SEQUENTIAL SUBSPACE NOISE INJECTION PREVENTS ACCURACY COLLAPSE IN CERTIFIED UNLEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Certified unlearning via differential privacy provides formal guarantees but remains impractical due to the severe accuracy collapse observed in current noisy fine-tuning (NFT) approaches. We propose Block-wise Noisy Fine-Tuning, which utilizes sequential subspace noise injection to partition the parameter space into orthogonal blocks and update only one block per step. This modification redistributes the noise budget over time; by restricting perturbations to lower-dimensional subspaces, we significantly reduce per-step distortion and prevent noise from overwhelming the training signal. We extend the privacy analysis to our block-wise schedule, proving that the same (ε, δ) privacy budget is retained. Empirical results on image classification benchmarks demonstrate that our approach prevents accuracy collapse, bridging the gap between rigorous guarantees and practical utility.

1 INTRODUCTION

Machine unlearning aims to remove the influence of specified training data from a trained model, motivated by legal requirements such as the GDPR (European Commission, 2016) and practical needs such as removing sensitive or malicious data (Nguyen et al., 2024). Retraining on the retained data is the gold standard but is often computationally prohibitive. Existing methods are commonly grouped into exact, certified, and empirical approaches (Guo et al., 2020; Neel et al., 2021; Fan et al., 2024; Jia et al., 2023): empirical methods can preserve utility but lack formal removal guarantees, while certified methods provide guarantees at the cost of greater conservatism.

This paper focuses on the *certified* setting, where the goal is to obtain a formal distributional guarantee rather than only strong empirical forgetting scores. Among certified approaches, differential-privacy-inspired methods are prominent (Dwork & Roth, 2014; Balle et al., 2020; Liu et al., 2023; Allouah et al., 2025). Recent noisy fine-tuning (NFT) with gradient clipping (Koloskova et al., 2025) provides an (ε, δ) certificate for arbitrary models and losses, but its Gaussian noise can severely degrade utility. On CIFAR-10 with ViT-Tiny, for example, NFT drops from 88% to below 20% test accuracy during unlearning and does not recover after subsequent fine-tuning (Figure 1).

To address this limitation, we propose *Block-wise Noisy Fine-Tuning* (Block-wise NFT), a variant of NFT that improves utility while preserving certified guarantees. We partition the parameter space into mutually orthogonal subspaces and apply the noisy updates sequentially, injecting noise only in the active subspace while freezing the rest. The scalar noise variance required by the certificate is not reduced; the gain comes from perturbing only a lower-dimensional block at each step rather than the full parameter vector. In our experiments, we use random orthogonal blocks, while other structured partitions (for example, layer-wise groupings) are possible.

We also address a second source of conservatism: worst-case initialization calibration. Standard NFT accounts for arbitrary pairs of starting models, whereas certified unlearning only requires comparison between the full-data model and its retained-data reference. We capture this relevant distance with a high-probability proximity parameter $\Delta(\rho)$ under a fixed coupling of the training runs, replacing the worst-case radius with a calibration tailored to retraining.

Together, these modifications address two complementary bottlenecks in NFT while preserving its certified-unlearning guarantees: full-space noise injection and worst-case initialization clipping. Figure 1 illustrates the combined effect of proximity calibration and block-wise scheduling relative to standard NFT. Our work makes the following contributions:

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

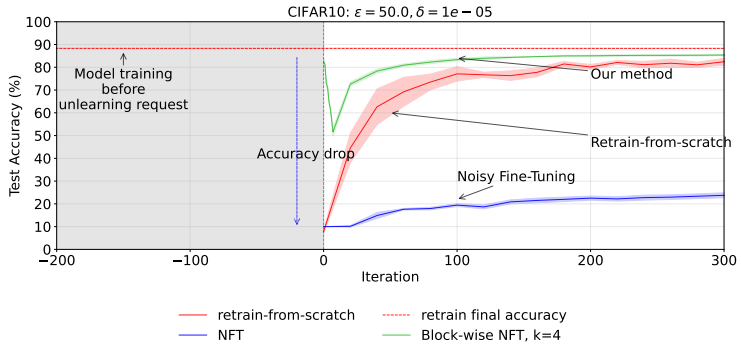


Figure 1: **Severe accuracy drop under noisy fine-tuning.** On CIFAR-10 with ViT-Tiny, standard noisy fine-tuning (NFT, Koloskova et al., 2025) test accuracy drops sharply from 88% to below 20% once the unlearning begins, and does not recover even after 300 subsequent fine-tuning steps.

- **Method.** We introduce Block-wise NFT, which applies noisy updates sequentially across orthogonal parameter blocks, reducing per-step distortion while retaining full-parameter coverage over the schedule.
- **Certification.** We extend the NFT certification analysis to the block-wise schedule and prove that the same (ε, δ) unlearning guarantee is preserved. We further show that worst-case model clipping can be replaced by the calibration parameter $\Delta(\rho)$.
- **Experiments.** On MNIST and CIFAR-10, Block-wise NFT consistently mitigates the accuracy collapse of standard NFT under the same $\Delta(\rho)$ calibration, with smaller degradation during unlearning and stronger recovery under subsequent fine-tuning.
- **Calibration bottleneck.** We identify a structural limitation of worst-case, initialization-agnostic certification. If a certified procedure achieves high utility in substantially fewer steps than retraining when started from the trained model, then a guarantee that holds uniformly over arbitrary initializations implies comparable performance when started from an unrelated initialization, effectively yielding a faster retraining procedure. This motivates calibrating certification to the actual full-data and retained-data training outputs through $\Delta(\rho)$, rather than to arbitrary parameters.

2 Preliminaries and Problem Statement

In this section, we establish notation and recall the standard definitions used throughout the paper.

Setup. A (possibly randomized) learning algorithm $\mathcal{A}$ maps a dataset $\mathcal{D}$ to model parameters $\hat{\mathbf{x}} \in \mathbb{R}^d$, i.e., $\hat{\mathbf{x}} = \mathcal{A}(\mathcal{D})$. A deletion request specifies a subset $\mathcal{D}_f \subseteq \mathcal{D}$ to be removed; the retained data are $\mathcal{D}_r := \mathcal{D} \setminus \mathcal{D}_f$. An unlearning mechanism $\mathcal{U}$ takes $(\hat{\mathbf{x}}, \mathcal{D}, \mathcal{D}_f)$ and, using randomness, outputs updated parameters $\tilde{\mathbf{x}} = \mathcal{U}(\hat{\mathbf{x}}, \mathcal{D}, \mathcal{D}_f)$.

Definition 2.1 ((ε, δ)-unlearning; following Koloskova et al. (2025))**.** Let $\varepsilon \geq 0$, $\delta \in [0, 1]$. We say that $\mathcal{U}$ is an (ε, δ)-unlearning algorithm for $\mathcal{A}$ if there exists a certifying algorithm $\bar{\mathcal{A}}$ such that, for any forget dataset $\mathcal{D}_f \subset \mathcal{D}$ and any observation $O \subset \mathbb{R}^d$,

$$\Pr[\mathcal{U}(\mathcal{A}(\mathcal{D}), \mathcal{D}, \mathcal{D}_f) \in O] \leq e^{\varepsilon} \Pr\left[\bar{\mathcal{A}}(\mathcal{D} \setminus \mathcal{D}_f) \in O\right] + \delta,$$
$$\Pr\left[\bar{\mathcal{A}}(\mathcal{D} \setminus \mathcal{D}_f) \in O\right] \leq e^{\varepsilon} \Pr[\mathcal{U}(\mathcal{A}(\mathcal{D}), \mathcal{D}, \mathcal{D}_f) \in O] + \delta.$$

Thus, throughout the paper, "certified" refers to closeness to the retained-data reference distribution, not to passing a particular empirical audit. Following Koloskova et al. (2025), we take the certifying algorithm to be $\bar{\mathcal{A}}(\mathcal{D} \setminus \mathcal{D}_f) = \mathcal{U}(\mathcal{A}(\mathcal{D}_r), \mathcal{D}_r, \varnothing)$, applying the same unlearning procedure to a retrained model. Thus, the guarantee compares the unlearned model to this retained-data reference distribution. It certifies indistinguishability, but not high utility.

We build upon the noisy fine-tuning (NFT) method of Koloskova et al. (2025), which applies clipped, Gaussian-noisy updates to the retained data D_r:

Definition 2.2 (Noisy fine-tuning (Koloskova et al., 2025))**.**

$$\mathbf{x}_0 = \Pi_{C_0}(\hat{\mathbf{x}}), \quad \mathbf{x}_{t+1} = \mathbf{x}_t - \gamma\big(\Pi_{C_1}(g_t) + \lambda \mathbf{x}_t\big) + \boldsymbol{\xi}_{t+1}. \tag{1a}$$

where $\mathbf{x}_t$ are the parameters at iteration t, g_t is the gradient at step t (computed on $\mathcal{D}_r$), $\gamma > 0$ is the learning rate, $\lambda \geq 0$ is the weight decay parameter, $\boldsymbol{\xi}_{t+1} \sim \mathcal{N}(0, \sigma^2 I_d)$ is Gaussian noise, and Π_{C_0}, Π_{C_1} are clipping operators with radii $C_0, C_1 > 0$, defined as $\Pi_C(\mathbf{v}) := \mathbf{v} \cdot \min\{\frac{C}{\|\mathbf{v}\|}, 1\}$.

NFT applies these noisy updates for T steps, with T and σ chosen to satisfy the certificate, followed by guarantee-preserving fine-tuning on D_r.

For comparison, we also introduce notation for retrained models. Let $\hat{\mathbf{x}}' = \mathcal{A}(\mathcal{D}_r)$ denote the model parameters obtained by training on the retained dataset $\mathcal{D}_r = \mathcal{D} \setminus \mathcal{D}_f$ with the same algorithm and a suitably *fixed* coupling of randomness (e.g., matched random seeds or noise schedules). Accordingly, let $\mathbf{x}'_t$ denote the iterates produced by applying the updates from Definition 2.2 to the model $\hat{\mathbf{x}}'$.

3 Algorithm Motivation

In this section, we identify two reasons why accuracy may degrade under certified unlearning and motivate two corresponding remedies. We first motivate *block-wise* noise injection to reduce the destructive effect of full-space perturbations in NFT, and then motivate a *proximity* parameter $\Delta(\rho)$ to address the broader limitation of worst-case, initialization-agnostic certification.

3.1 Necessity of block-wise noise scheduling

A central challenge in certified unlearning based on noisy fine-tuning is the severe degradation of model accuracy observed during *the unlearning phase*. Empirically, test accuracy often drops sharply once unlearning begins and cannot be fully recovered by subsequent retain fine-tuning. The reason is that the injected noise is large enough to dominate the gradient signal across all parameters.

We now formalize this limitation with a lower bound on the per-step noise level. The bound is expressed in terms of $(q, \varepsilon^{\text{rényi}})$-Rényi Differential Privacy (RDP) (Mironov, 2017), where $\varepsilon^{\text{rényi}}$ denotes the privacy loss at order $q > 1$. As standard, an RDP guarantee can be converted into an (ε, δ)-guarantee (Definition 2.1) via

$$\varepsilon = \varepsilon^{\text{rényi}} + \tfrac{\log(1/\delta)}{q-1}.$$

Theorem 3.1 (Per-step noise lower bound)**.** *Let $\gamma > 0$ be the learning rate and $\lambda \geq 0$ the weight decay parameter, with $\gamma\lambda < 1$. Consider Noisy Fine-Tuning with gradient clipping radii $C_0, C_1 > 0$ and Gaussian perturbations, certified via Rényi DP. Then, within this shifted-Rényi certification analysis, any noise scale σ that enables (ε, δ)-unlearning must satisfy*

$$\sigma^2 \begin{cases} \geq \ \gamma(2-\gamma\lambda)\frac{2q}{\varepsilon^{\text{rényi}}}\Big(2 - \frac{\lambda C_0}{C_1}\Big)C_0 C_1, & \text{if } \frac{\lambda C_0}{C_1} \in (0,1), \\ > \ \gamma(2-\gamma\lambda)\frac{2q}{\varepsilon^{\text{rényi}}}\frac{C_1^2}{\lambda}, & \text{if } \frac{\lambda C_0}{C_1} \in [1,\infty). \end{cases}$$

This inequality holds for any *number of unlearning steps T. The full dependence of the minimal step count $T(\sigma^2)$ on the noise level is derived in Appendix F.1. In particular, in the first regime $\frac{\lambda C_0}{C_1} \in (0,1)$ and for the minimal feasible noise $\sigma^2_{\min}$, the required number of steps is*

$$T(\sigma^2_{\min}) \ = \ \frac{\log\big(1 - \frac{\lambda C_0}{C_1}\big)}{\log(1-\gamma\lambda)}.$$

Interpretation. Theorem 3.1 is a *necessary condition within the proof framework of (Koloskova et al., 2025).* If σ^2 falls below the stated threshold, the divergence bound (Theorem A.9 in their work) cannot be satisfied, and the mechanism cannot be certified as (ε, δ)-unlearning by this analysis. This does not rule out that other algorithms or analyses might achieve valid guarantees with smaller noise: the result is proof-technique limited, not an information-theoretic impossibility.

This observation reveals a practical bottleneck of NFT in over-parameterized networks: within this framework, a non-negligible amount of Gaussian noise must be injected at *every* unlearning step. The key remaining choice is therefore how to distribute the noise across coordinates during optimization.

From global noise to blocks. To avoid perturbing all parameters simultaneously, we use a block-wise (subspace-wise) schedule: at each step, we inject noise and apply the clipped update only within

one orthogonal block, keeping the others fixed. Redistributing the same certified noise budget across blocks and time reduces per-step distortion and makes subsequent fine-tuning more stable.

Intuition. Even at the minimal feasible noise level $\sigma_{\min}$, every coordinate receives Gaussian noise at each step, so the noise vector typically has ℓ_2-norm about $\sigma_{\min}\sqrt{d}$. For networks with millions of parameters, this perturbation causes the additive noise to dominate the *clipped* update across many coordinates, explaining the sharp accuracy degradation observed in practice. The two regimes $\frac{\lambda C_0}{C_1} < 1$ versus $\frac{\lambda C_0}{C_1} \geq 1$ reflect whether gradient clipping or weight decay dominates the dynamics.

Taken together, the theorem highlights why NFT struggles in over-parameterized models: certification forces the injection of noise into *all* coordinates at each step. This motivates our adaptation based on *sequential subspace injection*, where the same noise budget is redistributed across orthogonal subspaces instead of being applied globally.

3.2 Retrained model localization

Independent of how noise is injected, a broader source of conservatism arises when the certification analysis holds uniformly over the input model. For NFT, the proof depends only on the worst-case separation between the two initial models and therefore applies to arbitrary initialization pairs. This is stronger than certified unlearning requires, which only compares the full-data and retained-data training outputs. The same issue is not unique to NFT: for example, output perturbation calibrates its guarantee to a worst-case bound over the model domain rather than to the specific full-data/retrained pair (Koloskova et al., 2025). We illustrate this broader limitation with an NFT thought experiment, motivating our replacement of the worst-case radius C_0 with the proximity parameter $\Delta(\rho)$.

Setup. Let T_{retrain} denote the minimal number of training steps required to retrain a model from scratch on the retain set D_r so as to achieve the same performance guarantee as that certified for NFT after T_{unl} unlearning steps. By construction, any procedure that attains this guarantee in fewer than T_{retrain} steps would constitute a faster retraining algorithm, contradicting the definition of T_{retrain}.

In the standard NFT analysis, the number of unlearning steps T_{unl} is fixed in advance as a function of the privacy budget (ε, δ), the learning rate, and the clipping parameters. Crucially, T_{unl} does *not* depend on the initialization of the model parameters. Moreover, the certified guarantee is formulated in the worst case: it requires NFT to produce indistinguishable outcomes for *any* two initializations.

Thought experiment. Suppose that NFT, when initialized at the fully trained model $\hat{\mathbf{x}}$, reaches performance at least α with probability p after T_{unl} steps, where $T_{\text{unl}} \ll T_{\text{retrain}}$. Since the guarantee is initialization-agnostic and holds for any model, we may instead initialize NFT from a randomly initialized model $\mathbf{x}_{\text{init}}$. Let $\mathcal{E}_\alpha$ denote the event that the resulting model has performance below α. Applying the (ε, δ) guarantee gives

$$\Pr\big[\mathcal{U}(\mathbf{x}_{\text{init}}) \in \mathcal{E}_\alpha\big] \leq e^{\varepsilon} \Pr\big[\mathcal{U}(\hat{\mathbf{x}}) \in \mathcal{E}_\alpha\big] + \delta \leq e^{\varepsilon}(1-p) + \delta.$$

Thus, after the same T_{unl} steps, NFT started from $\mathbf{x}_{\text{init}}$ reaches performance at least α with probability at least $1 - e^{\varepsilon}(1-p) - \delta$. Viewed as a retraining procedure, this would achieve the same performance threshold from a random initialization in only T_{unl} steps, albeit with a slightly weaker success probability. If $T_{\text{unl}} \ll T_{\text{retrain}}$, this contradicts the definition of T_{retrain} as the minimal retraining time for the corresponding performance guarantee. Hence, under this worst-case, initialization-agnostic analysis, NFT cannot simultaneously guarantee high utility and require substantially fewer steps than retraining. The intuition is illustrated in Fig. 2.

Remark 3.2. The argument assumes comparable retraining time for the slightly weakened performance guarantee; we formalize this condition and discuss its limitations in Appendix C.1.

Implication. The contradiction highlights that the model clipping-based guarantee is *too strong*: it enforces indistinguishability across all possible initializations, even completely random ones, which is not required by the definition of certified unlearning. In practice, the distance between the fully trained model $\hat{\mathbf{x}}$ and the retrained model $\hat{\mathbf{x}}'$ is much smaller than the clipping diameter $2C_0$. We replace the worst-case separation $2C_0$ induced by initial clipping with the proximity bound $\Delta(\rho)$, yielding significantly tighter and more practical guarantees in the subsequent analysis.

Definition 3.3 (High-Probability Initial Discrepancy)**.** For any failure probability $\rho \in (0, 1]$, we define the *initial discrepancy* $\Delta(\rho)$ between the fully trained model $\hat{\mathbf{x}}$ and the retrained model $\hat{\mathbf{x}}'$ as

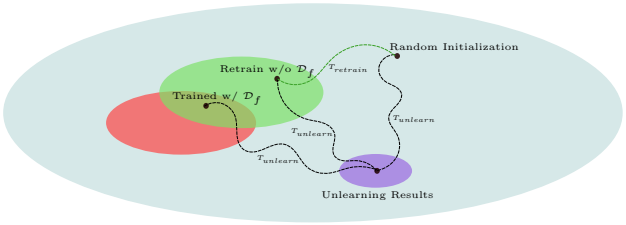


Figure 2: **Illustration of the initialization-agnostic bottleneck for Noisy Fine-Tuning.** The illustration shows fully-trained model $\mathbf{x}_0$, retrained model $\mathbf{x}_0'$ and $\mathbf{x}_{\text{init}}$ after model clipping. Unlearning trajectories require $T_{\text{unlearn}} < T_{\text{retrain}}$ steps. However, we need at least T_{retrain} steps of unlearning on the $\mathbf{x}_{\text{init}}$ to reach good quality (the green region). Therefore, results of unlearning (T_{unlearn} steps from $\mathbf{x}_{\text{init}}$) cannot obtain a good quality.

the smallest value satisfying

$$\Pr\big[\|\hat{\mathbf{x}} - \hat{\mathbf{x}}'\| \le \Delta(\rho)\big] \ge 1 - \rho. \tag{2}$$

Remark 3.4. Discrepancy $\Delta(\rho)$ for $\rho = 0$ is connected to the sensitivity (Dwork & Roth, 2014) of the model's output. However, we are only considering a particular joint distribution (X, X') while sensitivity takes supremum over all possible X and X'.

Remark 3.5. Different couplings may yield different values of $\Delta(\rho)$; we take the coupling to be part of the definition of the certifying procedure and hold it fixed throughout the analysis.

Discussion. Conceptually, this approach replaces a uniform guarantee over all initialization pairs in the C_0-ball with a guarantee for the specific coupled distribution of initializations induced by full-data training and retained-data training, which is exactly what the certified unlearning definition requires. This can improve the relative scale of the injected noise both through tighter calibration and by avoiding the initial model projection. The following corollary of Theorem 3.1 quantifies these effects when C_1/C_0 is kept fixed.

Corollary 3.6. *Suppose that C_1 is chosen so that the ratio C_1/C_0 remains fixed when $2C_0$ is replaced by $\Delta(\rho)$. For the minimal feasible noise scales in Theorem 3.1, measuring the Gaussian noise scale relative to the norm of the model before the noisy updates,*

$$\frac{\sigma_\Delta \sqrt{d}/\|\hat{\mathbf{x}}\|}{\sigma_{\text{NFT}} \sqrt{d}/\|\Pi_{C_0}(\hat{\mathbf{x}})\|} = \frac{\Delta(\rho)}{2\max\{C_0, \|\hat{\mathbf{x}}\|\}}.$$

Thus, whenever $\Delta(\rho)$ is small relative to the scale of the trained model, specifically $\Delta(\rho) < 2\max\{C_0, \|\hat{\mathbf{x}}\|\}$, proximity calibration reduces the injected noise relative to the model scale.

In this work, we treat $\Delta(\rho)$ as a *conditioning parameter* and interpret both the analysis and the experiments for a given instantiated value of $\Delta(\rho)$. The unlearning procedure remains certified under the stated proximity condition. Empirically, we instantiate $\Delta(\rho)$ using practical estimation procedures (rather than assuming an oracle bound).

4 Algorithm

In our approach, we fix the number of subspaces (blocks) $k \in \mathbb{N}$ and partition the weight space $\mathbb{R}^d$ into k mutually orthogonal components. To construct this partition, choose integers $r_1, \dots, r_k \ge 0$ with $\sum_{i=1}^k r_i = d$ and build matrices $A_i \in \mathbb{R}^{d \times r_i}$ whose columns are orthonormal. Define

$$A := [\, A_1 \ \cdots \ A_k \,] \in \mathbb{R}^{d \times d}, \qquad \text{so that } A^\top A = I_d. \tag{3}$$

Then the subspaces are $V_i := \operatorname{span}(A_i)$; they are mutually orthogonal and $\mathbb{R}^d = \bigoplus_{i=1}^k V_i$.

Note that this construction is fully general: the subspaces V_i may be chosen *arbitrarily* subject to orthogonality. Moreover, since $A = [A_1 \cdots A_k]$ is orthogonal, every $W \in \mathbb{R}^d$ decomposes uniquely as $W = \sum_{i=1}^k A_i B_i$, where $B_i = A_i^\top W$; see Appendix F. Algorithm 1 then applies noisy fine-tuning sequentially to these blocks and finally performs standard fine-tuning on the full model.

Algorithm 1 Block-wise noisy fine-tuning for unlearning

Require: model $\hat{\mathbf{x}}$, parameters $\gamma, \lambda, \Delta(\rho), C_1$, number of blocks k and privacy budget parameters (ε, δ).

1: Define projection matrices $A_1, \ldots, A_k$.
2: Decompose the model weights $\hat{\mathbf{x}}$ as $\hat{\mathbf{x}} = \sum_{i=1}^{k} A_i B_i$ (the A_i are fixed and not trained).
3: **for** $i = 1, \ldots, k$ **do**
4: Freeze all parameters except B_i.
5: Calculate the noise variance σ^2 and the number of steps T using the formula from Theorem 3.1 with C_0 replaced by $\Delta(\rho)/2$.
6: Apply noisy fine-tuning to B_i without the initial model-clipping step C_0.
7: **end for**
8: Run several standard fine-tuning steps with all model parameters unfrozen.

Utility intuition. The $\Delta(\rho)$ calibration and block-wise scheduling address complementary sources of utility degradation. The former reduces the perturbation relative to the model scale, as quantified in Corollary 3.6. At a fixed calibration, the latter localizes each noisy update to the active subspace while still covering all parameter directions over the complete schedule. Its sequential structure provides an additional optimization effect: because gradients are recomputed between blocks, later updates can partially correct perturbations introduced by earlier ones.

We make this effect explicit in a linear-regression setting in Appendix K. Starting from the retained-data optimum and using the same per-coordinate noise scale, Block-wise NFT with two blocks and one update per block has no greater expected excess loss than one full-space NFT update whenever $\gamma \leq 1/L_2$, with L_2 denoting the second-block smoothness constant; the inequality is strict when the empirical data covariance couples the blocks. Section 5 evaluates the method beyond this quadratic setting.

4.1 Theoretical guarantees

We also prove that our algorithm preserves theoretical unlearning guarantees. The same statements hold in the unconditional clipping-based formulation with radius C_0.

Theorem 4.1. *For the decomposition $W = \sum_{i=1}^{k} A_i B_i$ and noisy fine-tuning algorithm with privacy budget parameters (ε_i, δ) there is an (ε, δ) unlearning guarantee, where*

$$\varepsilon = \sum_{i=1}^{k} \varepsilon_i^{\textit{rényi}} + \frac{\log(1/\delta)}{q-1}.$$

Below we provide a brief proof sketch and clarify the role of the main technical components; full details are given in Appendix F.2. We extend the proof of Koloskova et al. (2025), which relies on a sequence of privacy amplification inequalities with shifted Rényi divergence (Balle et al., 2020) as the key tool. The block decomposition induces a *multi-dimensional* notion of shift, and our proof is organized around adapting the shifted-divergence framework to this setting.

Specifically, we generalize Wasserstein distance and shifted Rényi divergence to a vector-valued (block-wise) shift. In this setting, we adapt two key steps: the Shift Reduction Lemma (Feldman et al., 2018) (Lemma 4.5, proved in the main text) and a decomposed version of Lemma A.7 from Koloskova et al. (2025) (proved in Appendix F.2). Together, these decomposed tools allow us to proceed block by block in the Rényi divergence inequalities: each block update contributes an incremental privacy loss $\varepsilon_i^{\text{rényi}}$ that depends only on the parameters used for this block. Summing these per-block increments yields the overall $(\sum_{i=1}^{k} \varepsilon_i^{\text{rényi}}, q)$-RDP guarantee and, consequently, Theorem 4.1.

Definition 4.2 (Decomposition gap)**.** Let A_i for $i = 1, \ldots, k$ be a fixed set of matrices as defined in (3). Let

$$W = \textstyle\sum_{i=1}^{k} A_i B_i, \qquad W' = \sum_{i=1}^{k} A_i B_i'.$$

We define the *decomposition gap* between W and W' as

$$\mathcal{G}(W, W') := (z_1, \ldots, z_k), \qquad z_i := \|B_i - B_i'\|.$$

For two such vectors $z^{(1)}$ and $z^{(2)}$, we write $z^{(1)} \preceq z^{(2)}$ if the inequality holds coordinate-wise.

This leads to the definitions of the ∞-Wasserstein distance and the shifted Rényi divergence.

Definition 4.3 (Decomposed Wasserstein distance)**.** We say that $W_d(\mu, \mu') \preceq (z_1, \ldots, z_k)$ if there exists a coupling $w \in \Gamma(\mu, \mu')$ such that, almost surely for $w \sim (x, x')$,

$$\mathcal{G}(x, x') \preceq z.$$

Definition 4.4 (Decomposed shifted Rényi divergence)**.** For any $z \in \mathbb{R}_+^k$, $q \geq 1$, and two distributions μ, ν defined on $\mathbb{R}^d$, we define

$$D_q^{(z)}(\mu \,\|\, \nu) := \inf_{\mu' : W_d(\mu', \mu) \preceq z} D_q(\mu' \,\|\, \nu). \tag{4}$$

The new divergence retains many properties of the original. In particular, with zero shift, it reduces to the standard Rényi divergence. Moreover, the Shift Reduction Lemma can be adapted to bound the divergence before and after adding Gaussian noise.

Lemma 4.5 (Decomposed Shift Reduction Lemma for Gaussians)**.** *Let $q \geq 1$, $z, a \geq 0$, and X, Y be arbitrary random variables and matrix A_i as described in Definition 4.2. If $\xi, \xi' \sim \mathcal{N}(0, \sigma^2 I_{r_i})$ with $\sigma > 0$, then*

$$D_q^{(z)}(X + A_i\xi \,\|\, Y + A_i\xi') \leq D_q^{(z + ae_i)}(X \,\|\, Y) + \frac{qa^2}{2\sigma^2}. \tag{5}$$

Proof. The original proof for the unshifted case is adapted by modifying the first step of the inequality. In particular, instead of considering $(X + W, -W + \xi)$ and (Y, ξ'), we consider $(X + A_iW, -W + \xi)$ and (Y, ξ'), so as to ensure the shift $(0, \ldots, a, \ldots, 0) = ae_i$. Indeed, $X + A_i\xi$ and $Y + A_i\xi'$ can be obtained from $(X + A_iW, -W + \xi)$ and (Y, ξ') by post-processing $f(x, y) = x + A_iy$.

For the non-zero shift z, we adapt the proof by redefining W_1. Rather than the original choice $W_1 = h_z(W)$ (with $h_z(x) = x$ if $|x| \leq z$ and $h_z(x) = \frac{x}{|x|}z$ otherwise), we set

$$W_1 = \sum A_i h_{z_i}(B_i).$$

In this case, we observe that W_1 satisfies $G(W_1, 0) \preceq z$. Moreover, $G(W, W_1) \preceq ae_i$ whenever $G(W, 0) \preceq z + ae_i$. The remainder of the proof then follows directly from the original argument. □

Role of the proximity bound. The block-wise composition result is conditional on a valid bound on the initial shift required by the underlying NFT analysis, and is otherwise agnostic to how that bound is obtained. Proximity calibration and block decomposition therefore address complementary bottlenecks: the former controls how conservative the certified update must be, while the latter controls how the required perturbation is distributed across the parameter space.

4.2 Subspace design strategies

The certification in Theorem 4.1 applies to any admissible orthogonal decomposition, so the block construction can be chosen independently of the proof. In our experiments, we use a layer-wise random orthogonal decomposition: for each layer, we sample a Gaussian matrix, orthogonalize it, and split the resulting basis into k approximately equal blocks. This gives balanced random blocks without relying on a hand-designed architectural partition, but requires quadratic memory in the decomposed layer dimension. Other admissible decompositions trade off structure and cost: random permutations reduce this overhead to linear memory, while layer-wise grouping has no additional projection cost but relies more strongly on architectural structure. For nearly equal blocks, isotropic noise decomposes into independent block-wise noises with the same scalar variance. Hence assigning each block a Rényi budget $\varepsilon^{\text{renyi}}/k$ preserves the same overall (ε, δ) budget after composition. Balanced random blocks also have update norms that scale roughly as $1/\sqrt{k}$, so the clipping bounds scale approximately as $C_0/\sqrt{k}$ and $C_1/\sqrt{k}$. Thus, each noisy update perturbs only one block while keeping the per-block certification comparable to the baseline.

In the resulting protocol, we perform T noisy updates for each of the k blocks, followed by F standard full-model fine-tuning steps. The decomposition is implemented layer-wise and can be reused across deletion requests. For a layer $W_\ell \in \mathbb{R}^{a_\ell \times b_\ell}$, let $r_{\ell,i}$ be the dimension assigned to block

Table 1: Computational cost of the block constructions. Per-step overhead is additional to the ordinary model forward/backward pass.

Subspace Design	Construction cost	Initial conversion	Per-step overhead	Persistent storage
Random orthogonal	$O(\sum_\ell a_\ell^3)$	$O(\sum_\ell a_\ell^2 b_\ell)$	$O(\sum_\ell a_\ell r_{\ell,i} b_\ell + p_i)$	$O(\sum_\ell a_\ell^2)$
Random permutation	$O(\sum_\ell a_\ell)$	$O(\sum_\ell a_\ell b_\ell)^\dagger$	$O(p_i)$ + indexing	$O(\sum_\ell a_\ell)$
Cyclic grouping	$O(L)$	None	$O(p_i)$	$O(L)^\ddagger$

†Only if reordered weights are explicitly materialized; otherwise this is indexing/data movement. ‡Can be $O(1)$ when the layer assignment is represented implicitly.

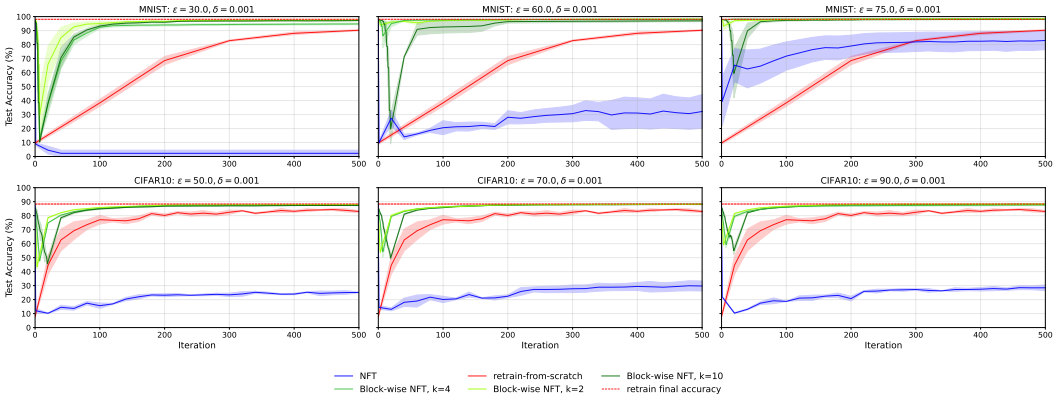


Figure 3: **Random 10% deletion on MNIST and CIFAR-10.** We compare $\Delta(\rho)$-calibrated Noisy Fine-Tuning (NFT) with Block-wise NFT ($k = 2, 4, 10$) with the final retrain accuracy shown for reference. Across privacy budgets Block-wise NFT shows smoother, more stable unlearning and better post–fine-tuning recovery.

i; then $p_i = \sum_{\ell=1}^{L} r_{\ell,i} b_\ell$ is the number of active parameters. Table 1 summarizes the one-time costs and the additional overhead per block update.

Random orthogonal blocks require dense basis operations, whereas random permutations and cyclic grouping avoid them. The per-step overheads in Table 1 exclude the ordinary forward/backward computation. Let C_{step} denote the cost of a standard full-model optimization step. For the latter two constructions, each block update remains $O(C_{\text{step}})$. Excluding the reusable setup costs,

$$C_{\text{noisy}} = O(kTC_{\text{step}}), \qquad C_{\text{post-FT}} = \Theta(FC_{\text{step}}), \qquad \frac{C_{\text{noisy}}}{C_{\text{post-FT}}} = O\left(\frac{kT}{F}\right).$$

Thus, the noisy phase is negligible when $kT \ll F$. In the ViT-Tiny benchmark reported in Appendix J.4, $kT/F = 8/1492 \approx 0.54\%$. Empirically, the permutation-based noisy phase takes 1.10s compared with 131.52s for post-unlearning fine-tuning, and the end-to-end runtime is within 1% of standard NFT for all three constructions and about $20\times$ lower than full retraining. Detailed setup, auxiliary-memory, and wall-clock measurements are provided in Appendix J.

4.3 Calibration and Re-certification

Calibrating $\Delta(\rho)$. We treat $\Delta(\rho)$ as a regime-level calibration parameter, fixed for a given architecture, training pipeline, coupling of randomness, and deletion setting. Formally, it corresponds to a high quantile of the distance between coupled full-data and retained-data training runs, and a statistically certified calibration procedure should provide a one-sided confidence guarantee for this quantile; we discuss this distinction in Appendix I. Once calibrated, $\Delta(\rho)$ instantiates the initial-shift bound in our certification analysis, with $C_0 = \Delta(\rho)/2$ in Theorem 3.1.

In our experiments, we instead use the maximum observed distance across 10 coupled runs as a practical empirical calibration. This estimate can be reused across unlearning requests within the same regime, but should be recalibrated when the architecture, training pipeline, data distribution, or deletion regime changes materially. Such setting dependence is common in practical unlearning

evaluations, where method-specific hyperparameters are typically selected for the considered deletion scenario (Ebrahimpour-Boroojeny et al., 2025). We do not claim that our empirical instantiation is optimal; alternative couplings or theoretical bounds may yield smaller values and tighter budgets. Appendix D discusses theoretical bounds based on argument stability.

Re-certification. Because $\Delta(\rho)$ is instantiated empirically, its estimate may change as more calibration evidence becomes available. Importantly, this does not necessarily require re-running unlearning: the optimization trajectory is fully determined by $(\lambda, \gamma, C_1, \sigma^2, T)$, and remains unchanged as long as these parameters are fixed. Thus, if a previously used estimate of $\Delta(\rho)$ is later deemed too optimistic, one can conservatively *recalculate* the corresponding certified *budget* for the same run without modifying it. We provide the formal statement and its proof in Appendix H.

5 Experiments

We empirically evaluate Block-wise NFT against retraining from scratch and an NFT baseline using the same $\Delta(\rho)$ calibration, thereby isolating the effect of the block-wise schedule. We evaluate random 10% deletion on MNIST (LeCun et al., 1998) with a 4.36M-parameter fully connected network and on CIFAR-10 (Krizhevsky, 2009) with ViT-Tiny (Dosovitskiy et al., 2020). Full experimental details are provided in Appendix G.

Procedure. For NFT-based methods, we fix (ε, δ) per plot with $\delta = 10^{-5}$ and vary ε across plots. For each setting, we compute the minimal feasible noise scale σ (Theorem 3.1), derive the corresponding step budget T, and run unlearning for T steps (Block-wise NFT applies this sequentially across blocks), followed by standard fine-tuning. We cap the *total* number of unlearning+fine-tuning iterations at 1000 ($\leq$1.5 epochs at batch size 64 on 90% of the data). In practice, methods typically reach their peak accuracy well before the end of fine-tuning. We use *random blocks*; other designs are deferred to the Appendix. Results are averaged over 5 runs to ensure statistical reliability.

Comparison design. Figure 1 compares the complete approach—proximity calibration together with block-wise scheduling—against original NFT. In the experiments below, both NFT and Block-wise NFT use the same $\Delta(\rho)$ calibration, isolating the additional effect of the block-wise schedule.

Figure 3 compares Block-wise NFT with $\Delta(\rho)$-calibrated NFT across privacy budgets. Across both datasets, Block-wise NFT exhibits a smaller accuracy drop during unlearning and stronger recovery under subsequent fine-tuning. For instance, at the tighter budgets on MNIST, NFT fails to recover even after fine-tuning, while Block-wise NFT retains non-trivial accuracy. Varying the number of blocks ($k \in 2, 4, 10$) yields comparable behavior, indicating that the gains are not strongly sensitive to k in this range. Additional ablations over k, block construction, runtime, and sensitivity to the instantiated $\Delta(\rho)$ are provided in Appendix G.3.

6 Conclusion

We studied utility limitations in perturbation-based certified unlearning. In particular, we showed that, for a predefined number of unlearning steps, the standard clipping-based analysis of NFT is overly conservative: it creates a tension between maintaining accuracy and improving over full retraining. This motivates our reformulation in terms of the proximity parameter $\Delta(\rho)$, which captures the practically relevant distance between the full-data model and its retained-data retraining counterpart.

We further proposed Block-wise Noisy Fine-Tuning, which applies certified noisy updates sequentially across orthogonal parameter blocks. Under our instantiated calibration of $\Delta(\rho)$, this block-wise schedule stabilizes unlearning and substantially reduces the transient accuracy drop relative to standard NFT, while remaining consistent with membership-inference auditing. Overall, our results suggest that NFT-style certified unlearning can be made substantially less destructive in the regimes we study, while retaining formal certified-unlearning guarantees.

More broadly, our findings suggest that worst-case certified-unlearning analyses can be misaligned with the practical comparison between full-data and retained-data training. Future work should develop certificates that better reflect training dynamics while retaining formal guarantees, and further study how calibration, block decompositions, and auditing protocols interact across different deletion regimes.

AI USE STATEMENT

We used generative AI tools to improve the clarity and readability of the manuscript, to assist with writing and editing parts of the experimental code, and to identify potentially relevant related work. All AI-assisted text was reviewed and edited by the authors. AI-assisted code was manually inspected and tested for correctness, and literature suggested by AI tools was verified against the original sources before being cited. We take responsibility for the final content of this work, including text, code, claims, and results produced with the aid of generative AI.

ETHICS STATEMENT

This work does not involve human subjects or newly collected personal data and uses only established public benchmark datasets. The proposed method is intended to support privacy-preserving data removal by improving the practicality of certified machine unlearning. We do not anticipate additional ethical concerns beyond those generally associated with machine learning and data privacy.

REPRODUCIBILITY STATEMENT

Algorithm 1 specifies Block-wise NFT, and the appendix provides complete proofs of the theoretical results and additional details on the proximity calibration. It also reports the model architectures, training and unlearning hyperparameters, data-processing procedures, alternative block constructions, ablations, computational complexity, and wall-clock measurements used in our experiments on MNIST and CIFAR-10. The implementation and experiment configurations are available through the anonymous repository listed in the Code Availability section.

REFERENCES

Youssef Allouah, Joshua Kazdan, Rachid Guerraoui, and Sanmi Koyejo. The utility and complexity of in- and out-of-distribution machine unlearning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=HVFMooKrHX.

Borja Balle, Gilles Barthe, Marco Gaboardi, Justin Hsu, and Tetsuya Sato. Hypothesis testing interpretations and renyi differential privacy. In Silvia Chiappa and Roberto Calandra (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 2496–2506. PMLR, 26–28 Aug 2020. URL https://proceedings.mlr.press/v108/balle20a.html.

Raef Bassily, Vitaly Feldman, Cristóbal Guzmán, and Kunal Talwar. Stability of stochastic gradient descent on nonsmooth convex losses, 2020. URL https://arxiv.org/abs/2006.06914.

Eli Chien, Haoyu Peter Wang, Ziang Chen, and Pan Li. Langevin unlearning: A new perspective of noisy gradient descent for machine unlearning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=3LKuC8rbyV.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020. URL https://arxiv.org/abs/2010.11929.

Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3–4):211–407, August 2014. ISSN 1551-305X. doi: 10.1561/0400000042. URL https://doi.org/10.1561/0400000042.

Ali Ebrahimpour-Boroojeny, Hari Sundaram, and Varun Chandrasekaran. Amun: Adversarial machine unlearning, 2025. URL https://arxiv.org/abs/2503.00917.

European Commission. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection

Regulation) (Text with EEA relevance), 2016. URL https://eur-lex.europa.eu/eli/reg/2016/679/oj.

Chongyu Fan, Jiancheng Liu, Yihua Zhang, Eric Wong, Dennis Wei, and Sijia Liu. Salun: Empowering machine unlearning via gradient-based weight saliency in both image classification and generation. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=gn0mIhQGNM.

Vitaly Feldman, Ilya Mironov, Kunal Talwar, and Abhradeep Thakurta. Privacy amplification by iteration. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 521–532. IEEE, October 2018. doi: 10.1109/focs.2018.00056. URL http://dx.doi.org/10.1109/FOCS.2018.00056.

Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. Certified data removal from machine learning models. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 3832–3842. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/guo20c.html.

Moritz Hardt, Ben Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In Maria Florina Balcan and Kilian Q. Weinberger (eds.), *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 1225–1234, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/hardt16.html.

Jiyan He, Xuechen Li, Da Yu, Huishuai Zhang, Janardhan Kulkarni, Yin Tat Lee, Arturs Backurs, Nenghai Yu, and Jiang Bian. Exploring the limits of differentially private deep learning with group-wise clipping, 2022. URL https://arxiv.org/abs/2212.01539.

Jinghan Jia, Jiancheng Liu, Parikshit Ram, Yuguang Yao, Gaowen Liu, Yang Liu, Pranay Sharma, and Sijia Liu. Model sparsity can simplify machine unlearning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=0jZH883i34.

Anastasia Koloskova, Youssef Allouah, Animesh Jha, Rachid Guerraoui, and Sanmi Koyejo. Certified unlearning for neural networks. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=3rWQlV3s1I.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Yann LeCun, Corinna Cortes, and Christopher JC Burges. The mnist database of handwritten digits. *http://yann.lecun.com/exdb/mnist*, 1998.

Yunwen Lei, Rong Jin, and Yiming Ying. Stability and generalization analysis of gradient methods for shallow neural networks. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 38557–38570. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/fb8fe6b79288f3d83696a5d276f4fc9d-Paper-Conference.pdf.

Jiaqi Liu, Jian Lou, Zhan Qin, and Kui Ren. Certified minimax unlearning with generalization rates and deletion capacity. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=6H8Md75kAw.

Tongliang Liu, Gábor Lugosi, Gergely Neu, and Dacheng Tao. Algorithmic stability and hypothesis complexity. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2159–2167. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/liu17c.html.

Ilya Mironov. Rényi differential privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pp. 263–275. IEEE, August 2017. doi: 10.1109/csf.2017.11. URL http://dx.doi.org/10.1109/CSF.2017.11.

Seth Neel, Aaron Roth, and Saeed Sharifi-Malvajerdi. Descent-to-delete: Gradient-based methods for machine unlearning. In Vitaly Feldman, Katrina Ligett, and Sivan Sabato (eds.), *Proceedings of the 32nd International Conference on Algorithmic Learning Theory*, volume 132 of *Proceedings of Machine Learning Research*, pp. 931–962. PMLR, 16–19 Mar 2021. URL https://proceedings.mlr.press/v132/neel21a.html.

Thanh Tam Nguyen, Thanh Trung Huynh, Zhao Ren, Phi Le Nguyen, Alan Wee-Chung Liew, Hongzhi Yin, and Quoc Viet Hung Nguyen. A survey of machine unlearning, 2024. URL https://arxiv.org/abs/2209.02299.

Toan N. Nguyen, Phuong Ha Nguyen, Lam M. Nguyen, and Marten Van Dijk. Batch clipping and adaptive layerwise clipping for differential private stochastic gradient descent, 2023. URL https://arxiv.org/abs/2307.11939.

Dominic Richards and Ilja Kuzborskij. Stability & generalisation of gradient descent for shallow neural networks without the neural tangent kernel. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 8609–8621. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/483101a6bc4e6c46a86222eb65fbcb6a-Paper.pdf.

Ayush Sekhari, Jayadev Acharya, Gautam Kamath, and Ananda Theertha Suresh. Remember what you want to forget: Algorithms for machine unlearning. In *Advances in Neural Information Processing Systems 34 (NeurIPS 2021)*, NeurIPS '21, pp. 18075–18086. Curran Associates, Inc., December 2021. URL https://arxiv.org/abs/2103.03279.

Ross Wightman. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019.

Binchi Zhang, Yushun Dong, Tianhao Wang, and Jundong Li. Towards certified unlearning for deep neural networks. In *Forty-first International Conference on Machine Learning*, 2024.

A Related work

Certified Machine Unlearning. Certified machine unlearning has emerged as a practical alternative to full retraining, offering rigorous data-removal guarantees at much lower cost. Classical approaches estimate the retrained solution via single–step Newton and influence surrogates (Guo et al. (2020); Sekhari et al. (2021); Zhang et al. (2024)), or use projected/perturbed gradient methods (Neel et al. (2021); Chien et al. (2024)), coupled with randomized mechanisms to ensure statistical indistinguishability. Many of these certificates are explicitly *DP-inspired*, applying noisy updates and privacy amplification ideas from differential privacy Dwork & Roth (2014); Balle et al. (2020). However, recent methods typically rely on strong assumptions (e.g., smoothness/strong convexity, knowledge of Hessian eigenvalues, or even a unique minimizer), which limit applicability to modern deep networks Zhang et al. (2024); Liu et al. (2023); Allouah et al. (2025). In this work we remain within the certified DP line, but we specifically adapt the noisy fine-tuning framework of Koloskova et al. (2025) to preserve certificates *without* convexity or uniqueness assumptions while mitigating the accuracy loss usually observed in DP-based unlearning.

Group-wise and layer-wise DP optimization. Our construction is related to group-wise and layer-wise DP-SGD methods (He et al., 2022; Nguyen et al., 2023), which split parameters or gradients into groups to reduce clipping bias or exploit layer heterogeneity in private training. The setting is different: these methods train privately from a clean initialization, whereas certified unlearning starts from a model already trained on the forget set and must certify closeness to retained-data retraining. Thus, we use block structure not only as an optimization device, but as a sequential certified-unlearning schedule whose privacy analysis composes across blocks while covering the full parameter space.

Fixed-subspace PEFT. Unlike fixed-subspace PEFT, which restricts optimization to a persistent parameter subspace, Block-wise NFT updates all parameter directions over the unlearning trajectory. This distinction is important for unlearning, since permanently freezing the complementary subspace would require an additional argument that it contains no forget-set information.

B CODE AVAILABILITY

We release our implementation at `https://anonymous.4open.science/r/block-nft-DFBE`. The repository includes the code needed to reproduce our experiments.

C THOUGHT EXPERIMENT FORMALIZATION

C.1 LIMITATIONS AND EXTENSIONS

Thought experiment limitations. The argumentation of our thought experiment implicitly relies on the assumption that the number of steps required to retrain from scratch to the original performance guarantee (α, p) is comparable to the number of steps required to reach the slightly weaker guarantee $(\alpha,\, 1 - e^{\varepsilon}(1-p) - \delta)$. While this is a natural assumption when (ε, δ) are small and the certified guarantee is tight, it need not hold universally. In particular, for some models or training regimes, small degradations in success probability may correspond to substantially different retraining times. Our thought experiment should therefore be interpreted as highlighting a structural tension induced by worst-case, initialization-agnostic guarantees, rather than as a strict impossibility result under all settings.

Scope beyond NFT. The argument is not specific to the update rule of NFT. It uses only that the certified procedure provides a uniform indistinguishability guarantee over arbitrary starting models and that its computational budget is fixed independently of the initialization. Under these conditions, transferring good performance from a trained initialization to an unrelated random initialization can imply a procedure for reaching comparable retained-data performance within the same computational budget.

We stated the argument in terms of step counts T_{unl} and T_{retrain}, although the per-step computational cost may differ across procedures. The same reasoning can be formulated in terms of wall-clock time or total compute, provided the algorithm's computational cost is independent of the model weights.

More generally, this observation suggests that certified unlearning need not protect uniformly over unrelated parameter pairs when the relevant comparison is between the actual outputs of full-data and retained-data training. The proximity parameter $\Delta(\rho)$ provides one concrete formulation of this principle. The block-wise construction, in contrast, is deliberately developed within the NFT framework.

D THEORETICAL BOUNDS ON $\Delta(\rho)$ VIA ARGUMENT STABILITY

The key quantity in our certificates is the discrepancy $\Delta(\rho)$ between the full retrain and the retraining-from-scratch. While obtaining a tight bound on $\Delta(\rho)$ for modern deep models in full generality is challenging, this question is well-studied in the literature on *argument stability* (also known as parameter stability). Below we summarize regimes where such bounds are known or can be derived under standard assumptions.

Algorithmic stability has long been used to analyze how sensitive learning algorithms are to changes in the training data. Liu et al. (2017) formalized *argument stability*, which upper-bounds the parameter deviation between hypotheses trained on neighboring datasets. Several subsequent works derive explicit bounds on this deviation under standard assumptions.

Smooth and Lipschitz losses. Hardt et al. (2016) provide a trajectory-level recursion for SGD under β-smooth and L-Lipschitz losses in three regimes: (i) non-convex, (ii) convex, (iii) γ-strongly convex. When the replaced index is known—as in our full-vs-retained setting—their recursion yields a closed-form upper bound on

$$\|\theta^{(t)} - \theta'^{(t)}\|,$$

and hence on our proximity $\Delta(\rho)$.

Example (strongly convex case). For completeness, we briefly illustrate how a bound on $\Delta(\rho)$ follows from the recursion of Hardt et al. (2016). Suppose the loss is γ-strongly convex and β-smooth, and SGD uses a constant stepsize $\alpha \leq 1/\beta$. Let $\theta^{(t)}$ and $\theta'^{(t)}$ denote the SGD iterates obtained from

the full and retained datasets, respectively, and let $\delta_t = \|\theta^{(t)} - \theta'^{(t)}\|$. Since both runs start from the same initialization, we have $\delta_0 = 0$.

Hardt et al. (2016) give the following one-step inequalities:

1. If the minibatches coincide:
$$\delta_{t+1} \;\le\; (1 - \alpha\gamma)\,\delta_t.$$
2. If the minibatches differ:
$$\delta_{t+1} \;\le\; (1 - \alpha\gamma)\,\delta_t + 2\alpha L.$$

Unrolling this recursion yields the explicit bound

$$\delta_T \;\le\; 2\alpha L \sum_{k \in B} (1 - \alpha\gamma)^k,$$

where B is the set of iteration indices for which the minibatches differ. This provides a closed-form upper bound on δ_T, and therefore on $\Delta(\rho)$ in this regime.

Nonsmooth convex losses. Bassily et al. (2020) prove argument stability for SGD without requiring smoothness. Using the monotonicity of subgradients together with L-Lipschitzness, they bound the deviation $\|\theta^{(t)} - \theta'^{(t)}\|$ at every iteration, directly giving an upper bound on $\Delta(\rho)$.

Neural networks. For certain neural architectures, stability of gradient-based methods has also been established. The work of Lei et al. (2022) proves stability-based generalization bounds for shallow ReLU networks. Complementarily, Richards & Kuzborskij (2021) analyze the dynamics of gradient descent and obtain stability and generalization guarantees for shallow networks beyond the NTK regime. In both cases, the analysis controls how the learned parameters change under small perturbations of the training data, providing architecture-specific upper bounds on the parameter deviation and hence on $\Delta(\rho)$ for these models.

Together, these results show that $\Delta(\rho)$ can be bounded in several well-understood regimes—including smooth non-convex objectives, nonsmooth convex objectives, and specific neural-network architectures for which parameter-stability analyses exist. A general closed-form bound is not available for all training pipelines and architectures, but many practical settings already fall into one of the regimes above.

E NOTATIONS

We summarize the main notation used throughout the paper in Table 2.

Table 2: Notation used throughout the paper

Symbol	Description
$\mathcal{A}$	Learning algorithm mapping a dataset to model parameters
$\mathcal{D}$	Training dataset
d	Dimension of the parameter (weight) space $\mathbb{R}^d$
$\hat{\mathbf{x}}$	Model parameters obtained from $\mathcal{A}(\mathcal{D})$
$\mathcal{D}_f$	Subset of $\mathcal{D}$ to be forgotten
$\mathcal{D}_r$	Retained dataset, $\mathcal{D} \setminus \mathcal{D}_f$
$\mathcal{U}$	Unlearning mechanism updating parameters after deletion request
$\tilde{\mathbf{x}}$	Model parameters output by $\mathcal{U}$
$\bar{\mathcal{A}}$	Certifying algorithm in the definition of (ε, δ)-unlearning
ε	Privacy/unlearning parameter controlling multiplicative slack
δ	Privacy/unlearning parameter controlling additive slack
$\mathbf{x}_0$	Model parameters after initial clipping of $\hat{\mathbf{x}}$ with radius C_0
$\mathbf{x}_t$	Model parameters at iteration t of noisy fine-tuning
g_t	Gradient at step t computed on the retained data $\mathcal{D}_r$
$\boldsymbol{\xi}_t$	Gaussian noise $\sim \mathcal{N}(0, \sigma^2 I_d)$ added at step t
Π_C	Clipping operator with radius C, $\Pi_C(\mathbf{v}) = \mathbf{v} \cdot \min\{\frac{C}{\|\mathbf{v}\|}, 1\}$

$\hat{\mathbf{x}}'$	Model parameters obtained by training on the retained dataset $\mathcal{D}_r$
$\mathbf{x}'_t$	Iterates produced by unlearning updates when initialized at $\hat{\mathbf{x}}'$
σ^2	Variance of the Gaussian noise in noisy fine-tuning
T_{retrain}	Minimal number of retraining steps required to reach good accuracy on $\mathcal{D}_r$
T	Number of unlearning steps in NFT
α	Target accuracy level in the thought experiment
p	Probability of achieving accuracy at least α after T steps
$\mathbf{x}_{\text{init}}$	Randomly initialized model parameters
$\Delta(\rho)$	High-probability bound on the distance between $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}'$
ρ	Failure probability in the definition of $\Delta(\rho)$
q	Order of Rényi Differential Privacy
$\varepsilon^{\text{rényi}}$	Privacy loss at order q in Rényi DP
σ^2	Variance of Gaussian perturbations in noisy fine-tuning
$T(\sigma^2)$	Minimal number of unlearning steps required for a given noise variance σ^2
$\sigma^2_{\min}$	Minimal feasible noise variance ensuring (ε, δ)-unlearning
k	Number of subspaces (blocks) in the partition of $\mathbb{R}^d$
r_i	Dimension of the i-th subspace, with $\sum_{i=1}^k r_i = d$
$A_i \in \mathbb{R}^{d\times r_i}$	Matrix with orthonormal columns spanning subspace V_i
A	Concatenation $[A_1 \cdots A_k] \in \mathbb{R}^{d\times d}$ with $A^\top A = I_d$
B_i	Coordinate vector in $\mathbb{R}^{r_i}$ corresponding to V_i in the decomposition of W
$\mathcal{G}(W, W')$	Decomposition gap between two weight vectors
$z^{(1)} \preceq z^{(2)}$	Coordinate-wise inequality between two vectors in $\mathbb{R}^k$
$W_d(\mu, \mu')$	Decomposed Wasserstein distance between distributions μ, μ'
$D_q^{(z)}(\mu \,\|\, \nu)$	Decomposed shifted Rényi divergence with shift vector z and order q
e_i	i-th standard basis vector in $\mathbb{R}^k$

F PROOFS

Proof of uniqueness of the decomposition. Since $A = [A_1 \ \cdots \ A_k]$ is orthogonal, we have $A^\top A = I_d$. For any $W \in \mathbb{R}^d$ set $B = A^\top W$ and partition it into blocks $B^\top = [B_1^\top, \ldots, B_k^\top]$. Then

$$AB = AA^\top W = W,$$

which expands to $W = \sum_{i=1}^k A_i B_i$. Uniqueness follows from orthogonality: if $\sum_i A_i B_i = \sum_i A_i B'_i$, then $A^\top(\sum_i A_i(B_i - B'_i)) = \sum_i B_i - B'_i = 0$, hence $B_i = B'_i$. □

F.1 PROOF OF THEOREM 3.1

We first recall the shifted Rényi divergence (Feldman et al., 2018):

Definition F.1 (Rényi divergence)**.** Let $q > 0$, $q \neq 1$. The q-Rényi divergence between two probability distributions μ and ν is defined as

$$D_q(\mu \,\|\, \nu) := \frac{1}{q-1} \log \mathbb{E}_{X\sim\nu} \left(\frac{\mu(X)}{\nu(X)} \right)^q .$$

Let us consider the reasoning presented in (Koloskova et al., 2025). Their proof contains a minor indexing mismatch in the recursion expansion, which did not affect the final result for their asymptotics, but it will be important for us. We corrected the indices in the product p_t accordingly.

Let us revisit the reasoning in (Koloskova et al., 2025). In their proof, the transition from equation (23) to equation (24) involves a minor computational error in solving the recursion. While this did not affect their asymptotic conclusions, it becomes relevant for our analysis. We correct this step by adjusting the indices in the product p_t accordingly. The corrected parts are highlighted in green.

Theorem F.2 (Koloskova A.9, fixed indices in recursion)**.** *Let $T, q \geq 1$, $\gamma_0, \ldots, \gamma_{T-1} \geq 0$, $\sigma_0, \ldots, \sigma_{T-1} > 0$, $\lambda \geq 0$ and consider the two sequences $\{x_t\}_{0\leq t\leq T}, \{x'_t\}_{0\leq t\leq T}$ as defined above. Denote by D_q the Rényi divergence of order q. Assume that for every $t \in \{0, \ldots, T-1\}$, $\gamma_t\lambda < 1$. Denote for every $t \in \{0, \ldots, T-1\}$,*

$$s_t := 2\gamma_t C_1, \qquad \rho_t := 1 - \gamma_t\lambda. \tag{6}$$

If there exist $a_0, \ldots, a_{T-1} \in \mathbb{R}_{\geq 0}$ *such that*

$$\sum_{t=0}^{T-1} \left(\prod_{k=t+1}^{T-1} \rho_k \right) a_t = \left(\prod_{t=0}^{T-1} \rho_t \right) 2C_0 + \sum_{t=0}^{T-1} \left(\prod_{k=t+1}^{T-1} \rho_k \right) s_t, \tag{7}$$

then

$$D_q(x_T \,\|\, x'_T) \;\leq\; \sum_{t=0}^{T-1} \frac{q a_t^2}{2\sigma_t^2}. \tag{8}$$

Let us attempt to find the optimal parameters. We will search for a solution under the following constraints:

- Suppose there exists a maximum allowable noise level at any given step, $\sigma \geq \sigma_i$.
- Suppose the weight decay λ and learning rate γ are the same for all steps.
- We will also seek a solution for fixed C_0, C_1, and RDP-budget $(\varepsilon^{\text{rényi}}, q)$.

For simplicity, let us denote $\alpha_t = \prod_{k=t+1}^{T-1} \rho_k$, then the condition on a can be rewritten as

$$\sum_{t=0}^{T-1} \alpha_t a_t = \rho_0 \alpha_0 2C_0 + \sum_{t=0}^{T-1} \alpha_t s_t, \tag{9}$$

In fact, determining T is equivalent to finding the first moment when the left-hand side exceeds the right-hand side:

$$\sum_{t=0}^{T-1} \alpha_t a_t \;\geq\; \rho_0 \alpha_0 2C_0 + \sum_{t=0}^{T-1} \alpha_t s_t. \tag{10}$$

Moreover, the unlearning budget ε is computed directly from the constraint on the Rényi divergence:

$$D_q(x_T \,\|\, x'_T) \;\leq\; \sum_{t=0}^{T-1} \frac{q a_t^2}{2\sigma_t^2} \;\leq\; \varepsilon^{\text{rényi}}. \tag{11}$$

Let y be the vector with coordinates $y_i = \frac{a_i}{\sigma_i}$. By the constraint, we are searching for a solution such that

$$\|y\|_2^2 \;\leq\; \tfrac{2\varepsilon^{\text{rényi}}}{q}. \tag{12}$$

Observe that by proportionally increasing a_i until the inequality becomes an equality, we do not increase the minimal number of epochs T required for condition equation 10. Hence, at the optimum we may assume

$$\|y\|_2 = \sqrt{\tfrac{2\varepsilon^{\text{rényi}}}{q}}. \tag{13}$$

Next, consider the optimal choice of a_i for our condition. By the Cauchy–Schwarz inequality,

$$\sum_{t=0}^{T-1} \alpha_t a_t \;\leq\; \sum_{t=0}^{T-1} (\alpha_t \sigma_t) \left(\tfrac{a_t}{\sigma_t} \right) \;\leq\; \|\alpha\sigma\| \cdot \|y\| \;\leq\; \sqrt{\tfrac{2\varepsilon^{\text{rényi}}}{q}} \sqrt{\sum_{i=0}^{T-1} \alpha_i^2 \sigma_i^2}. \tag{14}$$

Equality holds when the two vectors are proportional, which determines the optimal values of a_i.

Rewriting inequality equation 10, we obtain

$$\rho_0 \alpha_0 2C_0 + \sum_{t=0}^{T-1} \alpha_t s_t \;\leq\; \sqrt{\tfrac{2\varepsilon^{\text{rényi}}}{q}} \sqrt{\sum_{i=0}^{T-1} \alpha_i^2 \sigma_i^2} \;\leq\; \sqrt{\tfrac{2\varepsilon^{\text{rényi}} \sigma^2}{q}} \sqrt{\sum_{i=0}^{T-1} \alpha_i^2}. \tag{15}$$

Thus, T does not decrease if we set $\sigma_i = \sigma$ for every step.

For convenience, define

$$C_b := \sqrt{\frac{2\varepsilon^{\text{rényi}}\sigma^2}{q}}. \tag{16}$$

We now expand the inequality we aim to obtain, using the fact that

$$\alpha_t = \prod_{k=t+1}^{T-1} \rho_k = (1-\gamma\lambda)^{T-1-t}. \tag{17}$$

The left-hand side is

$$\begin{aligned} &2C_0(1-\gamma\lambda)^T + 2\gamma C_1 \sum_{t=0}^{T-1}(1-\gamma\lambda)^{T-1-t} \\ &= 2C_0(1-\gamma\lambda)^T + 2\gamma C_1 \tfrac{1-(1-\gamma\lambda)^T}{\gamma\lambda}, \end{aligned} \tag{18}$$

while the right-hand side is

$$C_b\sqrt{\sum_{t=0}^{T-1}(1-\gamma\lambda)^{2T-2-2t}} \;=\; C_b\sqrt{\tfrac{1-(1-\gamma\lambda)^{2T}}{1-(1-\gamma\lambda)^2}}. \tag{19}$$

Observe that both sides of the inequality are positive, hence it suffices to require that the square of the left-hand side exceeds the square of the right-hand side.

Introduce the variable

$$x = (1-\gamma\lambda)^T. \tag{20}$$

Then T can be recovered from x as

$$T = \tfrac{\log(x)}{\log(1-\gamma\lambda)}. \tag{21}$$

Thus, the problem of finding the minimal T is equivalent to finding the *maximal* x, subject to the constraint $0 < x \leq 1$.

By rewriting the difference of squares of the left- and right-hand sides, the problem reduces to finding the maximal root in the half-interval $(0, 1]$ of the equation

$$\left(2C_0 x + 2\gamma C_1 \tfrac{1-x}{\gamma\lambda}\right)^2 - C_b^2 \cdot \tfrac{1-x^2}{1-(1-\gamma\lambda)^2} = 0. \tag{22}$$

Equivalently,

$$\left(\left(\tfrac{2C_0}{C_b} - \tfrac{2C_1}{\lambda C_b}\right)x + \tfrac{2C_1}{\lambda C_b}\right)^2 - \tfrac{1-x^2}{1-(1-\gamma\lambda)^2} = 0. \tag{23}$$

Define

$$\zeta := \tfrac{1}{1-(1-\gamma\lambda)^2}, \qquad \beta_0 := \tfrac{2C_1}{\lambda C_b}, \qquad \beta_1 := \tfrac{2C_0}{C_b} - \tfrac{2C_1}{\lambda C_b} = \beta_0\big(\tfrac{\lambda C_0}{C_1} - 1\big). \tag{24}$$

Then the quadratic equation can be written as

$$(\beta_1^2 + \zeta)x^2 + 2\beta_0\beta_1 x + (\beta_0^2 - \zeta) = 0. \tag{25}$$

Its discriminant is

$$\begin{aligned} \Delta &= 4\beta_0^2\beta_1^2 - 4(\beta_1^2+\zeta)(\beta_0^2-\zeta) \\ &= 4\zeta(\zeta + \beta_1^2 - \beta_0^2). \end{aligned} \tag{26}$$

The largest root of the quadratic is given by

$$x_{\max} = \frac{-2\beta_0\beta_1 + \sqrt{4\zeta(\zeta+\beta_1^2-\beta_0^2)}}{2(\beta_1^2+\zeta)}. \tag{27}$$

We now state the condition for the existence of a root:

$$\zeta \;\geq\; \beta_0^2 - \beta_1^2 = \beta_0^2 - \left(\tfrac{2C_0}{C_b} - \beta_0\right)^2 = \tfrac{8C_0C_1}{C_b^2\lambda} - \tfrac{4C_0^2}{C_b^2}. \tag{28}$$

Equivalently,

$$\frac{1}{1-(1-\gamma\lambda)^2} \;\geq\; \frac{8C_0C_1}{C_b^2\lambda} - \frac{4C_0^2}{C_b^2} = \frac{2q}{\sigma^2\varepsilon^{\text{rényi}}}\Big(\frac{2}{\lambda}C_0C_1 - C_0^2\Big). \tag{29}$$

This implies

$$\sigma^2 \;\geq\; \gamma(2-\gamma\lambda)\cdot\frac{2q}{\varepsilon^{\text{rényi}}}\cdot\Big(2-\frac{\lambda C_0}{C_1}\Big)\cdot C_0C_1, \tag{30}$$

In the case $\frac{\lambda C_0}{C_1} \in (0,1)$, the right-hand side of the inequality is positive, which yields a bound on the minimal σ in this regime.

To obtain the bound on T, it suffices to substitute all original variables into the formula 27 for $x_{\max}$.

In the case of the minimal σ, the discriminant vanishes, and the expression simplifies to

$$\begin{aligned} x_{\max} &= \frac{-\beta_0\beta_1}{\beta_1^2+\zeta} = \frac{-\beta_0\beta_1}{\beta_0^2} = \frac{-(2\lambda C_0 - 2C_1)}{2C_1} \\ &= 1-\frac{\lambda C_0}{C_1} \in (0,1]. \end{aligned} \tag{31}$$

Therefore, substituting into

$$T = \frac{\log(x)}{\log(1-\gamma\lambda)}, \tag{32}$$

we obtain the desired estimate for T.

Furthermore, for $\frac{\lambda C_0}{C_1} \in [1,2)$, the bound on the noise obtained from the discriminant is also positive; however, in this case, the corresponding $x_{\max}$ would fall outside the interval $(0,1)$, as required.

To analyze the regime $\frac{\lambda C_0}{C_1} \geq 1$, we observe that $\beta_1 \leq 0$, implying that the vertex of the parabola lies at a negative coordinate. Consequently, the value at 0 must be negative (since the value at 1 is positive). From the inequality $\zeta > \beta_0^2$, one therefore derives a bound on σ^2. It is worth noting, however, that in the case of equality, $x_{\max}$ becomes zero, which corresponds to an infinite T.

Formula for minimal σ with given T

Given argumentation above, we can express σ in terms of $x_{\max}$:

$$\text{Let} \quad z := 1-\frac{\lambda C_0}{C_1}, \qquad x := x_{\max} \in (0,1]. \tag{33}$$

We now rewrite the equation 27 for $x_{\max}$ in terms of σ, which leads to a quadratic equation in $s := 1/\sigma^2$:

$$as^2 + bs + c = 0, \tag{34}$$

where

$$\begin{aligned} a &= \left(\frac{2q\,C_1^2}{\varepsilon^{\text{rényi}}\,\lambda^2}\right)^2 z^2\,(1-xz)^2, \\ b &= \left(\frac{2q\,C_1^2}{\varepsilon^{\text{rényi}}\,\lambda^2}\right)\frac{1}{\gamma\lambda(2-\gamma\lambda)}\Big[1-2xz+(2x^2-1)z^2\Big], \\ c &= \frac{x^2-1}{\big(\gamma\lambda(2-\gamma\lambda)\big)^2}. \end{aligned} \tag{35}$$

Observe that $a > 0$ and $c < 0$, hence we are interested in the largest root of the quadratic.

The (positive) solution for σ^2 can then be written as

$$\sigma^2(x) = \frac{2a}{-b+\sqrt{b^2-4ac}}. \tag{36}$$

Or the equivalent expression

$$\sigma^2(x) = \frac{-b-\sqrt{b^2-4ac}}{2c}. \tag{37}$$

F.2 PROOF OF THE THEOREM 4.1

The proof proceeds analogously to that of the original theorem (Koloskova et al., 2025).

We adapt Lemma A7 from the original proof, using adapted definitions of ∞-Wasserstein distance and shifted Rényi divergence.

Lemma F.3 (Decomposed Lemma A7 (Koloskova et al., 2025)). *Let $q \geq 1$, $z, \rho, s \geq 0$, $\psi : \mathbb{R}^d \to \mathbb{R}^d$, and X, Y be arbitrary random variables. If ψ satisfies, for all $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^d$ (for a single component i, while for the others nothing changes),*

$$\|\psi(\mathbf{x}') - \psi(\mathbf{x})\| \;\leq\; \rho\|\mathbf{x}' - \mathbf{x}\| + s,$$

then

$$D_q^{(z_1,\ldots,(\rho z_i+s),\ldots,z_k)}(\psi(X) \,\|\, \psi(Y)) \;\;\leq\;\; D_q^{(z_1,\ldots,z_i,\ldots,z_k)}(X \,\|\, Y).$$

Proof. To adapt the original proof, it suffices to observe that for the decomposed Wasserstein distance the following inequality holds:

from $W_d(\mu, \nu) \preceq z = (0, \ldots, z_i, \ldots, 0)$

$$W_d(\psi_\#(\mu), \psi_\#(\nu)) \preceq (0, \ldots, (\rho z_i + s), \ldots, 0),$$

when we modify the i-th component. This inequality indeed holds due to the assumption on ψ: for every pair (x, x'), the condition ensures the bound componentwise. □

Using the lemma above and decomposed Shift Reduction Lemma (4.5), we can proceed block by block. By zeroing out the coordinates sequentially, we bound the *increment* in Rényi divergence contributed by block i by a quantity $\varepsilon_i^{\text{rényi}}$. Summing these contributions yields an overall

$$(\textstyle\sum_{i=1}^k \varepsilon_i^{\text{rényi}},\, q)\text{-RDP}.$$

Since we already have an estimate for the number of steps required for unlearning in each component, summing them gives the bound on the total number of steps.

Applying the standard conversion from Rényi DP to (ε, δ)-DP (Mironov, 2017) gives

$$\varepsilon(\delta) \;=\; \sum_{i=1}^k \varepsilon_i^{\text{rényi}} \;+\; \frac{\log(1/\delta)}{q-1} \;=\; \sum_{i=1}^k \varepsilon_i \;-\; (k-1)\frac{\log(1/\delta)}{q-1}, \tag{38}$$

where we define $\varepsilon_i := \varepsilon_i^{\text{rényi}} + \frac{\log(1/\delta)}{q-1}$ for notational convenience.

We emphasize that we do not claim each block is itself an (ε_i, δ)-mechanism; the equality above is an algebraic rewriting of the single $(\sum_i \varepsilon_i^{\text{rényi}}, q)$-RDP bound after conversion.

Equal blocks corollary.

To guarantee the same budget as the baseline algorithm, it suffices to set

$$\varepsilon_i^{\text{renyi}} = \frac{\varepsilon^{\text{renyi}}}{k}.$$

Moreover, due to the randomness in the distribution of the model weights and gradients, the group norms are approximately equal. Hence we may treat

$$\frac{1}{\sqrt{k}}C_0 \quad \text{and} \quad \frac{1}{\sqrt{k}}C_1$$

as the individual clipping bounds for B_i.

Keeping the same noise level (not necessarily minimal) does not change the number of steps T for each group, since the factors of $\sqrt{k}$ compensate. Thus, the total cost is exactly kT steps for the whole algorithm. Finally, we may choose a larger noise level, thereby reducing T. Since in each step we add substantially less noise to the model than in the baseline algorithm, we have the flexibility to increase the noise.

G EXPERIMENTS

Listing 1: Architecture of the model used for MNIST.

```
import torch.nn as nn
import torch.nn.functional as F

class LinearNet(nn.Module):
   def __init__(self, num_classes: int = 10):
      super().__init__()
      self.flatten = nn.Flatten()
      self.fc1 = nn.Linear(28 * 28, 2048)
      self.fc2 = nn.Linear(2048, 1024)
      self.fc3 = nn.Linear(1024, 512)
      self.fc4 = nn.Linear(512, 256)
      self.fc5 = nn.Linear(256, num_classes)

   def forward(self, x):
      x = self.flatten(x)
      x = F.relu(self.fc1(x))
      x = F.relu(self.fc2(x))
      x = F.relu(self.fc3(x))
      x = F.relu(self.fc4(x))
      x = self.fc5(x)
      return x
```

Table 3: *Training* hyperparameters for the fully trained MNIST model and the retrain baseline (identical settings).

Parameter	**Fully trained**	**Retrain baseline**
Optimizer	SGD	SGD
Learning rate	0.005	0.005
Momentum	0.9	0.9
Weight decay	1×10^{-5}	1×10^{-5}
Batch size	64	64

Table 4: Training hyperparameters for the ViT-Tiny model and the retrain baseline (identical settings). The model is instantiated via `timm.create_model("vit_tiny_patch16_224", pretrained=True)` (Wightman, 2019).

Parameter	**Pretrained ViT-Tiny**	**Retrain baseline**
Model architecture	ViT-Tiny (patch 16)	same
Pretrained initialization	Yes	Yes
Optimizer	AdamW	AdamW
Learning rate	3×10^{-4}	3×10^{-4}
Weight decay	0.05	0.05
Batch size	64	64
Epochs	50	50
LR scheduler	CosineAnnealingLR ($T_{\max} = 50$)	same
Warmup	none	none
Image resolution	224 (ViT input)	224
Data augmentation	Resize(224) + RandomHorizontalFlip + ToTensor + Normalize(μ, σ)	same

Table 5: Unlearning hyperparameters for Block-wise NFT and NFT under random 10% deletion.

Parameter	**ViT-Tiny / CIFAR-100**	**MNIST**
Initial distance bound $\Delta(\rho)$	0.25	0.5
Gradient clipping C_1	33	5000
Unlearning step size η	0.002	1×10^{-4}
Weight decay λ (unlearning)	50	100
Failure probability δ	10^{-5}	10^{-5}
Number of steps T (per block)	2	2
Fine-tuning optimizer	AdamW	SGD
Fine-tuning learning rate	5×10^{-3}	1×10^{-3}
Fine-tuning weight decay	0	1×10^{-5}
Fine-tuning momentum	–	0.9

G.1 Theory-motivated hyperparameter initialization

For NFT-based methods, the certificate suggests a principled initialization for the noise scale and clipping radius. For fixed step budget T, learning rate γ, weight decay λ, proximity parameter $\Delta(\rho)$, and target (ε, δ), one can choose the Rényi parameters $(q, \varepsilon^{\mathrm{renyi}})$ to minimize the induced noise term under the conversion

$$\varepsilon = \varepsilon^{\mathrm{renyi}} + \frac{\log(1/\delta)}{q-1}.$$

This yields

$$\sigma^2 = \lambda\gamma(2-\lambda\gamma)\frac{1+(1-\lambda\gamma)^T}{1-(1-\lambda\gamma)^T}\frac{8q}{\varepsilon^{\mathrm{renyi}}}\Delta(\rho)^2, \qquad C_1 = \frac{2\lambda\Delta(\rho)}{1-(1-\lambda\gamma)^T}.$$

These expressions are not a universal rule, but a certificate-compatible warm start that can narrow the search space.

G.2 Experiments with different block structure

We further compare 4 constructions of the block mixing matrix A (see Section 4.2):

- a random A with equal-sized blocks;
- a permutation-matrix A with equal-sized blocks;
- a cyclic layer-wise grouping into k blocks, assigning layer ℓ to block $\ell \bmod k$
- a two-block partition on head and the rest of the model (only for ViT-Tiny).

We provide results of experiments for both datasets (MNIST and CIFAR-10). For forget set $\mathcal{D}_f$, we use random 10% deletion task.

Results on MNIST.

Figure 4 shows MNIST trajectories at privacy budgets $\varepsilon \in \{30., 60.\}$, $\delta = 10^{-5}$ and number of blocks $k = 2$. All three schemes exhibit very similar behavior during unlearning and subsequent fine-tuning; the main difference is the magnitude of the initial accuracy drop, which is slightly smaller for the odd-even split than for random splitting.

Results on ViT-Tiny.

Figure 5 shows CIFAR-10 trajectories at privacy budgets $\varepsilon = 50$, $\delta = 10^{-5}$ and number of blocks $k = 4$. All schemes demonstrate comparable stability during unlearning and recovery under fine-tuning.

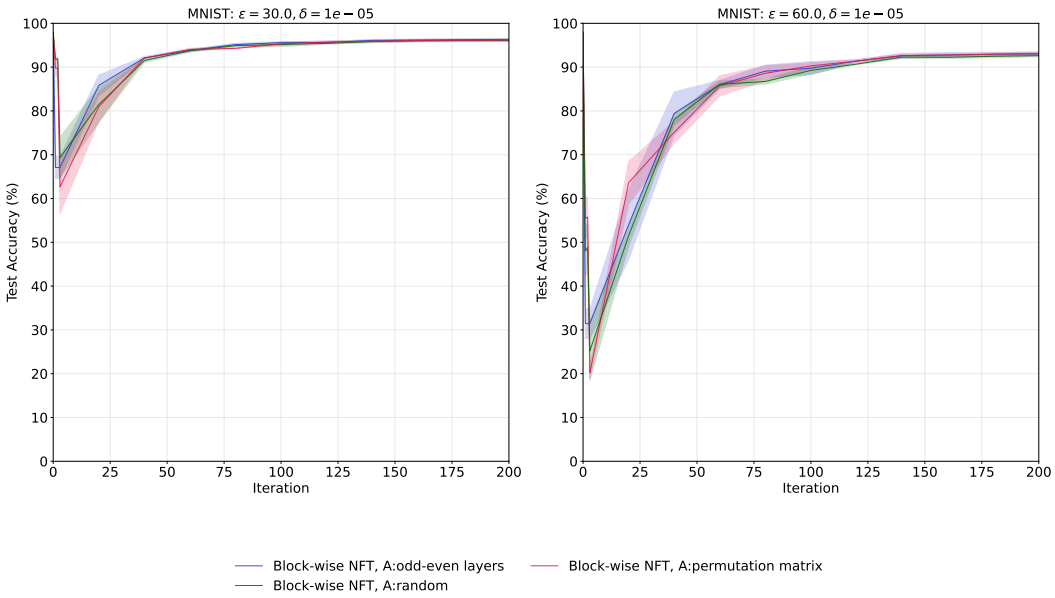


Figure 4: **Block-construction schemes** for Block-wise NFT on MNIST at $\varepsilon \in \{30, 60\}$ and $\delta = 10^{-5}$.

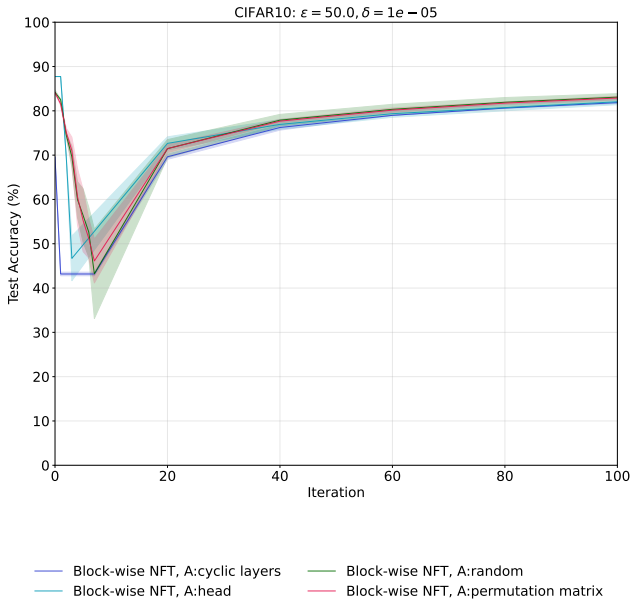


Figure 5: **Block-construction schemes** for Block-wise NFT on CIFAR-10 for ViT-Tiny at $\varepsilon = 50$ and $\delta = 10^{-5}$.

G.3 Ablations

Table 6: Effect of the number of blocks k on ViT-Tiny/CIFAR-10 for privacy budget $\varepsilon = 50$. Min Acc. = minimum accuracy during unlearning before post-fine-tuning; Final Acc. = accuracy after post-fine-tuning; RTE is reported in *seconds*; RTE (blocks only) excludes post-fine-tuning.

k	Min Acc.	Final Acc.	RTE	RTE (blocks only)	ε
2	37.6±7.71	89.69±0.49	90.77±0.93	1.46±0.3	50
4	41.69±6.06	89.89±0.56	91.37±0.51	2.33±0.34	50
8	40.12±1.09	89.74±0.2	92.01±0.67	3.97±0.07	50
16	34.25±8.2	89.58±0.66	95.27±0.55	9.01±0.21	50
32	34.38±10.11	89.62±0.17	108.18±0.32	24.66±0.11	50

Table 7: Effect of the number of blocks k on ViT-Tiny/CIFAR-10 for privacy budget $\varepsilon = 30$. Min Acc. = minimum accuracy during unlearning before post-fine-tuning; Final Acc. = accuracy after post-fine-tuning; RTE is reported in *seconds*; RTE (blocks only) excludes post-fine-tuning.

k	Min Acc.	Final Acc.	RTE	RTE (blocks only)	ε
2	21.77±3.93	88.45±1.1	91.9±0.51	1.45±0.3	30
4	20.62±5.63	89.27±0.68	91.4±0.58	2.03±0.06	30
8	24.34±9.09	89.02±0.33	92.81±0.32	3.97±0.11	30
16	13.94±4.01	88.52±1.29	95.87±0.76	9.01±0.08	30
32	13.47±0.43	88.74±0.55	108.65±0.33	24.59±0.03	30

Table 8: Effect of block construction and the number of blocks k on ViT-Tiny/CIFAR-10 at privacy budget $\varepsilon = 50$. Min Acc. = minimum accuracy during unlearning before post-fine-tuning; Final Acc. = accuracy after post-fine-tuning; RTE is reported in *seconds*; RTE (blocks only) excludes post-fine-tuning.

k	Block mode	Min Acc.	Final Acc.	RTE	RTE (blocks only)
2	random	36.39±15.99	89.81±0.52	92.37±0.32	1.28±0.02
2	random permutation	39.74±3.3	89.53±0.29	90.98±0.75	0.7±0.03
2	layer-wise	38.94±1.06	89.9±0.22	91.25±0.69	0.8±0.29
4	random	41.69±6.06	89.89±0.56	92.38±0.18	2.15±0.13
4	random permutation	36.35±1.46	89.67±0.33	91.37±0.58	1.39±0.02
4	layer-wise	35.54±5.11	89.73±0.16	91.52±0.32	1.3±0.01
8	random	40.12±1.09	89.74±0.2	93.42±0.14	3.95±0.06
8	random permutation	44.41±2.19	89.98±0.19	93.03±0.19	3.3±0.19
8	layer-wise	34.34±6.56	89.73±0.12	92.05±0.34	2.89±0.02

Table 9: Sensitivity to randomness in block partitioning for $k = 2$ on ViT-Tiny/CIFAR-10, reported for different privacy budgets ε. Min Acc. = minimum accuracy during unlearning before post-fine-tuning; Final Acc. = accuracy after post-fine-tuning.

k	ε	Min Acc.	Final Acc.
2	50	38.92±7.1	89.87±0.49
2	70	51.58±2.37	90.09±0.41
2	90	57.56±0.0	90.52±0.0

G.4 Additional experiments

We also report results for a more challenging setup in which 50% of the training data are selected uniformly at random for removal.

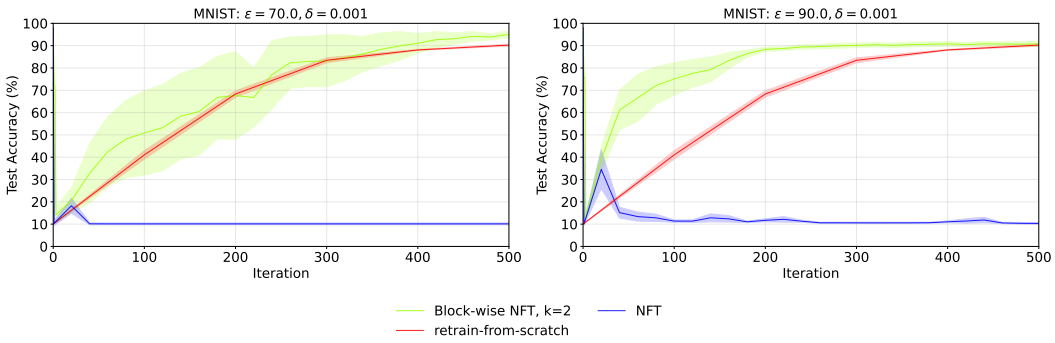


Figure 6: **Random 50% deletion.** We compare standard NFT with Block-wise NFT with two blocks, and show the final accuracy of retraining from scratch for reference. Despite the larger forget set, Block-wise NFT achieves accuracy recovery.

H RECALCULATION OF $\Delta(\rho)$

For each block, the NFT algorithm is parameterized by $\lambda, \gamma, \Delta(\rho), C_1$ and the budget parameters $\varepsilon_{\text{rényi}}$ and q.

Given these parameters, the noise variance σ^2 and the number of unlearning steps T are determined by the algorithm 1.

Importantly, only the parameters $(\lambda, \gamma, C_1, \sigma^2, T)$ affect the optimization trajectory of the algorithm. In particular, if two NFT instantiations share the same values of $(\lambda, \gamma, C_1, \sigma^2, T)$, then they generate identical parameter updates (with shared random seed) and therefore follow the same optimization path.

This observation allows us to state the following result.

Proposition H.1. *Consider an NFT algorithm with parameters $\lambda, \gamma, \Delta(\rho), C_1$ and budget parameter $B := q/\varepsilon_{rényi}$, with noise variance σ^2 and number of unlearning steps T. Define*

$$\Delta^{new}(\rho) := m \cdot \Delta(\rho), \qquad B^{new} := \frac{1}{m^2} \cdot B,$$

for some $m \geq 1$.

Then if we run the algorithm 1 with parameters $\lambda, \gamma, \Delta^{new}(\rho), C_1, B^{new}$, while keeping the same values of σ^2 and T, *then the resulting algorithm follows exactly the same optimization trajectory and is certified with a* different budget, *correspondent to B^{new}.*

Proof. In Algorithm 1, the noise variance and the number of unlearning steps are computed according to

$$\sigma^2 = \gamma(2-\gamma\lambda)\frac{q}{\varepsilon^{\text{rényi}}}\left(2 - \frac{\lambda\Delta(\rho)}{2C_1}\right)\Delta(\rho)C_1,$$

and

$$T(\sigma^2_{\min}) \;=\; \frac{\log\!\left(1 - \frac{\lambda\Delta(\rho)}{2C_1}\right)}{\log(1-\gamma\lambda)}.$$

Under the transformation $\Delta(\rho) \to \Delta^{\text{new}}(\rho)$, $C_1 \to m \cdot C_1$, and $B \to B^{\text{new}}$, the values of σ^2 and T remain unchanged.

The resulting algorithm is therefore parameterized by $(\lambda, \gamma, mC_1, \sigma^2, T)$ and admits a valid certification. In the certification proof (Koloskova et al., 2025), the only property required of the gradient clipping constant is that the norm of the clipped gradient does not exceed this constant. For this reason, for the algorithm with the stricter clipping threshold C_1, the certification may safely use the larger value mC_1.

Consequently, an algorithm parameterized by $(\lambda, \gamma, C_1, \sigma^2, T)$ is certified with budget correspondent to B^{new}, and its optimization trajectory—fully determined by $(\lambda, \gamma, C_1, \sigma^2, T)$—coincides exactly with the original trajectory.

□

What does this mean in practice? Throughout our discussion, certification has always been understood as being conditional on the chosen value of $\Delta(\rho)$.

The proposition above shows that even if the selected $\Delta(\rho)$ turns out to be inaccurate, this does not invalidate the certification of the algorithm. The algorithm itself remains certified; however, the resulting privacy budget may differ from the one originally anticipated. In such a case, the budget can be recalculated using the updated estimate of $\Delta(\rho)$.

Importantly, this recalculation does not alter the optimization trajectory of the algorithm. The sequence of updates remains unchanged. Consequently, the certification budget can be viewed as a function of the chosen estimate of $\Delta(\rho)$, while the underlying algorithmic behavior stays fixed.

H.1 How to calculate a new budget

To decrease $\dfrac{1}{B}$ by m^2 times, we need to understand connection between $\dfrac{1}{B}$ and unlearning budget (ε, δ).

For given (ε, δ), the pair $\varepsilon_{\text{rényi}}, q$ is calculated to minimize σ^2, or, equally, B.

$$B = \frac{q}{\varepsilon_{\text{rényi}}} \to \min, \quad \text{where} \quad \varepsilon = \varepsilon^{\text{rényi}} + \tfrac{\log(1/\delta)}{q-1}$$

Denote $a := \log(1/\delta)$. From the constraint,

$$\varepsilon_{\text{rényi}}(q) = \varepsilon - \frac{a}{q-1}, \qquad B(q) = \frac{q}{\varepsilon - \frac{a}{q-1}}, \qquad q > 1 + \frac{a}{\varepsilon}.$$

Let $t = q - 1$. Minimizing B is equivalent to minimizing

$$B(t) = \frac{t(t+1)}{\varepsilon t - a}, \qquad t > \frac{a}{\varepsilon}.$$

The unique minimizer satisfies

$$\varepsilon t^2 - 2at - a = 0 \quad \Rightarrow \quad t^\star = \frac{a + \sqrt{a(a+\varepsilon)}}{\varepsilon}.$$

Hence

$$q^\star = 1 + t^\star = 1 + \frac{a + \sqrt{a(a+\varepsilon)}}{\varepsilon},$$

and

$$\varepsilon^\star_{\text{rényi}} = \varepsilon - \frac{a}{t^\star} = \varepsilon + a - \sqrt{a(a+\varepsilon)}.$$

Therefore,

$$q^\star = 1 + \frac{\log(1/\delta) + \sqrt{\log(1/\delta)\big(\log(1/\delta) + \varepsilon\big)}}{\varepsilon}, \qquad \varepsilon^\star_{\text{rényi}} = \varepsilon + \log(1/\delta) - \sqrt{\log(1/\delta)\big(\log(1/\delta) + \varepsilon\big)}.$$

The graph 7 illustrates dependency ε and $1/B$ with fixed $\delta = 10^{-5}$.

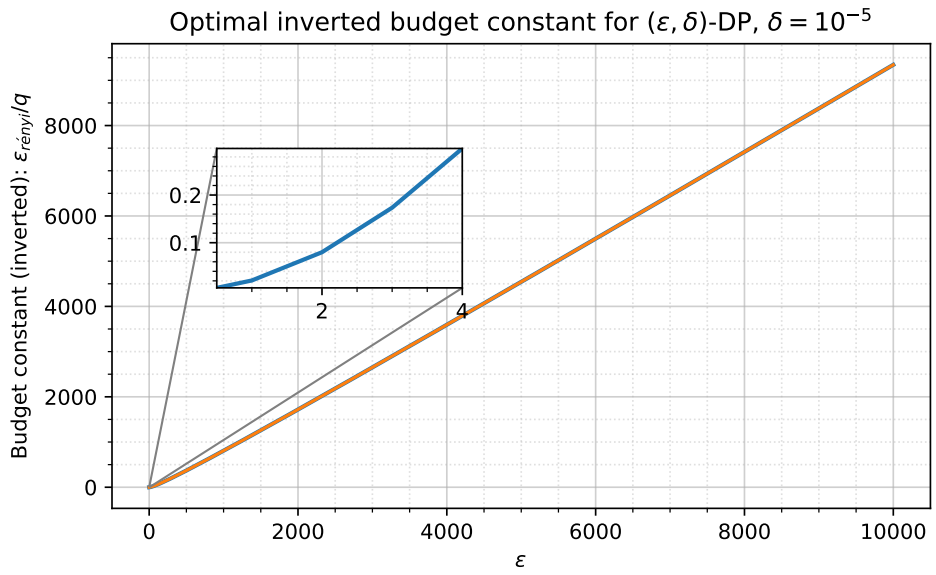


Figure 7: Budget constant connection to budget ε

I Calibration and Estimation of $\Delta(\rho)$

I.1 Calibration of the Discrepancy Parameter

Role of calibration in the theoretical guarantee. The discrepancy parameter $\Delta(\rho)$ instantiates the initial-shift bound required by the NFT analysis. Formally, let S be an independent sample of coupled training runs. A one-sided calibration procedure should return an estimate $\widehat{\Delta}(\rho; S)$ satisfying

$$\Pr_S\left[\Pr\left(\|\hat{x} - \hat{x}'\| \le \widehat{\Delta}(\rho; S)\right) \ge 1 - \rho\right] \ge 1 - \beta.$$

Thus, with confidence at least $1 - \beta$ over the calibration sample, the estimated value satisfies the ρ-proximity condition assumed by the theory.

For every calibration outcome for which this condition holds, Algorithm 1 substitutes $C_0 = \widehat{\Delta}(\rho; S)/2$ into Theorem 3.1 to determine σ^2 and T for each block. Theorem 4.2 then composes the resulting per-block privacy losses. Calibration and block-wise composition therefore play separate roles: calibration provides a value satisfying the initial-shift premise, while the composition theorem is agnostic to how this value is obtained. Here, ρ is the failure probability in the proximity condition, whereas β quantifies the confidence that the calibration procedure has produced a valid bound.

In our experiments, we use the maximum observed discrepancy across 10 coupled training runs as a practical empirical approximation rather than as a formal one-sided confidence bound.

Dependence on the magnitude of $\Delta(\rho)$. Holding C_1 fixed, for a fixed target (ε, δ), a larger $\Delta(\rho)$ leads to a more conservative noisy phase. In the first regime of Theorem 3.1,

$$\sigma^2_{\min} \propto \Delta(\rho)\left(2 - \frac{\lambda\Delta(\rho)}{2C_1}\right), \qquad T = \frac{\log\left(1 - \frac{\lambda\Delta(\rho)}{2C_1}\right)}{\log(1 - \gamma\lambda)}.$$

Hence, both the required noise and the number of noisy updates increase with $\Delta(\rho)$, and Block-wise NFT consequently performs kT noisy block updates.

Conversely, for a fixed completed trajectory (σ^2, T), revising $\Delta(\rho)$ upward does not change the model updates, but yields a weaker certified budget. As shown in Appendix H, this budget can be recomputed without rerunning unlearning.

Calibration across deployment regimes. For a fixed architecture, training pipeline, coupling, and deletion regime, $\Delta(\rho)$ is an upper $(1 - \rho)$-quantile of $\|\hat{x} - \hat{x}'\|$. It can therefore be estimated before

deployment from independent coupled training pairs using a one-sided confidence bound. Calibration is performed before receiving deletion requests and can be reused within the same regime.

The calibration cost consists of m paired training runs, which can be parallelized and amortized over subsequent requests within that regime. Recalibration is needed only when the architecture, training pipeline, data distribution, or deletion regime changes materially. This regime dependence is not specific to our method: practical unlearning evaluations commonly select method-specific quantities for the deletion setting under consideration, and these quantities must likewise be revisited when the deployment regime changes.

I.2 Empirical Estimation of $\Delta(\rho)$

Empirical sensitivity estimation for $\Delta(\rho)$. To instantiate $\Delta(\rho)$ in our experiments, we perform an architecture-level sensitivity estimation procedure *prior* to receiving any unlearning request. The goal is to quantify how far the trained parameters can move under small, controlled changes of the training set, under a coupled comparison of training trajectories.

Coupled training protocol. We run training multiple times (10 runs in our experiments) under fixed training randomness: we fix all seeds and the data order, and keep the optimization hyperparameters unchanged. We then construct a new training stream by applying a controlled dataset modification while preserving the same mini-batch structure as much as possible. Concretely, for *random* 10% *deletion* we remove a randomly selected 10% subset of training examples. The removed samples are replaced by retained samples so that the overall training pipeline remains coupled across the two runs.

Distance measurement and aggregation. For each run, we train on the original dataset and on its perturbed counterpart, and measure the resulting parameter distance. We aggregate these distances across the 10 runs and use the resulting value as an empirical upper estimate for $\Delta(\rho)$ for the considered architecture and deletion setting.

As illustrated by the calibration plots, the *time evolution* of the proximity—i.e., how the distance between coupled trajectories grows and stabilizes over training—is often consistent across runs under fixed coupling. This suggests that, for a given architecture and training pipeline, one can characterize this trajectory in advance and use it as a practical guide (e.g., for selecting conservative instantiations of $\Delta(\rho)$), while trading a fully distribution-free bound for a fast, architecture-specific calibration.

In addition, storing intermediate checkpoints during training can be beneficial: in practice, later checkpoints with comparable accuracy may exhibit smaller estimated proximity, and initializing unlearning from such a checkpoint can reduce the resulting certified budget for the same unlearning trajectory.

Figures 8–13 visualize the parameter distance Δ (right axis) together with test accuracy (left axis) over training steps for different deletion types and architectures. Here $\Delta := \Delta(1)$ denotes the ℓ_2 distance between the parameters of the two coupled runs. Notably, these trajectories often exhibit a regime where accuracy is already high while Δ is still relatively small, motivating the use of earlier checkpoints when a tighter proximity (and thus a smaller certified budget) is desired. On MNIST, varying the learning rate can noticeably change the magnitude of Δ even when the final accuracy is similar (Figures 8 and 9).

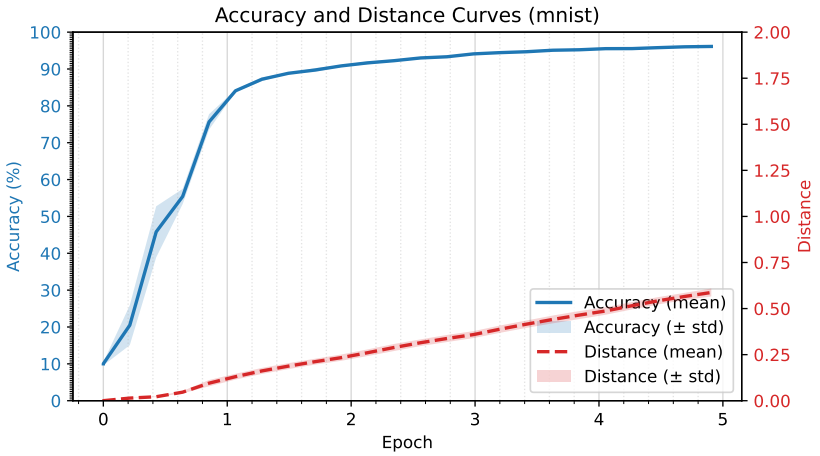


Figure 8: **MNIST (4-layer MLP), random 10% deletion, lr=0.005.** Test accuracy (left) and parameter distance Δ (right) over training steps.

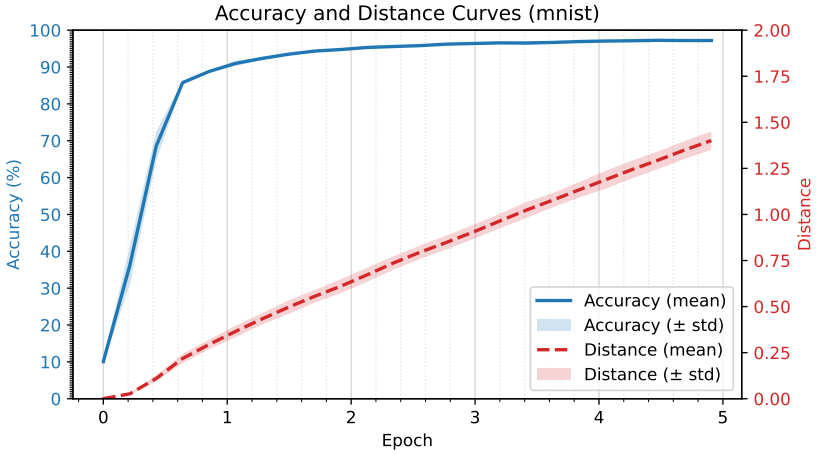


Figure 9: **MNIST (4-layer MLP), random 10% deletion, lr=0.01.** Test accuracy (left) and parameter distance Δ (right) over training steps.

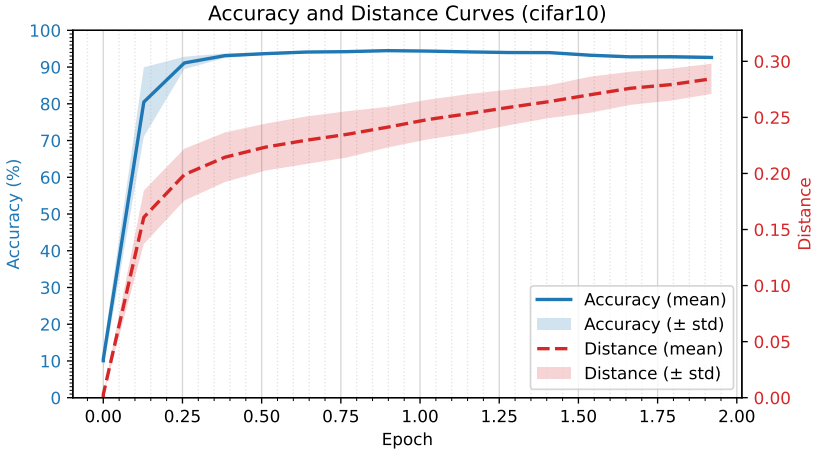


Figure 12: **CIFAR-10 (ViT-Tiny), random 10% deletion, lr=3e-4.** Test accuracy (left) and parameter distance Δ (right) over training steps.

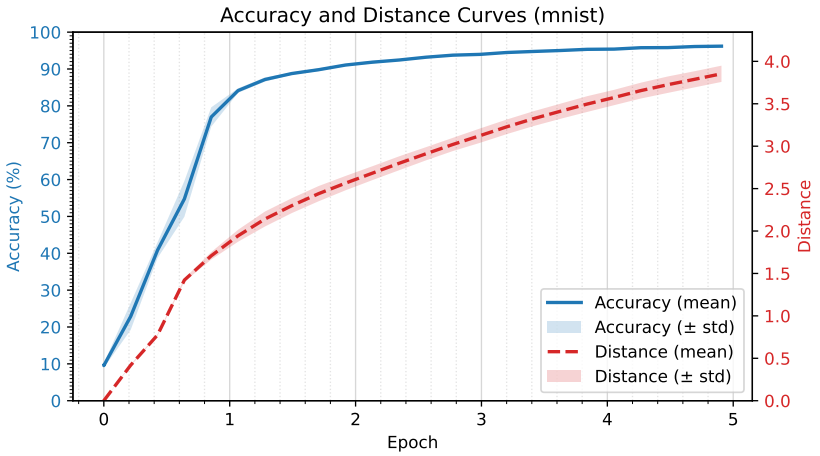


Figure 10: **MNIST (4-layer MLP), class deletion, lr=0.005.** Test accuracy (left) and parameter distance Δ (right) over training steps.

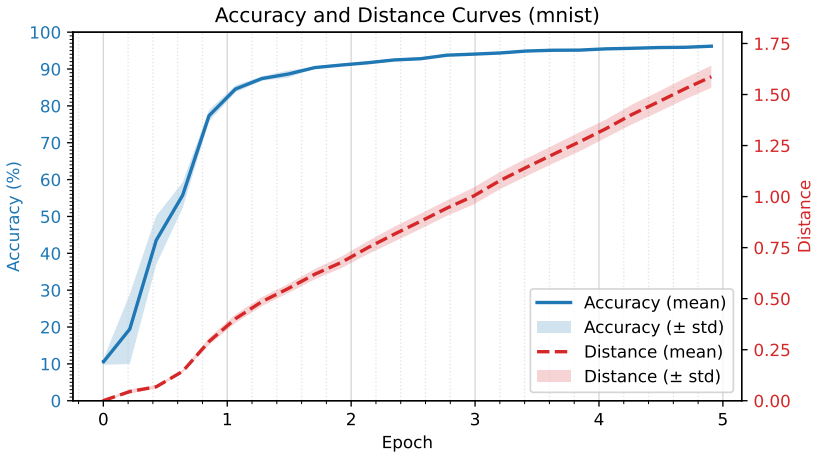


Figure 11: **MNIST (4-layer MLP), random 50% deletion, lr=0.005.** Test accuracy (left) and parameter distance Δ (right) over training steps.

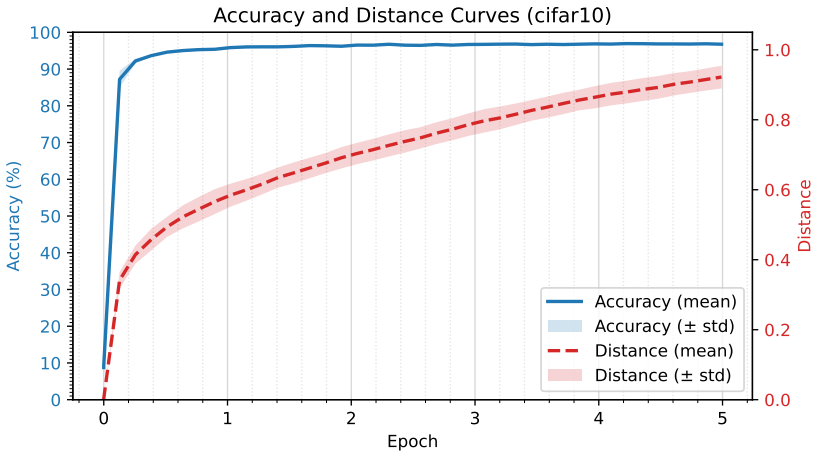


Figure 13: **CIFAR-10 (ViT-Tiny), class deletion, lr=3e-4.** Test accuracy (left) and parameter distance Δ (right) over training steps.

I.3 Sensitivity to proximity calibration

We additionally study how sensitive Block-wise NFT is to the instantiated value of $\Delta(\rho)$. Recall that $\Delta(\rho)$ enters only through the certified noise calibration: larger values correspond to a more conservative estimate of the initial discrepancy between the full-data and retained-data training runs.

Figure 14 shows the effect of overestimating $\Delta(\rho)$ by a multiplicative factor m. Moderate overestimation has limited effect on the final model: the final accuracy remains stable for small increases of $\Delta(\rho)$, while larger overestimation eventually makes the certified noisy phase too conservative and degrades utility. This supports the view that the method does not require an exactly optimal calibration, although overly pessimistic values of $\Delta(\rho)$ can still harm performance.

Figure 15 illustrates the complementary re-certification perspective. If a completed run is later evaluated under a larger value $m\Delta(\rho)$, the same trajectory can be certified with a larger privacy/unlearning budget as described in Appendix H. The increase is smooth for moderate changes in $\Delta(\rho)$, which means that small calibration corrections do not require rerunning unlearning; they can instead be accounted for by recalculating the certified budget.

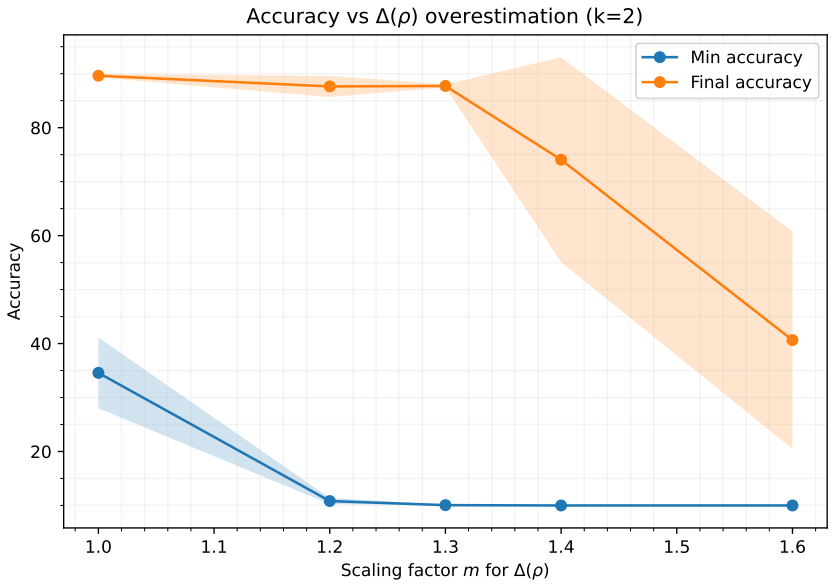


Figure 14: Sensitivity to $\Delta(\rho)$ overestimation ($k = 2$) for ViT-Tiny/CIFAR-10. The x-axis shows the factor m in $\Delta(\rho) \to m\Delta(\rho)$, while the y-axis reports accuracy. Here, Min accuracy denotes the accuracy immediately after the unlearning steps, before post-fine-tuning, i.e., the worst accuracy reached during the unlearning process. Moderate overestimation (up to $\sim$30%) preserves high final accuracy, while larger values lead to degradation.

J Computational Complexity and Runtime

We distinguish between the one-time, reusable cost of constructing the block decomposition and the additional computation incurred for each unlearning request.

Layer-wise block parameterization. As described in Section 4.2, our implementation applies the decomposition independently to each trainable layer rather than constructing a global $d \times d$ basis over the full parameter vector. Let the reshaped weight tensor of layer ℓ be

$$W_\ell \in \mathbb{R}^{a_\ell \times b_\ell},$$

and let

$$A_{\ell,i} \in \mathbb{R}^{a_\ell \times r_{\ell,i}}, \qquad \sum_{i=1}^{k} r_{\ell,i} = a_\ell,$$

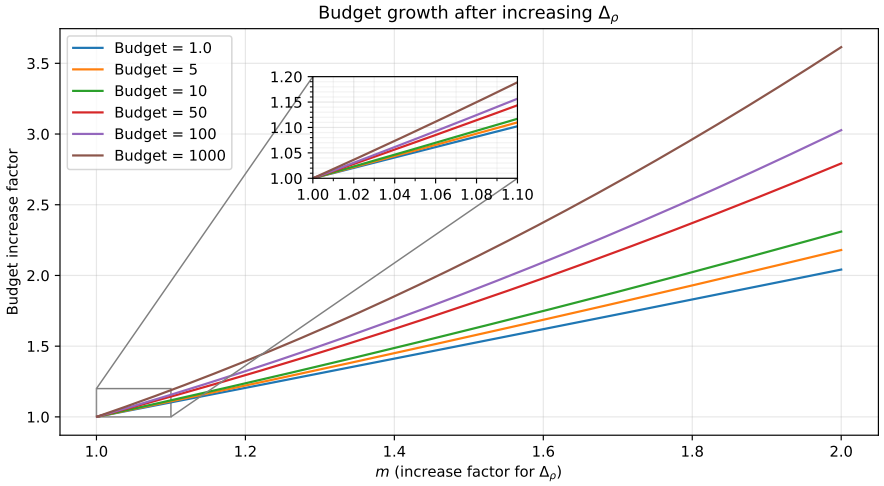


Figure 15: Budget recalculation under larger values of $\Delta(\rho)$. For a fixed unlearning trajectory, increasing the discrepancy estimate to $m\Delta(\rho)$ increases the certified budget by a smooth multiplicative factor. Thus, moderate corrections to $\Delta(\rho)$ can be handled by re-certification rather than rerunning unlearning.

Table 10: One-time setup cost for the block constructions considered in our experiments.

Block construction	Construction time	Persistent storage	Initial coordinate conversion
Random orthogonal	$\mathcal{O}\left(\sum_{\ell=1}^{L} a_\ell^3\right)$	$\mathcal{O}\left(\sum_{\ell=1}^{L} a_\ell^2\right)$	$\mathcal{O}\left(\sum_{\ell=1}^{L} a_\ell^2 b_\ell\right)$
Random permutation	$\mathcal{O}\left(\sum_{\ell=1}^{L} a_\ell\right)$	$\mathcal{O}\left(\sum_{\ell=1}^{L} a_\ell\right)$	$\mathcal{O}\left(\sum_{\ell=1}^{L} a_\ell b_\ell\right)$
Cyclic layer grouping	$\mathcal{O}(L)$	$\mathcal{O}(L)$ indices, or $\mathcal{O}(1)$ if implicit	None

denote the layer-wise block matrices. The decomposition is therefore applied as

$$W_\ell = \sum_{i=1}^{k} A_{\ell,i} B_{\ell,i}.$$

We use this notation below to describe the setup and per-request costs of the different block constructions.

J.1 One-time setup cost

For a trained model, the setup consists of constructing the layer-wise block structure and, when required, converting the model weights to the corresponding block coordinates. The resulting costs are summarized in Table 10.

For random orthogonal blocks, constructing an orthogonal basis for an $a_\ell \times a_\ell$ matrix costs $\mathcal{O}(a_\ell^3)$, while storing the basis requires $\mathcal{O}(a_\ell^2)$ memory. Computing the initial block coordinates through $A_\ell^\top W_\ell$ costs $\mathcal{O}(a_\ell^2 b_\ell)$.

For random permutation blocks, the dense basis operations are replaced by an $\mathcal{O}(a_\ell)$ permutation vector. If the reordered weights are explicitly materialized, the initial conversion requires $\mathcal{O}(a_\ell b_\ell)$ data movement. Cyclic layer grouping only assigns existing layers to blocks and therefore requires no coordinate conversion.

The orthogonal bases, permutations, or layer assignments depend only on the architecture and sampled block construction, rather than on a particular deletion request, and can therefore be reused. The conversion of W_ℓ to block coordinates is performed once for a particular trained model. For sequential deletion requests, this conversion need not be repeated provided that the updated model is retained in the same block parameterization.

J.2 Additional cost during unlearning

Block-wise NFT directly optimizes the active block while all remaining blocks are frozen. In particular, it does not compute a full-model gradient and then explicitly project it onto the active subspace.

Freezing parameters does not remove the ordinary model computation: frozen parameters still participate in the forward pass, and backward propagation may pass through frozen layers to reach the active block. We therefore distinguish the ordinary forward/backward cost from the additional operations introduced by Block-wise NFT.

Let

$$p_i = \sum_{\ell=1}^{L} r_{\ell,i} b_\ell$$

denote the number of trainable scalar parameters in block i. Clipping, noise generation, weight decay, and the optimizer update are applied only to the active block and require $\mathcal{O}(p_i)$ additional time and auxiliary memory.

For random orthogonal blocks, the block parameterization additionally requires multiplication by the fixed basis. While block i is active, we write

$$W_\ell = W_{\ell,-i} + A_{\ell,i} B_{\ell,i}, \qquad W_{\ell,-i} = \sum_{j \neq i} A_{\ell,j} B_{\ell,j}.$$

Since $W_{\ell,-i}$ remains fixed throughout the T updates of block i, it can be computed once and reused. With this caching, the additional parameterization cost per optimization step is

$$\mathcal{O}\left(\sum_{\ell=1}^{L} a_\ell r_{\ell,i} b_\ell \right).$$

Without caching, a conservative bound is

$$\mathcal{O}\left(\sum_{\ell=1}^{L} a_\ell^2 b_\ell \right).$$

For random permutation blocks, the additional parameterization cost is limited to linear indexing or data movement, with no dense basis multiplication. Cyclic layer grouping introduces no additional parameterization operation. Thus, all constructions require $\mathcal{O}(p_i)$ active-block operations for clipping, noise generation, weight decay, and optimization, while only the random orthogonal construction requires dense basis multiplication.

The corresponding auxiliary-memory costs are $\mathcal{O}(p_i)$ for the active-block operations. For random orthogonal blocks, caching $W_{\ell,-i}$ additionally requires $\mathcal{O}(\sum_\ell a_\ell b_\ell)$ memory. Random permutation may also require a temporary reordered tensor if it is explicitly materialized.

J.3 Total per-request cost

For one unlearning request, Block-wise NFT performs T noisy updates for each of the k blocks. Let $C_{\mathrm{fb},i}$ denote the cost of the ordinary forward/backward computation while block i is active, and let $C_{\mathrm{param},i}$ denote the additional cost induced by the chosen block parameterization. The block-wise phase therefore costs

$$C_{\mathrm{block}} = T \sum_{i=1}^{k} \left(C_{\mathrm{fb},i} + C_{\mathrm{param},i} + \mathcal{O}(p_i) \right),$$

where the $\mathcal{O}(p_i)$ term accounts for clipping, noise generation, weight decay, and the parameter update.

There are two implementation choices for the subsequent F full-model fine-tuning steps. First, the model can remain in block coordinates, with all blocks unfrozen:

$$C_{\text{post-FT}} = F \left(C_{\mathrm{full}} + C_{\mathrm{param,full}} \right).$$

Table 11: Runtime profiling on ViT-Tiny/CIFAR-10 with $\varepsilon = 50$, $k = 4$, and $T = 2$. Times are reported in seconds; persistent storage is additional storage introduced by the block construction.

Method	Construction	Reparam.	Noisy phase	Post-FT	Total	Storage
Random orthogonal	0.67±0.03	0.09±0.01	1.59±0.02	131.73±0.23	134.08±0.25	85.08 MiB
Random permutation	0.06±0.02	0.06±0.02	1.10±0.01	131.52±0.14	132.74±0.15	0.36 MiB
Cyclic layer grouping	0.006±0.0002	0.008±0.0001	1.08±0.02	132.14±0.91	133.24±0.89	0.36 MiB
NFT	–	–	0.42±0.04	132.54±0.45	132.96±0.43	–
Retraining	–	–	–	–	2743.90±18.73	–

Alternatively, the standard weights W_ℓ can be reconstructed once, after which fine-tuning proceeds in the original parameterization:

$$C_{\text{post-FT}} = C_{\text{map}} + FC_{\text{full}}.$$

Hence, the total per-request cost is

$$C_{\text{request}} = C_{\text{block}} + C_{\text{post-FT}}.$$

For the first request, the one-time block-construction and initial reparameterization costs must additionally be included.

J.4 Empirical runtime

We additionally profile the computational overhead on ViT-Tiny/CIFAR-10 with $\varepsilon = 50$, $k = 4$, and $T = 2$, using the same GPU and implementation for all methods in Table 11. We report mean $\pm$ standard deviation over three seeds and separately measure block construction, initial model reparameterization, the noisy/block-wise phase, and post-unlearning fine-tuning. For this profiling experiment, we use $F = 1500 - kT$ post-unlearning fine-tuning steps and train the retrained reference model for 30 epochs.

The setup and block-wise phases are small relative to post-unlearning fine-tuning, which dominates the total runtime. Across the three block constructions, the end-to-end runtime differs from standard NFT by less than 1% and is approximately $20\times$ lower than full retraining. Random permutation and cyclic layer grouping achieve this with negligible additional persistent storage, while random orthogonal blocks require additional storage for the dense orthogonal bases.

K A Utility Illustration for Sequential Block Updates

This section gives a simple illustration of how the block-wise construction can improve utility. We consider linear regression with zero weight decay and a common scalar noise scale σ, comparing one full-space NFT update with one complete Block-wise NFT pass consisting of one update for each of two coordinate-wise blocks. The result is not a general utility guarantee, but isolates an additional benefit of applying the noisy updates sequentially.

Model. Consider the least-squares objective

$$F(\theta) = \frac{1}{2n}\|X\theta - y\|^2, \qquad X \in \mathbb{R}^{n\times d}, \quad y \in \mathbb{R}^n,$$

and let $\theta^\star \in \arg\min_\theta F(\theta)$. Writing $H := X^\top X/n$, the normal equations give

$$F(\theta) - F(\theta^\star) = \frac{1}{2}(\theta - \theta^\star)^\top H(\theta - \theta^\star), \qquad \nabla F(\theta) = H(\theta - \theta^\star).$$

Fix the two orthogonal blocks used by Block-wise NFT, with dimensions d_1 and d_2 such that $d_1 + d_2 = d$, and express the parameter error and Hessian in the corresponding orthonormal coordinates:

$$\theta - \theta^\star = \begin{pmatrix} u \\ v \end{pmatrix}, \qquad H = \begin{pmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{pmatrix},$$

where $u \in \mathbb{R}^{d_1}$, $v \in \mathbb{R}^{d_2}$, $H_{ij} \in \mathbb{R}^{d_i \times d_j}$, and $H_{21} = H_{12}^\top$. We denote the second-block smoothness constant by

$$L_2 := \lambda_{\max}(H_{22}).$$

Deletion setting. We consider an idealized deletion regime in which the initial full-data model coincides with the retrained optimum, $\widehat{\theta} = \theta^\star$. This can represent uniform deletion at the population level, where removing an independent subset leaves the underlying objective and its optimum unchanged. The assumption removes any initial discrepancy and isolates the effect of the two update schedules.

Comparison. We compare one full-space NFT update with one complete Block-wise NFT pass consisting of one noisy clipped-gradient update per block. Both procedures start from $\theta^\star$ and use the same per-coordinate Gaussian noise scale σ.

Proposition K.1. *Suppose $L_2 > 0$ and $0 < \gamma \le 1/L_2$. Then*

$$\mathbb{E}[F(\theta_{\mathrm{block}}) - F(\theta^\star)] \le \mathbb{E}[F(\theta_{\mathrm{NFT}}) - F(\theta^\star)]\,.$$

The inequality is strict whenever $H_{21} \neq 0$.

Proof. We first consider the dynamics of NFT. Let

$$\xi_1 \sim \mathcal{N}(0, \sigma^2 I_{d_1}), \qquad \xi_2 \sim \mathcal{N}(0, \sigma^2 I_{d_2})$$

be independent. Since $\nabla F(\theta^\star) = 0$, one full-space NFT update gives

$$\theta_{\mathrm{NFT}} = \theta^\star + \begin{pmatrix} \xi_1 \\ \xi_2 \end{pmatrix}.$$

Using the quadratic representation of the excess loss,

$$\mathbb{E}[F(\theta_{\mathrm{NFT}}) - F(\theta^\star)] = \frac{1}{2}\mathbb{E}\big[\xi_1^\top H_{11}\xi_1 + 2\xi_1^\top H_{12}\xi_2 + \xi_2^\top H_{22}\xi_2\big] = \frac{\sigma^2}{2}\left(\mathrm{tr}(H_{11}) + \mathrm{tr}(H_{22})\right).$$

For Block-wise NFT, the first-block update gives

$$\theta_1 = \theta^\star + \begin{pmatrix} \xi_1 \\ 0 \end{pmatrix}.$$

The full gradient at this point is

$$\nabla F(\theta_1) = H(\theta_1 - \theta^\star) = \begin{pmatrix} H_{11}\xi_1 \\ H_{21}\xi_1 \end{pmatrix}.$$

Since the second block is updated next, let

$$q := \Pi_{C_1}(H_{21}\xi_1),$$

where $C_1 > 0$ is the clipping radius. Applying this clipped update and adding the second-block noise gives

$$\theta_{\mathrm{block}} = \theta^\star + \begin{pmatrix} \xi_1 \\ \xi_2 - \gamma q \end{pmatrix}.$$

Therefore,

$$\begin{aligned}\mathbb{E}[F(\theta_{\mathrm{block}}) - F(\theta^\star)] &= \frac{1}{2}\mathbb{E}\big[\xi_1^\top H_{11}\xi_1 + 2\xi_1^\top H_{12}(\xi_2 - \gamma q) + (\xi_2 - \gamma q)^\top H_{22}(\xi_2 - \gamma q)\big] \\ &= \frac{\sigma^2}{2}\left(\mathrm{tr}(H_{11}) + \mathrm{tr}(H_{22})\right) - \gamma\mathbb{E}\big[q^\top H_{21}\xi_1\big] + \frac{\gamma^2}{2}\mathbb{E}\big[q^\top H_{22} q\big]\,,\end{aligned}$$

where we used $H_{12} = H_{21}^\top$ and the independence and zero mean of ξ_2. Subtracting the NFT expression yields

$$\mathbb{E}[F(\theta_{\mathrm{block}}) - F(\theta^\star)] - \mathbb{E}[F(\theta_{\mathrm{NFT}}) - F(\theta^\star)] = -\gamma\mathbb{E}\big[q^\top H_{21}\xi_1\big] + \frac{\gamma^2}{2}\mathbb{E}\big[q^\top H_{22} q\big]\,.$$

By the definition of clipping,

$$q = \alpha H_{21}\xi_1, \qquad \alpha := \min\left\{1, \frac{C_1}{\|H_{21}\xi_1\|}\right\} \in (0, 1],$$

with $\alpha = 1$ when $H_{21}\xi_1 = 0$. Hence,

$$q^\top H_{21}\xi_1 = \alpha\|H_{21}\xi_1\|^2, \qquad q^\top H_{22}q \le L_2\alpha^2\|H_{21}\xi_1\|^2.$$

It follows that

$$\mathbb{E}[F(\theta_{\mathrm{block}}) - F(\theta^\star)] - \mathbb{E}[F(\theta_{\mathrm{NFT}}) - F(\theta^\star)] \le -\gamma\mathbb{E}\left[\alpha\left(1 - \frac{\gamma\alpha L_2}{2}\right)\|H_{21}\xi_1\|^2\right] \le -\frac{\gamma}{2}\mathbb{E}\big[\alpha\|H_{21}\xi_1\|^2\big] \le 0,$$

where the second inequality follows from $\gamma \le 1/L_2$ and $\alpha \le 1$. If $H_{21} \neq 0$, then $\|H_{21}\xi_1\| > 0$ with positive probability, so the inequality is strict. □

Discussion. Both procedures use the same per-coordinate noise scale and perturb every parameter direction over their complete update. The improvement therefore does not come from reducing the total injected noise: the first block's perturbation induces a gradient in the second block, which the subsequent clipped update partially corrects before adding the second-block noise. The condition $\gamma \le 1/L_2$ is the standard conservative stepsize condition for a gradient update on an L_2-smooth block objective, rather than a restriction specific to the block-wise construction. The result is deliberately limited to a quadratic objective, two blocks, one pass, and initialization at the retrained optimum; it serves only as an exact illustration of this sequential adaptation mechanism.