WHEN MICROBATCHING CHANGES THE TRAINING RULE: STATISTIC SCOPE AND GRADIENT BIAS

Anonymous authors
Paper under double-blind review

ABSTRACT

Gradient accumulation is widely used to fit a large training batch into memory and is normally assumed to preserve the update. This equivalence silently relies on splitting the batch without changing the quantities used by the loss. Many modern training rules violate this condition: reward and advantage normalization, contrastive losses, and metric-based objectives compute statistics shared across examples. If those statistics are recomputed inside each microbatch, the compute partition also changes the training rule—even when the global examples, effective batch size, and optimizer step are unchanged. We isolate this effect by varying only the set of examples used to compute the shared statistic, which we call its statistic scope. Two intuitive mechanisms explain the resulting mean-gradient shift: self-inclusion, where an example helps determine the statistic used to evaluate itself, and nonlinear plug-in bias from applying nonlinear functions to estimated statistics. We show that the shift grows predictably as statistic scope narrows and corresponds, to leading order, to optimizing a different effective objective. Across controlled non-decomposable objectives and frozen Qwen3-4B rollouts, the measured effect follows the predicted scaling. On Qwen3-4B, four separately normalized microbatches of 32 produce a 14.74% mean-gradient gap relative to one statistic over all 128 examples, and 20 paired SGD steps produce a path separation equal to 20.4% of the global-statistic displacement. Finally, global statistics can be computed over the full batch while backpropagation remains chunked, preserving the exact global-statistic gradient with chunk-sized activation memory.

1 INTRODUCTION

Gradient accumulation lets a large training batch fit in limited memory. For a sum of per-example losses, this changes memory use without changing the batch gradient: gradients from smaller chunks add to the gradient of their union. Many modern training rules also use quantities shared across examples. Batch Normalization makes each example depend on minibatch moments (Ioffe & Szegedy, 2015), while Batch Renormalization was introduced specifically to reduce this minibatch dependence (Ioffe, 2017). Ghost Batch Normalization deliberately changes the normalization group size (Hoffer et al., 2017). Contrastive learning couples examples through in-batch negatives and global interaction terms (Chen et al., 2022; Yuan et al., 2022), and recent LLM policy optimization uses group- or batch-level reward and advantage normalization (Shao et al., 2024; Xu & Ding, 2026). These rules introduce an additional implementation choice: which examples define the shared quantity. We call this set the *statistic scope*.

Consider the same 128 examples and the same optimizer step. One implementation computes a reward-normalization statistic once over all 128 examples. Another splits the batch into four microbatches of 32, recomputes the statistic inside each microbatch, and accumulates the resulting gradients. For an additive loss, the two procedures coincide. With a shared statistic, the partition changes which examples define the statistic seen by each row and can therefore change the *mean* update. Microbatching can alter both the memory schedule and the training rule.

We characterize this effect while holding the global examples fixed. To leading order, the mean-gradient gap grows with the difference between the inverse microbatch and global statistic scopes; equivalently, the two implementations follow different effective objectives. Two simple mechanisms explain the gap. First, a sample can help construct the statistic used to evaluate that same sample,

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.

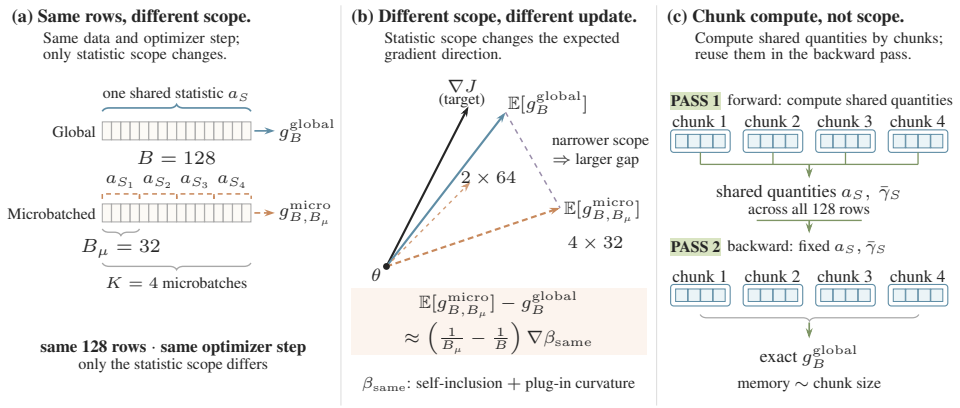

Figure 1: Statistic scope changes the training rule with the data and optimizer step fixed. (a) The same 128 examples use either one global statistic or four separately recomputed microbatch statistics. (b) Narrower scope creates a larger systematic mean-update shift through self-inclusion and nonlinear plug-in effects. (c) Global statistics are compatible with chunked backward computation.

creating a self-inclusion effect. Second, a noisy statistic can pass through a nonlinear transformation, creating nonlinear plug-in bias. Mean-centering isolates the first mechanism, while standard-deviation normalization contains both.

The effect is large enough to matter in a modern ML setting. On frozen Qwen3-4B rollouts, changing reward-statistic scope from one global statistic over 128 examples to four separately normalized microbatches of 32, with rollout rows and optimizer step fixed, changes the mean gradient by 14.74% of the global-statistic gradient norm. Over 20 paired SGD steps through the same ordered batches, the resulting parameter paths separate by 20.4% of the global-statistic displacement. Controlled Optical Digits and Fama–French objectives, where exact population references are available, reproduce the predicted leading-order scaling across statistic scopes.

The theory also gives a direct practical implementation. A global statistic can be computed once over the full batch while the expensive backward computation is performed in chunks, preserving the global-statistic gradient with chunk-sized activation memory.

Related work. Finite-batch computation can change the optimization problem in several ML settings. In contrastive learning, minibatch negatives induce gradient bias (Chen et al., 2022); SogCLR optimizes a global contrastive objective with memory-efficient stochastic updates (Yuan et al., 2022), and AUC-CL targets batch-size-robust unbiased gradients (Sharma et al., 2024). Nonlinear minibatch aggregates create Jensen gaps (Xu et al., 2025), while mini-batch quadratics systematically distort local geometry (Tatzel et al., 2025). In LLM policy optimization, SPO replaces groupwise normalization with batch-global advantage normalization (Xu & Ding, 2026), making normalization scope an explicit algorithmic choice. These results establish that batch-dependent aggregates matter. We study the gradient-accumulation intervention in which the global rows and optimizer step are fixed while only the examples defining the shared statistic change. This yields a general self-inclusion/plug-in decomposition, the $1/b - 1/B$ scaling law, and a global-statistic implementation compatible with chunked backpropagation. Appendix B gives a detailed comparison. Figure 1 summarizes the paper's argument.

Contributions.

- **Statistic scope changes expected updates.** We study the fixed-global-batch intervention and show that narrowing the statistic scope while holding the global batch fixed systematically changes the expected gradient. The leading correction separates self-inclusion from nonlinear plug-in bias, yields an effective-objective interpretation, and predicts the microbatch scaling.

- **Evidence and training consequence.** Controlled non-decomposable objectives follow the predicted scaling, while frozen Qwen3-4B rollouts show a 14.74% mean-gradient gap and a 20-step path separation equal to 20.4% of the global-statistic displacement. Paired estimation and an unrolled response verify that the shift is measurable and persists over multiple updates.
- **Global statistics with chunk-sized activations.** We compute the intended statistic globally while chunking the expensive backward computation, preserving the global-statistic batch gradient with chunk-sized activation memory.

2 Shared Statistics in Microbatched Training

The Introduction identifies a practical implementation choice: the optimizer step may use the same global examples while the statistic seen by each row changes with the microbatch partition. Reward normalization makes this choice concrete. With one global batch, every row uses the same reward mean and scale; with per-microbatch recomputation, the same data and optimizer step are paired with separate normalizers for each chunk. Similar coupling appears in non-decomposable metrics (Narasimhan et al., 2015), data-dependent training constraints (Cotter et al., 2019), and finite-sum coupled compositional objectives (Wang & Yang, 2022). We use one abstraction for these cases and call the set of examples defining the shared quantity its *statistic scope*.

2.1 A shared-statistic objective and two scopes

The common structure is an outer row loss evaluated using an aggregate estimated from many rows. For model parameters θ, let $m_i(\theta) \in \mathbb{R}^d$ be row i's contribution to a low-dimensional shared statistic. The population statistic and target objective are

$$a(\theta) = \frac{1}{n}\sum_{i=1}^{n} m_i(\theta), \qquad J(\theta) = \frac{1}{n}\sum_{i=1}^{n} \ell_i(\theta, a(\theta)) \,. \tag{1}$$

The defining feature is that many row losses reuse the same aggregate $a(\theta)$.

Draw one global batch S of size B and partition it into K equal microbatches $S_1, \ldots, S_K$ of size $b = B/K$. Let a_S be the statistic over all rows in S and a_{S_k} the statistic recomputed inside chunk k. We compare

$$g_B^{\text{global}} = \nabla_\theta \left[\frac{1}{B} \sum_{i \in S} \ell_i(\theta, a_S) \right], \tag{2}$$

$$g_{B,b}^{\text{micro}} = \frac{1}{K} \sum_{k=1}^{K} \nabla_\theta \left[\frac{1}{b} \sum_{i \in S_k} \ell_i(\theta, a_{S_k}) \right]. \tag{3}$$

Both rules use the same global examples and produce one accumulated update. For an additive loss they coincide; with a shared statistic, recomputing a after partitioning can change the gradient.

2.2 Examples

Reward normalization. In the Qwen experiments, row i is weighted by $-s_i(\theta)(r_i - \mu)/\sqrt{\nu - \mu^2 + \epsilon}$ using reward moments (μ, ν) from the chosen scope. Group-relative policy optimization popularized within-group reward normalization for reasoning models (Shao et al., 2024), while recent work has replaced groupwise normalization with batch-global alternatives (Xu & Ding, 2026). Mean-centering creates self-inclusion because each reward helps set its own baseline; scale normalization additionally applies a nonlinear function to estimated moments.

Smooth F1. The loss depends on batch-level true positives, predicted rate, and prevalence, a standard source of non-decomposability in direct metric optimization (Eban et al., 2017); Smooth Macro-F1 repeats the same construction classwise. **Mean–volatility.** The portfolio objective depends on the first two moments of model-dependent returns (Zhang et al., 2020; Kubo & Nakagawa, 2025). These objectives share a common structure: each reuses a statistic estimated from the rows being optimized.

We next ask why narrowing that statistic scope can systematically change the expected update.

3 How Statistic Scope Changes the Mean Update

Section 2 defines two implementations that use the same global rows with different statistic scopes. The next question is what stochastic optimization object this change induces. Fixing the global rows and optimizer step attributes the resulting mean shift directly to how the shared statistic is estimated and reused. Reward normalization exposes the key intuition. A sample can help define the statistic used to score itself, and nonlinear functions of an estimated statistic can turn ordinary finite-batch fluctuations into a systematic shift. We first make these two effects concrete, then state the general result and return to the global-versus-microbatch comparison from the Introduction.

3.1 Warm-up: reward normalization

We begin with mean-centering, $\ell_i(\theta, a) = -s_i(\theta)(r_i - a)$ and $m_i = r_i$, where the batch mean a is computed from the same examples used in the loss. Because the loss is affine in a, the nonlinear plug-in term is exactly zero, isolating the self-inclusion effect:

$$\mathbb{E}[g_{\mathrm{same},B}] = (1 - \kappa_B)\,\nabla J, \qquad \kappa_B = \frac{n - B}{B(n-1)} \approx \frac{1}{B}. \tag{4}$$

Each sample helps construct the baseline against which that same sample is scored, slightly shrinking the expected policy gradient. This is a pure *self-inclusion* effect; leave-one-out baselines such as RLOO remove the same-example feedback (Ahmadian et al., 2024).

Standard-deviation normalization adds a second effect. The estimated scale now enters through the nonlinear map $1/\sigma$, so the nonlinear transformation converts finite-batch fluctuations in the reward moments into an additional systematic contribution after averaging. Analyses of group-based policy optimization identify normalization and groupwise weighting as sources of objective distortion (Liu et al., 2025; Fontana et al., 2026); SPO instead normalizes advantages globally across the batch (Xu & Ding, 2026). The general case below formalizes these two mechanisms: self-inclusion and nonlinear plug-in bias.

3.2 Two sources of finite-batch bias

Let S be a uniformly sampled batch of size B, with $a_S = B^{-1}\sum_{i\in S} m_i$. The usual rule reuses S both to estimate the statistic and to evaluate the row losses. To separate self-inclusion from nonlinear plug-in bias, we use an analytical sample-splitting control in which an independent batch R supplies the statistic while S supplies the outer examples:

$$J_{\mathrm{same},S} = \frac{1}{B}\sum_{i\in S} \ell_i(\theta, a_S), \qquad\qquad J_{\mathrm{split},S,R} = \frac{1}{B}\sum_{i\in S} \ell_i(\theta, a_R). \tag{5}$$

Both rules have the same finite-batch fluctuation in the statistic. In same-batch reuse, an example also directly influences the statistic used in its own loss; sample splitting removes this direct self-inclusion. We write $g_e = \nabla_\theta J_e$ for $e \in \{\mathrm{same}, \mathrm{split}\}$. For sampling without replacement, the leading finite-batch scale is $\rho_B = 1/B - 1/n$; Appendix C gives the covariance identity behind the expansion.

The two leading corrections mirror the reward example. The self-inclusion correction is $\beta_{\mathrm{coup}}(\theta) = \mathrm{tr}[\mathrm{Cov}(\nabla_a \ell_i, m_i)]$. The nonlinear plug-in correction is $\beta_{\mathrm{plug}}(\theta) = \frac{1}{2}\,\mathrm{tr}(\bar{H}_a\,\mathrm{Cov}(m_i))$, where $\bar{H}_a = n^{-1}\sum_i \nabla^2_{aa}\ell_i(\theta, a)$. Let $\beta_{\mathrm{same}} = \beta_{\mathrm{coup}} + \beta_{\mathrm{plug}}$ and $\beta_{\mathrm{split}} = \beta_{\mathrm{plug}}$.

Theorem 1 (Finite-batch bias decomposition)**.** *Assume the row loss is smooth in the shared statistic, with bounded derivatives through third order, bounded centered row moments, and fixed statistic dimension. Then*

$$\mathbb{E}[J_{\mathrm{split}}] - J = \rho_B\,\beta_{\mathrm{plug}} + O(\rho_B^{3/2}), \tag{6}$$

$$\mathbb{E}[J_{\mathrm{same}}] - J = \rho_B\big(\beta_{\mathrm{coup}} + \beta_{\mathrm{plug}}\big) + O(\rho_B^{3/2}). \tag{7}$$

The hidden constant is determined by the smoothness and moment bounds and is independent of B.

The theorem says exactly what the warm-up suggests. With an independent statistic batch, the linear self-inclusion term vanishes while nonlinear use of the noisy statistic produces the plug-in

correction. Reusing the same rows adds the self-inclusion term. If the loss is affine in the shared statistic, $\beta_{\text{plug}} = 0$: sample splitting matches the target objective and gradient, while same-batch reuse retains the self-inclusion correction. Appendix C gives the Taylor-expansion proof.

3.3 From gradient bias to an effective objective

Training follows gradients, so we translate the objective expansion into the mean-gradient shift that governs optimization. Define the mean-gradient shift at batch size B as $\Delta g_{e,B} = \mathbb{E}[g_e] - \nabla J$.

Theorem 2 (Expected-gradient bias)**.** *Under the conditions of Theorem 1 and the derivative regularity conditions in Appendix C,*

$$\Delta g_{e,B} = \rho_B \nabla \beta_e + O(\rho_B^{3/2}), \tag{8}$$

$$J_{e,B}^{\text{eff}}(\theta) = J(\theta) + \rho_B \beta_e(\theta). \tag{9}$$

Thus, to leading order, a finite-batch implementation follows the gradient of a batch-dependent *effective objective*. Different statistic-estimation rules can therefore have different mean updates even when they target the same population objective. Around a nondegenerate stationary point, the same expansion implies an $O(\rho_B)$ shift in the nearby zero of the expected update; Appendix C gives the formal statement.

3.4 Fixed global batch: microbatch scaling

We now return to the practical intervention in Figure 1: with the global rows and optimizer step fixed, we vary the statistic scope. Normalization work already treats the number of examples defining a statistic as an algorithmically meaningful choice: Ghost Batch Normalization deliberately uses smaller groups (Hoffer et al., 2017), while Batch Renormalization targets the instability caused by minibatch-dependent moments (Ioffe, 2017). Here we derive the corresponding mean-update scaling for the general shared-statistic objective. Partition a size-B global batch into K equal microbatches of size $b = B/K$.

Corollary 1 (Microbatch scaling)**.** *Let g_B^{global} compute the shared statistic once over the full global batch, and let $g_{B,b}^{\text{micro}}$ recompute it inside each microbatch before averaging the microbatch gradients. Then*

$$\mathbb{E}[g_{B,b}^{\text{micro}} - g_B^{\text{global}}] = \left(\frac{1}{b} - \frac{1}{B}\right) \nabla \beta_{\text{same}} + O(\rho_b^{3/2}). \tag{10}$$

For an additive loss, the two gradients are identical for every batch and partition.

The leading gap grows with $1/b - 1/B$: narrowing the statistic scope changes the *mean* update with the sampled rows fixed. At $B = 256$, $b = 128, 64, 32$ gives relative factors 1:3:7, which Section 6 tests directly. With a global statistic, backward-pass chunk size can vary independently while the mathematical batch gradient remains unchanged.

We next need to estimate this mean shift under minibatch noise and determine whether it persists over multiple updates.

4 Measuring the Shift and Its Training Effect

Section 3 predicts a systematic shift in the *mean* gradient. Empirically, that signal can be much smaller than ordinary minibatch variation, and its training consequence depends on how later optimization transports it. This section therefore separates two supporting questions: how to estimate the local mean shift precisely, and how to propagate it along a training trajectory. The first uses paired sampling on the same rows; the second uses an unrolled linear response.

4.1 Paired estimation of the mean shift

Our controlled settings provide the full-population statistic and its derivative. For each sampled batch, we therefore pair the implemented gradient g_e with a reference gradient g_{cv} built from the *same rows*

and the known population quantity; Appendix D gives the construction. Since $\mathbb{E}[g_{\mathrm{cv}}] = \nabla J$,

$$\mathbb{E}[g_e - g_{\mathrm{cv}}] = \mathbb{E}[g_e] - \nabla J = \Delta g_{e,B}. \tag{11}$$

The pairing leaves the estimand unchanged while canceling much of the row-level sampling noise shared by the two gradients. This is a standard common-random-numbers control variate; Section 6 reports the resulting Monte Carlo precision.

4.2 From a local shift to path separation

Let θ_t be the reference full-population trajectory, $\Delta g_e(\theta_t)$ the implementation's mean-gradient shift at step t, and $H_t = \nabla^2 J(\theta_t)$. Linearizing the expected implementation trajectory around the reference path gives

$$\widehat{\delta}_T^e = -\sum_{t=0}^{T-1} \left[\prod_{s=t+1}^{T-1} \left(I - \eta_s H_s\right) \right] \eta_t \, \Delta g_e(\theta_t). \tag{12}$$

This is the standard unrolled linear response obtained by differentiating through gradient-descent steps (Koh & Liang, 2017; Bae et al., 2024). Influence-function work has also extended training-data attribution to non-decomposable losses (Deng et al., 2025). Each local shift is transported through the later updates, so curvature can contract, rotate, or amplify its contribution to the final parameter difference. A naive sum of the local shifts ignores this transport and serves as our baseline. Appendix D bounds the linearization error.

Together, paired estimation resolves the local mean shift and the unrolled response predicts how it propagates. We next show how global statistics combine with memory-efficient gradient accumulation while preserving the intended update.

5 Gradient Accumulation with Global Statistics

The previous sections identify the training change caused by per-microbatch statistic recomputation. Memory-efficient training can preserve the global-statistic update by separating statistic aggregation from expensive differentiation. Several ML systems use this broader pattern, including gradient caching for contrastive learning (Gao et al., 2021) and exact memory-efficient backpropagation for long-sequence LLM training (Luo et al., 2025). Global-normalizer optimization is likewise an explicit design problem in large-scale contrastive learning (Wei et al., 2026). In our setting the shared statistic is low-dimensional, yielding a simple decomposition: compute it over the full training batch and process the expensive forward/backward computation in smaller chunks.

5.1 Chunking the backward computation

For one global batch S of size B, let a_S be its shared statistic and define the average sensitivity $v_S = B^{-1} \sum_{i \in S} \nabla_a \ell_i(\theta, a_S)$. Writing $J_{m_i}(\theta)$ for the Jacobian of the row statistic, the chain rule gives

$$\nabla_\theta L_S = \frac{1}{B} \sum_{i \in S} \left[\partial_\theta \ell_i(\theta, a_S) + J_{m_i}(\theta)^\top v_S \right], \qquad L_S = \frac{1}{B} \sum_{i \in S} \ell_i(\theta, a_S). \tag{13}$$

Once a_S and v_S are known, every term in the sum is row-local. The rows can therefore be backpropagated in arbitrary chunks while preserving the same g_B^{global} as the unchunked batch. The compute chunk size controls activation memory independently of statistic scope.

5.2 Implementation and cost

The computation separates naturally into three stages: accumulate a_S over forward chunks; keep a_S fixed and accumulate the shared sensitivity v_S; then revisit the chunks and backpropagate the row-local terms in Equation 13. For Smooth F1, the first two stages can share a small set of aggregates, while for our frozen-reward Qwen setting the reward moments are parameter-independent. The decomposition is especially efficient when the shared statistic is low-dimensional or otherwise cheap to aggregate; parameter-dependent statistics additionally require the shared-sensitivity pass. Activation storage remains chunk-sized.

The principle is to preserve the intended full-batch interaction while chunking the expensive computation, as in gradient caching (Gao et al., 2021); StreamBP provides a different exact decomposition for reducing activation memory in LLM backpropagation (Luo et al., 2025). On a 45,000-row Smooth-F1 stress test, a specialized two-pass implementation matches the unchunked gradient to machine precision while reducing peak allocated memory by 9.79× at 1.87× runtime. Appendix E gives the memory bound, extensions to transformed or implicit statistics, and the full benchmark.

The experiments now test the complete claim: how much the mean update changes when statistic scope narrows, how that change propagates along training, and how the global-statistic rule performs with chunk-sized activation memory.

6 EXPERIMENTS

The theory makes two empirical predictions: narrowing statistic scope should produce a systematic mean-gradient gap with the leading $1/b - 1/B$ scaling, and the local gap should propagate through later optimization in a predictable way. We test the first prediction in a language-model setting, then use controlled objectives with exact population references to measure the scaling law and the finite-horizon response. In every comparison, we fix the sampled global rows and optimizer settings and vary the scope used to compute the shared statistic.

Setup. We use Qwen3 (Yang et al., 2025) with LoRA adaptation (Hu et al., 2022) on frozen GSM8K rollout populations (Cobbe et al., 2021). Qwen3-4B tests the main statistic-scope effect and 20-step path separation, while Qwen3-0.6B provides an estimator-precision stress test. Optical Digits (Alpaydin & Kaynak, 1998) and FF25 (French, 2026) provide controlled non-decomposable objectives with exact reference gradients, and Binary Smooth F1 provides an additional leading-order calibration. Paired comparisons reuse the same sampled rows. Appendix F gives checkpoints, draw counts, hyperparameters, and provenance.

6.1 FIXED-GLOBAL-BATCH SCALING

Corollary 1 predicts that, with the global rows fixed, the leading mean-gradient gap grows with $1/b - 1/B$. We test the effect first on Qwen3-4B and then ask whether the same scaling appears in controlled objectives where the population target is known exactly.

Qwen3-4B. Each of 256 paired draws reuses the same 128 frozen rollout rows and compares three reward-statistic scopes: one global statistic over 128 rows, 2×64, or 4×32. The mean-gradient gap grows from 4.14% of the global mean-gradient norm at 2×64 to 14.74% at 4×32; the latter is resolved at 6.20 MC-SE. Tripling the leading scope factor increases the measured gap by $3.56\times$, and the predicted–measured cosine rises from 0.931 to 0.988. At the narrowest scope, the first-order theory captures the direction and ordering, while the measured/predicted norm ratio of 1.90 quantifies a material higher-order correction.

Controlled objectives. With Optical Digits, $B = 256$ and $b \in \{128, 64, 32\}$ give the predicted relative factors 1:3:7. Across six model–checkpoint pairs, the median measured factors are 1:3.16:8.28 and the median predicted–measured cosine is 0.982. FF25 repeats the test with $B = 64$ and $b \in \{32, 16, 8\}$ across three checkpoints and two non-overlapping periods; the median factors are 1:3.00:7.31 with median cosine 0.958. The same leading scaling appears across reward normalization, Smooth Macro-F1, and a mean–volatility objective, supporting a mechanism shared across these objectives.

6.2 FINITE-HORIZON TRAINING EFFECTS

We next ask whether the local mean-gradient shift survives subsequent optimization. Qwen3-4B provides the direct paired path comparison; the controlled objectives compare the unrolled response in Equation 12 with a simple cumulative-bias baseline.

Qwen3-4B. Starting from the same LoRA state, we run 20 paired SGD steps ($\eta = 0.05$) through the same ordered $B = 128$ batches, comparing global-128 with 4×32 reward normalization. The final branch gap is 1.75×10^{-3}, equal to 20.4% of the global-statistic displacement. The two total displacement vectors remain nearly collinear (cosine 0.9966), yet the microbatch-statistic branch

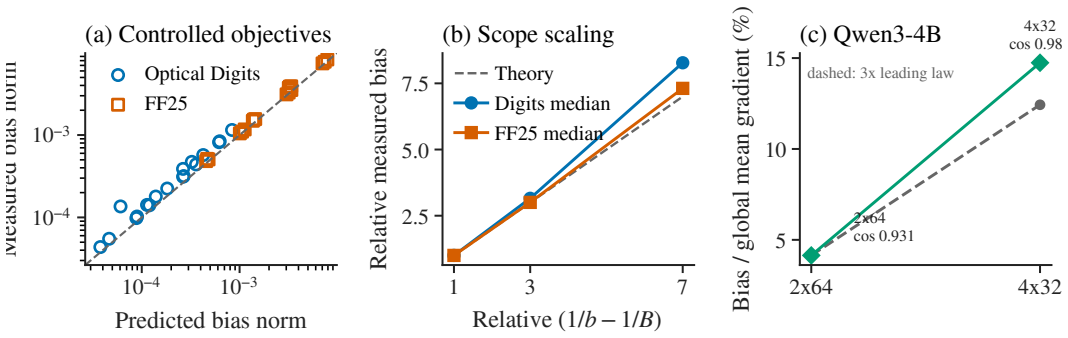

Figure 2: Narrowing statistic scope changes the mean gradient with the predicted leading scaling. Qwen3-4B shows a 14.74% gap at 4×32; Optical Digits and FF25 provide controlled tests of the 1:3:7 law.

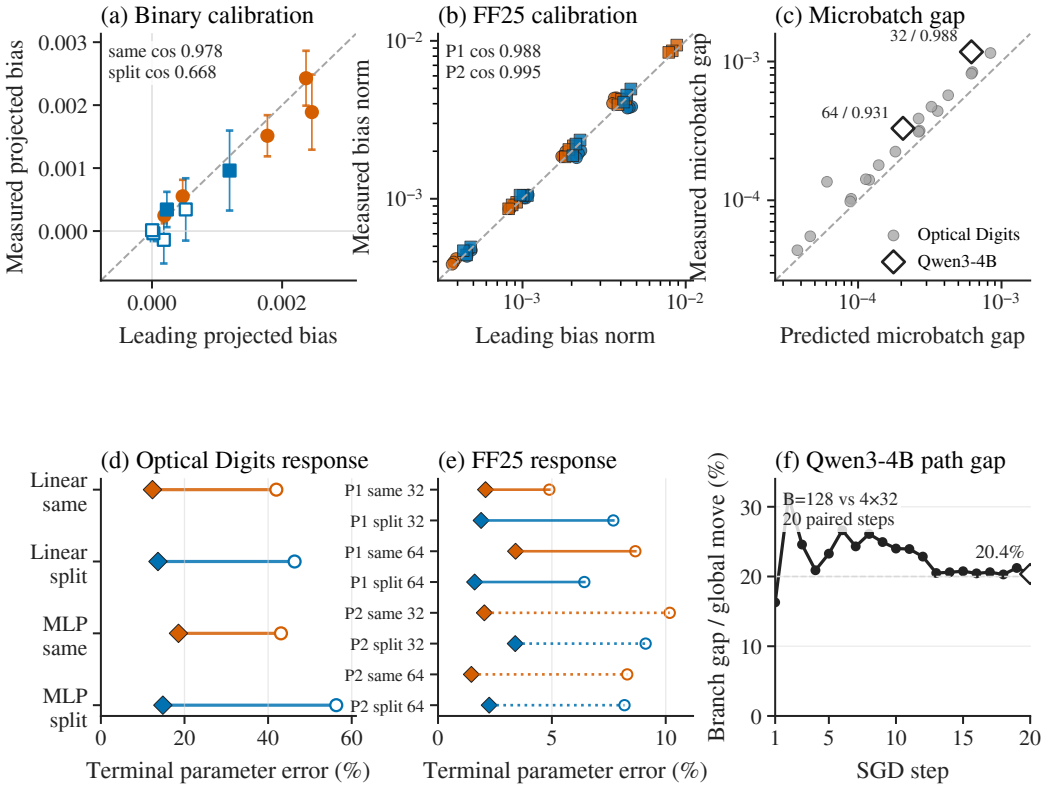

Figure 3: Local calibration and finite-horizon propagation. Panels (a,b) check the leading expansion; (c) shows fixed-global-batch scope scaling; (d,e) compare the unrolled response with naive cumulative bias; and (f) shows the paired Qwen3-4B path separation.

travels about 19% less far (6.97 versus 8.60×10^{-3}). Thus a substantial path separation can appear before a coarse directional comparison makes the two trajectories look different.

Controlled response tests. On Optical Digits, the unrolled response predicts terminal displacement with 12.34–18.56% relative error and cosine 0.9828–0.9924, compared with 41.97–56.30% error for the naive cumulative-bias baseline. On FF25, response error is 1.46–3.40% across two periods, versus 4.89–10.18% for the naive baseline. These results support the interpretation that later curvature matters for how local statistic-scope shifts accumulate.

6.3 SUPPORTING CALIBRATION AND PRECISION CHECKS

Binary Smooth-F1 provides an additional calibration of the leading expansion: across five non-control regimes, the same-batch shift has the predicted sign in all five cells and a median predicted–measured cosine of 0.978. This check evaluates transfer of the first-order expansion to an additional non-decomposable objective. Paired estimation addresses measurement precision by separating a small systematic mean shift from ordinary minibatch noise while preserving the estimand. On Optical Digits, pairing raises the number of settings resolved above 2 MC-SE from 2/36 to 32/36 and reduces trace variance by a median 19.28×; across the two FF25 periods, paired SNR exceeds direct SNR in all 48 settings. The Qwen3-0.6B stress test shows the same variance-reduction effect in a language-model gradient, with 10.45–16.38× trace-VRF. In all cases the paired and direct estimators agree within Monte Carlo error. Together, these checks establish calibration and measurement precision for the main fixed-row comparisons. Appendix Figure 5 and Appendix F give the cell-level results.

Global-statistic accumulation. Finally, we benchmark the implementation from Section 5 on a 45,000-row Smooth-F1 stress test with compute chunks of 512 rows. The specialized two-pass implementation reproduces the unchunked global-statistic gradient to machine precision while reducing peak allocated memory from 18.15 GiB to 1.85 GiB (9.79×), with median runtime increasing from 2.21 s to 4.14 s (1.87×). The additional runtime reflects a second data/forward pass that reconstructs chunkwise forward quantities while the shared statistic and sensitivity remain global. Thus statistic scope and activation chunk size can be controlled independently, exposing a direct memory–compute tradeoff without changing the intended global-statistic update. Appendix E reports the full comparison across implementations and population sizes.

7 DISCUSSION

The experiments connect the fixed-row comparison in Section 2 to training dynamics: statistic scope changes the mean update in controlled objectives and Qwen3, and the resulting shift can accumulate over multiple steps. This unifies two familiar ML observations. Normalization methods treat the examples defining batch statistics as an algorithmic choice (Hoffer et al., 2017; Ioffe, 2017), while non-decomposable and contrastive objectives show that finite-batch approximations can change the stochastic optimization target (Chen et al., 2022; Sharma et al., 2024). Recent LLM policy optimization makes the same issue concrete for reward and advantage normalization (Xu & Ding, 2026).

The practical implication is that effective batch size alone does not fully specify training with shared statistics. One quantity controls how many examples contribute to the optimizer step; another controls how many examples define the shared statistic. Training descriptions should therefore report statistic scope—global batch, microbatch, device, leave-one-out construction, or tracked state—alongside batch size, accumulation factor, and optimizer settings. In distributed training, per-device, per-microbatch, and globally reduced statistics can define different updates at the same nominal global batch size. The mechanisms also point to standard design choices: leave-one-out baselines remove direct same-example feedback (Ahmadian et al., 2024); tracked or corrected statistics stabilize normalization (Ioffe, 2017); and global normalizers preserve the intended cross-example interaction (Wei et al., 2026). Section 5 provides an exact global-statistic implementation with chunked expensive computation.

Across the tested scopes, the leading expansion captures the direction and ordering of the effect; at narrow reward scopes, larger higher-order corrections appear, with a measured/predicted norm ratio of 1.90 for Qwen3-4B at 4×32. Fixed finite populations and frozen rollouts let us isolate this implementation effect directly. A natural next setting combines statistic-scope changes with on-policy data refresh, adaptive optimizers, and repeated seeds to study their interaction with evolving trajectories; recent LLM policy-gradient work shows that normalization, baselines, and gradient-estimation choices materially alter reasoning training (Xu & Ding, 2026; Chu et al., 2026). The central message is simple: microbatching changes the training rule when it changes the scope of a shared statistic. The same distinction applies whenever examples interact through a batch-computed mean, scale, normalizer, metric surrogate, or other shared aggregate. Treating statistic scope as part of the mathematical specification of training makes these objectives explicit across single-device, accumulated, and distributed implementations.

AI USE STATEMENT

We used ChatGPT to assist with identifying potentially relevant related work and with language editing. We used Codex for limited assistance in implementing and running author-designed analyses. The authors specified the research questions, methods, estimands, and analytical choices, and manually reviewed and verified all AI-assisted code, calculations, citations, analyses, and text. All scientific interpretations and final decisions were made by the authors, who take full responsibility for the content of the paper.

REPRODUCIBILITY STATEMENT

The main text and appendix state the assumptions and proofs for the finite-population, dynamic-response, and chunked-gradient results. The anonymous supplementary package includes the frozen binary Smooth-F1 confirmatory outputs and selection protocol; rerunnable Optical Digits and FF25 experiments, including the fixed-global-batch FF25 microbatch grid with frozen seeds; the Qwen3-0.6B audit configuration and outputs; the frozen Qwen3-4B rollout population, initial trainable state, static microbatch audit, and 20-step trajectory code, configurations, and outputs; figure-generation utilities; and the executable replay benchmark with its configuration, 15 method-by-population summaries, and canonical timing and CUDA-memory table. Public model weights and datasets are referenced by pinned revisions or checksums at their original sources.

REFERENCES

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12248–12267. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.662.

Ethem Alpaydin and C. Kaynak. Optical recognition of handwritten digits. UCI Machine Learning Repository, 1998.

Juhan Bae, Wu Lin, Jonathan Lorraine, and Roger Grosse. Training data attribution via approximate unrolling. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Mathieu Blondel, Quentin Berthet, Marco Cuturi, Roy Frostig, Stephan Hoyer, Felipe Llinares-López, Fabian Pedregosa, and Jean-Philippe Vert. Efficient and modular implicit differentiation. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Changyou Chen, Jianyi Zhang, Yi Xu, Liqun Chen, Jiali Duan, Yiran Chen, Son Tran, Belinda Zeng, and Trishul Chilimbi. Why do we need large batchsizes in contrastive learning? a gradient-bias perspective. In *Advances in Neural Information Processing Systems*, volume 35, pp. 33860–33875, 2022.

Xiangxiang Chu, Hailang Huang, Xiao Zhang, Fei Wei, and Yong Wang. GPG: A simple and strong reinforcement learning baseline for model reasoning. In *International Conference on Learning Representations*, 2026.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Andrew Cotter, Maya Gupta, Heinrich Jiang, Nathan Srebro, Karthik Sridharan, Serena Wang, Blake Woodworth, and Seungil You. Training well-generalizing classifiers for fairness metrics and other data-dependent constraints. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1397–1405, 2019.

Junwei Deng, Weijing Tang, and Jiaqi W. Ma. A versatile influence function for data attribution with non-decomposable loss. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 13256–13276, 2025.

Elad Eban, Mariano Schain, Alan Mackey, Ariel Gordon, Rif A. Saurous, and Gal Elidan. Scalable learning of non-decomposable objectives. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 832–840, 2017.

Aleksandar Fontana, Marco Simoni, Giulio Rossolini, Paolo Mori, and Andrea Saracino. On the hidden objective biases of group-based reinforcement learning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 109–121, 2026. doi: 10.18653/v1/2026.acl-short.11.

Kenneth R. French. 25 portfolios formed on size and book-to-market (5 x 5). Kenneth R. French Data Library, 2026. URL `https://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html`. Accessed September 2026.

Luyu Gao, Yunyi Zhang, Jiawei Han, and Jamie Callan. Scaling deep contrastive learning batch size under memory limited setup. In *Proceedings of the 6th Workshop on Representation Learning for NLP*, pp. 316–321. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.repl4nlp-1.31.

Elad Hoffer, Itay Hubara, and Daniel Soudry. Train longer, generalize better: Closing the generalization gap in large batch training of neural networks. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Sergey Ioffe. Batch renormalization: Towards reducing minibatch dependence in batch-normalized models. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 448–456, 2015.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1885–1894, 2017.

Kenji Kubo and Kei Nakagawa. Portfolio optimization using deep learning with risk aversion utility function. *Finance Research Letters*, 74:106761, 2025. doi: 10.1016/j.frl.2025.106761.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. In *Conference on Language Modeling*, 2025.

Qijun Luo, Mengqi Li, Lei Zhao, and Xiao Li. Streambp: Memory-efficient exact backpropagation for long sequence training of llms. In *Advances in Neural Information Processing Systems*, volume 38, pp. 182495–182521, 2025. doi: 10.52202/085713-5488.

Harikrishna Narasimhan, Purushottam Kar, and Prateek Jain. Optimizing non-decomposable performance measures: A tale of two classes. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 199–208, 2015.

Nikhil Nayak, Julia White, Urchade Zaratiana, Kelton Zhang, Henrijs Princis, Dhruv Atreja, Henry Fawcett, Matthew Thomas, George Hurn-Maloney, and Ash Lewis. Correcting stochastic update bias in preconditioned language model optimizers, 2026.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, pp. 8024–8035, 2019.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Rohan Sharma, Kaiyi Ji, Zhiqiang Xu, and Changyou Chen. AUC-CL: A batchsize-robust framework for self-supervised contrastive representation learning. In *International Conference on Learning Representations*, 2024.

Lukas Nicola Tatzel, Bálint Mucsányi, Osane Hackel, and Philipp Hennig. Debiasing mini-batch quadratics for applications in deep learning. In *International Conference on Learning Representations*, 2025.

Bokun Wang and Tianbao Yang. Finite-sum coupled compositional stochastic optimization: Theory and applications. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 23292–23317, 2022.

Xiyuan Wei, Chih-Jen Lin, and Tianbao Yang. NeuCLIP: Efficient large-scale CLIP training with neural normalizer optimization. In *International Conference on Learning Representations*, 2026.

Chen Xu, Yuxin Li, Wenjie Wang, Liang Pang, Jun Xu, and Tat-Seng Chua. Bridging jensen gap for max-min group fairness optimization in recommendation. In *International Conference on Learning Representations*, 2025.

Zhongwen Xu and Zihan Ding. Single-stream policy optimization. In *International Conference on Learning Representations*, 2026.

An Yang, Anfeng Li, Baosong Yang, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Zhuoning Yuan, Yuexin Wu, Zi-Hao Qiu, Xianzhi Du, Lijun Zhang, Denny Zhou, and Tianbao Yang. Provable stochastic optimization for global contrastive learning: Small batch does not harm performance. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 25760–25782, 2022.

Zihao Zhang, Stefan Zohren, and Stephen Roberts. Deep learning for portfolio optimization. *The Journal of Financial Data Science*, 2(4):8–20, 2020. doi: 10.3905/jfds.2020.1.042.

A NOTATION

The table lists symbols used in more than one section; symbols local to a single proof are defined where they appear.

Symbol	Meaning
Population and sampling	
n, i	population size; row index
$\theta \in \mathbb{R}^p$	model parameters
S, B	sampled batch and its size (uniform, without replacement)
R	independently drawn subset used by split sampling
S_k, K, b	microbatches of S, their number, and their size $b = B/K$
ρ_B	finite-population scale $1/B - 1/n$ (Equation 14)
Σ_{xy}	finite-population covariance of row arrays x_i, y_i (denominator $n-1$)
Objective and shared statistic	
$m_i(\theta) \in \mathbb{R}^d$	row moment; d is the (fixed) statistic dimension
$a(\theta)$, a_S, Δ_S	population statistic, sample statistic, and deviation $a_S - a$
$\ell_i(\theta, a)$, $J(\theta)$	row loss and target objective (Equation 1)
e; J_e, g_e	implementation $e \in \{\mathrm{same}, \mathrm{split}\}$; its minibatch objective and gradient
γ_i, $\bar{\gamma}$	statistic sensitivity $\nabla_a \ell_i$ and its population average
H_i, $\bar{H}_a$	statistic Hessian $\nabla^2_{aa} \ell_i$ and its population average
Bias	
$\Delta g_{e,B}$	mean-gradient shift $\mathbb{E}[g_e] - \nabla J$
β_e	leading effective-objective correction (Equation 8); β_{coup} is same-batch coupling and β_{plug} nonlinear plug-in bias
$J^{\mathrm{eff}}_{e,B}$	effective objective $J + \rho_B \beta_e$ (Equation 9)
$\theta^\star$, $H^\star$; $\theta^\star_{e,B}$	stationary point of J and its Hessian; shifted zero of $\mathbb{E}[g_e]$ (Eq. 15)
g^{global}_B, $g^{\mathrm{micro}}_{B,b}$	gradient with one statistic over S; average of per-microbatch gradients
κ_B	shrinkage factor $(n-B)/[B(n-1)]$ for mean-centered rewards (Equation 4)
Estimation and dynamics	
g_{cv}	matched control variate with known mean ∇J (Appendix D)
VRF, MC-SE, SNR	trace variance-reduction factor; vector Monte Carlo standard error; bias norm divided by MC-SE
η_t, H_t	step size; Hessian $H_t = \nabla^2 J(\theta_t)$ along the reference trajectory
$\widehat{\delta}^e_T$, $\Delta\theta^e_T$	predicted (unrolled) and actual displacement at step T
Gradient accumulation with global statistics	
v_S	average sensitivity $B^{-1} \sum_{i \in S} \nabla_a \ell_i(\theta, a_S)$ of one global batch
C; $q = h(a) \in \mathbb{R}^s$	chunk size; transformed shared statistic (Appendix E)

B RELATED WORK ON BATCH-DEPENDENT OBJECTIVES AND STATISTICS

Several ML settings already show that minibatching can alter more than variance. Contrastive learning provides a direct example: finite batches change the negative set and can bias the stochastic gradient of the intended global objective (Chen et al., 2022). Global-contrastive optimization addresses the same large-interaction target with memory-efficient stochastic methods (Yuan et al., 2022), while AUC-CL explicitly constructs a batch-size-robust contrastive objective with unbiased stochastic gradients (Sharma et al., 2024). More generally, coupled compositional optimization formalizes objectives whose outer terms depend on sample-dependent inner aggregates (Wang & Yang, 2022).

A different line of work studies nonlinear consequences of minibatch substitution. Max-min fairness objectives can exhibit a minibatch Jensen gap (Xu et al., 2025). Mini-batch quadratic approximations can distort local geometry through same-data selection effects (Tatzel et al., 2025). Stochastic preconditioners create a related coupling between a noisy gradient and a nonlinear scaling rule (Nayak et al., 2026). NeuCLIP focuses on accurately maintaining a global contrastive normalizer under limited batch sizes (Wei et al., 2026). Our intervention isolates statistic scope by fixing the sampled global rows and optimizer step, providing a direct measurement of how the shared-statistic construction changes the mean update.

Normalization itself also provides close empirical precedents. Ghost Batch Normalization intentionally changes the group used to compute moments (Hoffer et al., 2017), whereas Batch Renormalization aims to reduce dependence on small or non-i.i.d. minibatches (Ioffe, 2017). In LLM policy optimization, SPO reports gains from replacing groupwise advantage normalization with normalization across the whole batch (Xu & Ding, 2026). Together, these works establish statistic scope as a practically meaningful implementation choice; our fixed-global-batch comparison measures its mean-gradient effect directly.

Table 1: Comparison with related batch-dependent optimization settings. Our intervention fixes the sampled global rows and optimizer step and varies the scope of a shared statistic.

Work	Controlled quantities	Varied quantity	Main object	Mechanism / consequence
Contrastive / global (Chen et al., 2022; Yuan et al., 2022)	model and data source	negative set / global interaction estimate	objective or gradient	non-decomposability; batch-dependent target
Jensen gap (Xu et al., 2025)	model and full-data target	minibatch aggregate inside nonlinear loss	expected objective	nonlinear plug-in bias; shifted optimum
Mini-batch quadratics (Tatzel et al., 2025)	checkpoint and model	batch used to define local geometry	slope / curvature	regression to mean; distorted geometry
Stochastic preconditioning (Nayak et al., 2026)	parameter and population target	gradient / preconditioner estimates	update vector	same-batch coupling + nonlinear inverse
Ours	**global rows + optimizer step**	**statistic scope**	**mean gradient / effective objective**	**self-inclusion + plug-in bias; path shift**

C Proofs for Section 3

For centered finite-population arrays x_i and y_i,

$$\mathbb{E}[\bar{x}_S \bar{y}_S^\top] = \rho_B \Sigma_{xy}, \qquad \rho_B = \frac{1}{B} - \frac{1}{n}. \tag{14}$$

Sampling identity. For centered finite-population arrays $x_i \in \mathbb{R}^r$ and $y_i \in \mathbb{R}^s$, Equation 14 follows by writing the sample means with inclusion indicators $I_i = \mathbf{1}\{i \in S\}$, using $\mathbb{E}I_i = B/n$ and $\mathbb{E}[I_i I_j] = B(B-1)/[n(n-1)]$ for $i \neq j$, and the centering identity $\sum_{i \neq j} x_i y_j^\top = -\sum_i x_i y_i^\top$. This yields $(n-B)[Bn(n-1)]^{-1} \sum_i x_i y_i^\top = \rho_B \Sigma_{xy}$.

Theorem 1. For scalar centered x_i with $|x_i| \leq M$, expanding $(\sum_{i \in S} x_i)^4$ and grouping terms by index-collision pattern gives $\mathbb{E}[\bar{x}_S^4] \leq 8M^4 \rho_B^2$. Applied coordinatewise, this implies $\mathbb{E}\|\Delta_S\|^3 = O(d^{3/2} M_m^3 \rho_B^{3/2})$ and the joint moment bound needed after differentiating the Taylor remainder. For the proof, write $\gamma_i = \nabla_a \ell_i$, $H_i = \nabla_{aa}^2 \ell_i$, $\bar{H}_a = n^{-1} \sum_i H_i$, and let $\Sigma_{\gamma m}$ and Σ_{mm} denote the corresponding finite-population covariances. Expanding each $\ell_i(\theta, a + \Delta)$ to second order gives the terms in Theorem 1: independence removes the split linear term; Equation 14 makes the same-batch linear term $\rho_B \operatorname{tr}(\Sigma_{\gamma m}) = \rho_B \beta_{\text{coup}}$; and the population Hessian part of the quadratic term is $\frac{\rho_B}{2} \operatorname{tr}(\bar{H}_a \Sigma_{mm}) = \rho_B \beta_{\text{plug}}$. The centered Hessian–moment coupling and the third-order Taylor remainder are $O(\rho_B^{3/2})$.

Theorem 2. Assume in addition that the centered coordinates of $D_\theta m_i$ are bounded, $\|D_\theta a\|_F$ is bounded, and derivatives through order four in (θ, a) are uniformly bounded. Differentiating the integral third-order remainder produces direct θ differentiation, movement of the evaluation point, and differentiation of the three copies of Δ, with integral coefficients $1/6$, $1/24$, and $1/2$. The fourth-moment and joint moment bounds control the resulting $\|\Delta\|^3$ and $\|\Delta\|^2 \|D_\theta \Delta\|_F$ terms, which gives the $O(\rho_B^{3/2})$ remainder. Because the population has finitely many subsets, differentiation commutes with the sampling expectation.

The local stationary-point consequence stated in the main text is

$$\theta_{e,B}^\star = \theta^\star - \rho_B (H^\star)^{-1} \nabla \beta_e(\theta^\star) + O(\rho_B^{3/2}), \qquad H^\star = \nabla^2 J(\theta^\star). \tag{15}$$

Stationary-point expansion (Eq. 15). Write $F(\theta) = \mathbb{E}[g_e](\theta) = \nabla J(\theta) + \rho_B \nabla \beta_e(\theta) + r(\theta)$ with $\|r\| \leq C \rho_B^{3/2}$ near $\theta^\star$; F is continuous as a finite average of continuous functions. Let

$\theta_0 = \theta^\star - \rho_B (H^\star)^{-1} \nabla \beta_e(\theta^\star)$ and $T(\theta) = \theta - (H^\star)^{-1} F(\theta)$. For θ in the ball of radius $K\rho_B^{3/2}$ around θ_0,

$$T(\theta) - \theta_0 = -(H^\star)^{-1}\big[\nabla J(\theta) - H^\star(\theta - \theta^\star)\big] - \rho_B (H^\star)^{-1}\big[\nabla \beta_e(\theta) - \nabla \beta_e(\theta^\star)\big] - (H^\star)^{-1} r(\theta).$$

Since $\|\theta - \theta^\star\| = O(\rho_B)$, the first two terms are $O(\rho_B^2)$ by the C^3 and Lipschitz assumptions, and the last is at most $\|(H^\star)^{-1}\| C \rho_B^{3/2}$. Choosing $K > \|(H^\star)^{-1}\| C$, T maps the ball into itself for small ρ_B, and Brouwer's theorem gives a fixed point, which is a zero of F.

Corollary 1. Each block S_k is marginally a size-b sample without replacement, so $\mathbb{E}[g_{B,b}^{\text{micro}}] = \mathbb{E}[g_{\text{same},b}]$ by symmetry, while $\mathbb{E}[g_B^{\text{global}}] = \mathbb{E}[g_{\text{same},B}]$. Subtracting and applying Theorem 2 gives Equation 10, using $\rho_b - \rho_B = 1/b - 1/B$ and $\rho_B \le \rho_b$. For additive objectives, the average of the K microbatch gradients equals the gradient over their union for every draw.

C.1 Mean-centered reward normalization

Let $\ell_i = -s_i(\theta)(r_i - a)$ with fixed rewards, and let $C = (n-1)^{-1} \sum_i (\nabla s_i - \overline{\nabla s})(r_i - \bar r)$. Then $\nabla J = -n^{-1} \sum_i \nabla s_i (r_i - \bar r) = -\frac{n-1}{n} C$. For a sample S, $g_{\text{same},S} = -B^{-1} \sum_{i \in S} \nabla s_i \, r_i + \overline{\nabla s}_S \, \bar r_S$. The first term has mean $-n^{-1} \sum_i \nabla s_i r_i$, and by Equation 14, $\mathbb{E}[\overline{\nabla s}_S \, \bar r_S] = \overline{\nabla s}\, \bar r + \rho_B C$. Hence $\mathbb{E}[g_{\text{same},B}] = -\big(\frac{n-1}{n} - \rho_B\big) C = (1 - \kappa_B) \nabla J$ with $\kappa_B = \rho_B \frac{n}{n-1} = \frac{n-B}{B(n-1)}$. The affine loss and parameter-independent m_i make this expression exact. For 4×32 versus one statistic of 128 rows, $\kappa_{32} - \kappa_{128} = \frac{n}{n-1}\big(\frac{1}{32} - \frac{1}{128}\big) \approx 0.023$.

D Control Variate and Response Bound

Exact control-variate construction. Evaluate derivatives at the population statistic $a = a(\theta)$ and write $U_i = \partial_\theta \ell_i(\theta, a)$, $D_i = J_{m_i}(\theta)$, and $\bar\gamma = n^{-1} \sum_i \nabla_a \ell_i(\theta, a)$. The population gradient is $\nabla J = \bar U + \bar D^\top \bar\gamma$. For the same sampled rows used by implementation e, define

$$g_{\text{cv}}(S, R) = \frac{1}{B} \sum_{i \in S} U_i + \Big(\frac{1}{B} \sum_{j \in R} D_j\Big)^\top \bar\gamma, \tag{16}$$

with $R = S$ for same-batch reuse and an independent size-B sample R for the sample-split reference. Both sample means have their population means, so $\mathbb{E}[g_{\text{cv}}] = \nabla J$ and Eq. equation 11 follows exactly. Unbiasedness follows from the sample means having their population means and holds for any coupling of S and R. Using common random numbers makes g_e and g_{cv} strongly correlated; the usual variance identity $\text{Var}(g_e - g_{\text{cv}}) = \text{Var}(g_e) + \text{Var}(g_{\text{cv}}) - 2\text{Cov}(g_e, g_{\text{cv}})$ explains the reduction reported in Section 6.

Response bound. Let the full-population and expected-minibatch-gradient trajectories start from the same θ_0, and let $\Delta\theta_t^e = \theta_t^e - \theta_t$ be the actual displacement. If J has an M_H-Lipschitz Hessian on a tube containing both trajectories and Δg_e is L_Δ-Lipschitz, then

$$\begin{aligned} \Delta\theta_{t+1}^e &= A_t \Delta\theta_t^e - \eta_t \Delta g_e(\theta_t) + \xi_t, \qquad A_t = I - \eta_t \nabla^2 J(\theta_t), \\ \|\xi_t\| &\le \eta_t \Big(L_\Delta \|\Delta\theta_t^e\| + \frac{M_H}{2} \|\Delta\theta_t^e\|^2 \Big). \end{aligned} \tag{17}$$

Iterating the linear part gives $\widehat\delta_T^e$ in Equation 12, and iterating the remainder gives

$$\|\Delta\theta_T^e - \widehat\delta_T^e\| \le \sum_{t=0}^{T-1} \Big(\prod_{s=t+1}^{T-1} \|A_s\| \Big) \eta_t \Big(L_\Delta \|\Delta\theta_t^e\| + \frac{M_H}{2} \|\Delta\theta_t^e\|^2 \Big). \tag{18}$$

If independent control-variate estimates $\widehat{\Delta g}_t$ are used at the checkpoints, the plug-in response is unbiased for $\widehat\delta_T^e$; its covariance is obtained by transporting each checkpoint covariance through the same later-step products that appear in Eq. equation 12.

E TRANSFORMED AND IMPLICIT SHARED STATISTICS

Section 5 treats losses that depend on a directly. If the loss depends on a transformed statistic $q = h(a) \in \mathbb{R}^s$, let $v_S^q = B^{-1} \sum_{i \in S} \nabla_q \ell_i(\theta, q)$; the statistic sensitivity in Equation 13 becomes $J_h(a_S)^\top v_S^q$. For a regular implicit equilibrium $F(q, a) = 0$, solve

$$F_q^\top \lambda = v_S^q \tag{19}$$

and use $-F_a^\top \lambda$ as the statistic sensitivity. This is the standard implicit-function adjoint (Blondel et al., 2022); the chunked-backward pattern is unchanged, and the retained state gains $s^2 + M_{\text{solve}}(s)$ for the adjoint solve.

Figure 4 compares full-graph differentiation, checkpointing, cached row outputs, and generic (three-scan) and specialized (two-scan) chunked differentiation on Smooth-F1 target sets of 5,000 to 45,000 rows. The specialized implementation exploits the fact that its shared sensitivity can be accumulated with a few additional row averages; the generic construction retains the three logical stages described in Section 5.

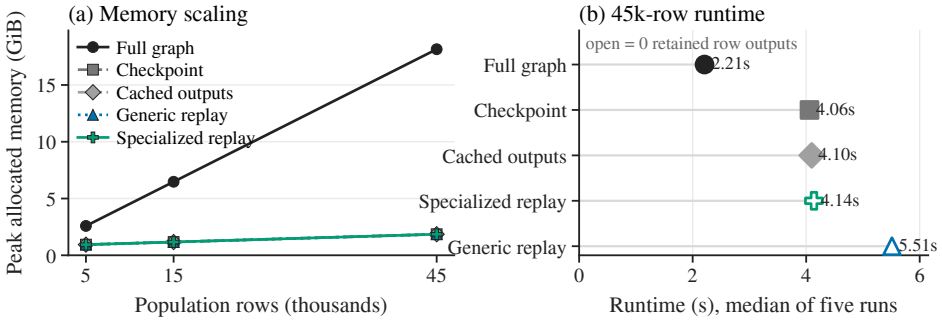

Figure 4: Chunked global-statistic differentiation trades extra passes for lower retained state while preserving the exact target gradient. Peak CUDA memory (left) and runtime (right) are reported for Smooth-F1 target sets up to 45,000 rows.

F EXTENDED EXPERIMENTAL DETAILS

F.1 BINARY SMOOTH-F1 CONFIRMATORY CALIBRATION

The Binary Smooth-F1 confirmatory design crosses requested population sizes 5,000 and 45,000, positive-class retentions 0.25, 0.50, and 1.00, and requested batch fractions 0.01, 0.05, and 0.20. Six regimes are frozen using deterministic theory-based criteria, followed by fresh confirmatory sampling. Each same-batch or split method–condition cell uses 2,500 fresh fixed-size draws without replacement in float64 at the registered frozen CNN parameter state. Condition 11 provides the small-theory control, and both projected intervals contain zero as expected.

Table 2: Theory-selected binary Smooth-F1 confirmatory regimes. Rows report the selection criterion, condition ID, realized population size, batch size B, and ρ_B; condition 11 is the small-theory control.

Order	Criterion	Cond.	Rows	B	ρ_B
1	largest same 1	6	5000	50	1.980×10^{-2}
2	largest same 2	3	4763	48	2.062×10^{-2}
3	largest split 1	7	5000	250	3.800×10^{-3}
4	largest split 2	0	4644	46	2.152×10^{-2}
5	largest angle	9	41636	416	2.380×10^{-3}
6	small-theory control	11	41636	8327	9.607×10^{-5}

Across the five signal regimes, same-batch projected signs agree with theory in 5/5 cells, all 5/5 projected 95% Monte Carlo intervals exclude zero, and the median theory–empirical cosine is 0.978. For split sampling, the corresponding counts are 3/5 and 2/5, with median cosine 0.668. The supplementary artifact contains all 12 fresh method–condition rows and the complete cell-level diagnostics.

F.2 Optical Digits Smooth Macro-F1

Table 3: Optical Digits Smooth Macro-F1 control-variate audit. Each row summarizes nine checkpoint-by-batch settings for one model and implementation; columns report the trace-VRF range and median.

Model	Implementation	VRF range	Median VRF
Linear	Same	18.05–53.18	31.39
Linear	Split	5.27–14.70	8.56
MLP	Same	21.65–104.91	48.41
MLP	Split	5.68–22.66	11.07

Table 4: Unrolled response on Optical Digits Smooth Macro-F1. Errors are terminal parameter relative errors (%) of the unrolled response and of the naive cumulative-bias baseline; the cosine measures directional agreement of the unrolled response.

Model	Implementation	Response err.	Naive err.	Response cosine
Linear	Same	12.34	41.97	.9924
Linear	Split	13.66	46.34	.9909
MLP	Same	18.56	43.07	.9828
MLP	Split	14.83	56.30	.9897

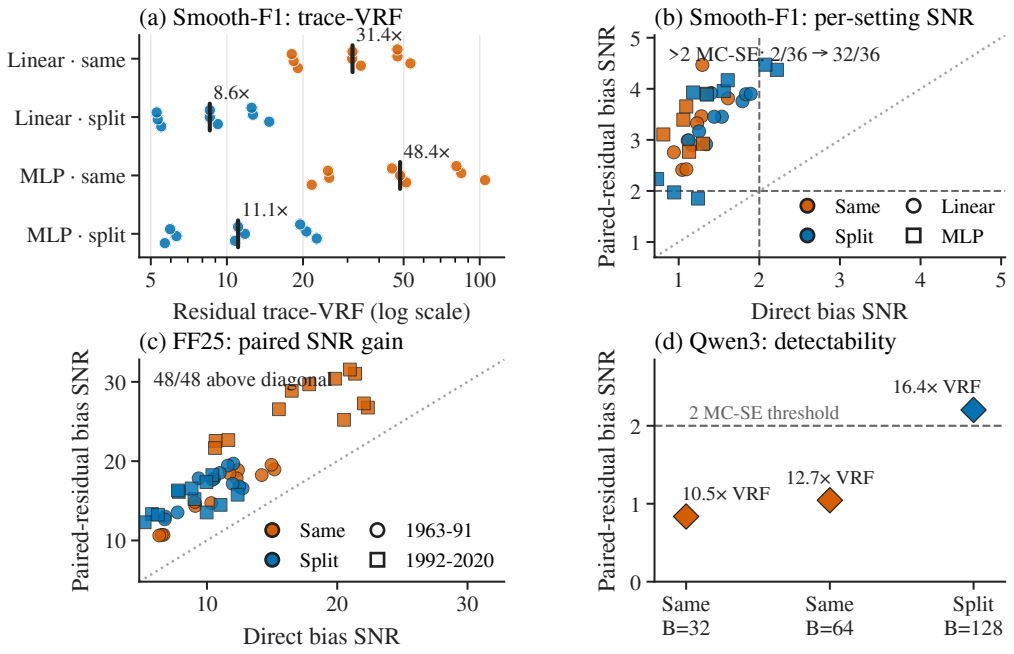

Figure 5: The control variate sharpens gradient-bias estimates. (a,b) Optical Digits trace-VRF and direct/control-variate SNR; (c) FF25 in both periods; (d) Qwen3-0.6B residual SNR with VRF labels. Orange/blue denote same/split; dashed lines mark 2 MC-SE.

F.3 FF25 SETTINGS AND CROSS-PERIOD SUMMARY

The official Kenneth R. French file contains 25 value-weighted portfolio-return columns. Returns are converted from percentages to decimals. Each period is treated as one fixed finite population. Both periods use 24 free softmax logits, the mean–volatility objective from Section 2, same/split sampling without replacement, $B \in \{8, 16, 32, 64\}$, full-population checkpoints 0/6/12, step size 5, $\epsilon = 10^{-8}$, 20,000 static draws per setting, and float64 CPU computation. Dynamic audits use 20,000 control-variate draws per step and 50,000 expected-minibatch-gradient draws per step.

Table 5: FF25 cross-period replication. SNR is bias norm divided by vector MC-SE; VRF is the ratio of direct to control-variate trace variance.

Quantity	1963–1991	1992–2020
Months	342	348
Raw SNR range	6.34–15.18	5.22–22.36
Control-variate SNR range	10.63–19.69	12.30–31.55
Control-variate SNR $>$ raw	24/24	24/24
VRF range	1.56–3.60	1.47–5.36
Median VRF	2.53	3.11
Theory cosine range	.869–.999	.971–1.000
Median theory cosine	.988	.995
Median empirical/theory norm	1.022	1.045
Response error	1.61–3.40%	1.46–3.39%
Naive error	4.89–8.67%	8.19–10.18%

F.4 FF25 FIXED-GLOBAL-BATCH MICROBATCH GRID

We additionally fix the global batch at $B = 64$ and partition each sampled batch into $K \in \{2, 4, 8\}$ equal microbatches ($b \in \{32, 16, 8\}$). For each of the two non-overlapping periods and each full-population checkpoint $t \in \{0, 6, 12\}$, 20,000 paired global-versus-microbatch draws use the same sampled rows and a fresh random within-batch permutation. Across all 18 cells, median theory–empirical cosine is 0.958, median measured/theory norm ratio is 1.051, and median SNR is 52.1. Normalizing by the $b = 32$ gap within each period/checkpoint, the leading law predicts relative factors 1:3:7 and the median measured factors are 1:3.00:7.31. The released runner fixes the random seeds for every period–checkpoint pair so that these Monte Carlo summaries are reproduced exactly. Period-specific summaries are shown below.

Table 6: FF25 fixed-global-batch microbatch grid ($B = 64$).

Period	Median cosine	Median norm ratio	$16/32$ gap ratio	$8/32$ gap ratio
1963–1991	0.903	1.100	3.08	7.71
1992–2020	0.987	1.041	2.98	7.07

F.5 FROZEN-ROLLOUT QWEN3 SETTINGS

The 0.6B audit fixes checkpoint 0 and 2,048 rollouts (256 GSM8K prompts $\times 8$); all settings share buffer, seed, reward definition, LoRA configuration, and 512 draws.

Table 7: Qwen3-0.6B control-variate audit. SNR is residual bias norm divided by residual vector MC-SE.

Setting	Residual SNR	VRF	Theory cosine
same/$B = 32$	0.836	10.452	0.200
same/$B = 64$	1.045	12.699	0.179
split/$B = 128$	2.203	16.380	0.892

The 4B experiments use a 2,048-rollout population from the same prompt IDs (8 each), with thinking enabled and a 32,768-token cap; all rollouts remain below the cap and 2,028/2,048 parse successfully. The static audit fixes $B = 128$, one initial LoRA state, and 256 paired draws per microbatch size.

Table 8: Qwen3-4B microbatch audit. Bias/global is the estimated microbatch-bias norm divided by the global mean-gradient norm.

Microbatch	Bias/MC-SE	Theory cos.	Emp./theory	Bias/global
2×64	2.563	0.931	1.604	4.14%
4×32	6.198	0.988	1.903	14.74%

The 20-step trajectory reuses the same 4B population and LoRA state, freezes 20 batches of $B = 128$, and runs plain SGD at learning rate 0.05; the global branch uses global-128 reward normalization and the microbatch branch uses 4×32 normalization. At step 20, the global and micro displacements are 8.596×10^{-3} and 6.967×10^{-3}, the branch gap is 1.750×10^{-3} (20.36%), the displacement cosine is 0.99658, and the maximum importance ratios are 1.0091 and 1.0082.

Numerical verification. FF25 analytic/autograd and exact-enumeration checks agree to numerical precision (maximum reported relative error 1.68×10^{-15}) (Paszke et al., 2019). The 4B branches share a frozen, hashed initial state.