features and creating a pathway for privacy leakage without explicit protection.

THE STANDARD DP-SGD PIPELINE

Deep learning with differential privacy 10446 2016
M Abadi, A Chu, I Goodfellow, HB McMahan, I Mironov, K Talwar, L Zhang
Proceedings of the 2016 ACM SIGSAC conference on computer and communications …

1. SUBSAMPLE EXAMPLES
Randomly select examples independently

Sample examples independently via

$$m_{t,i} \sim \text{Bernoulli}(q)$$

sampling rate $q = L/N$

2. COMPUTE PER-EXAMPLE GRADIENTS

For each sampled example x_i

$$g_t(x_i) = \nabla_\theta \ell(\theta_t; x_i)$$

where $i \in B_t$

CLIP GRADIENTS TO NORM C

Scale gradients to maximum norm C

$$\tilde{g}_t(x_i) = \frac{g_t(x_i)}{\max(1, \|g_t(x_i)\|_2 / C)}$$

4. AGGREGATE CLIPPED GRADIENTS

Sum the processed gradients across the batch

$$\dot{s}_t = \sum_{i \in B_t} \tilde{g}_t(x_i)$$

5. ADD GAUSSIAN NOISE

Ensure differential privacy via a Gaussian mechanism

$$\tilde{s}_t = \dot{s}_t + \mathcal{N}(0, \sigma^2 C^2 I)$$

6. MODEL UPDATE

Normalize by expected batch size L and form the private model update

$$\theta_{t+1} = \theta_t - \eta(\tilde{s}_t / L)$$

Track cumulative privacy loss across training.

Cumulative Rényi DP: (α, ε_R) at each step.
Total Privacy and conversion to (ε, δ)-DP.

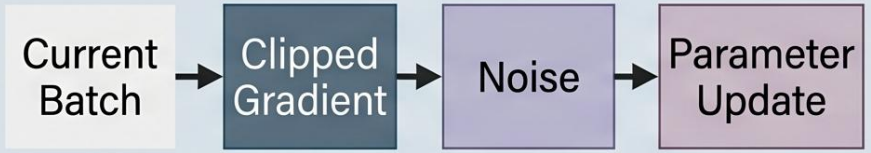
Report Final (ε, δ)-DP

KEY LIMITATIONS OF STANDARD DP-SGD

1. Ignores past updates (DP-SGD is memoryless). 2. Relies only on current-step information. 3. No use of momentum or optimization history.

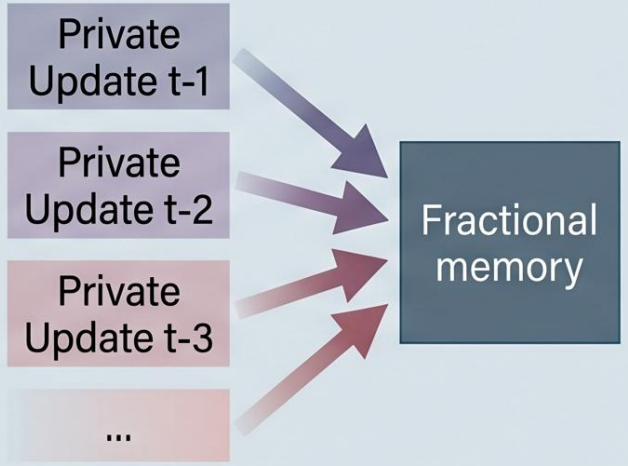
How Fractional Memory Extends Standard DP-SGD

Standard DP-SGD
Memoryless



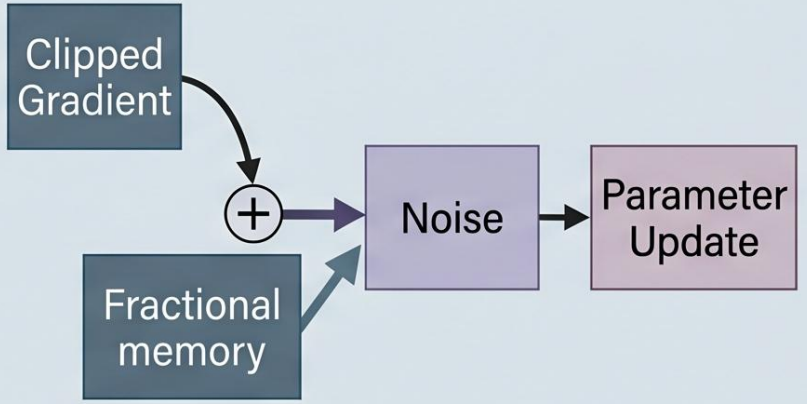
Uses only current-step information

New fractional idea



Fractional memory from past private releases

FO-DP-SGD
History-aware



Combines current information with weighted private history

FO-DP-SGD keeps the private learning pipeline, but enriches the current update with fractional memory from past private releases.

From Memoryless DP-SGD to Fractional Memory

Fractional memory preserves useful history through gradual power-law decay.

Memoryless Baseline (DP-SGD)

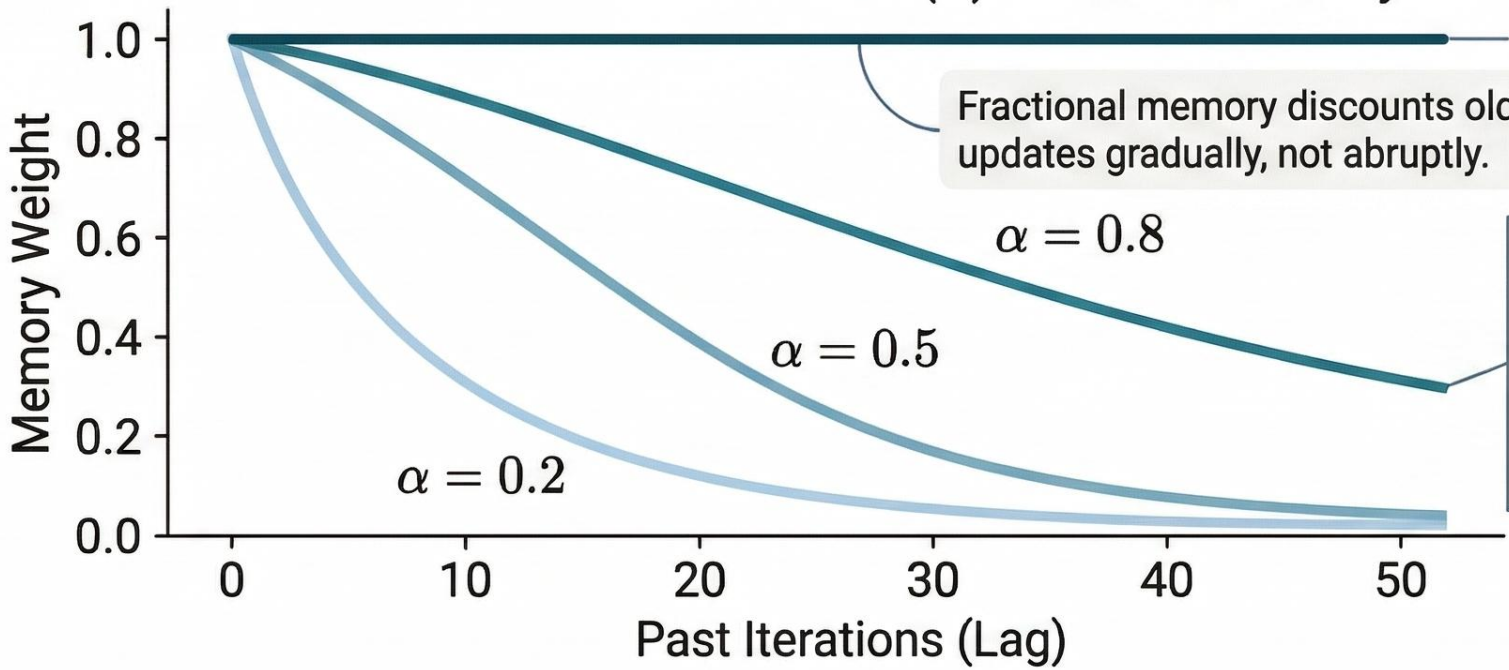
- Uses only the current noisy update
- No explicit long-range memory encoded
- Corresponds to $\alpha = 1$

Inject fractional memory while preserving the private release structure.

Optimization Benefit

- Smooths noisy descent
- Retains useful historical structure
- Stabilizes private gradient flow

Fractional Order (α) Controls Memory Retention

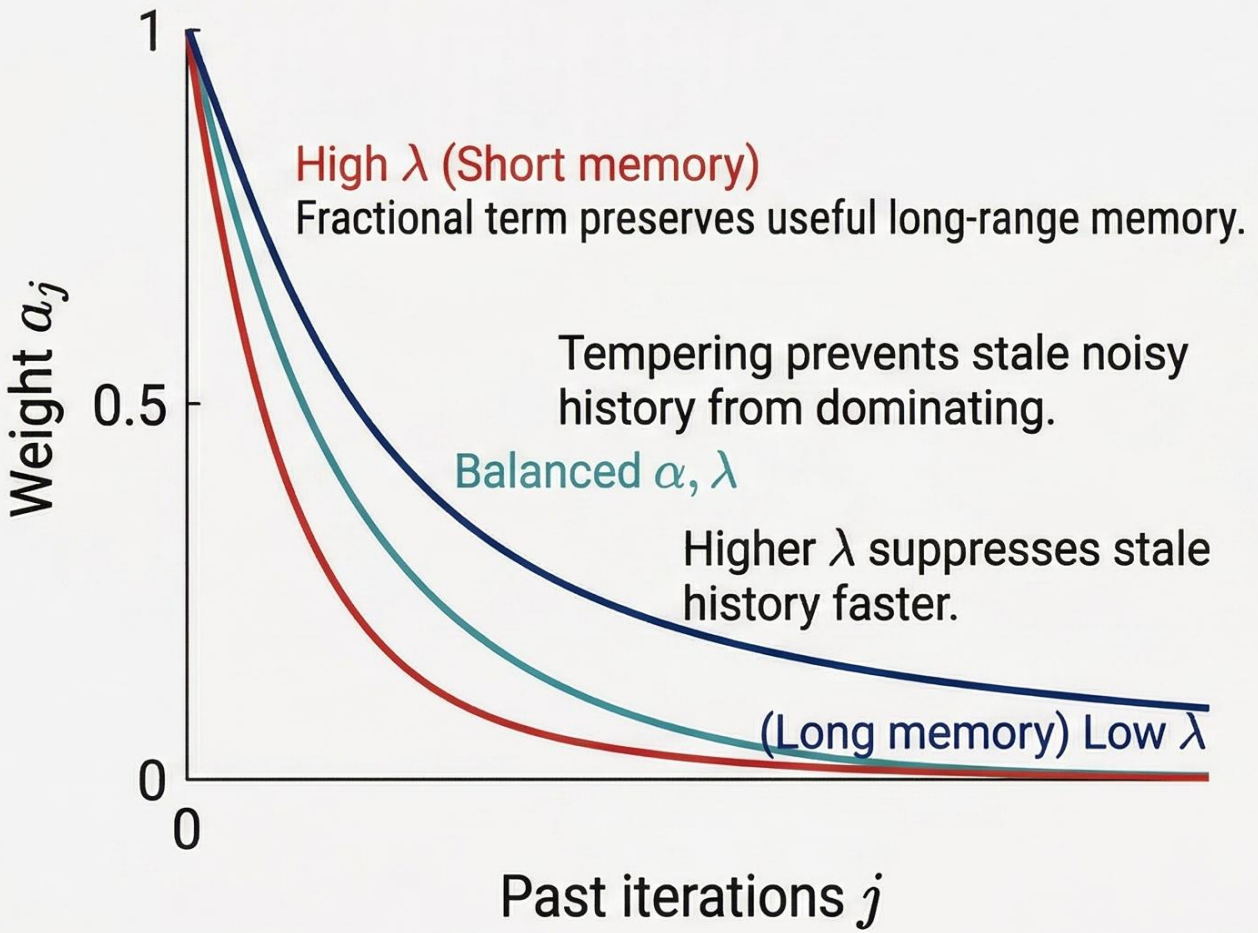


Fractional Memory Curves

Smaller α: Slower decay, longer history.
Larger α: Quicker decay, shorter history.

Tempered Fractional Memory in FO-DP-SGD

FO-DP-SGD uses **tempered fractional memory** to retain useful **history** while suppressing stale noise.



Tempered Memory Construction

1. Active Memory Horizon

$$K_t = \min\{K, t+1\}$$

2. Tempered Fractional Kernel

$$a_j(\alpha, \lambda) = (j+1)^{\alpha-1} e^{-\lambda j}$$

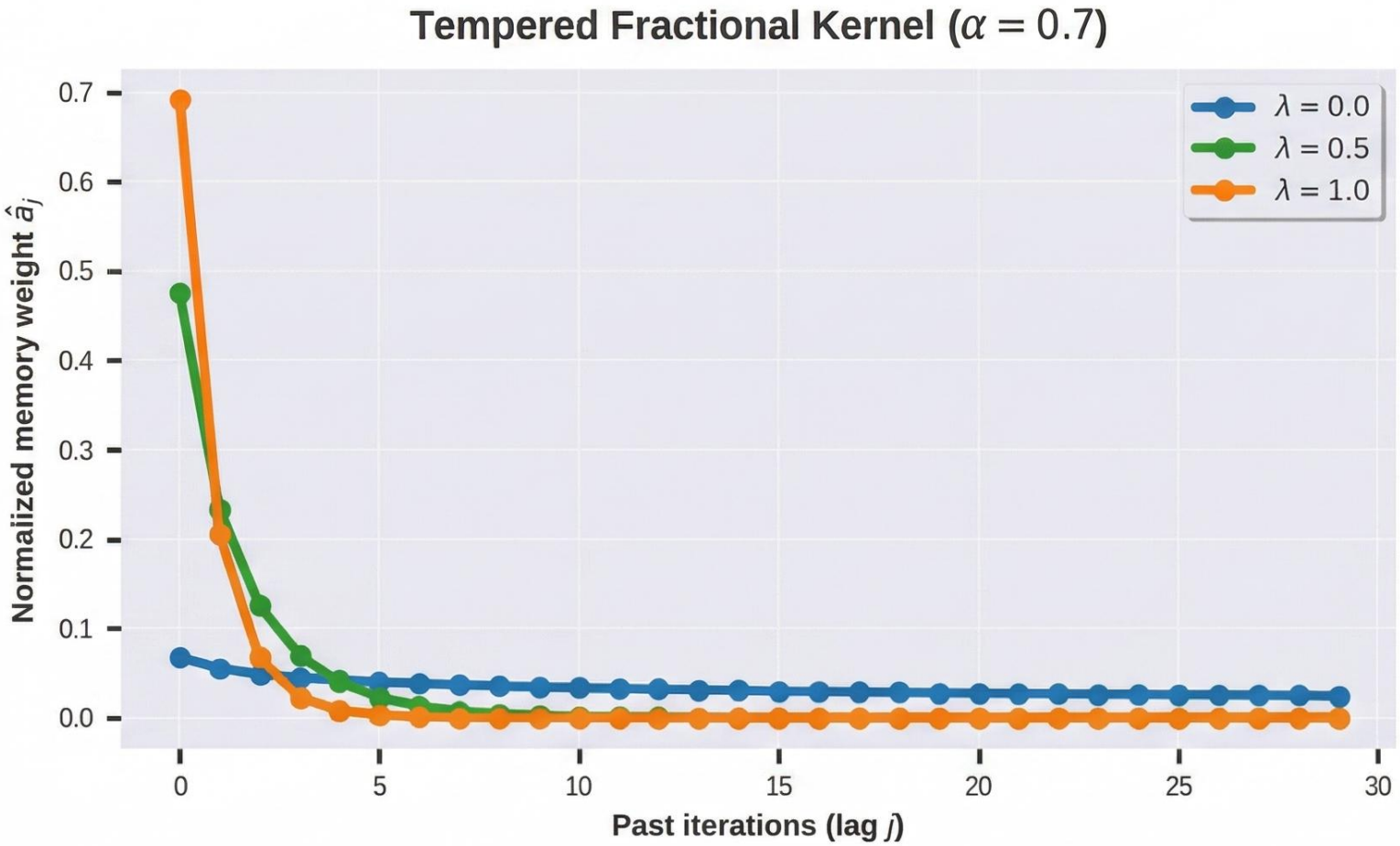
3. Normalized Memory Weights

$$\hat{a}_j(\alpha, \lambda) = \frac{a_j(\alpha, \lambda)}{\sum_{m=0}^{K_t-2} a_m(\alpha, \lambda)}$$

α memory shape | λ tempering / forgetting rate
K_t active memory horizon | $\hat{a}_j$ normalized lag contribution

IMPACT OF TEMPERING PARAMETER λ ON MEMORY WEIGHTS

Analysis of Decay Characteristics with Fixed $\alpha = 0.7$ in FO-DP-SGD



Normalized Weight Expression

$$a_j(\alpha, \lambda) = (j+1)^{\alpha-1} \exp(-\lambda j)$$

Normalized memory weight for lag j.

Decay Profile and Lambda Influence

- $\lambda = 0.0$ (Blue): Heavy-tailed power-law decay. Long-lived memory. Stale updates have non-negligible influence.
- $\lambda = 0.5$ (Green): Combined decay modes. Faster, hybrid power-law and exponential suppression of historical data.
- $\lambda = 1.0$ (Orange): Aggressive exponential suppression. Very short effective memory length. Stale updates are strongly discounted.

KEY FINDING

Increasing the tempering parameter λ accelerates memory decay, shortening effective memory and suppressing stale gradient information in FO-DP-SGD.

FO-DP-SGD: Formal Workflow Under Fixed-History Conditioning

Formal Workflow Equations (Privacy Analysis-Aware)

Key definition for privacy analysis

$$h_t = (\tilde{s}_0, \tilde{s}_1, \ldots, \tilde{s}_{t-1})$$

Condition on the realized past private releases in h_t, treated as fixed at step t.

"Fixed" = previously released privatized quantities from earlier iterations, viewed as **realized constants** during the **step-t** sensitivity argument.

Filtered/Tempered Private Memory

$$u_{t-1}(h_t) = \sum_{k=1}^{K_t-1} \hat{a}_{k-1}(\alpha, \lambda)\tilde{s}_{t-k}$$

computed from past private releases

Recursive Query (Key Analysis Step)

$$r_t(D; m_t, h_t) = \underbrace{[\beta s_t(D; m_t)]}_{\text{fresh data-dependent term}} + \underbrace{[(1-\beta)u_{t-1}(h_t)]}_{\text{history-conditioned term}}$$

At step t, the memory term is fully **determined** by the past private releases in h_t; hence only $s_t(D; m_t)$ contributes fresh data dependence

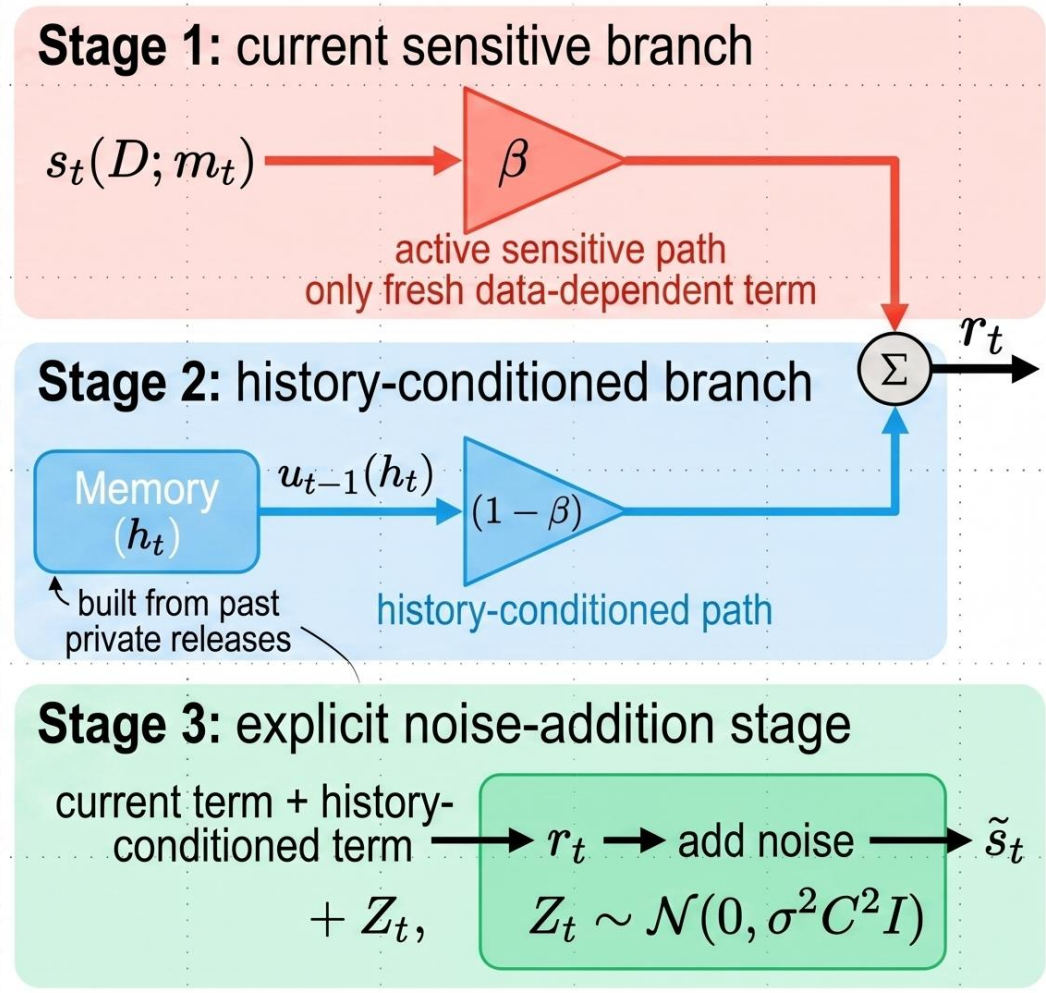
Therefore, the recursive-query sensitivity is driven only by the current term and scales through β

Noisy Release

$$\tilde{s}_t = r_t(D; m_t, h_t) + Z_t, \qquad Z_t \sim \mathcal{N}(0, \sigma^2 C^2 I)$$

$$r_t \longrightarrow + Z_t \longrightarrow \tilde{s}_t$$

System Block Diagram



Sensitivity takeaway: At step t, the past private releases in h_t are conditioned on as fixed. therefore only $s_t(D; m_t)$ introduces **fresh data** dependence, yielding $\Delta_r(m_t, h_t) \leq \beta C$.

Sensitivity Bounds: Classical DP-SGD vs. FO-DP-SGD

Analysis Condition: $D \sim D'$ under add/remove adjacency and fixed history h_t where applicable.

Classical DP-SGD: Clipped-Sum Sensitivity

Query: $$s_t(D; m_t) = \sum_{i=1}^{N} m_{t,i} \bar{g}_t(x_i)$$

$$\Delta_s(m_t) = \sup_{D \sim D'} \|s_t(D; m_t) - s_t(D'; m_t)\|_2$$

Difference under Adjacency:

$$s_t(D; m_t) - s_t(D'; m_t) = \pm \bar{g}_t(x^*)$$

Clipping Bound: $\|\bar{g}_t(x^*)\|_2 \leq C$

Conclusion:

Sensitivity Bound:
$$\Delta_s(m_t) \leq C$$

Conditioning Isolates the Fresh Term

$$C \longrightarrow \beta C$$

FO-DP-SGD: Recursive-Query Sensitivity

Recursive Query:

$$r_t(D; m_t, h_t) = \beta s_t(D; m_t) + (1-\beta) u_{t-1}(h_t)$$

Conditioning on fixed h_t.

Because h_t is fixed, $u_{t-1}(h_t)$ is constant and does not contribute fresh sensitivity.

Difference reduces to fresh term only:

$$r_t(D; m_t, h_t) - r_t(D'; m_t, h_t)$$
$$= \beta(s_t(D; m_t) - s_t(D'; m_t))$$

Only the fresh term remains.

Sensitivity Bound:
$$\Delta_r(m_t, h_t) \leq \beta C$$

Proof sketch continues on next slide.

Takeaway: Conditioning on fixed history isolates the fresh data-dependent term, reducing recursive-query sensitivity bound from C to βC.

Theorem: History-Conditioned Recursive Sensitivity

Under add/remove adjacency, for every fixed sampling mask m_t and every realizable prior transcript h_t,

$$\Delta_r(m_t, h_t) \leq \beta C$$

where $$\Delta_r(m_t, h_t) = \sup_{D \sim D'} \|r_t(D; m_t, h_t) - r_t(D'; m_t, h_t)\|_2.$$

and $\Delta_s(m_t) = \sup_{D \sim D'} \|s_t(D; m_t) - s_t(D'; m_t)\|_2 \leq C$ is the classical bound.

Definition $$\Delta_r(m_t, h_t) = \sup_{D \sim D'} \|r_t(D; m_t, h_t) - r_t(D'; m_t, h_t)\|_2$$

Substitute query $$= \sup_{D \sim D'} \|\beta s_t(D; m_t) + (1-\beta) u_{t-1}(h_t) - [\beta s_t(D'; m_t) + (1-\beta) u_{t-1}(h_t)]\|_2$$

fixed transcript term

Cancel fixed history $$= \sup_{D \sim D'} \|\beta(s_t(D; m_t) - s_t(D'; m_t))\|_2$$

Factor out β $$= \beta \sup_{D \sim D'} \|s_t(D; m_t) - s_t(D'; m_t)\|_2$$

Apply clipped-sum bound $$= \beta \cdot \sup_{D \sim D'} \underbrace{\|s_t(D; m_t) - s_t(D'; m_t)\|_2}_{\Delta_s(m_t) \leq C} \leq \beta C$$

$$\Delta_r(m_t, h_t) \leq \beta C$$

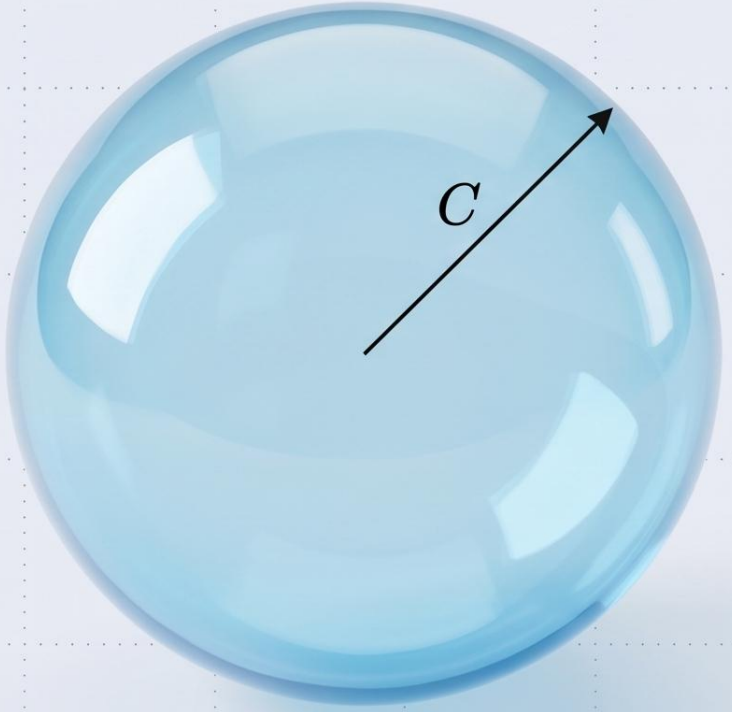
The recursive query inherits the clipped-sum sensitivity scaled by β.

From C to βC: Visualizing Sensitivity Scaling in FO-DP-SGD

With fixed private history h_t: • Only the current term remains sensitive. • The effective radius shrinks from C to βC.

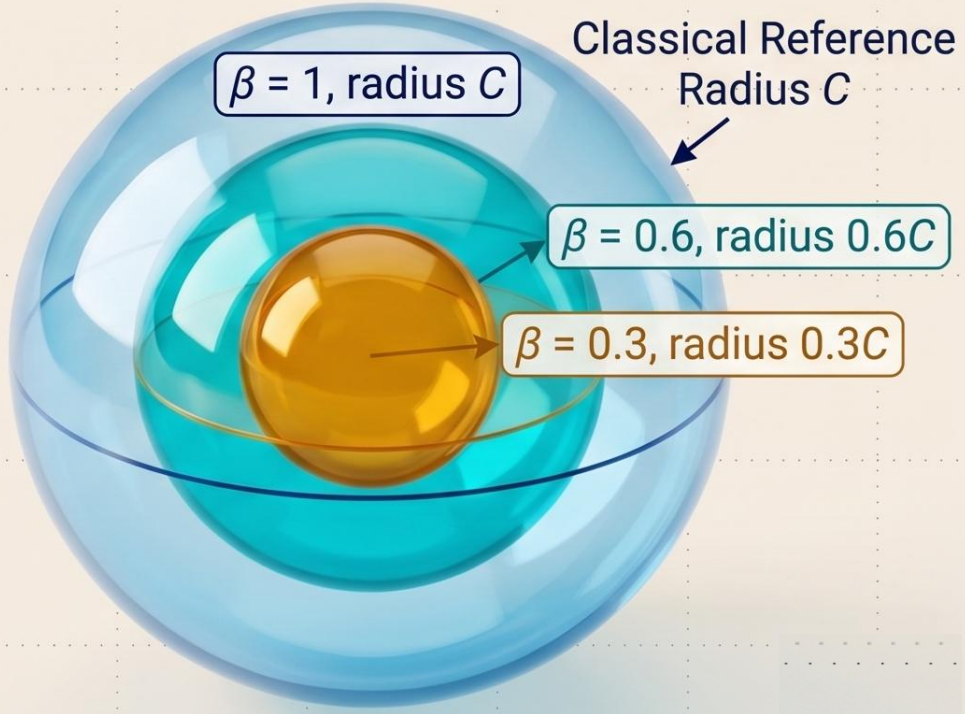
Classical DP-SGD Sensitivity

$$\Delta_s(m_t) \leq C$$



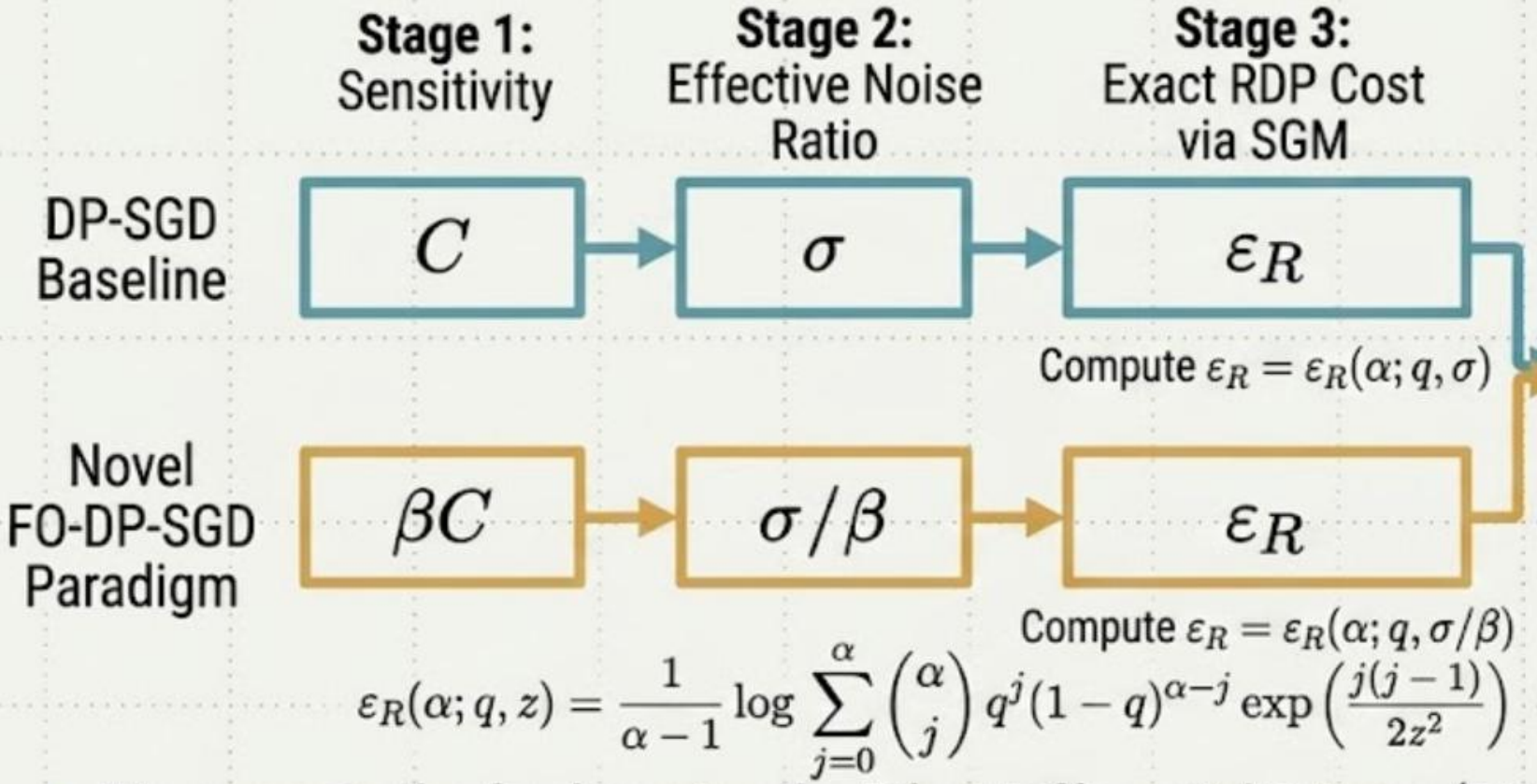
Recursive FO-DP-SGD Sensitivity

$$\Delta_r(m_t, h_t) \leq \beta C$$



Privacy Accounting: From Sensitivity to Final (ε, δ)-DP

Per-step privacy costs are tracked with RDP and then composed to obtain the final privacy budget ε.



Rényi Differential Privacy (RDP) and final accounting

Composition across all iterations

$$\varepsilon_{\text{tot}} = \sum_{t=0}^{T-1} \varepsilon_R$$

Final Conversion to (ε,δ)-DP

$$\varepsilon = \min_{\alpha>1} \left\{ \varepsilon_{\text{tot}}(\alpha) + \log \frac{1/\delta}{\alpha - 1} \right\}$$

- RDP accountant update per step → final ε after composition.

$$\varepsilon_R(\alpha; q, z) = \frac{1}{\alpha - 1} \log \sum_{j=0}^{\alpha} \binom{\alpha}{j} q^j (1-q)^{\alpha-j} \exp\left(\frac{j(j-1)}{2z^2}\right)$$

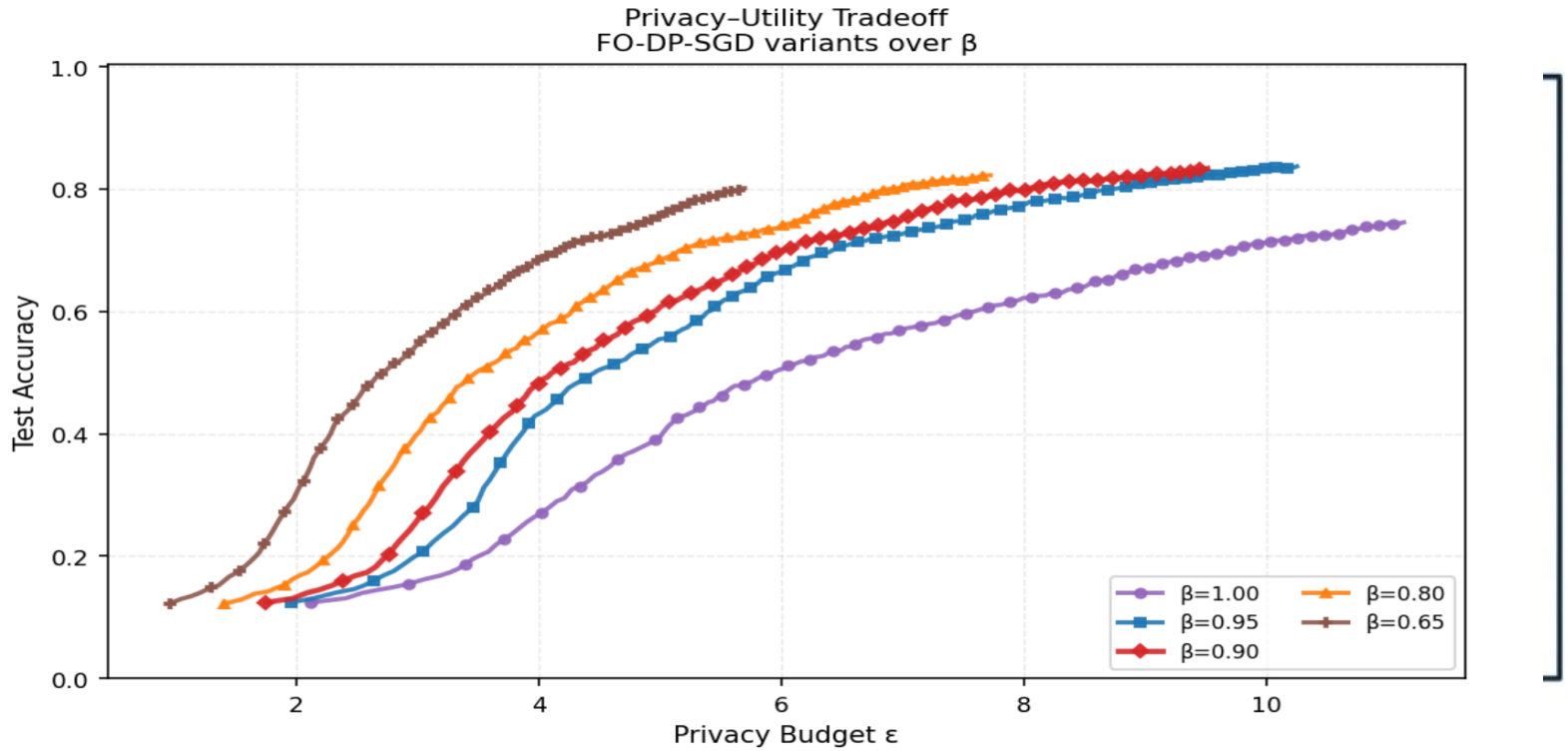
Exact SGM-based RDP formula for integer order α

- We compute the final ε using the Rényi Differential Privacy (RDP) accounting framework.
- By reducing sensitivity from C to βC, FO-DP-SGD improves the effective noise ratio while maintaining the same final privacy guarantee.

Experimental validation proves enhanced privacy-utility tradeoffs

Setup Details

- **Datasets:** MNIST, CIFAR-10
- **Architectures:** CNN, ResNet
- **Baselines:** Classical DP-SGD
- **Metrics:** Test Accuracy vs. Privacy Budget (ϵ)



- FO-DP-SGD consistently outperforms classical DP-SGD at strict privacy budgets (low ϵ).

- The fractional-order memory provides superior gradient smoothing, preventing the typical accuracy collapse seen in high-noise regimes.

- Convergence is achieved in fewer epochs due to structural momentum-like effects.

Ablation on the Fractional Order α

Ablation on the fractional order α with fixed $\beta = 0.7$, $K = 5$, $C = 2$, and $\sigma = 1.5$.

α	**CIFAR-10**	**CIFAR-100**
0.3 (stronger decay)	42.76 ± 0.39	35.55 ± 0.43
0.5 (proposed)	$\mathbf{42.89 \pm 0.38}$	$\mathbf{35.62 \pm 0.41}$
0.8	42.71 ± 0.37	35.49 ± 0.42
1.0 (uniform)	42.55 ± 0.39	35.35 ± 0.43
DP-Adam-AC (baseline)	40.35 ± 1.06	33.47 ± 1.11
DP-SGD (classical)	33.68 ± 1.17	27.95 ± 1.22

Consistent gains across datasets

Best at $\alpha = 0.5$

Outperforms strong baselines

Far above classical DP-SGD

Conclusion and Future Directions

Key contributions and next research directions

Conclusion

Method

Integrated fractional-order recursion directly into DP-SGD query.

Theory

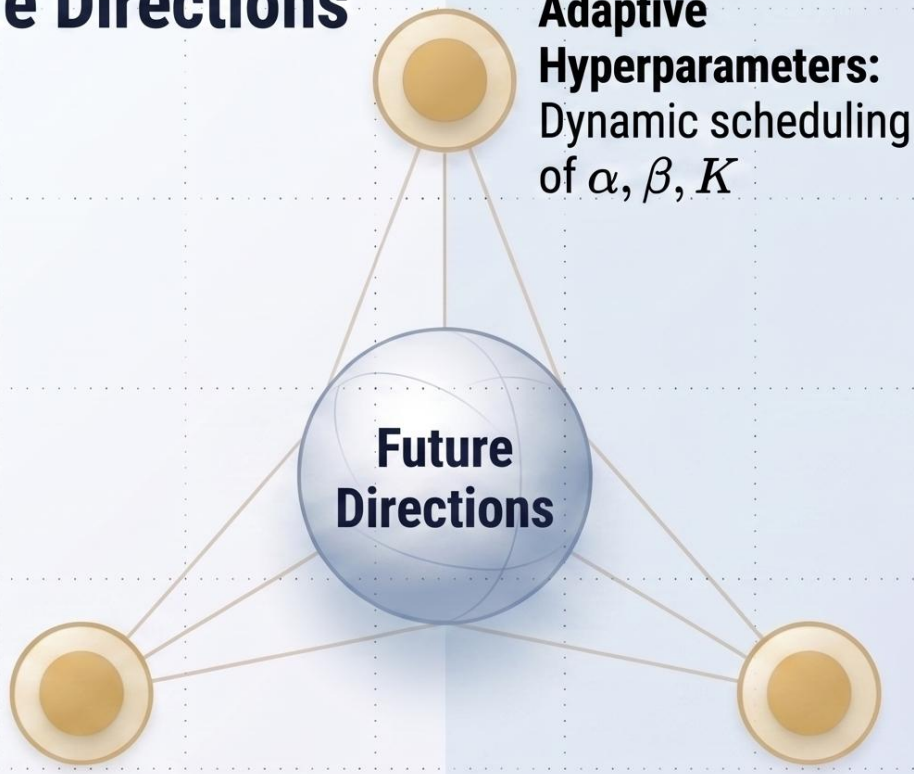
$$\Delta_r(m_t, h_t) \leq \beta C$$

Conditioned Sensitivity Bound

Impact

Reduced effective sensitivity supports improved privacy accounting and optimization utility.

Future Directions



Adaptive Hyperparameters: Dynamic scheduling of α, β, K

Federated Learning: Extending FO-DP-SGD to multi-client settings

Broader Optimization: Applications to LLMs and advanced optimizers

FO-DP-SGD shows that structured recursive memory can reduce sensitivity while retaining useful historical information.

Thank You

For Your Attention

Any questions?