
Is Relative Curvature Normalization Redundant in 1.5-SPSA?

A Scale-Invariance Analysis with Matched-Budget Tests

Han (Harrison) Lee
Stanford University
hanl@stanford.edu

Abstract

1.5-SPSA re-weights random probe directions in zero-order training by a power of their estimated directional curvature; its authors leave batch-normalized or relative curvature estimates to future work. We ask whether such normalization reduces sensitivity to loss scaling and batch composition beyond what a tuned scalar learning rate already provides. We first derive what positive loss rescaling does to curvature-weighted probe updates: every weighting rule decomposes into a per-step scalar, which is a learning rate, and relative probe weights, which set the direction; the published rule’s direction depends on the loss scale only through its absolute floor $\lambda_{\text{reg}} = 1$, and median normalization makes the relative weights exactly scale-free. We then compare, at matched forward-evaluation counts with paired probe seeds and equal tuning budgets, the original method, 1SPSA and MeZO at the same budget, scalar normalizations, and relative-median variants with floors, clipping, reliability shrinkage and per-batch/pooled scopes, on stiff quadratics with exact Hessians and controlled noise across condition numbers $1\text{--}10^3$ and loss scales $10^{-4}\text{--}10^4$, and on two small neural tasks (MNIST MLP; prompt-based SST-2 fine-tuning of GPT-2) with validation-only selection and a first-order reference. Loss-scale robustness turns out to be entirely a scalar effect: a Newton-unit scalar on 1SPSA weights is already exactly invariant, while the published rule is the only one whose retuned behaviour varies with the scale. The published weighting ($\alpha = 0.1$) is indistinguishable from 1SPSA everywhere; full Newton weights ($\alpha = 1$) help on ill-conditioned noise-free quadratics whose stiffness is concentrated in one direction (the upstream stress test) but not on a log-uniform spectrum with the same condition number, and collapse below unit curvature SNR unless floored in relative units, after which they match but do not beat 1SPSA; no shrinkage or scope variant improves on the plainest relative rule; and MeZO-style sequential updates dominate all averaging rules at equal evaluations. On the neural tasks the derived loss-scale behaviour is reproduced exactly; a floored relative $\alpha = 1$ rule beats 1SPSA by a small margin on an MNIST MLP at the protocol batch size but is worse at micro-batch sizes 32 and 512, while on prompt-based GPT-2 fine-tuning no zero-order rule is reliably better than 1SPSA at our budget (confirmed by a re-run with a corrected, label-balanced selection split), and a first-order optimizer is far better at $2.2\times$ the peak memory. We conclude that relative curvature normalization is redundant with scalar learning-rate normalization for stability, and report the negative result with the derivation that explains it.

1 Introduction

Zero-order (ZO) optimization trains a model from loss evaluations alone, perturbing all parameters along random probe directions. Because no activations, gradients or optimizer moments are stored, memory equals inference memory, which is why MeZO-style methods are attractive for fine-tuning large language models [Malladi et al., 2023]. 1.5-SPSA [Chaubard et al., 2026a] improves on the one-probe estimator by using many probes per step and re-weighting each probe by a power of its directional curvature, estimated from a three-point stencil at the cost of one extra shared forward pass. Its authors leave “batch-normalized or relative curvature estimates” to future work, suggesting that normalizing the curvature estimates across probes or batches could further stabilize updates without new hyper-parameters or memory.

This paper tests that suggestion. The question is less innocent than it looks, because any rule that multiplies every probe weight by a common factor is a learning-rate change in disguise: a method that appears more stable under a change of loss scale may simply have a differently calibrated step size, and a learning-rate sweep for the baseline would erase the difference. We therefore begin (Section 3) with a derivation of what is and is not invariant under positive rescaling of the loss, decompose every weighting rule into a per-step scalar (a learning rate) and relative probe weights (a direction), and identify the single mechanism through which the published rule’s *direction* depends on the loss scale: an absolute floor $\lambda_{\text{reg}} = 1$ whose units are those of the loss. Normalizing curvatures by their per-step median makes the relative weights exactly scale-free, and normalizing the overall step by the same median (“Newton units”) makes the whole update invariant.

We then run the comparison the derivation calls for. On stiff quadratics with exact Hessians and two controlled noise models we compare, at matched forward-evaluation counts (exact $2P$ versus $2P+1$ accounting, paired probe seeds, perturbations regenerated from seeds), the original 1.5-SPSA, 1SPSA and MeZO at the same evaluation budget, scalar learning-rate normalizations, and relative-median variants with floors, clipping, reliability shrinkage and per-batch/per-step/pooled scopes, each with the same learning-rate tuning budget, under both a *retuned* and a *transfer* protocol. We confirm on two small neural tasks (an MNIST MLP on CPU and prompt-based SST-2 fine-tuning of GPT-2 on a single GPU) with validation-only selection, a first-order reference, and memory accounting.

Findings. (1) Loss-scale sensitivity in probe-space ZO methods is a learning-rate phenomenon: after retuning, every rule except the published floor is exactly invariant, and with a single transferred learning rate only the rules with a $1/m$ scalar (and $\alpha = 1$ weights while their floor is inactive) survive a $10^{\pm 2}$ rescaling; the simplest scalar baseline already achieves this. (2) The published weighting ($\alpha = 0.1$, $\lambda_{\text{reg}} = 1$) is statistically indistinguishable from 1SPSA in every setting we tested; its absolute floor is the only component that makes its behaviour depend on the loss scale, and it silently turns the method into 1SPSA at small loss scales. (3) Full probe-space Newton weights ($\alpha = 1$) do help on the upstream ill-conditioned quadratic (1.5–5 orders of magnitude lower loss than 1SPSA at matched evaluations), a gain that a pre-registered second Hessian family with the same condition numbers but a log-uniform spectrum reduces to ≤ 0.13 orders of magnitude, as the derivation predicts from the smaller across-probe curvature spread. They also collapse when the curvature signal-to-noise ratio drops below one unless a *relative* floor or clipping is used; even then they only recover 1SPSA, and none of the robustification variants (shrinkage, per-batch, pooled) beats the plainest relative rule. (4) At matched forward evaluations, MeZO-style sequential single-probe updates beat every multi-probe averaging rule on quadratics. (5) On the neural tasks the loss-scale transfer behaviour predicted by the derivation is reproduced exactly (scale-invariant rules agree to four digits across $s \in \{0.01, 1, 100\}$ on GPT-2). On the MNIST MLP a relative $\alpha = 1$ rule with a floor at the median is the only ZO rule that beats 1SPSA, by a small margin that survives ten paired seeds (-0.011 [$-0.015, -0.006$] validation loss, 9/10 seeds) but reverses at micro-batch sizes 32 and 512, so no variant gives a batch-size-robust gain; on SST-2 with GPT-2 no ZO rule is reliably better than 1SPSA on held-out accuracy at our budget, both with our original dev split (which turned out to be label-imbalanced) and in a corrected re-run with a balanced split, and Adam at the same forward-equivalent budget is far better at $2.2\times$ the peak GPU memory. We conclude that relative curvature normalization is redundant with a scalar learning-rate normalization for the stability properties claimed for it, and useful only as a scale-free way of expressing the floor when strong ($\alpha \approx 1$) curvature weighting is wanted.

2 Background and related work

Zero-order optimization of neural networks. Simultaneous-perturbation stochastic approximation [Spall, 1992] estimates a descent direction from two loss evaluations along a random direction; its second-order form 2SPSA [Spall, 2000] estimates the Hessian from additional perturbations. MeZO [Malladi et al., 2023] made the one-probe estimator practical for fine-tuning large language models by regenerating the perturbation from a seed, so that memory equals inference memory; it motivated analyses of why ZO works on pretrained models (low effective rank of the Hessian) and a line of follow-ups on variance reduction, sparsity and preconditioning [Zhang et al., 2024]. 1.5-SPSA [Chaubard et al., 2026a] sits between 1SPSA and 2SPSA: it keeps many probes per step and uses the second difference along each probe (one extra shared centre evaluation) as a directional curvature estimate, weighting probe i by $1/\max(\lambda_{\text{reg}}, |\hat{c}_i|^\alpha)$ with $\alpha = 0.1$. Its Section 6 names batch-normalized or relative curvature estimates as future work; the companion model-sharding paper [Chaubard et al., 2026b] addresses distribution, not weighting. The present work tests exactly that future-work item.

Preconditioning and normalization in stochastic optimization. Diagonal preconditioners such as Adagrad/Adam [Kingma and Ba, 2015] and normalized or sign-based updates make the step size insensitive to the gradient scale; the equivalence between a common rescaling of all preconditioner entries and a learning-rate change is well known for first-order methods, and our Section 3 transfers it to probe-space weights. Robust scale estimates (median absolute deviation) and James–Stein shrinkage towards a pooled value are classical statistical tools [Huber, 1981]; we use them to normalize and to shrink noisy curvature estimates. Curvature-matching step sizes for SPSA-type methods were also studied by Spall [2000] and in the Hessian-free literature, but we are not aware of prior work on loss-scale invariance of probe-space curvature weights.

Evaluation protocol. Prior ZO papers compare at equal optimizer steps; the 1.5-SPSA paper’s Appendix A.3 counts forward passes ($2P$ versus $2P+1$) and charges backpropagation three forward-equivalents. We adopt that accounting throughout and additionally report wall-clock and peak memory, following the handoff’s requirement that equal optimization steps are not equal training compute.

3 What loss scaling does to curvature-weighted probe updates

Setting. Let $L : \mathbb{R}^d \rightarrow \mathbb{R}$ be the (mini-batch) loss, $\theta \in \mathbb{R}^d$ the parameters, $\varepsilon > 0$ the finite-difference scale and $z_1, \dots, z_P \in \{-1, +1\}^d$ i.i.d. Rademacher probes regenerated from a seed. For each probe the stencil $\{L(\theta + \varepsilon z_i), L(\theta), L(\theta - \varepsilon z_i)\}$ gives the two scalars

$$\begin{aligned} \Delta \ell_i &= L(\theta + \varepsilon z_i) - L(\theta - \varepsilon z_i) \approx 2\varepsilon z_i^\top \nabla L(\theta), \\ \hat{c}_i &= \frac{L(\theta + \varepsilon z_i) - 2L(\theta) + L(\theta - \varepsilon z_i)}{\varepsilon^2} \approx z_i^\top \nabla^2 L(\theta) z_i. \end{aligned} \tag{1}$$

All methods we study are of the form

$$\theta \leftarrow \theta - \frac{\eta}{2\varepsilon P} \sum_{i=1}^P \Delta \ell_i w_i z_i, \quad w_i = w(\hat{c}_1, \dots, \hat{c}_P, \Delta \ell_1, \dots, \Delta \ell_P), \tag{2}$$

with $w_i \equiv 1$ for 1SPSA (and MeZO, which uses $P = 1$ Gaussian probes and P sequential sub-steps), and

$$w_i^{1.5} = \frac{1}{\max(\lambda_{\text{reg}}, |\hat{c}_i|^\alpha)}, \quad \alpha = 0.1, \lambda_{\text{reg}} = 1 \tag{3}$$

for 1.5-SPSA as published [Chaubard et al., 2026a]; we verified that the reference implementation (commit 68036ff) implements exactly Eqs. 1–3 with η a free learning rate (the paper ties $\eta = \varepsilon$). The centre evaluation is shared across probes, so a step costs $2P+1$ forward passes per micro-batch versus $2P$ for 1SPSA; we count both exactly.

A decomposition. Any weight vector factors as $w_i = S \cdot r_i$ with a per-step scalar $S > 0$ and *relative weights* r_i normalized so that $\text{med}_i r_i = 1$. The scalar S multiplies the whole step and is

indistinguishable from a (possibly step-dependent) learning rate; only r changes the *direction* of the update inside the probe span. This gives a clean test for whether a weighting rule does anything a learning rate could not: compare it with the *scalar baseline* that uses the same S but $r_i \equiv 1$.

Positive loss scaling. Replace L by sL with $s > 0$ (a change of units, a different reduction over tokens, a temperature, or a batch of harder examples). Then $\Delta\ell_i \rightarrow s \Delta\ell_i$ and $\hat{c}_i \rightarrow s \hat{c}_i$ exactly, for every probe, because the stencil is linear in L . The consequences for the rules above follow by inspection.

Proposition 1 (Scale equivariance of the published rule). *Under $L \rightarrow sL$ the 1.5-SPSA step with weights (3) satisfies: (i) if $|\hat{c}_i|^\alpha \geq \lambda_{\text{reg}}$ for all i both before and after scaling, the step is multiplied by $s^{1-\alpha}$ and the relative weights r are unchanged; (ii) if $|\hat{c}_i|^\alpha < \lambda_{\text{reg}}$ for all i (floor active everywhere) the step is multiplied by s and equals a 1SPSA step; (iii) otherwise the relative weights change with s : $r_i \propto 1/\max(\lambda_{\text{reg}}, s^\alpha |\hat{c}_i|^\alpha)$, which is not a function of $|\hat{c}_i|/\text{med}_j |\hat{c}_j|$ alone.*

Proof. Immediate from $w_i(s) = 1/\max(\lambda_{\text{reg}}, s^\alpha |\hat{c}_i|^\alpha)$ and $\Delta\ell_i(s) = s \Delta\ell_i$. In case (i) $w_i(s) = s^{-\alpha} w_i(1)$ for all i ; in case (ii) $w_i(s) = 1/\lambda_{\text{reg}}$ for all i . $\square$

Cases (i) and (ii) are *learning-rate equivalences*: a tuned learning rate at each loss scale removes the effect entirely, and with $\alpha = 1$ in case (i) the step is already scale-invariant. Case (iii) is the only regime in which the published rule’s direction depends on the loss scale, and it is created by the absolute floor $\lambda_{\text{reg}} = 1$, whose units are those of $|\hat{c}|^\alpha$ (loss units to the power α). Retuning λ_{reg} together with η also removes it, so the floor is a second scale-dependent hyper-parameter rather than a direction-changing mechanism per se.

Proposition 2 (Relative normalization). *Let $m = \text{med}_i |\hat{c}_i|$ (per step) and $r_i = 1/\max(\lambda, (|\hat{c}_i|/m)^\alpha)$ for a dimensionless floor $\lambda \geq 0$. Then r is invariant under $L \rightarrow sL$, the step with $w_i = r_i$ is multiplied by s (a pure learning-rate effect), and the step with $w_i = r_i/m$ is invariant. Moreover, when $\lambda = 0$ and no floor is active in (3), $r_i = m^\alpha w_i^{1.5}$: relative normalization equals the published rule up to the per-step scalar $S = m^\alpha$.*

Corollary 1 (Redundancy on deterministic quadratics). *If L is quadratic with Hessian H and noise-free, $\hat{c}_i = z_i^\top H z_i$ does not depend on θ , so m is a step-independent random variable with mean $\text{tr } H$ and relative fluctuations $O(\|H\|_F / \text{tr } H)$. In this regime every per-step scalar $S(m)$ is a learning-rate change up to small random fluctuations, and the only non-redundant effect of any curvature rule is in r . If in addition H is diagonal in the coordinate basis, $z_i^\top H z_i = \text{tr } H$ for every Rademacher probe, $r_i \equiv 1$, and every curvature weighting is exactly 1SPSA with a rescaled learning rate.*

The corollary explains why the upstream stress test uses a *rotated* stiff paraboloid: the probe curvature $z^\top H z = \text{tr } H + \sum_{j \neq k} H_{jk} z_j z_k$ varies across probes only through off-diagonal Hessian entries in the probe basis. For $H = I + (k-1)uu^\top$ (one stiff direction u , $\|u\| = 1$) one obtains $z^\top H z = d + (k-1)(u^\top z)^2$: probes aligned with the stiff direction have larger curvature and are down-weighted, which is the mechanism by which 1.5-SPSA can take larger steps in the flat subspace.

Noise in the curvature estimate. With i.i.d. observation noise of standard deviation σ_L on each loss evaluation, $\text{Var}(\hat{c}_i) = 6\sigma_L^2/\varepsilon^4$ (shared centre, marginal variance), while the across-probe spread of the true curvature is $\text{Var}_z(z^\top H z) = 2\|H\|_F^2 - 2\sum_j H_{jj}^2$. The ratio of these two variances is a *curvature signal-to-noise ratio* that is invariant under loss scaling (both scale as s^2) but decays as ε^{-4} . Below unit SNR, curvature weights are noise-driven re-weightings of 1SPSA; this is where a reliability-based shrinkage towards the step-level value is justified, and it is also where $\alpha = 1$ weights ($1/|\hat{c}_i|$) can become arbitrarily large when a noisy $\hat{c}_i$ passes near zero, which the published floor prevents only in absolute units.

4 Normalized variants

All variants keep the update form (2), cost $2P+1$ forward passes per micro-batch (the same as 1.5-SPSA), store no model-sized state, and are controlled by the dimensionless statistics $\rho_i = |\hat{c}_i|/m$ with $m = \text{med}_i |\hat{c}_i|$. They were fixed before the held-out runs (`src/zo/methods.py`).

Scalar baselines (LR-only). $w_i = m^{-\alpha}$ (SCALAR-POW), $w_i = 1/m$ (SCALAR-INV, “Newton units”: the step $\Delta\ell_i/(\varepsilon m)$ is invariant to loss scale, so one learning rate serves all scales), and $w_i = 1/\text{med}_j |\Delta\ell_j/2\varepsilon|$ (SCALAR- $\Delta\ell$, needs no centre evaluation). Their relative weights are $r_i \equiv 1$: whatever they gain over 1SPSA is attributable to a per-step scalar, i.e. to an adaptive learning rate.

Relative-median weighting. $w_i = 1/\max(\lambda, \rho_i^\alpha)$ (REL-MEDIAN) with $\alpha \in \{0.1, 0.5, 1\}$ and a dimensionless floor $\lambda \in \{0, 0.5, 1, 2\}$; $\lambda = 1$ means probes with below-median curvature are not up-weighted. REL-MEDIAN $\times 1/m$ adds the Newton-unit scalar.

Clipping. ρ_i is clipped to $[1/C, C]$ before weighting, $C \in \{3, 10\}$: it bounds the dynamic range of the weights so that a single noisy near-zero $\hat{c}_i$ cannot dominate the step.

Reliability shrinkage. When a step uses $B \geq 2$ micro-batches, the stencil is evaluated on each of them and yields $\hat{c}_{ib}$ for free. We decompose the across-probe variance of $\hat{c}_i = \frac{1}{B} \sum_b \hat{c}_{ib}$ into signal and noise, $\hat{\rho} = \max(0, \widehat{\text{Var}}_i \hat{c}_i - \widehat{\text{Var}}_{\text{within}}/B)/\widehat{\text{Var}}_i \hat{c}_i \in [0, 1]$, and shrink $\hat{c}_i \leftarrow m + \hat{\rho}(\hat{c}_i - m)$ before weighting (James–Stein-type shrinkage towards the step-level value). It costs no extra evaluations and one scalar of state.

Normalization scope. *per-step* (m from the current step’s P probes), *per-batch* (m_b from each micro-batch’s curvatures, weights averaged over b), and *pooled* (m an exponential moving average over steps, $\beta = 0.9$). Per-step normalization has no memory; pooled normalization has one scalar.

Implementation and accounting. Probe directions are regenerated from (seed, step, probe) and applied in place; the update is applied probe-by-probe after the weights are known, so no model-sized gradient buffer exists (the reference implementation accumulates one fp32 buffer of model size). Every forward pass on a micro-batch counts as one evaluation; first-order baselines are charged three forward-equivalents per micro-batch (backward ≈ 2 forwards, following the upstream paper’s Appendix A.3) and we also report wall-clock. Probe seeds, batch-composition seeds and initialization are paired across methods (same seed list), so comparisons at the same seed differ only in the weighting rule.

5 Experimental setup

Analytical objectives (synthetic, labelled as such). $L(\theta) = \frac{s}{2} \theta^\top H \theta$ with $H = R^\top \text{diag}(\kappa, 1, \dots, 1) R$ for a random rotation R (the upstream stress test), $d = 100$, $\kappa \in \{1, 10, 100, 1000\}$, loss scale $s \in \{10^{-4}, 10^{-2}, 1, 10^2, 10^4\}$. Noise model A adds i.i.d. Gaussian observation noise $s\sigma \xi$ to each evaluation (σ is a relative level, so SNR is scale-invariant). Noise model B is a linear least-squares problem with rows $x \sim \mathcal{N}(0, \Sigma)$, $\Sigma = H$, and mini-batches of 8 examples arranged as $B = 4$ micro-batches per step, so per-batch Hessians $H_b = X_b^\top X_b/8$ differ in composition; the $(+, 0, -)$ stencil shares a batch, as in the reference code. $P = 20$, $\varepsilon = 10^{-3}$, $\theta_0 \sim \mathcal{N}(0, I)$, budget 20,000 evaluations (80,000 for model B). The reported metric is the exact (noise-free) loss ratio $L(\theta_T)/L(\theta_0)$, an oracle diagnostic that is never used by the optimizers and never counted; a run is declared diverged when the ratio exceeds 10^3 and receives $\log_{10} = 6$ in averages.

Tuning protocol (frozen before the held-out runs; configs/quad_protocol*.json). Every method receives the same 41-point quarter-decade learning-rate grid $10^{-7} \dots 10^3$ at every condition; the grid is selected on three development seeds (Hessian rotation, noise and probe seeds all differ from evaluation seeds) by mean $\log_{10}$ final loss ratio, and the selected rate is run on five held-out seeds, paired across methods. Two protocols are compared: *retuned* (selection at every loss scale) and *transfer* (selection at $s = 1$ only, the same rate applied at all scales).

Neural task 1: MNIST MLP from scratch (CPU). A 784–64–10 MLP (50,890 parameters), cross-entropy times $s \in \{0.01, 1, 100\}$, two micro-batches of 128 per step, $P = 20$, $\varepsilon = 10^{-3}$, budget 10^5 forward passes. Train/validation/test = 50,000/10,000 (fixed index split of the MNIST training set) / 10,000 (MNIST test). Learning rates are tuned per method and loss scale on a development seed with a shared 17-point half-decade grid by final validation loss; three held-out seeds (initialization and batch order paired across methods; ten at $s = 1$ after round 1, Table 8) are then

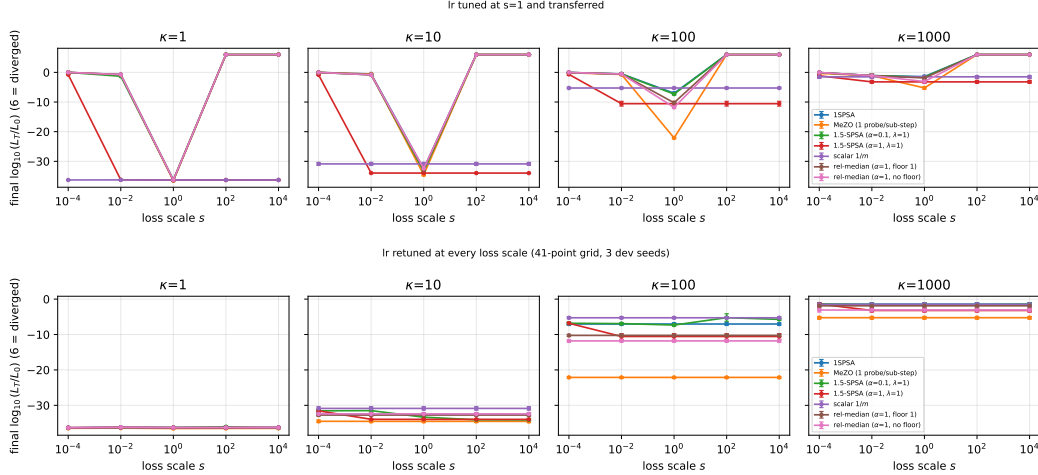


Figure 1: Final $\log_{10}(L_T/L_0)$ after 20,000 evaluations on the rotated stiff paraboloid versus loss scale s (5 held-out seeds, 95% bootstrap CIs; 6 marks divergence). **Top (transfer):** learning rate selected at $s = 1$ and reused. Every rule whose weights are $O(1)$ (1SPSA, MeZO, published 1.5-SPSA, relative-median) diverges for $s \geq 100$ and stalls for $s \leq 10^{-2}$; the Newton-unit scalar $1/m$, the relative weights $\times 1/m$ and 1.5-SPSA with $\alpha = 1$ (floor inactive) are exactly invariant, as predicted by Propositions 1–2. **Bottom (retuned):** with the rate retuned at every scale all curves are flat except where the published absolute floor $\lambda_{\text{reg}} = 1$ changes the active set (1.5-SPSA at $s \leq 10^{-2}$: at $s = 10^{-4}$ every $|\hat{c}|^\alpha$ is below the floor and the method is 1SPSA).

run with the selected rate; test accuracy is computed once on the final model of each held-out run. First-order baselines: SGD with momentum 0.9 and Adam at the same forward-equivalent budget.

Neural task 2: SST-2 with GPT-2 (124M) on an L4 GPU. Prompt-based fine-tuning as in MeZO: input “sentence It was”, label words “terrible”/“great”, loss = cross-entropy over the two label-word logits at the last position; all 124M parameters are perturbed. Train = 1,000 examples per class sampled from GLUE SST-2 train, dev = 500 further disjoint training examples (used for all selection; a construction artefact made it 76% positive, see Section 6.6; the corrected run uses 250 per class), test = the GLUE SST-2 validation split (872 examples; the GLUE test labels are not public), evaluated once per final run. Two micro-batches of 8, $P = 10$, $\varepsilon = 10^{-3}$, fp32, budget 33,600 micro-batch forward passes (≈ 800 steps). Four decade-spaced learning rates per method centred on each method’s natural unit (Sec. 4; $O(1)$ weights $\rightarrow 10^{-7} \dots 10^{-4}$, $1/m$ weights with $m \approx 10^3$ from the pilot $\rightarrow 10^{-4} \dots 10^{-1}$; Adam $10^{-7} \dots 10^{-4}$) are selected on a development seed by final dev loss; three held-out seeds follow. Peak GPU memory, optimizer state size and wall-clock are recorded per run. The reference repository selected checkpoints by *test* accuracy; we never do.

Statistics. Uncertainty is reported as 95% bootstrap confidence intervals over independent units (seeds); differences between methods are paired over seeds (common/stats.py). All numbers in tables and figures are produced by scripts/analyze_quad.py and scripts/analyze_neural.py from the raw per-run JSON records under results/.

6 Results

6.1 Quadratics: loss-scale sensitivity is a learning-rate phenomenon

Transfer. Figure 1 (top) and Table 2 confirm the derivation exactly: 1SPSA, MeZO, the published 1.5-SPSA ($\alpha = 0.1$) and all relative-median variants with $O(1)$ weights fail identically when the loss is rescaled by $10^{\pm 2}$ or more (10/25 runs diverged, the rest stall), because their step is proportional to s (or $s^{0.9}$). The three rules whose step is invariant by construction—scalar $1/m$, relative weights $\times 1/m$, and 1.5-SPSA with $\alpha = 1$ while its floor is inactive—reproduce the $s = 1$ result

Table 1: Retuned protocol, noise-free, $s = 1$: mean final $\log_{10}(L_T/L_0)$ over 5 held-out seeds [95% CI]. Last column: range over the five loss scales of the per-scale-retuned mean at $\kappa = 100$ (0.00 = exactly loss-scale invariant after retuning; non-zero values for the published rule come from the absolute floor, and for scalar $m^{-0.1}$ from the finite grid, see text).

method	$\kappa = 10, s = 1$	$\kappa = 100, s = 1$	$\kappa = 1000, s = 1$	$\max_s - \min_s (\kappa=100)$
1SPSA	-32.56 [-32.89,-32.15]	-7.03 [-7.37,-6.71]	-1.42 [-1.94,-0.99]	0.00
MeZO (1 probe/sub-step)	-34.53 [-34.89,-34.18]	-22.11 [-22.38,-21.89]	-5.25 [-5.67,-4.90]	0.00
1.5-SPSA ($\alpha=0.1, \lambda=1$)	-33.35 [-33.52,-33.10]	-7.32 [-7.65,-7.04]	-1.60 [-2.08,-1.12]	2.08
1.5-SPSA ($\alpha=1, \lambda=1$)	-33.96 [-34.13,-33.81]	-10.56 [-11.37,-9.75]	-3.20 [-3.71,-2.70]	3.74
scalar $m^{-0.1}$	-33.30 [-33.49,-33.07]	-6.39 [-6.89,-5.96]	-1.47 [-1.96,-1.02]	1.58
scalar $1/m$	-30.87 [-31.37,-30.37]	-5.27 [-5.59,-4.97]	-1.50 [-2.04,-1.11]	0.00
scalar $1/\text{med} \Delta\ell $	-3.94 [-4.13,-3.70]	-3.20 [-3.37,-2.95]	-2.13 [-2.72,-1.54]	0.07
rel-median ($\alpha=0.1$)	-31.53 [-31.93,-31.10]	-7.01 [-7.31,-6.79]	-1.56 [-2.06,-1.10]	0.00
rel-median ($\alpha=1$, floor 1)	-32.74 [-33.05,-32.36]	-10.25 [-10.59,-9.95]	-1.88 [-2.40,-1.42]	0.00
rel-median ($\alpha=1$, no floor)	-32.43 [-32.82,-32.06]	-11.81 [-12.16,-11.48]	-3.08 [-3.66,-2.66]	0.00
rel-median $\times 1/m$ ($\alpha=1$)	-33.99 [-34.12,-33.87]	-11.09 [-11.40,-10.78]	-2.44 [-2.72,-2.16]	0.00

Table 2: Transfer protocol at $\kappa = 100$: learning rate selected at $s = 1$, applied at every scale (mean final $\log_{10}$ ratio over 5 seeds; div. = all five runs diverged).

method	$s = 0.0001$	$s = 0.01$	$s = 1$	$s = 100$	$s = 10000$	diverged
1SPSA	-0.04	-0.51	-7.03	div.	div.	10/25
MeZO (1 probe/sub-step)	-0.12	-0.71	-22.11	div.	div.	10/25
1.5-SPSA ($\alpha=0.1, \lambda=1$)	-0.06	-0.56	-7.32	div.	div.	10/25
1.5-SPSA ($\alpha=1, \lambda=1$)	-0.67	-10.56	-10.56	-10.56	-10.56	0/25
scalar $m^{-0.1}$	-0.09	-0.56	-6.39	div.	div.	10/25
scalar $1/m$	-5.27	-5.27	-5.27	-5.27	-5.27	0/25
scalar $1/\text{med} \Delta\ell $	-3.13	-3.18	-3.20	-3.15	-3.13	0/25
rel-median ($\alpha=0.1$)	-0.03	-0.51	-7.01	div.	div.	10/25
rel-median ($\alpha=1$, floor 1)	-0.03	-0.54	-10.25	div.	div.	10/25
rel-median ($\alpha=1$, no floor)	-0.03	-0.56	-11.81	div.	div.	10/25
rel-median $\times 1/m$ ($\alpha=1$)	-11.09	-11.09	-11.09	-11.09	-11.09	0/25

to the last digit at every scale (0/25 divergences). The published rule with $\alpha = 1$ loses this property at $s = 10^{-4}$, where the absolute floor $\lambda_{\text{reg}} = 1$ becomes active (-0.67 versus -10.56). In other words, *loss-scale robustness is obtained by a scalar*, and the simplest scalar baseline (1SPSA weights divided by the median probe curvature) already has it.

Retuning removes the sensitivity for every rule except the absolute floor. With the learning rate retuned at each scale (Figure 1, bottom; Table 1) the per-scale means of 1SPSA, MeZO, scalar $1/m$ and all relative-median rules are identical across the five loss scales (range 0.00), as the selected rates scale exactly as $1/s$ on the quarter-decade grid. The published rule is the exception: its retuned performance at $\kappa = 100$ ranges over 2.1 ($\alpha = 0.1$) and 3.7 ($\alpha = 1$) orders of magnitude across scales because the floor silently turns curvature weighting off at small s . The non-zero range of scalar $m^{-0.1}$ (1.58) is a tuning-grid artefact that we flag rather than hide: its effective rate is $\eta m^{-0.1}$, which shifts the grid by a factor ≈ 0.6 relative to 1SPSA, and on this noise-free problem the final loss after 500 steps is extremely sensitive to the rate near the stability edge. This sets the resolution of the retuned comparison: differences below ≈ 1.5 in $\log_{10}$ between rules that are LR-equivalent should be read as grid effects.

Does the direction help? At $s = 1$ the published $\alpha = 0.1$ weighting is indistinguishable from 1SPSA at every condition number (e.g. -7.32 vs -7.03 at $\kappa = 100$, overlapping CIs), consistent with $|\hat{c}|^{0.1} \in [1.6, 2.3]$ producing at most a $1.4\times$ reweighting. Full probe-space Newton weights ($\alpha = 1$) do help on the noise-free rotated paraboloid: -10.6 (published floor), -10.3 (relative, floor at median) and -11.8 (relative, no floor) versus -7.0 for 1SPSA at $\kappa = 100$, and -3.1 to -3.2 versus -1.4 at $\kappa = 1000$. The scalar $1/m$ baseline is *worse* than 1SPSA (-5.3), so the gain of the relative rules over their own scalar baseline (-11.1 vs -5.3 for the invariant pair) is attributable to the relative weights, not to the adaptive step size. However, the line-searched one-step diagnostic (Figure 2) shows that the *per-step direction* of $\alpha = 1$ weights is only $\leq 2\%$ better than 1SPSA's;

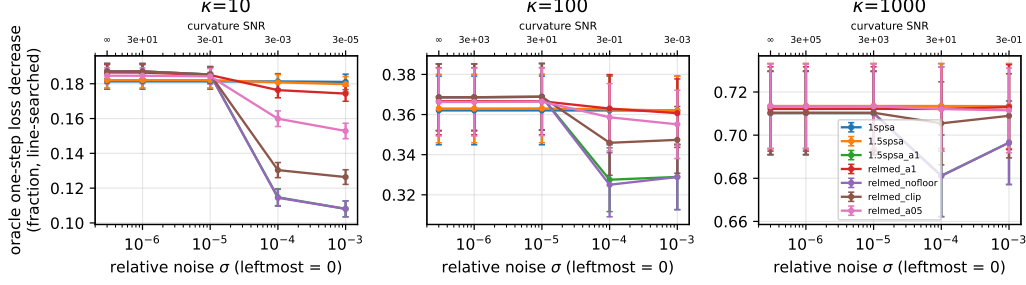


Figure 2: Oracle one-step diagnostic (200 random states, $P = 20$): fraction of the loss removed by an exact line search along the update direction of each rule. Curvature weights barely change the per-step direction when the curvature SNR is high and degrade it when $\text{SNR} < 1$ (top axis: $\text{Var}_z(z^\top H z)/(6\sigma^2/\varepsilon^4)$); a floor at the median or clipping limits the damage.

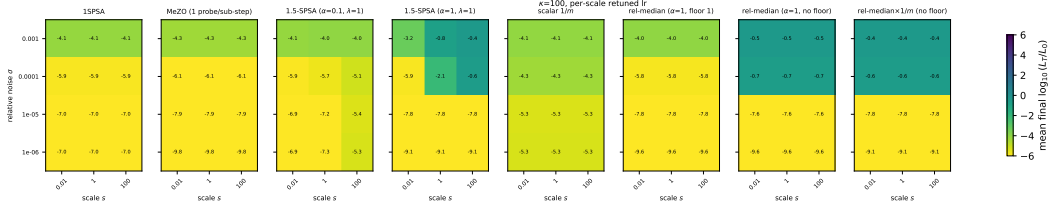


Figure 3: Loss-scale/noise phase diagram at $\kappa = 100$ with per-scale retuned learning rates (mean final $\log_{10}$ ratio over 5 seeds; numbers printed). Columns of a loss-scale-invariant rule are constant along s . The published rule changes behaviour along s at every noise level (its floor is in absolute loss units); $\alpha = 1$ weights without a floor collapse once the curvature SNR drops below one ($\sigma \geq 10^{-4}$); a relative floor restores 1SPSA-level robustness.

the multi-step benefit comes from stability—down-weighting probes aligned with the stiff direction admits a larger learning rate in the flat subspace—not from a better direction. Finally, at matched forward evaluations MeZO-style sequential single-probe updates dominate every P -probe averaging rule in the noise-free setting (-22.1 at $\kappa = 100$): twenty sequential steps use the same 40 evaluations more effectively than one averaged step. (On the $\kappa = 1$ problem all rules coincide, as the corollary predicts: $z^\top H z = \text{tr } H$ for every probe.)

6.2 Noise phase diagram: a floor in the wrong units

Figure 3 and Table 3 vary the relative observation noise. Three regimes appear. (i) At high curvature SNR ($\sigma \leq 10^{-5}$) relative $\alpha = 1$ weights keep their $\hat{2}$ – 2.6 order-of-magnitude advantage over 1SPSA. (ii) Once $\text{SNR} < 1$ ($\sigma \geq 10^{-4}$) weights $1/\hat{c}_i$ without a floor are driven by noisy near-zero curvatures and the optimizer stalls (-0.66 versus -5.86 for 1SPSA); the same happens to the published rule with $\alpha = 1$ (-2.08) because its floor $\lambda_{\text{reg}} = 1$ is far below the curvature scale $m \approx 150$ and therefore never active—the floor protects only when the loss happens to be measured in units where curvatures are $O(1)$. A floor at the median (-5.84), clipping to $[0.1, 10]$ (-5.58) or a floor at $2m$ (-5.87) restore 1SPSA-level behaviour, but none beats 1SPSA. (iii) At $\sigma = 10^{-3}$ every rule, including MeZO, converges to the same noise floor (≈ -4). The published $\alpha = 0.1$ rule tracks 1SPSA within CI throughout. Along the loss-scale axis the picture of Section 6.1 repeats under noise: only the published rule’s columns vary with s .

6.3 Batch-composition noise and the robustification variants

Table 4 uses the least-squares model, in which the stencil shares a batch (so batch noise cancels to first order in $\hat{c}$) and curvature estimates vary across micro-batches through H_b . The ordering is the same as before: the published rule equals 1SPSA (-3.55 vs -3.43 at $\kappa = 100$), relative $\alpha = 1$ weights gain 1.2 – 1.9 orders of magnitude (-4.6 to -5.3), and MeZO is best (-5.4). Among the variants proposed for this setting, none improves on the plainest relative rule: reliability shrinkage

Table 3: Additive observation noise at $\kappa = 100$, $s = 1$, retuned learning rates: mean final $\log_{10}$ ratio [95% CI]. The curvature SNR is 3.2×10^3 , 32, 0.32 and 3.2×10^{-3} for the four columns.

method	$\sigma = 1e-06$	$\sigma = 1e-05$	$\sigma = 0.0001$	$\sigma = 0.001$
1SPSA	-7.03 [-7.37,-6.71]	-6.96 [-7.30,-6.61]	-5.86 [-6.26,-5.50]	-4.12 [-4.48,-3.78]
MeZO (1 probe/sub-step)	-9.75 [-10.15,-9.40]	-7.95 [-8.30,-7.64]	-6.07 [-6.44,-5.72]	-4.27 [-4.61,-3.95]
1.5-SPSA ($\alpha=0.1$, $\lambda=1$)	-7.33 [-7.65,-7.04]	-7.23 [-7.54,-6.92]	-5.69 [-5.94,-5.45]	-4.00 [-4.37,-3.66]
1.5-SPSA ($\alpha=1$, $\lambda=1$)	-9.11 [-9.52,-8.79]	-7.81 [-8.23,-7.41]	-2.08 [-2.45,-1.69]	-0.82 [-1.14,-0.54]
scalar $1/m$	-5.27 [-5.59,-4.98]	-5.32 [-5.60,-5.03]	-4.27 [-4.60,-3.96]	-4.05 [-4.42,-3.70]
rel-median ($\alpha=1$, floor 1)	-9.60 [-9.97,-9.28]	-7.76 [-8.16,-7.39]	-5.84 [-6.20,-5.51]	-4.01 [-4.38,-3.66]
rel-median ($\alpha=1$, no floor)	-9.64 [-10.03,-9.27]	-7.63 [-8.04,-7.24]	-0.66 [-0.94,-0.40]	-0.47 [-0.84,-0.13]
rel-median $\times 1/m$ (no floor)	-9.11 [-9.52,-8.79]	-7.81 [-8.23,-7.41]	-0.64 [-0.85,-0.43]	-0.43 [-0.73,-0.14]
rel-median clip 10	-9.64 [-10.03,-9.27]	-7.63 [-8.04,-7.24]	-5.58 [-5.94,-5.23]	-3.72 [-4.06,-3.39]
rel-median floor 2	-6.37 [-6.69,-6.12]	-6.35 [-6.69,-6.09]	-5.87 [-6.23,-5.55]	-4.02 [-4.40,-3.67]

Table 4: Linear least squares with mini-batches of 8 (4 micro-batches per step, per-batch Hessians), 80,000 evaluations, retuned rates: mean final $\log_{10}$ ratio [95% CI] over 5 seeds.

method	$\kappa = 10$	$\kappa = 100$	$\kappa = 1000$
1SPSA	-8.24 [-8.55,-8.03]	-3.43 [-3.88,-2.98]	-1.33 [-1.89,-0.76]
MeZO (1 probe/sub-step)	-9.61 [-9.80,-9.45]	-5.40 [-5.77,-5.03]	-1.87 [-2.33,-1.42]
1.5-SPSA ($\alpha=0.1$, $\lambda=1$)	-8.24 [-8.47,-8.01]	-3.55 [-4.02,-3.07]	-1.36 [-1.92,-0.80]
1.5-SPSA ($\alpha=1$, $\lambda=1$)	-7.84 [-8.09,-7.58]	-5.34 [-5.86,-4.97]	-2.53 [-2.98,-2.07]
scalar $1/m$	-7.92 [-8.18,-7.69]	-3.43 [-3.82,-3.04]	-1.39 [-1.90,-0.87]
rel-median ($\alpha=1$, floor 1)	-7.78 [-8.09,-7.56]	-4.61 [-5.04,-4.18]	-1.43 [-1.95,-0.94]
rel-median ($\alpha=1$, no floor)	-7.88 [-8.11,-7.66]	-5.01 [-5.40,-4.67]	-1.97 [-2.51,-1.48]
rel-median shrink	-8.07 [-8.43,-7.81]	-4.79 [-5.27,-4.39]	-1.85 [-2.40,-1.33]
rel-median pooled	-7.95 [-8.20,-7.70]	-5.19 [-5.65,-4.86]	-0.49 [-2.54,+2.89] (1/5 div.)
rel-median per-batch	-8.03 [-8.44,-7.68]	-4.85 [-5.38,-4.51]	-2.08 [-2.60,-1.60]
rel-median clip 10	-7.88 [-8.11,-7.66]	-5.01 [-5.40,-4.67]	-1.97 [-2.51,-1.48]
rel-median clip 3	-7.89 [-8.11,-7.66]	-4.89 [-5.24,-4.61]	-1.74 [-2.24,-1.24]
rel-median ($\alpha=0.5$)	-8.06 [-8.39,-7.81]	-3.98 [-4.52,-3.49]	-1.56 [-2.07,-1.06]
rel-median floor 0.5	-7.92 [-8.15,-7.68]	-4.99 [-5.38,-4.66]	-1.68 [-2.20,-1.15]
rel-median floor 2	-7.71 [-7.99,-7.49]	-3.56 [-4.02,-3.06]	-1.40 [-1.92,-0.87]

(-4.79 ; the estimated reliability $\hat{\rho}$ is high because per-batch curvatures of a quadratic are consistent), per-batch normalization (-4.85), clipping (-5.01 , identical to no floor because no curvature exceeds $10m$), $\alpha = 0.5$ (-3.98) and a floor at $2m$ (-3.56 , which disables most weights). Pooled (EMA) normalization is the only variant that diverged (1/5 runs at $\kappa = 1000$), because a lagging scale estimate can mis-size a step after the curvature spectrum seen by the probes shifts. The extra noise sweep in this model (Appendix B, quad_E4b) reproduces the collapse of floor-less $\alpha = 1$ weights at $\sigma \geq 10^{-4}$.

6.4 A second Hessian family: the $\alpha = 1$ gain is a property of rank-one stiffness

Corollary 1 says that relative weights can only matter through the across-probe spread of $z^\top H z$ relative to $\text{tr } H$. That spread is unusually large for the upstream stress test ($\text{sd}/\text{tr } H = 0.69$ at $\kappa = 100$ and 1.26 at $\kappa = 1000$) and small for a log-uniform spectrum with the same condition number (0.16 and 0.22), so the derivation predicts that the $\alpha = 1$ gains of Sections 6.1–6.3 should largely disappear on the second family. Table 5 confirms this. On the log spectrum, noise-free, the relative $\alpha = 1$ rule with a median floor is *worse* than 1SPSA at $\kappa = 100$ ($+0.22$ [$+0.09$, $+0.33$] $\log_{10}$; it was -3.22 [-3.67 , -2.86] on the upstream family) and equal at $\kappa = 1000$ ($+0.01$ [-0.01 , $+0.03$]); without a floor it gains 0.13 [0.07 , 0.19] and 0.06 [0.02 , 0.09] $\log_{10}$, i.e. $1.3\times$ and $1.15\times$ lower loss instead of $10^{4.8\times}$. The published $\alpha = 0.1$ rule is again within a fraction of an order of magnitude of 1SPSA (worse by 0.72 at $\kappa = 100$, 0.10 at $\kappa = 1000$), and MeZO’s advantage shrinks from 15 to 0.4 – 0.5 orders of magnitude. The noise result transfers in the form the theory states, in terms of curvature SNR rather than σ : the large trace of the log spectrum raises the SNR at $\sigma = 10^{-4}$ to 2.1 ($\kappa = 100$) and 179 ($\kappa = 1000$), and no rule collapses there; at $\kappa = 10$ the SNR is

Table 5: Upstream Hessian family (“paper”: one stiff direction, $H = R^\top \text{diag}(\kappa, 1, \dots, 1)R$) versus a log-uniform spectrum (“log”: d eigenvalues log-uniform in $[1, \kappa]$, random rotation), $s = 1$, retuned rates, mean final $\log_{10}(L_T/L_0)$ over 5 held-out seeds [95% CI]. Last row: paired difference of the relative $\alpha = 1$ rule (median floor) to 1SPSA. The log-spectrum experiment (E6) was pre-registered in `configs/quad_protocol_E6.json` with the same grid, seeds and budget.

method	paper $\kappa=100$	noise-free log $\kappa=100$	log $\kappa=1000$	paper $\kappa=100$	$\sigma = 10^{-4}$ log $\kappa=100$	log $\kappa=1000$
1SPSA	-7.03 [-7.37,-6.71]	-6.98 [-7.08,-6.88]	-3.21 [-3.30,-3.13]	-5.86 [-6.26,-5.50]	-6.38 [-6.43,-6.33]	-3.21 [-3.30,-3.13]
MeZO (1 probe/sub-step)	-22.11 [-22.38,-21.89]	-7.48 [-7.60,-7.35]	-3.61 [-3.82,-3.44]	-6.07 [-6.44,-5.72]	-6.71 [-6.80,-6.63]	-3.61 [-3.82,-3.44]
1.5-SPSA ($\alpha=0.1$, $\lambda=1$)	-7.32 [-7.65,-7.04]	-6.26 [-6.44,-6.07]	-3.12 [-3.19,-3.06]	-5.69 [-5.94,-5.45]	-6.18 [-6.34,-6.01]	-3.12 [-3.19,-3.06]
1.5-SPSA ($\alpha=1$, $\lambda=1$)	-10.56 [-11.37,-9.75]	-6.40 [-6.59,-6.23]	-3.28 [-3.35,-3.20]	-2.08 [-2.45,-1.69]	-6.31 [-6.42,-6.19]	-3.28 [-3.36,-3.21]
scalar $1/m$	-5.27 [-5.59,-4.97]	-6.27 [-6.46,-6.09]	-3.08 [-3.16,-3.00]	-4.27 [-4.60,-3.96]	-6.19 [-6.36,-6.04]	-3.09 [-3.17,-3.00]
rel-median ($\alpha=1$, floor 1)	-10.25 [-10.59,-9.95]	-6.76 [-6.88,-6.63]	-3.20 [-3.28,-3.11]	-5.84 [-6.20,-5.51]	-6.46 [-6.57,-6.37]	-3.20 [-3.29,-3.11]
rel-median ($\alpha=1$, no floor)	-11.81 [-12.16,-11.48]	-7.10 [-7.23,-6.97]	-3.27 [-3.34,-3.19]	-0.66 [-0.94,-0.40]	-6.54 [-6.59,-6.47]	-3.27 [-3.34,-3.19]
rel-median clip 10	—	-7.10 [-7.23,-6.97]	-3.27 [-3.34,-3.19]	-5.58 [-5.94,-5.23]	-6.54 [-6.59,-6.47]	-3.27 [-3.34,-3.19]
Δ rel-median ($\alpha=1$, floor 1) – 1SPSA	-3.22 [-3.67,-2.86]	+0.22 [+0.09,+0.33]	+0.01 [-0.01,+0.03]	+0.02 [-0.12,+0.13]	-0.08 [-0.16,-0.03]	+0.01 [-0.01,+0.03]

Table 6: MNIST MLP (51k parameters, 10^5 forward passes, 3 held-out seeds, 95% bootstrap CIs; validation loss capped at 10 for averaging). Learning rate selected on a development seed at $s = 1$; “transfer” applies it unchanged at $s = 0.01$ and $s = 100$. State/model = optimizer-state bytes relative to model bytes.

method	lr ($s=1$)	dev loss, $s=1$	test acc, $s=1$	transfer $s=0.01$	transfer $s=100$	state/model
1SPSA	0.0316	0.411 [0.406,0.419]	0.888 [0.887,0.890]	2.241 [2.223,2.251]	div. 3/3	0
MeZO (1 probe/sub-step)	0.001	0.412 [0.411,0.414]	0.891 [0.890,0.892]	2.265 [2.249,2.274]	div. 3/3	0
1.5-SPSA ($\alpha=0.1$, $\lambda=1$)	0.0316	0.418 [0.413,0.424]	0.889 [0.887,0.892]	2.243 [2.225,2.252]	div. 3/3	0
1.5-SPSA ($\alpha=1$, $\lambda=1$)	1	0.522 [0.489,0.565]	0.851 [0.840,0.860]	0.610 [0.601,0.620]	div. 2/3	0
scalar $1/m$	3.16	0.451 [0.444,0.458]	0.875 [0.874,0.876]	0.442 [0.433,0.454]	0.437 [0.433,0.440]	0
rel-median ($\alpha=1$, floor 1)	0.0316	0.394 [0.386,0.399]	0.893 [0.890,0.898]	2.253 [2.235,2.263]	div. 3/3	0
rel-median ($\alpha=1$, no floor)	0.01	div. 3/3	0.098 [0.098,0.098]	div. 3/3	div. 3/3	0
rel-median $\times 1/m$ (no floor)	0.0001	div. 3/3	0.098 [0.098,0.098]	div. 3/3	div. 3/3	0
rel-median shrink	0.0316	0.399 [0.385,0.411]	0.888 [0.882,0.895]	2.242 [2.223,2.252]	div. 3/3	0
Adam (first-order)	0.000316	0.110 [0.108,0.111]	0.973 [0.972,0.974]	0.110 [0.109,0.111]	0.110 [0.108,0.111]	$2 \times$
SGD+mom. (first-order)	0.01	0.098 [0.096,0.100]	0.975 [0.973,0.977]	0.454 [0.450,0.458]	2.031 [1.795,2.302]	$1 \times$

0.02 and the floor-less relative rule (-0.31) and the published $\alpha = 1$ rule (-1.22) collapse while the median-floored rule matches 1SPSA (-6.18 versus -6.19). The positive $\alpha = 1$ result on quadratics is therefore specific to Hessians whose stiffness is concentrated in a few directions, and the paper’s conclusions about relative normalization (redundant for scale; a relative floor is what makes strong weighting safe) hold on both families.

6.5 Neural task 1: MNIST MLP

Tables 6 and 7 and Figure 4 reproduce every prediction of Section 3 on a non-quadratic objective. *Transfer*: with the rate selected at $s = 1$, 1SPSA, MeZO, the published 1.5-SPSA, rel-median and shrinkage all stall at $s = 0.01$ (validation loss ≈ 2.24 , i.e. barely below the initial 2.30) and all diverge at $s = 100$ (3/3 seeds); the scalar $1/m$ rule gives the same result at every scale up to fp32 round-off (0.44–0.45), as does Adam, whose own normalization makes it scale-invariant, whereas SGD is not. The published rule with $\alpha = 1$ shows the floor effect again: at $s = 0.01$ the transferred run is worse (0.61) but does not stall, because the absolute floor becomes active and the effective step grows. *Retuning* restores exact invariance for every rule whose weights are scale-free (1SPSA 0.411 at all three scales with rates 3.16, 0.0316, 3.16×10^{-4} ; rel-median 0.394 at all three), while the published rule’s retuned loss varies from 0.412 to 0.449 and its $\alpha = 1$ form from 0.404 to divergence (2/3 seeds at $s = 100$, where the dev seed selected a rate that the held-out seeds could not tolerate).

Does the direction help here? The relative rule with $\alpha = 1$ and a floor at the median is the only ZO variant that beats 1SPSA on this task, by a small but consistent margin. With the three seeds of the original protocol the paired validation-loss difference was -0.016 $[-0.021, -0.008]$ at every scale; because three paired units give a coarse, anti-conservative percentile interval, we added seven held-out seeds at $s = 1$ at the frozen learning rates (Table 8). With $n = 10$ the difference shrinks by a third but remains: -0.0106 $[-0.0147, -0.0064]$ (paired- t interval $[-0.0155, -0.0056]$), better on 9/10 seeds, and test accuracy $+0.42$ $[+0.30, +0.54]$ points. The same ten seeds put shrinkage at -0.0054 $[-0.0109, -0.0006]$ (paired- t interval includes zero; better on 6/10 seeds) and the published 1.5-

Table 7: MNIST MLP, learning rate retuned at every loss scale (selected rate in parentheses): final validation loss [95% CI] over 3 held-out seeds.

method	$s=0.01$ (lr)	$s=1$ (lr)	$s=100$ (lr)
1SPSA	0.411 [0.406,0.419] (3.16)	0.411 [0.406,0.419] (0.0316)	0.411 [0.406,0.419] (0.000316)
MeZO (1 probe/sub-step)	0.412 [0.411,0.414] (0.1)	0.412 [0.411,0.414] (0.001)	0.412 [0.411,0.414] (1e-05)
1.5-SPSA ($\alpha=0.1$, $\lambda=1$)	0.412 [0.406,0.421] (3.16)	0.418 [0.413,0.424] (0.0316)	0.449 [0.433,0.475] (0.001)
1.5-SPSA ($\alpha=1$, $\lambda=1$)	0.404 [0.399,0.411] (3.16)	0.522 [0.489,0.565] (1)	div. 2/3 (1)
scalar $1/m$	0.442 [0.433,0.454] (3.16)	0.451 [0.444,0.458] (3.16)	0.437 [0.433,0.440] (3.16)
rel-median ($\alpha=1$, floor 1)	0.394 [0.387,0.399] (3.16)	0.394 [0.386,0.399] (0.0316)	0.394 [0.385,0.399] (0.000316)
rel-median ($\alpha=1$, no floor)	div. 3/3 (0.0001)	div. 3/3 (0.01)	div. 1/3 (0.0001)
rel-median $\times 1/m$ (no floor)	div. 1/3 (0.316)	div. 3/3 (0.0001)	div. 3/3 (0.316)
rel-median shrink	0.402 [0.384,0.418] (3.16)	0.399 [0.385,0.411] (0.0316)	0.399 [0.384,0.411] (0.000316)
Adam (first-order)	0.110 [0.109,0.111] (0.000316)	0.110 [0.108,0.111] (0.000316)	0.110 [0.108,0.111] (0.000316)
SGD+mom. (first-order)	0.098 [0.096,0.100] (1)	0.098 [0.096,0.100] (0.01)	0.098 [0.097,0.100] (0.0001)

Table 8: MNIST MLP at $s = 1$ with 10 held-out seeds (seeds 0–2 from the original protocol, 3–9 added in round 1 at the frozen validation-selected rates): final validation loss [95% bootstrap CI], test accuracy, and the paired difference to 1SPSA (bootstrap and paired- t 95% intervals; “better seeds” = seeds with lower validation loss than 1SPSA on the same initialization and batch order).

method	val loss	test acc	Δ val loss vs 1SPSA	paired- t CI	better seeds	Δ test acc vs 1SPSA
1SPSA	0.413 [0.409,0.417]	0.8858	—	—	—	—
MeZO (1 probe/sub-step)	0.412 [0.409,0.415]	0.8905	-0.001 [-0.006,+0.005]	[-0.007,+0.006]	4/10	+0.47 pt
1.5-SPSA ($\alpha=0.1$, $\lambda=1$)	0.418 [0.415,0.421]	0.8880	+0.005 [+0.002,+0.009]	[+0.001,+0.010]	2/10	+0.22 pt
1.5-SPSA ($\alpha=1$, $\lambda=1$)	0.522 [0.500,0.545]	0.8521	+0.109 [+0.086,+0.134]	[+0.079,+0.139]	0/10	-3.37 pt
scalar $1/m$	0.446 [0.440,0.453]	0.8750	+0.034 [+0.027,+0.040]	[+0.026,+0.041]	0/10	-1.08 pt
rel-median ($\alpha=1$, floor 1)	0.402 [0.397,0.406]	0.8900	-0.011 [-0.015,-0.006]	[-0.016,-0.006]	9/10	+0.42 pt
rel-median ($\alpha=1$, no floor)	div. 8/10	0.2237	—	—	—	—
rel-median $\times 1/m$ (no floor)	div. 10/10	0.0980	—	—	—	—
rel-median shrink	0.407 [0.401,0.414]	0.8862	-0.005 [-0.011,-0.001]	[-0.012,+0.001]	6/10	+0.04 pt
Adam (first-order)	0.110 [0.109,0.111]	0.9728	-0.303 [-0.307,-0.299]	[-0.308,-0.298]	10/10	+8.69 pt
SGD+mom. (first-order)	0.099 [0.097,0.101]	0.9739	-0.314 [-0.318,-0.309]	[-0.319,-0.308]	10/10	+8.80 pt

SPSA at $+0.0054$ [$+0.0016$, $+0.0091$] in validation loss, i.e. slightly worse than 1SPSA, while its test accuracy is $+0.22$ [$+0.07$, $+0.38$] points higher; we read the published rule as indistinguishable from 1SPSA in practice. The same relative weights *without* a floor diverge for every learning rate on the grid (3/3 seeds at every scale, including after the boundary extension; 8/10 of the ten seeds at $s = 1$): on a neural loss the three-point curvature estimate of some probe is always close to zero, and $1/|\hat{c}_i|$ then produces an unbounded step. Reliability shrinkage without a floor does not diverge (0.399; $\hat{\rho}$ pulls unreliable curvatures towards the median and acts as a soft floor) but is at most marginally better than 1SPSA (three seeds: -0.011 [-0.021 , $+0.004$]; ten seeds: above). The published $\alpha = 0.1$ rule is indistinguishable from 1SPSA at $s = 0.01$ (where it *is* 1SPSA) and slightly worse at $s \geq 1$ ($+0.005$ at $s = 1$ with ten seeds, $+0.038$ at $s = 100$ with three). The scalar $1/m$ baseline is 0.03–0.04 worse than 1SPSA, so the rel-median gain is attributable to the relative weights rather than to the adaptive scalar. MeZO equals 1SPSA here (unlike on quadratics: sequential single-probe steps gain nothing once per-evaluation noise dominates). All ZO rules keep zero optimizer state; the first-order references reach 0.10–0.11 validation loss at the same forward-equivalent budget with $1\text{--}2\times$ model-sized state.

Batch composition (micro-batch 32 / 128 / 512). The research question asks about batch composition as well as loss scale, and the quadratic batch model (Section 6.3) could not exercise shrinkage because its per-batch curvatures are too reliable. Table 9 varies the micro-batch size on the MLP with per-setting retuning. The relative $\alpha = 1$ gain is specific to the protocol batch size: -0.013 [-0.019 , -0.007] (5/5 seeds) at 128, but $+0.036$ [$+0.032$, $+0.039$] (0/5) at 32 and $+0.009$ [$+0.007$, $+0.011$] (0/5) at 512. Reliability shrinkage, whose intended regime is small, noisy batches, is within ± 0.008 of 1SPSA at every size and slightly worse at 32 ($+0.004$ [$+0.002$, $+0.005$]). At 512 the published 1.5-SPSA is better than 1SPSA (-0.016 [-0.020 , -0.011], 5/5), but its own scalar baseline at the same rate obtains -0.013 [-0.018 , -0.007]: 1SPSA’s optimum lies between two grid points, and the $m^{-0.1}$ scalar shifts the published rule’s effective rate into that gap (the grid-offset effect of Section 6.1). The direction component of the published rule—1.5-SPSA minus its scalar baseline, paired—is -0.0065 [-0.0111 , -0.0016], $+0.0035$ [$+0.0001$, $+0.0071$] and -0.0031 [-0.0041 , -0.0017] at 32, 128 and 512 (reports/tables/mlp_batch.md): consistent

MNIST MLP: validation loss vs compute ($s=1$, mean over seeds)

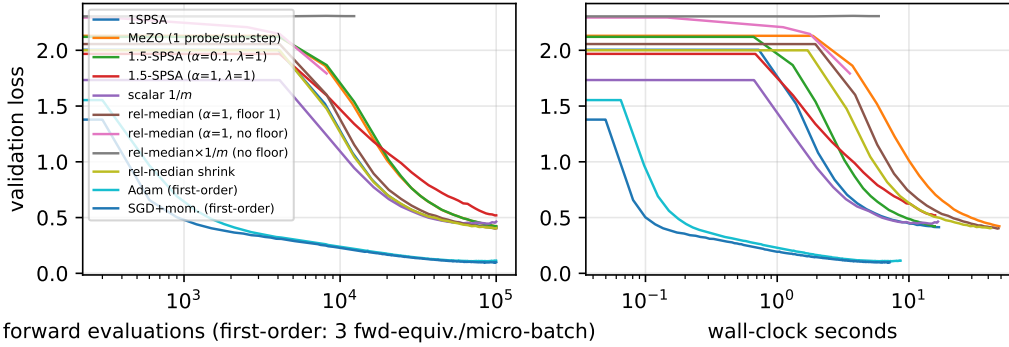


Figure 4: MNIST MLP at $s = 1$: validation loss versus forward evaluations (left) and wall-clock (right; CPU, shared machine—indicative only), mean over 3 seeds at the validation-selected learning rate.

Table 9: MNIST MLP, $s = 1$, micro-batch size varied at the same 10^5 forward passes (2 micro-batches per step), learning rate re-tuned per method and batch size on the development seed over the same 7-point half-decade grid, 5 held-out seeds. First row: 1SPSA validation loss [95% CI]; other rows: paired difference to 1SPSA (seeds better than 1SPSA; selected rate). Scalar $m^{-0.1}$ is the scalar baseline of the published rule (same per-step scalar, $r \equiv 1$) run at the published rule’s selected rate.

method	micro-batch 32	micro-batch 128 (protocol)	micro-batch 512
1SPSA	0.539 [0.530,0.549] (lr 0.01)	0.410 [0.404,0.416] (lr 0.0316)	0.353 [0.350,0.356] (lr 0.0316)
1.5-SPSA ($\alpha=0.1, \lambda=1$)	-0.000 [-0.019,+0.022] (3/5; lr 0.0316)	+0.006 [+0.001,+0.010] (1/5; lr 0.0316)	-0.016 [-0.020,-0.011] (5/5; lr 0.1)
scalar $m^{-0.1}$	+0.006 [-0.017,+0.033] (3/5; lr 0.0316)	+0.003 [-0.003,+0.008] (2/5; lr 0.0316)	-0.013 [-0.018,-0.007] (5/5; lr 0.1)
rel-median ($\alpha=1$, floor 1)	+0.036 [+0.032,+0.039] (0/5; lr 0.01)	-0.013 [-0.019,-0.007] (5/5; lr 0.0316)	+0.009 [+0.007,+0.011] (0/5; lr 0.0316)
rel-median shrink	+0.004 [+0.002,+0.005] (0/5; lr 0.01)	-0.008 [-0.017,+0.000] (3/5; lr 0.0316)	-0.000 [-0.002,+0.002] (4/5; lr 0.0316)

in magnitude (≤ 0.007 , under 2% of the loss) and not in sign. No normalization variant gives a batch-size-robust improvement over 1SPSA.

6.6 Neural task 2: SST-2 with GPT-2 (124M)

Table 10 and Figure 5 summarize the language-model task. Three observations are robust; a fourth is a protocol flaw we found after the runs and report rather than hide.

(i) *Loss-scale transfer behaves exactly as derived.* With the rate selected at $s = 1$, the scalar $1/m$ rule gives the same dev loss at $s \in \{0.01, 1, 100\}$ to four digits (0.4812, 0.4811, 0.4812) and so does Adam (0.3242, 0.3239–0.3241, 0.3241), whereas every rule with $O(1)$ weights changes with s : at $s = 100$ the published 1.5-SPSA diverges (its step grows as $s^{0.9}$), rel-median with shrinkage reaches 5.17, rel-median $\alpha = 1$ 1.17, MeZO 1.06 and 1SPSA 0.62, all with test accuracy at or below chance (0.47–0.51). At $s = 0.01$ these rules behave like a $100\times$ smaller learning rate. This reproduces the quadratic and MLP transfer results on a 124M-parameter pretrained model.

(ii) *Zero-order rules are indistinguishable from each other on dev loss at this budget.* The ZO dev-loss trajectories fluctuate by ± 0.05 –0.1 between evaluations (Figure 5, left) and the held-out-seed spread of 1SPSA’s final dev loss is 0.49–0.63. Paired over seeds against 1SPSA, the dev-loss differences of the published 1.5-SPSA (-0.031 [$-0.157, +0.052$]), MeZO ($+0.004$ [$-0.003, +0.009$]) and rel-median $\alpha = 1$ without a floor (-0.013 [$-0.124, +0.107$]) all include zero. Rel-median with shrinkage (-0.105 [$-0.150, -0.020$]) and scalar $1/m$ (-0.097 [$-0.145, -0.010$]) have intervals that exclude zero, but see (iv) for why lower dev loss is not a reliable signal here.

(iii) *On held-out test accuracy every ZO rule improves on the zero-shot model, and none beats 1SPSA reliably.* The untrained model predicts “great” for 99.3% of test sentences (test accuracy 0.516). After 800 ZO steps, 1SPSA reaches 0.766 [0.702, 0.802], MeZO 0.773, rel-median with shrinkage 0.804 [0.776, 0.822] (paired difference to 1SPSA $+0.038$ [$-0.018, +0.120$]), the published 1.5-

Table 10: SST-2, GPT-2 (124M), prompt-based, 33,600 micro-batch forward passes (≈ 800 ZO steps), fp32, one L4 per run. Learning rate selected by final dev loss on a development seed from four decade-spaced values; dev loss and test accuracy over 3 held-out seeds at $s = 1$ [95% bootstrap CI; with $n = 3$ these intervals are coarse]; “transfer” = the $s = 1$ rate applied at $s = 0.01$ and $s = 100$ (seed 0 only). Test = GLUE SST-2 validation (872), evaluated once per run. The zero-shot row is the untrained model (dev parity with the GPU runs: 0.5179 on both).

method	lr ($s=1$)	dev loss, $s=1$	test acc, $s=1$	transfer $s=0.01$	transfer $s=100$	state/model	peak GiB	s/run
zero-shot (no training)	—	0.518	0.516	—	—	0	—	—
1SPSA	1e-05	0.578 [0.491,0.627]	0.766 [0.702,0.802]	0.556 [0.556,0.556]	0.618 [0.618,0.618]	0	1.17	1001
MeZO (1 probe/sub-step)	1e-06	0.581 [0.497,0.624]	0.773 [0.727,0.805]	0.554 [0.554,0.554]	1.064 [1.064,1.064]	0	1.17	808
1.5-SPSA ($\alpha=0.1$, $\lambda=1$)	0.0001	0.547 [0.470,0.628]	0.726 [0.697,0.757]	0.585 [0.585,0.585]	div. 1/1	0	1.17	1003
scalar $1/m$	0.0001	0.481 [0.481,0.481]	0.525 [0.525,0.525]	0.481 [0.481,0.481]	0.481 [0.481,0.481]	0	1.17	972
rel-median ($\alpha=1$, no floor)	1e-07	0.565 [0.503,0.598]	0.666 [0.658,0.673]	0.515 [0.515,0.515]	1.169 [1.169,1.169]	0	1.17	995
rel-median shrink	1e-05	0.473 [0.465,0.482]	0.804 [0.776,0.822]	0.503 [0.503,0.503]	5.173 [5.173,5.173]	0	1.17	982
Adam (first-order)	1e-07	0.324 [0.323,0.324]	0.873 [0.873,0.874]	0.324 [0.324,0.324]	0.324 [0.324,0.324]	2×	2.60	790

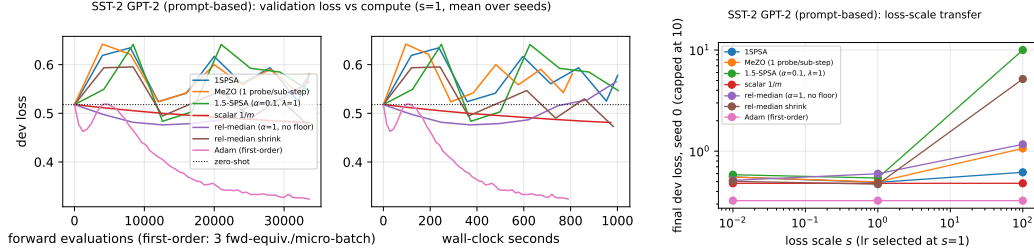


Figure 5: SST-2/GPT-2. **Left:** dev loss versus forward evaluations and wall-clock at $s = 1$ (mean of 3 seeds; dotted: zero-shot). ZO trajectories are dominated by step-to-step noise of ± 0.05 – 0.1 at this budget, while Adam descends steadily. **Right:** final dev loss of seed 0 when the $s = 1$ rate is reused at $s = 0.01$ and $s = 100$: the two scale-invariant rules (scalar $1/m$, Adam) are flat to four digits; every $O(1)$ -weight rule degrades at $s = 100$ and the published 1.5-SPSA diverges.

SPSA 0.726 (-0.040 [-0.070 , -0.005]) and rel-median $\alpha = 1$ without a floor 0.666 (-0.100 [-0.137 , -0.029]); the floor-less $\alpha = 1$ rule is also the one that degraded sharply (dev loss 3.4) or diverged at every rate $\geq 10^{-5}$ during tuning, as on the MLP. Adam at the same forward-equivalent budget reaches 0.873 [0.873, 0.874] at $2\times$ model-sized optimizer state and $2.2\times$ the peak GPU memory (2.60 versus 1.17 GiB for every ZO rule, which store no model-sized tensor).

(iv) *Protocol flaw: the dev split is label-imbalanced.* The dev split was built from the first 500 unused examples of the shuffled SST-2 training order after drawing 1,000 examples per class; because positives fill their quota first, the skipped examples are positives and the dev split is 76.4% positive (the test split is 50.9%). The zero-shot model, which almost always answers “great”, therefore has a deceptively good dev loss (0.518, accuracy 0.766), and selection by dev loss rewards rules and learning rates that stay close to it. The clearest case is scalar $1/m$: the median curvature on this task is $m \approx 4 \times 10^3$ – 3×10^4 , an order of magnitude above the pilot estimate used to centre its grid, so its selected rate (the grid minimum) barely moves the model; it attains the lowest ZO dev loss while its test accuracy (0.525, identical across seeds) is essentially zero-shot. The test split was never used for any decision, so the test numbers above remain unbiased estimates for the selected configurations, but the selection itself is weaker than intended. We therefore re-ran the tuning and held-out stages with a corrected split; see (v).

(v) *Corrected run with a balanced dev split.* With user approval of additional compute we re-ran the complete tuning stage (32 runs) and the held-out stage (21 runs) with a balanced dev split (Table 11); the transfer stage was not repeated. On the balanced split the zero-shot model’s dev loss is 0.958 instead of 0.518, so dev loss now penalizes the always-“great” answer, and every ZO rule improves on it. Four selections are unchanged (1SPSA 10^{-5} , MeZO 10^{-6} , rel-median shrinkage 10^{-5} , Adam 10^{-7}), and because training never touches the dev split these rules reproduce their earlier test accuracies to the last digit, a determinism check across two separate launches. Three selections move: the published 1.5-SPSA from 10^{-4} (grid maximum) to 10^{-5} , rel-median $\alpha = 1$ without a floor from 10^{-7} (grid minimum) to 10^{-6} , and scalar $1/m$ from 10^{-4} (grid minimum) to 10^{-1} (now the grid maximum). With the corrected selection the scalar $1/m$ rule trains normally (test 0.765 [0.761,

Table 11: Corrected SST-2/GPT-2 run (v7): the same frozen LM protocol re-run with a label-balanced dev split (250 examples per class from the unused training pool, exact-duplicate sentences of train/test excluded; dev_mode="balanced"). Learning rates re-selected by final dev loss on the development seed; 3 held-out seeds at $s = 1$; test split unchanged. Train/test data, seeds and batch order are identical to Table 10, so a rule whose selected rate did not change reproduces its earlier test accuracy exactly.

method	lr ($s=1$)	dev loss, $s=1$	test acc, $s=1$	state/model	peak GiB	s/run
zero-shot (no training)	—	0.958	0.516	0	—	—
1SPSA	1e-05	0.558 [0.546,0.571]	0.766 [0.702,0.802]	0	1.17	994
MeZO (1 probe/sub-step)	1e-06	0.557 [0.553,0.561]	0.773 [0.727,0.805]	0	1.17	849
1.5-SPSA ($\alpha=0.1$, $\lambda=1$)	1e-05	0.561 [0.558,0.563]	0.757 [0.750,0.763]	0	1.17	977
scalar $1/m$	0.1	0.549 [0.538,0.558]	0.765 [0.761,0.768]	0	1.17	956
rel-median ($\alpha=1$, no floor)	1e-06	0.570 [0.536,0.632]	0.753 [0.677,0.799]	0	1.17	965
rel-median shrink	1e-05	0.465 [0.460,0.473]	0.804 [0.776,0.822]	0	1.17	969
Adam (first-order)	1e-07	0.331 [0.330,0.332]	0.873 [0.873,0.874]	$2\times$	2.60	775

Table 12: Pre-stated success criterion (“ $\geq 20\%$ fewer forward evaluations than 1SPSA to a fixed target, or fewer failures to reach it”), evaluated on the held-out curves at $s = 1$: number of (condition, target) pairs where the criterion is met, out of the pairs where either rule reached the target (targets $L_T/L_0 \leq 10^{-2}, 10^{-4}, 10^{-6}$; “worse” = fewer seeds reach the target or the evaluation ratio’s lower CI exceeds 1.25). Per-pair ratios with CIs: reports/tables/e2t_quad.md.

method	noise-free, retuned	additive noise, retuned	batch model, retuned	log spectrum, retuned
MeZO (1 probe/sub-step)	5/12 (0 worse)	5/11 (0 worse)	2/7 (0 worse)	3/21 (3 worse)
1.5-SPSA ($\alpha=0.1$, $\lambda=1$)	0/10 (0 worse)	1/11 (0 worse)	0/6 (0 worse)	0/21 (3 worse)
1.5-SPSA ($\alpha=1$, $\lambda=1$)	3/10 (0 worse)	3/11 (5 worse)	3/7 (0 worse)	4/21 (3 worse)
scalar $1/m$	0/10 (1 worse)	0/11 (5 worse)	0/6 (0 worse)	2/21 (6 worse)
rel-median ($\alpha=0.1$)	0/10 (0 worse)	—	—	—
rel-median ($\alpha=1$, floor 1)	3/10 (0 worse)	6/11 (0 worse)	2/6 (0 worse)	0/21 (0 worse)
rel-median ($\alpha=1$, no floor)	4/11 (0 worse)	6/11 (5 worse)	2/6 (0 worse)	0/21 (3 worse)
rel-median $\times 1/m$ ($\alpha=1$)	3/10 (0 worse)	—	—	—
rel-median clip 10	—	6/11 (0 worse)	2/6 (0 worse)	0/21 (3 worse)
rel-median shrink	—	—	2/6 (0 worse)	—
rel-median per-batch	—	—	2/6 (0 worse)	—
rel-median pooled	—	—	2/7 (0 worse)	—

0.768] instead of the zero-shot-level 0.525), and every ZO rule lands within seed noise of 1SPSA on test accuracy: paired differences -0.009 [-0.052 , $+0.056$] (published 1.5-SPSA; previously -0.040 [-0.070 , -0.005], which was therefore a selection artefact), $+0.007$ [-0.008 , $+0.025$] (MeZO), -0.002 [-0.040 , $+0.067$] (scalar $1/m$), -0.013 [-0.125 , $+0.098$] (rel-median $\alpha = 1$ without a floor) and $+0.038$ [-0.018 , $+0.120$] (shrinkage). On the balanced dev split shrinkage has clearly the lowest ZO dev loss (0.465 versus 0.558 for 1SPSA; paired -0.092 [-0.108 , -0.084]) and the highest test accuracy (0.804 [0.776, 0.822]), but with three seeds its test advantage is not resolved. Adam remains far ahead (0.873, dev loss 0.331) at $2.2\times$ the peak memory. Scalar $1/m$ and Adam now sit on grid boundaries; the LM protocol has no boundary-extension rule and we did not add one after the fact.

6.7 The success criterion measured directly: evaluations to target and unstable runs

The brief for this project set a concrete bar: fewer unstable runs, or at least 20% fewer function evaluations to a fixed held-out target, across several loss scales. We measured it directly on the held-out curves (scripts/analyze_e2t.py; no re-selection), pairing each rule with 1SPSA on the same seed and treating runs that never reach the target as censored; the criterion was fixed before computing it (reports/reviews/round_3.md).

Unstable runs (Table 13). With per-scale retuning, essentially no rule diverges (0 of 100–190 held-out runs for every core rule; pooled normalization 1 of 30). With a single transferred learning rate every $O(1)$ -weight rule—1SPSA, MeZO, the published 1.5-SPSA and rel-median—diverges in 40 of 100 held-out runs, and every rule with a $1/m$ factor or with $\alpha = 1$ weights above the floor diverges in 0 of 100 (upper bound 3%). The “fewer unstable runs across loss scales” half of the

Table 13: Unstable (diverged) held-out runs on quadratics, pooled over all conditions of E1–E4b; in parentheses the Clopper–Pearson 95% upper bound on the divergence probability. Transfer = learning rate selected at $s = 1$, applied at $s \in \{10^{-4}, 10^{-2}, 1, 10^2, 10^4\}$.

method	retuned: diverged/runs	transfer: diverged/runs
1.5-SPSA ($\alpha=0.1, \lambda=1$)	0/190 (≤ 0.02)	40/100 (≤ 0.49)
1SPSA	0/190 (≤ 0.02)	40/100 (≤ 0.49)
MeZO (1 probe/sub-step)	0/175 (≤ 0.02)	40/100 (≤ 0.49)
1.5-SPSA ($\alpha=1, \lambda=1$)	0/190 (≤ 0.02)	0/100 (≤ 0.03)
rel-median ($\alpha=0.1$)	0/100 (≤ 0.03)	40/100 (≤ 0.49)
rel-median ($\alpha=1$, floor 1)	0/175 (≤ 0.02)	40/100 (≤ 0.49)
rel-median $\times 1/m$ ($\alpha=1$)	0/100 (≤ 0.03)	0/100 (≤ 0.03)
rel-median ($\alpha=1$, no floor)	0/190 (≤ 0.02)	40/100 (≤ 0.49)
rel-median pooled	1/30 (≤ 0.15)	–
scalar $1/\text{med} \Delta\ell $	0/100 (≤ 0.03)	0/100 (≤ 0.03)
scalar $1/m$	0/190 (≤ 0.02)	0/100 (≤ 0.03)
scalar $m^{-0.1}$	0/100 (≤ 0.03)	40/100 (≤ 0.49)

bar is therefore met, but it is met equally by the pure scalar $1/m$ baseline, i.e. by a learning-rate normalization; it is not a curvature-handling effect.

Evaluations to target (Table 12). The published 1.5-SPSA meets the bar in 1 of 48 (condition, target) pairs; its typical ratio is 0.95 [0.93, 0.97] at $\kappa = 100$ (5% fewer evaluations). Relative $\alpha = 1$ weights meet it on the upstream family at $\kappa = 100$ for every target when noise is low (ratio 0.65 [0.57, 0.73] to 10^{-2} ; 0.57–0.59 without a floor) and at $\kappa = 1000$ by reaching the target on more seeds; they never meet it on the log-spectrum family (0 of 21 pairs), consistent with Section 6.4. MeZO meets it most often on quadratics (ratio 0.30 at $\kappa = 100$, noise-free). On the MNIST MLP no zero-order rule reaches validation-loss targets 1.0, 0.6 or 0.45 with fewer evaluations than 1SPSA except scalar $1/m$ at the loosest target (0.77 [0.77, 0.77]); the relative $\alpha = 1$ rule, which ends at a lower validation loss, needs 1.02 – $1.10\times$ the evaluations to reach each target, and the published rule 1.40 – $1.55\times$ (reports/tables/e2t_mlp.md; curves are logged every 50 steps, so ratios are resolved to $\approx 2.5\%$). In short, the evaluation-count half of the bar is met only on one quadratic family by strong ($\alpha = 1$) weighting, and the stability half only by a scalar.

7 Ablations and diagnostics

Remove robust scaling / change the floor / clip. Tables 3 and 4 ablate the components of the relative rule individually: $\alpha \in \{0.1, 0.5, 1\}$, floor $\lambda \in \{0, 0.5, 1, 2\}$ (in units of the median), clipping $C \in \{3, 10\}$, reliability shrinkage, and per-step/per-batch/pooled scopes. Only α matters in the noise-free and batch-noise settings; the floor and clipping matter only under observation noise, where they act as safety devices (recovering 1SPSA behaviour) rather than as improvements. Shrinkage never changed the outcome measurably, because on quadratics the within-probe variance of per-batch curvatures is small relative to the across-probe spread ($\hat{\rho} \rightarrow 1$); its intended regime (unreliable curvature estimates) coincides with the regime where a floor is already needed. Pooled normalization is the only variant that introduced instability.

Normalize only the scalar. The scalar-only rules (SCALAR-POW, SCALAR-INV, SCALAR- $\Delta\ell$) isolate the learning-rate component. SCALAR-POW is 1SPSA up to the grid (Table 1); SCALAR-INV buys exact loss-scale invariance at a cost of 1–2 orders of magnitude of final loss on noise-free quadratics (its step shrinks whenever the sampled curvatures happen to be large) and no cost under noise (-4.05 vs -4.12 at $\sigma = 10^{-3}$); SCALAR- $\Delta\ell$ (normalized gradient, no centre evaluation) is scale-invariant but cannot converge below a fixed step-size floor (-3.2 to -4.2).

Change the loss scale / vary batch composition / perturb curvature with known noise. These are the axes of Figures 1–3, Table 4 and, on a neural task, Table 9; the known-noise perturbation of $\hat{c}$ is the additive model, whose curvature SNR is given in closed form in Section 3 and printed on the diagnostic axis of Figure 2.

Table 14: Null-update control (`scripts/drift_control.py`, CPU, fp32): the exact perturbation schedule of a ZO run (1SPSA pattern, P probes, $\theta \pm \varepsilon z$ and restore) with the update switched off, versus the parameter change of real steps at the validation-selected learning rates.

model	null steps	$\ \Delta\theta\ _2$	frac. touched	metric after null drift	real update $\ \Delta\theta\ _2$ (MLP: whole run; LM: median per step)
MLP (51k)	1250 (full budget)	1.71e-03	0.28	val loss 2.29308 $\rightarrow$ 2.29309	1spsa: 31.5; relmed_a1: 27.1; scalar_inv: 41.9; 1.5spsa: 19.2
GPT-2 (124M)	60 (fit $\propto t^{0.69}$; 800 steps ≈ 0.023)	3.87e-03	0.55	dev loss 0.517874 $\rightarrow$ 0.517873; test acc 0.5161 $\rightarrow$ 0.5161	1spsa: 3.21/step; scalar_inv: 0.00612/step; 1.5spsa: 4.85/step; relmed_shr: 5.67/step

Tuning cost. Every rule received the same number of configurations, so no rule is favoured by tuning. The practical argument for a scale-free rule is therefore not better tuned performance but *transfer*: a learning rate found at one loss scale remains valid at every other (Table 2), which removes one re-tuning sweep per change of loss units. The published rule needs both η and λ_{reg} retuned.

fp32 restore drift (null-update control). Restoring θ by $+\varepsilon z - 2\varepsilon z + \varepsilon z$ in fp32 is not bit-exact, and our probe-by-probe update adds P regeneration passes per step, so accumulated rounding could in principle masquerade as optimization when a rule takes very small steps. Table 14 measures it. On the MLP the full-budget null schedule moves the parameters by 1.7×10^{-3} in ℓ_2 (28% of entries touched) and changes the validation loss in the fifth decimal, versus 19–42 for the real runs. On GPT-2, 60 null steps move the 124M parameters by 3.9×10^{-3} (55% of entries touched; growth $\propto t^{0.69}$, so ≈ 0.02 over a full 800-step run) and leave dev loss ($\Delta = 10^{-6}$) and test accuracy unchanged, whereas one real step at the selected rate moves them by 3–6 for 1SPSA, 1.5-SPSA and rel-median shrinkage. The exception is scalar $1/m$, whose median curvature on this task ($m \approx 7 \times 10^3 - 2 \times 10^4$) makes each step only $\approx 6 \times 10^{-3}$: drift is then $\approx 1\%$ of a step and at most $\approx 15\%$ of its accumulated movement (if its steps were uncorrelated, $0.006\sqrt{800} \approx 0.17$ versus 0.02), so its near-zero-shot test accuracy (Section 6.6) reflects an effective learning rate that is $\sim 10^3 \times$ too small, not rounding. Drift is therefore not a confound for any comparison in this paper.

8 Limitations

- **Scale of the neural confirmation.** The two neural tasks are a 51k-parameter MLP trained from scratch and prompt-based fine-tuning of GPT-2 (124M) for ≈ 800 steps with $P = 10$ probes. The upstream results are for 8B–30B models with $P \geq 40$ and thousands of steps; we do not claim that our ordering of methods transfers to that regime. In particular, the published $\alpha = 0.1$ weighting is almost inert at our curvature magnitudes ($|\hat{c}|^{0.1} \in [1.5, 3]$), so the neural experiments mostly test the stronger $\alpha = 1$ family where normalization could matter.
- **LM selection split.** The original SST-2 dev split was 76% positive (Section 6.6). The corrected run with a balanced split (Table 11) replaces it for the held-out comparison at $s = 1$; the loss-scale transfer runs still use the learning rates selected on the original split. With three seeds per rule the LM comparison resolves only differences of several accuracy points.
- **Tuning granularity.** Matched tuning budgets were enforced with shared grids (41 quarter-decade points on quadratics, 17 half-decade points on the MLP, 4 decade-spaced points on the LM). Decade spacing on the LM is coarse; on the LM four of seven original selections lay on a grid boundary, two of seven in the corrected run (scalar $1/m$ at the maximum, Adam at the minimum); they were not extended, so those methods may be under-tuned. Quadratic and MLP selections at a grid boundary were extended by a uniform rule.
- **Noise models.** Additive i.i.d. observation noise and batch-composition noise in a linear least-squares model are stylized; neural batch noise has structure (shared centre, correlated stencils) that only the batch model captures, and the loss-scale invariance results hold exactly only for positive scalar rescaling, not for shifts or per-example reweighting.
- **Wall-clock on CPU.** MLP wall-clock was measured on a shared machine with other jobs and is reported but not used for conclusions; GPU wall-clock for the LM task was measured in dedicated containers.

- **First-order comparison.** Adam/SGD were given the same forward-equivalent budget with the paper’s $3\times$ convention and a matched number of learning-rate configurations; this is a reference point, not an exhaustive retuning, and it does not include memory-saving first-order variants.
- **Oracle diagnostics.** The exact loss ratio on quadratics and the line-searched one-step decrease are oracle quantities used only for reporting; the optimizers never see them.

9 Reproducibility statement

Code, configurations, raw per-run records and the scripts that produce every table and figure are in the project repository (src/, configs/, results/, scripts/). Protocols were frozen in configs/quad_protocol*.json, configs/lm_protocol.json and results/p07/mlp_E5/protocol.json before the held-out runs; development and evaluation seeds are disjoint and listed there. Tests in tests/ check exact forward counting ($2P$ vs $2P+1$ per micro-batch), bitwise seed regeneration of probes, pairing of probes across methods, loss-scale invariance of relative weights and non-invariance of the published floor, the learning-rate equivalence of Proposition 2, the absence of model-sized optimizer tensors, and train/validation/test disjointness for both neural tasks. The reference implementation was pinned at commit 68036ff and only read; the upstream README reports selecting checkpoints by test accuracy, which we did not reproduce. Modal GPU jobs record the exact configuration next to each output; costs are in cost_ledger.csv (list prices re-verified on 2026-10-21). The MNIST files were obtained from the ossci-datasets mirror (SHA-256 in sources.lock.json); SST-2 from the nyu-ml1/glue Hugging Face dataset; GPT-2 (124M, MIT) from the Hugging Face hub.

References

- F. Chaubard, M. J. Kochenderfer, and C. Ré. Probe-space preconditioning for fast and stable zero-order training (1.5-SPSA). arXiv:2609.38095, 2026. Reference code: <https://github.com/Fchaubard/1.5-SPSA-LLM> (commit 68036ff).
- F. Chaubard et al. Zero-order model sharding for memory-efficient training. arXiv:2609.37899, 2026.
- S. Malladi, T. Gao, E. Nichani, A. Damian, J. D. Lee, D. Chen, and S. Arora. Fine-tuning language models with just forward passes. In *NeurIPS*, 2023. <https://princeton-nlp.github.io/mezo/>.
- J. C. Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, 37(3):332–341, 1992.
- J. C. Spall. Adaptive stochastic approximation by the simultaneous perturbation method. *IEEE Transactions on Automatic Control*, 45(10):1839–1853, 2000.
- Y. Zhang, P. Li, J. Hong, J. Li, Y. Zhang, W. Zheng, P.-Y. Chen, J. D. Lee, W. Yin, M. Hong, Z. Wang, S. Liu, and T. Chen. Revisiting zeroth-order optimization for memory-efficient LLM fine-tuning: A benchmark. In *ICML*, 2024.
- D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.
- P. J. Huber. *Robust Statistics*. Wiley, 1981.
- A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. OpenAI technical report, 2019.
- R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, 2013.
- A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR*, 2019.
- Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

A Equation parity with the reference implementation

The reference `scripts/train.py` (commit 68036ff, lines 491–498 and 160–165) computes, per probe, `curv = |loss_plus - 2 loss_clean + loss_minus| / eps^2`, `curvature = max(curv ** alpha, lambda_reg)` and `grad_coeff = (loss_plus - loss_minus) / (2 eps n_perts curvature)`, accumulates `grad_coeff * z` into an fp32 buffer of model size, and finally applies `theta -= lr * grad`. This is exactly Eqs. (1)–(3) with $\eta = \text{lr}$; the paper’s Eq. 5 writes the update with $\eta = 1/(2P)$ after tying $\eta = \varepsilon$, and its Eq. 5 has a sign typo (plus) relative to Eq. 6 and Algorithm 1 (minus); we use the descent sign. The clean loss is evaluated once per step on the same micro-batches as the perturbed losses, and perturbations are regenerated from `torch.manual_seed(iteration*10000 + pert_idx)` as bit-packed Rademacher vectors. Our implementation differs only in applying the update probe-by-probe after the weights are known, which removes the model-sized buffer at the cost of P additional regeneration passes (measured at 8 ms per pass for 124M parameters on an L4, versus 22 ms per forward pass at batch 16).

B Additional tables

Full per-condition tables (selected learning rates, mean $\log_{10}$ final loss ratio with 95% bootstrap CIs, divergence counts, median evaluations to 10^{-2} , and paired differences against 1SPSA and 1.5-SPSA) are generated by `scripts/analyze_quad.py` into `reports/tables/`. Selected learning rates for the neural tasks are in `reports/tables/{mlp,lm}_selected_lrs.json`.