TRAINING CONVOLUTIONAL NETWORKS AT TEMPORARILY REDUCED INTERNAL RESOLUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Continuation methods train neural networks on a sequence of modified problems before recovering the original objective. We investigate spatial continuation for image classification with convolutional networks: the internal spatial resolution of feature maps is temporarily reduced, then progressively restored during training. This simple intervention improves mean test accuracy in all twenty no-augmentation settings, by 0.38 to 6.28 points, and most of this gain comes from the temporary reduction itself rather than from the progression. The settings span datasets, dataset sizes, architectures, activations, normalizations, optimizers and training budgets. The intervention is absent from the final network, so inference cost is unchanged. Its measured training time is comparable to standard training in the short reference setting and lower in a controlled long-SGD timing study. In matched comparisons with a prior smoothing curriculum, resolution reduction has higher mean accuracy in the no-augmentation recipes, while the augmented SGD comparison depends on BatchNorm calibration. The gains are more consistent with simple optimizers than with adaptive optimizers, where they remain present but vary more. Their magnitude, however, depends on the training recipe, and stronger data augmentation can substantially reduce it. Comparisons with Gaussian filtering of the feature maps, used here as a control, indicate that the gains cannot be attributed to smoothing alone. Geometric diagnostics show lower curvature in the reference setting, but comparisons across methods and training recipes do not support lower curvature as a general explanation of the gains. Overall, these results support temporary internal resolution reduction as a simple and useful intervention in unaugmented training, while showing that its advantage narrows under stronger recipes.

1 INTRODUCTION

An idea to minimize a difficult objective is to start from a simpler approximation and progressively recover the target problem. Continuation methods follow this principle by linking problems through a parameterized family and using the previous solution to initialize the next stage (Allgower & Georg, 1990; Mobahi & Fisher, 2015). Graduated optimization provides guarantees under specific geometric assumptions (Hazan et al., 2016). Constructing useful families for neural-network training remains less straightforward. A convolutional network offers one handle that is simple to act on: the spatial resolution at which its inner layers operate.

We study the family obtained from that handle. During training, we temporarily reduce the spatial resolution of the internal feature maps, then progressively restore the native resolution. A common intuition behind such curricula is that a model can first focus on large-scale structures before finer details; we test whether this progression, and its order, matter. Reduction acts on the grid rather than on its contents: each remaining position keeps the strongest response in its window, which is not the same as filtering fine detail away. Because the schedule ends at the original architecture, the reference network is recovered and no added operation remains at inference.

Other spatial curricula act on different handles. Gaussian filtering smooths the feature maps without changing their size (Sinha et al., 2020); progressive resizing changes the input images rather than the internal maps (Hoffer et al., 2019; Tan & Le, 2021). SDPoint also reduces internal resolution, but samples a block boundary and a reduction ratio at random on each iteration and leaves

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.

a downsampling configuration available at inference (Kuen et al., 2018). We instead use a single fixed site, a monotone coarse-to-fine schedule leading to an endpoint that is exactly the reference architecture. We study this intervention in its own right: we measure the size and consistency of the gain, identify the training recipes that preserve it, ask whether the progression and its coarse-to-fine order are themselves required, and test whether its effect can be explained by a simple change in loss landscape around the solutions reached by the procedure, at the end of the schedule.

Across twenty settings without augmentation (spanning datasets, dataset sizes, architectures, activations, normalizations, optimizers and training budgets) resolution reduction improves mean test accuracy in every one, by 0.38 to 6.28 points. Its measured running time is close to that of standard training in the reference setting; a controlled longer SGD run shows a reduction in training-loop time. A CIFAR-10 exploration fixes the reduction site and schedule, which we then keep across settings (Appendix B); Gaussian filtering of feature maps, and the combination of both, are carried along as comparison points. The gains are more consistent with simple optimizers such as SGD than with adaptive optimizers such as AdamW, where they remain present but vary more. We extend the comparison to alternative losses, data augmentation and longer training, and include Curriculum by Smoothing and SDPoint as direct comparators; these additional experiments locate the effect rather than merely bound it: the gains persist across optimizers, losses and budgets, but recede once the recipe is itself strongly regularized.

Contributions.

- We isolate temporary internal resolution reduction as a simple training intervention. Our results show that it improves mean test accuracy in all twenty no-augmentation transfer settings, with gains from 0.38 to 6.28 points, and leaves the inference network unchanged, with training time close to that of standard training in the reference recipe.
- We delimit where the effect holds and where it stops. The gains are most consistent with simple optimizers and weak regularization, while data augmentation or a longer 160-epoch recipe leaves little or no advantage over the baseline. Selecting the learning rate and stopping epoch on a held-out validation split, for both methods, leaves a gain of 2.49 ± 0.24 points. Reversing or mixing the reduced-resolution epochs at matched exposure also preserves the gain, and a single moderately reduced resolution retains most of it, so most of the gain comes from the temporary reduction rather than from a coarse-to-fine progression.
- We place the effect against existing spatial interventions. Under matched training budgets, resolution reduction has higher mean accuracy than both Curriculum by Smoothing variants in each of the three no-augmentation recipes. In the augmented SGD recipe, their ordering depends on the BatchNorm calibration policy. SDPoint, which reduces internal resolution at random rather than on a schedule, is behind under short SGD and stronger under AdamW without augmentation. Gaussian filtering does not reproduce the gains' consistency, and filtering more on the same resolution schedule does not uniformly improve on reduction alone.
- We examine the geometry of the resulting models. Resolution reduction lowers curvature in the reference setting, but comparisons with SDPoint, other recipes and normalization policies show that these diagnostics do not consistently track the accuracy gains.

2 Related work

Continuation and curricula. Classical continuation methods trace solutions through parameterized problems (Allgower & Georg, 1990). For nonconvex optimization, Mobahi & Fisher (2015) analyze Gaussian-smoothed objectives, while Hazan et al. (2016) identify a class for which graduated optimization converges globally. Spatial filtering inside a network is not equivalent to smoothing its loss in parameter space, so these guarantees do not directly apply here. Related approaches include Mollifying Networks, which anneal noise in the activations while gradually moving the network from identity or linear transformations toward its original nonlinear computation (Gülçehre et al., 2017). Curriculum learning progressively changes the selection or ordering of training examples (Bengio et al., 2009). Our methods keep the examples and labels fixed and change their spatial processing.

Spatial information during training. Curriculum by Smoothing (CBS) (Sinha et al., 2020) also follows a schedule: Gaussian kernels filter feature maps, and their scale decreases during training. SDPoint (Kuen et al., 2018) also reduces the resolution of internal feature maps. At each training iteration, it randomly selects a block boundary and a reduction ratio, then applies average pooling. Our procedure instead uses max-pooling at a fixed location and progressively restores the original resolution. While SDPoint allows the inference cost to be adjusted by choosing a downsampling configuration, our modification is temporary and ends at the original inference architecture. Other approaches change the resolution of the input images rather than the internal maps. For instance, Hoffer et al. (2019) use different image sizes, while Tan & Le (2021) progressively increase the resolution and adjust the regularization accordingly.

Frequency and aliasing. Rahaman et al. (2019) describe a bias towards low-frequency functions during neural-network learning. In this setting, frequency refers to variations of the learned function over its input space, which is different from the spatial frequencies of the feature maps we filter. For these feature maps, reducing resolution and removing high frequencies are also different things, and their order matters. Applying a low-pass filter before subsampling can make predictions more stable under small image translations (Zhang, 2019). The locations at which low-pass filters are inserted can be important for generalization (Vasconcelos et al., 2021). The picture is further complicated by nonlinearities, which can introduce new frequency components and aliasing (Michaeli et al., 2023). These distinctions matter for our Gaussian interventions: placing a blur after a strided convolution does not undo aliasing that has already occurred in that convolution.

Geometry and sensitivity. Our procedures recover the original architecture at the end of training, but follow different optimization paths. To examine the resulting models, we draw on the loss visualizations of Li et al. (2018), using filter-normalized loss slices and PCA projections of optimization trajectories. These plots require care: reparameterizing a network can change its apparent sharpness without changing its predictions (Dinh et al., 2017). Even for adaptive sharpness, its relation to generalization is not stable across training regimes and depends on the sharpness definition, including whether logits are normalized (Andriushchenko et al., 2023). Our average loss increase under random weight perturbations is related to expected sharpness (Neyshabur et al., 2017); we use symmetric, filter-normalized perturbations at several amplitudes. We also examine local curvature through Hessian–vector products, which can be computed without forming the full Hessian (Pearlmutter, 1994), as used in scalable Hessian-spectrum analyses of neural networks (Ghorbani et al., 2019). Here, we use these tools to compare the trained models, without using trajectory projections as evidence, and do not claim that the measured differences explain their accuracy gains.

3 SPATIAL CONTINUATION

During training, we progressively modify the spatial resolution of the internal feature maps and, when filtering is used, the scale of the Gaussian filters. Each configuration is kept for a fixed number of optimization updates before moving to the next, while preserving both the network weights and the optimizer state. We do not require the intermediate training problems to be solved to convergence. The schedule eventually restores the initial resolution and removes all added filtering, so that training ends with the original network and objective. We use continuation to describe this sequence of modified problems ending at the original one. Since resolution changes are discrete and max-pooling is nonsmooth, we do not claim to follow an actual smooth path of minimizers.

Our exploratory CIFAR-10 study, detailed in Appendix B, considered several reduction operators, insertion points, schedules, and filtering configurations. From this exploration, we retained three procedures for the subsequent experiments: resolution reduction alone (RESOLUTION, R), Gaussian filtering alone (GAUSSIAN, G), and their combination (COMBINED, RG). R is the intervention studied in this paper, while G and RG provide comparison points: G asks whether the gains require smoothing rather than a change of resolution, and RG whether filtering more, on the same resolution schedule, helps further. Plain denotes standard training without either intervention. Table 1 summarizes the three procedures.

Resolution reduction (R). R reduces the spatial size of the feature maps using adaptive max-pooling, applied independently to each channel. In ResNet-20, the reduction is inserted after block

Table 1: Procedures compared in ResNet-20. R is the intervention studied in this paper; G and RG are the comparison points used throughout. Plain has no intervention.

Method	Reduction	Added Gaussian
R	After block 1	None
G	None	10 post-ReLU sites
RG	After block 1	19 conv outputs

1 (the second residual block; blocks are indexed from 0), and the map side progressively increases from 16 to 24 and finally to its native size 32. The following layers operate directly on the smaller maps, without upsampling. Global average pooling keeps the classifier input dimension unchanged, and no trainable parameter shapes are modified. Since max-pooling retains local maxima, the operation reduces spatial resolution without being a linear low-pass filter. Exact pooling windows and schedule boundaries are given in Appendix A.3 and Appendix A.4.

Gaussian filtering (G). G keeps the spatial resolution unchanged and instead smooths intermediate feature maps with channelwise Gaussian filters. The filtering strength is progressively decreased and is removed before the end of training. In ResNet-20, filters are applied after the initial ReLU and after the final ReLU of each residual block, for ten filtering sites in total. The filters introduce no trainable parameters. Sites, kernel and schedule are given in Appendix A.3 and Appendix A.4.

Combined procedure (RG). RG follows the same resolution schedule as R while also applying Gaussian filtering. In ResNet-20, the filters are placed at the outputs of all 19 convolutions, before BatchNorm, rather than at the ten post-ReLU sites used by G. That placement was selected rather than assumed: without reduction, post-ReLU filtering scored higher, whereas with reduction after block 1 the order reversed in every seed (Appendix B). Their scale is additionally adjusted according to the current resolution, which makes the effective filtering scale non-monotone. RG is therefore not the literal superposition of R and G, but a separately selected training procedure. The exact scale rule and schedule are given in Appendix A.4.

4 Experimental study

4.1 Setup and evaluation

The reference setting, unless stated otherwise, is CIFAR-10 (Krizhevsky, 2009) with ResNet-20 (He et al., 2016) and BatchNorm (Ioffe & Szegedy, 2015), trained for 30 epochs of SGD without augmentation. In that setting R reaches $80.37 \pm 0.61\%$ test accuracy against $75.43 \pm 0.76\%$ for Plain. An exploration on the same setting fixed the reduction site and schedule; Appendix B reports it together with the comparisons and failures that led to these choices.

We evaluate each model after the final scheduled epoch, through the original network with the added operations disabled and the saved normalization statistics. Gains over Plain are formed within a seed, with matching initial weights and training-example order, and results from separate campaigns are kept separate. Gains quoted in the text are these within-seed means; $\pm$ denotes one sample standard deviation (SD) over seeds. Section 4.4 also recalibrates BatchNorm.

4.2 Transfer across datasets, architectures and optimizers

Every one of the sixty seed-level comparisons between R and Plain is positive, with mean gains from 0.38 to 6.28 percentage points across the twenty settings (Figure 1). This includes SVHN, STL-10, GELU, SiLU, GroupNorm and VGG-11-BN (Netzer et al., 2011; Coates et al., 2011; Hendrycks & Gimpel, 2016; Ramachandran et al., 2017; Wu & He, 2018; Simonyan & Zisserman, 2015). Averaged over the ten SGD settings the gain of R is 4.03 points, against 2.92 over the ten adaptive ones, with a smaller spread under SGD, although the three largest adaptive gains all come from STL-10. One reading is that adaptive preconditioning already compensates part of the scale differences the reduction acts on, which would reduce its relative benefit; the learning rates and weight decay differ across these recipes, so we cannot test that here.

Table 2 reports the final accuracies and Table 11 the paired gains for all three procedures. Each setting changes the optimizer, activation, normalization, architecture, dataset, data size or budget compared with the reference. The transfer studies use three seeds each. The spatial ratios and schedule structure are preserved, with adaptations for map size and training budget. The protocols and architectural adaptations are given in Appendix C.

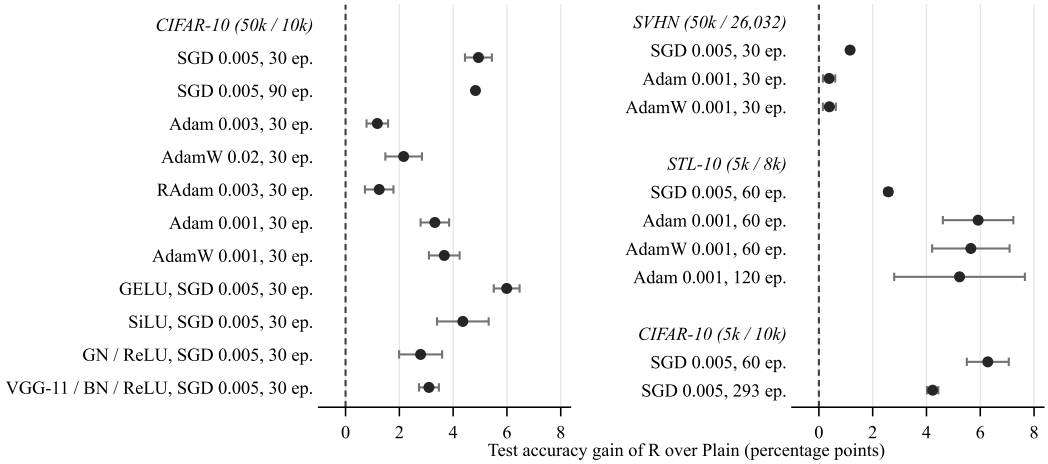


Figure 1: Within-seed test accuracy gain of R over Plain across the twenty no-augmentation transfer settings, in percentage points. Markers give the mean over three seeds and bars span one SD; the dashed line is Plain. Every setting improves, and all sixty seed-level comparisons are positive. Tables 2 and 11 give the accuracies and paired gains of all methods.

Table 2: Final test accuracy (%), mean ± sample SD over three seeds, without augmentation. Bold means exceed the matching Plain mean; this is descriptive, not a significance test. The dataset headings give training / test sizes. BN and GN denote BatchNorm and GroupNorm.

Model / norm / activation	Optimizer	LR	Epochs	Plain	R	G	RG
CIFAR-10 (50k / 10k) train / test							
ResNet-20 / BN / ReLU	SGD	0.005	30	75.43 ± 0.76	**80.37** ± 0.61	**79.91** ± 0.27	**81.51** ± 0.22
ResNet-20 / BN / ReLU	SGD	0.005	90	76.92 ± 0.18	**81.75** ± 0.20	**82.35** ± 0.56	**83.67** ± 0.44
ResNet-20 / BN / ReLU	Adam	0.003	30	84.13 ± 0.59	**85.31** ± 0.27	83.95 ± 0.07	**85.28** ± 0.26
ResNet-20 / BN / ReLU	AdamW	0.02	30	84.12 ± 0.49	**86.28** ± 0.22	83.71 ± 0.29	**86.21** ± 0.29
ResNet-20 / BN / ReLU	RAdam	0.003	30	84.38 ± 0.12	**85.63** ± 0.40	**84.75** ± 0.19	**85.26** ± 0.15
ResNet-20 / BN / ReLU	Adam	0.001	30	79.64 ± 0.19	**82.96** ± 0.67	**82.17** ± 0.43	**83.25** ± 0.27
ResNet-20 / BN / ReLU	AdamW	0.001	30	79.54 ± 0.55	**83.22** ± 0.55	**81.90** ± 0.54	**83.77** ± 0.34
ResNet-20 / BN / GELU	SGD	0.005	30	74.80 ± 0.59	**80.79** ± 0.42	**80.27** ± 0.36	**81.58** ± 0.13
ResNet-20 / BN / SiLU	SGD	0.005	30	76.55 ± 0.68	**80.91** ± 0.33	**80.44** ± 0.78	**81.68** ± 0.14
ResNet-20 / GN / ReLU	SGD	0.005	30	75.48 ± 0.88	**78.27** ± 0.68	**76.64** ± 0.43	**78.85** ± 0.21
VGG-11 / BN / ReLU	SGD	0.005	30	80.67 ± 0.61	**83.77** ± 0.41	**84.66** ± 0.25	**84.54** ± 0.45
SVHN (50k / 26,032) train / test							
ResNet-20 / BN / ReLU	SGD	0.005	30	93.10 ± 0.03	**94.26** ± 0.07	**94.26** ± 0.14	**94.85** ± 0.20
ResNet-20 / BN / ReLU	Adam	0.001	30	94.40 ± 0.19	**94.78** ± 0.08	**94.71** ± 0.19	**94.91** ± 0.05
ResNet-20 / BN / ReLU	AdamW	0.001	30	94.46 ± 0.24	**94.85** ± 0.10	**94.81** ± 0.09	**95.01** ± 0.11
STL-10 (5k / 8k) train / test							
ResNet-20 / BN / ReLU	SGD	0.005	60	57.35 ± 0.53	**59.93** ± 0.45	**58.44** ± 1.55	56.06 ± 0.44
ResNet-20 / BN / ReLU	Adam	0.001	60	62.79 ± 1.09	**68.72** ± 0.22	**63.33** ± 1.13	62.76 ± 0.38
ResNet-20 / BN / ReLU	AdamW	0.001	60	62.95 ± 0.96	**68.60** ± 0.81	**63.01** ± 0.89	62.63 ± 0.65
ResNet-20 / BN / ReLU	Adam	0.001	120	62.06 ± 1.40	**67.29** ± 1.06	**67.37** ± 0.36	**67.15** ± 0.58
CIFAR-10 (5k / 10k) train / test							
ResNet-20 / BN / ReLU	SGD	0.005	60	48.76 ± 1.05	**55.04** ± 0.51	**57.40** ± 0.38	**59.39** ± 1.34
ResNet-20 / BN / ReLU	SGD	0.005	293	50.77 ± 0.93	**55.00** ± 0.79	**60.88** ± 0.24	**62.55** ± 0.87

Each row is a separate recipe. The two Adam and AdamW rates belong to different campaigns, with different weight decay (Appendix C). All evaluations use saved normalization statistics and the native, filter-free path; GN has no running statistics. Paired gains appear in Table 11.

The Gaussian gives a less uniform result. Its mean gain is slightly negative at the originally selected Adam and AdamW rates (Kingma & Ba, 2015; Loshchilov & Hutter, 2019), small and positive

with RAdam (Liu et al., 2020), and 2.53 and 2.35 points on CIFAR-10 with Adam and AdamW at 0.001. These recipes differ in weight decay, and the earlier rates were chosen on Plain's seed-0 test accuracy, so the optimizer alone does not account for the difference.

The combination does not perform best on every dataset. On STL-10 with SGD for 60 epochs, R and G improve the mean but RG loses 1.29 points, with a negative difference in every seed. Since RG filters at different sites from G, this does not isolate an effect of combining the two interventions. With Adam for 120 epochs, all three instead gain more than five points. This recovery changes both the optimizer recipe and training horizon; it does not identify the reason for the earlier failure. On CIFAR-10, extending SGD training to 90 epochs preserves substantial gains, including 6.75 points for RG. With only 5,000 examples, RG gains 10.62 points at 60 epochs and 11.78 at 293 epochs. Sample count or duration alone therefore does not determine whether filtering helps.

The interventions also change the balance between training and test cross-entropy (CE). Across the seventeen no-augmentation settings with both losses available (Table 12), 48 method–baseline comparisons improve mean test accuracy; 44 of them show a higher CE on a fixed 500-image training probe together with a lower test CE, and the remaining four show a decrease in both. The separate five-seed reference study confirms the pattern on the full training set: all three methods have higher training CE, lower test CE and higher test accuracy than Plain in every seed (Table 27). The gains therefore cannot be explained by a lower final training loss; they often occur despite a higher one, although that increase is not necessary for better test performance.

Further controls retain the same exposure to each resolution and the same final native phase, while reversing or mixing the reduced-resolution epochs. Both preserve the 30-epoch SGD gain; the mixed schedule exceeds R by 1.33 ± 0.19 points under 160-epoch AdamW, while also changing transition frequency and BatchNorm history (Appendix G.1). These comparisons do not establish an advantage for monotone ordering. A single reduced side held for the same reduced phase tests whether the progression itself is needed (Appendix G.2). A side of 24 retains 4.32 of the 4.66 points gained by R over Plain in 30-epoch SGD (five seeds) and is within 0.06 points of R under 160-epoch AdamW (three seeds); a side of 16 costs 1.1 to 1.7 points relative to R and falls below Plain under AdamW. Most of the gain thus comes from a temporary, moderate reduction rather than from a multi-stage progression, although R remains above the single side of 24 in every short-SGD seed.

Separately, selecting the learning rate and stopping epoch on a 45k/5k training/validation split leaves a recalibrated test-accuracy gain of 2.49 ± 0.24 points for R on three new seeds. The search covers five learning rates from 0.0025 to 0.04 and the stopping epoch, on three selection seeds and validation accuracy only; the test set is used once, after the choices are fixed, and the gain is positive in all three pairs. Both methods select the largest tested rate, 0.04 (Appendix G.3); this subsequent control is separate from the twenty transfer settings. This estimate is a result for a limited search, not an upper bound on the gain or evidence that either method is optimally tuned.

4.3 Other objectives and augmentation

R's benefit is not confined to cross-entropy training. Label smoothing and focal loss (Szegedy et al., 2016; Lin et al., 2017) retain gains for all three procedures. Squared-error training on logits also gives gains at suitable learning rates, but they disappear at the largest tested rate (Table 16). We retain the full learning-rate sweep rather than treating its best result as independent validation.

Augmentation changes the picture more clearly. With SGD at 0.01 for 60 epochs, adding random crops and horizontal flips raises Plain from 80.27% to 88.27%. The paired gain for R falls from 3.25 to 0.19 points; the comparison points follow, from 2.57 to -0.44 for G and from 4.24 to 0.12 for RG (Tables 17 and 18). Two readings are consistent with this. Strong augmentation is itself a substantial regularizer and changes the objective seen during training, so part of what the intervention provides may already be supplied. The stronger baseline also leaves less room above it. Our experiments separate neither mechanism, and the two do not match in size: the room above Plain shrinks by a factor of 1.7, the paired gain for R by a factor of 17. The practical counterpart is that the gains are largest exactly where augmentation is absent, a regime that matters whenever domain-appropriate transformations are not available. Appendix D reports all loss and augmentation settings, including their controls.

4.4 Comparison with Curriculum by Smoothing and SDPoint

We compare R with two CBS configurations (Sinha et al., 2020) and SDPoint (Kuen et al., 2018) in a common ResNet-20 trainer, with G and RG carried alongside. CBS-P follows the released Gaussian operator and published decay schedule; CBS-B uses the same operator but removes filtering after 70% of the update budget. Both differ from G in filter size, boundary handling and placement, so G is not a reimplementation of CBS. SDPoint samples a residual-block boundary or no downsampling for each optimizer update, using average pooling when active. Its evaluated inference instance is fixed to full resolution. Appendix E specifies the adaptations and the checks against the authors' code. These are comparisons under shared recipes, not reproductions of the papers' headline results.

BatchNorm requires particular care here: SDPoint's saved statistics mix several training configurations. Our primary comparison, P1, recalibrates every model on all 50,000 clean training images. We also report P3, a pass through the training loader with the moving-average update used in the SDPoint authors' evaluation code. P3 changes several calibration choices together, including whether training augmentation is used. With augmentation, P1 falls below P3 for every method, by 1.9 to 7.3 points, whereas the two agree to within 0.13 points in the unaugmented 160-epoch recipe; recalibrating on clean images a network trained on augmented ones is the likely cause. Neither policy updates the learned weights, and both are applied to every method. The saved-statistics (P0) and calibration-batch-size (P2) controls are retained in the appendix. Stopping SDPoint when R returns to native resolution changes mean P1 accuracy by at most 0.16 points in three recipes, while saved-statistics differences are larger (Appendix E.4).

Against the better of CBS-P and CBS-B, R has higher mean accuracy in each of the three no-augmentation recipes in Table 3, by 1.75 to 3.42 percentage points under P1; the long-AdamW ordering is unchanged under P3. In the two augmented recipes, its margin ranges from -0.60 to 1.98 points across P1 and P3. The SGD ordering reverses between calibration policies, and three seeds do not support a precise ranking of the closer comparisons. G performs better than both CBS configurations in these no-augmentation comparisons, but this does not establish superiority over CBS at other budgets or under its original recipe. SDPoint shares the handle rather than the schedule. In the 30-epoch SGD comparison, R exceeds it by 0.63 ± 0.19 points and RG by 1.80 ± 0.08 under P1, with positive differences in every seed. With AdamW, SDPoint instead exceeds R by 0.92 ± 0.53 points. Its advantage persists at 160 epochs without augmentation: it exceeds R by 3.11 ± 0.55 points and RG by 2.39 ± 0.30.

Table 3: Comparison with CBS and SDPoint on CIFAR-10 (50k / 10k), ResNet-20-BN/ReLU. Final test accuracy (%), mean ± SD over three seeds. Bold means exceed Plain within the same column.

	30 epochs, no aug.		160 epochs					
	SGD	AdamW	SGD, aug.		AdamW, no aug.		AdamW, aug.	
Method	P1	P1	P1	P3	P1	P3	P1	P3
Plain	75.46 ± 0.85	84.28 ± 0.20	89.66 ± 0.17	91.53 ± 0.35	84.95 ± 0.26	84.86 ± 0.25	88.30 ± 0.65	90.84 ± 0.36
R	$\mathbf{80.46} \pm 0.51$	$\mathbf{86.65} \pm 0.14$	88.56 ± 1.28	$\mathbf{91.66} \pm 0.15$	$\mathbf{85.83} \pm 0.65$	$\mathbf{85.72} \pm 0.64$	$\mathbf{88.46} \pm 0.44$	90.53 ± 0.17
G	$\mathbf{79.94} \pm 0.03$	83.75 ± 0.23	85.67 ± 1.63	90.46 ± 0.39	84.74 ± 0.83	84.63 ± 0.84	81.18 ± 1.87	88.52 ± 0.31
RG	$\mathbf{81.63} \pm 0.27$	$\mathbf{86.24} \pm 0.36$	87.17 ± 1.13	90.70 ± 0.41	$\mathbf{86.55} \pm 0.32$	$\mathbf{86.42} \pm 0.29$	85.65 ± 0.71	89.44 ± 0.15
CBS-P	$\mathbf{78.01} \pm 0.61$	80.56 ± 0.24	88.85 ± 0.31	91.36 ± 0.34	83.33 ± 0.47	83.27 ± 0.30	86.48 ± 0.67	89.52 ± 0.15
CBS-B	$\mathbf{78.71} \pm 0.90$	83.23 ± 0.38	89.16 ± 0.33	$\mathbf{91.54} \pm 0.21$	83.71 ± 0.31	83.71 ± 0.26	86.40 ± 0.49	89.25 ± 0.07
SDPoint	$\mathbf{79.83} \pm 0.34$	$\mathbf{87.57} \pm 0.59$	88.43 ± 0.43	$\mathbf{91.90} \pm 0.31$	$\mathbf{88.94} \pm 0.12$	$\mathbf{88.84} \pm 0.12$	88.18 ± 0.46	$\mathbf{91.18} \pm 0.12$

CBS-P keeps the published five-epoch decay; CBS-B adapts it to the budget. Native inference is used: CBS-P retains filtering at 30 epochs. SGD learning rates are 0.005 (30 epochs) and 0.1 (160); AdamW uses 0.02. The 160-epoch SGD recipe also changes batch size, decay and scheduler. P0/P2 and full protocols appear in Appendix E.

With augmentation and 160 epochs, calibration can reverse the ordering. Under AdamW, R is 0.28 points above SDPoint with P1 and 0.65 points below it with P3. Under P3, the augmented SGD recipe reaches 91.53% for Plain, 91.66% for R and 91.90% for SDPoint; G and RG are below Plain. Under the same policy, R, G and RG are all below Plain with augmented AdamW; under P1 only G and RG are. The large original gains therefore do not carry over to these stronger recipes, which add augmentation to a longer schedule and may already supply part of what the intervention provides. The small remaining differences, based on three seeds, do not support a precise ranking. CE and

accuracy should be kept distinct here: with augmented AdamW under P3, R, G and RG all have lower test CE than Plain despite lower accuracy (Table 22).

4.5 COMPUTATIONAL COST

We compare the measured training times of the methods under matched update budgets. In the reference setting, R takes around as long as Plain, while G and RG add approximately 23% and 45% to total running time. The longer comparisons distinguish training time from evaluation overhead. In the 160-epoch SGD recipe R is the fastest of the seven procedures, ahead of Plain and of SDPoint; with AdamW it returns to the cost of Plain, while filtering stays above it in every recipe (Table 4). A separate campaign times this recipe by phase on a dedicated device: R's native phase costs what Plain costs, 14.65 against 14.60 seconds per epoch, so the saving comes from the 40% of epochs at reduced resolution alone (Appendix C.3). There the saved time does not buy accuracy: R is 0.21 ± 0.36 points below Plain under P3, and a variant given 36% more time is not resolved either. Smaller maps can therefore save time, but do not guarantee a saving in every implementation.

Table 4: Measured training-loop time (minutes) at 160 epochs on CIFAR-10 / ResNet-20-BN/ReLU, mean $\pm$ SD over three seeds on Tesla T4. Checkpoint writes are included; periodic evaluation and final calibration are excluded.

	SGD	AdamW	
Method	Aug.	No aug.	Aug.
Plain	37.2 ± 0.8	41.3 ± 0.9	42.8 ± 1.4
R	32.0 ± 0.4	43.4 ± 3.3	43.5 ± 2.3
G	47.7 ± 1.8	53.0 ± 0.7	53.7 ± 0.2
RG	48.9 ± 1.6	66.1 ± 1.7	65.1 ± 3.1
CBS-P	41.8 ± 0.6	48.8 ± 0.8	49.6 ± 1.9
CBS-B	41.6 ± 1.1	47.6 ± 2.7	48.1 ± 0.8
SDPoint	33.9 ± 0.7	42.8 ± 0.6	43.7 ± 1.2

These training-loop timers exclude evaluation but retain checkpoint writing. Total times and measured peak allocated GPU memory appear in Table 25 (Appendix E); the shared-GPU execution conditions prevent hardware-independent speed claims. Energy was not measured. At inference, R, G and RG disable every added operation, so their computation returns to that of the original network. CBS-P retains its native filtering, and our SDPoint comparison uses its full-resolution instance.

5 LOSS GEOMETRY AFTER SPATIAL CONTINUATION

We examine whether the accuracy gains are accompanied by lower curvature or weight-perturbation sensitivity. The reference study compares Plain, R, G and RG over five CIFAR-10 / ResNet-20 / SGD seeds, using the native filter-free network.

Perturbing the weights of a BatchNorm network also shifts its batch statistics, so curvature depends on how those statistics are treated. We use two conventions. For the Hessian, statistics are recalibrated once at the checkpoint and then held fixed (*centre-frozen*). For finite perturbations, we additionally recalibrate at every perturbed point (*pointwise recalibration*). Both give the same function at the checkpoint but not around it. All measures use fixed 1,000-image training and test probes; accuracies are reported, as in Section 4.4, after recalibration on the full training set. Sensitivity S is the mean symmetric, centre-subtracted CE increase along 20 filter-normalized directions; its definition and numerical checks are in Appendix F.

Under fixed statistics, R has smaller leading eigenvalues and traces of the Hessian $\boldsymbol{H}$ of the mean CE than Plain on both probes in every seed (Appendix F.4). Figure 5 (Appendix F) shows a corresponding reduction in finite sensitivity. Pointwise recalibration reduces the test-probe contrast; G then becomes more sensitive than Plain despite its better unperturbed accuracy. Recalibration thus changes the question of sensitivity, rather than revealing a unique ordering of the models.

The comparison with SDPoint further limits this interpretation (Table 5). In 30-epoch SGD, SDPoint has a smaller test-probe trace than R in all three paired seeds, while R has higher accuracy. In

160-epoch AdamW without augmentation, both methods have larger traces than Plain and better accuracy. With augmentation, the trace increases remain while accuracy gains are small or absent. The direction-weighted traces (defined in Appendix F.4) show the same recipe change in Appendix F.6, which also gives additional measures and the learning-rate control.

Table 5: Mean accuracy and test-probe trace for three matched seeds per recipe. Accuracy uses full-test P1 (50k calibration); the Hessian uses a 1k-image test probe and centre-frozen BatchNorm (2k calibration). Compare trace magnitudes within recipes. Paired spreads and further measures are in Appendix F.6.

	Test accuracy (%)			Test-probe $\mathrm{tr}(\boldsymbol{H})$		
Recipe	Plain	R	SDPoint	Plain	R	SDPoint
SGD 30, no aug.	75.46	80.46	79.83	20,772	11,414	5,817
AdamW 160, no aug.	84.95	85.83	88.94	6.31	9.85	7.58
AdamW 160, crop + flip	88.30	88.46	88.18	6.94	8.56	8.82

Andriushchenko et al. (2023) likewise find that sharpness correlations within ResNet training regimes do not necessarily persist across them. They distinguish invariance to weight rescaling (including BatchNorm-related symmetries) from sensitivity to logit scale, which they address by normalizing logits. Our filter-normalized perturbations and BatchNorm recalibration address different choices. We do not normalize logits or measure their elementwise adaptive sharpness. These comparisons do not establish lower curvature as the cause of R's gain.

6 Discussion and conclusion

Temporary internal resolution reduction is a simple and effective training intervention in unaugmented recipes: its mean accuracy gain is positive in all twenty no-augmentation transfer settings, from 0.38 to 6.28 points, including longer runs and alternative optimizers. It adds one pooling operation at a fixed site during training and leaves no added operation at inference. Its site and schedule were selected on CIFAR-10, then adapted to the map size and training budget of later settings. Training time is comparable to Plain in the short reference recipe and lower in the controlled 160-epoch SGD timing study. The additional controls locate where that gain ends: augmentation substantially reduces it, and the stronger 160-epoch recipes leave little or no accuracy advantage over Plain.

The continuation view motivated the schedule, but the gain does not require a coarse-to-fine progression: at matched exposure, reversing or mixing the reduced-resolution epochs preserves it, and a single moderately reduced resolution retains most of it. Both the ranking against other spatial interventions and the geometric diagnostics also change with the training recipe and the BatchNorm convention. R is ahead of both CBS configurations in each of the three no-augmentation recipes. Against SDPoint the outcome splits by optimizer: R is ahead under short SGD training, while SDPoint is stronger under AdamW without augmentation. BatchNorm calibration can reverse close comparisons with augmentation. Gaussian filtering helps in some regimes, including longer training on STL-10 with Adam, but can hurt in others.

The geometry study places the reference curvature reduction in context. SDPoint can have lower curvature than R while achieving lower accuracy, and the trace ordering against Plain changes under AdamW. Finite sensitivity also depends on BatchNorm treatment. These diagnostics do not identify the cause of the gains.

Our study remains exploratory: three choices bound what it can establish. Method selection and several learning-rate decisions used CIFAR-10 test results, and later experiments were motivated by earlier outcomes. Additional seeds do not remove that exposure. The subsequent validation control retains a gain, but selects the boundary of the tested learning-rate grid. The settings are targeted rather than a factorial benchmark, and three seeds cannot reliably rank small differences. Within this scope, the contribution is an empirical account of when temporary resolution reduction helps, which part of its schedule matters, when established alternatives do better, and the limits of geometric interpretations of its gains.

AI USE STATEMENT

In this work, we used generative AI tools to implement methods: they assisted in writing the code of the training, evaluation and analysis pipelines. They were also used to give feedback on the experimental methodology and on the interpretation of results: they reviewed protocols, suggested controls and checked that each claim in the text matched the recorded numerical exports. Which controls to run and the wording of every conclusion were decided by the authors. We have not used generative AI tools to assist with translation. Generating synthetic data, developing theoretical models, formulating mathematical claims or proofs, cleaning or reformatting datasets and qualitative data analysis are not applicable to this work, which makes no theoretical claim and uses only public benchmarks.

Additionally, we used generative AI tools to identify related work, which the authors then read and checked against the primary sources; to create and edit code, including the scripts that generate the figures and tables from the recorded runs; to format references; to draft parts of the manuscript, including sections of the appendices, which the authors then revised; and to edit the manuscript for readability, spelling and syntax.

We have reviewed all AI-assisted work. The runs were executed and audited by the authors, and the appendices record the protocols, the frozen numerical exports and the reproduction checks against which the reported numbers were verified. The research questions and the experimental design were decided by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

All experiments use public datasets: CIFAR-10, SVHN and STL-10. Section 3 defines the three procedures. Appendix A gives the reference training recipe, the insertion sites, the pooling windows and the executed schedules; Appendix C gives the per-setting adaptations and how the learning rates were chosen; Appendix D gives the objective, augmentation and input-resizing recipes; Appendix E gives the CBS and SDPoint ports, the comparison recipes and the four BatchNorm evaluation policies; Appendix F defines every geometric diagnostic, with its numerical checks; and Appendix G describes the order, single-resolution and validation controls. The supplementary material contains the anonymized code for training, evaluation and analysis, the run configurations, the per-run numerical exports from which every table and figure is generated, and the complete exploration catalogue.

REFERENCES

Eugene L. Allgower and Kurt Georg. *Numerical Continuation Methods: An Introduction*, volume 13 of *Springer Series in Computational Mathematics*. Springer, Berlin, Heidelberg, 1990.

Maksym Andriushchenko, Francesco Croce, Maximilian Müller, Matthias Hein, and Nicolas Flammarion. A modern look at the relationship between sharpness and generalization. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 840–902, 2023.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 41–48, 2009.

Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 215–223, 2011.

Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1019–1028, 2017.

Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry Vetrov, and Andrew Gordon Wilson. Loss surfaces, mode connectivity, and fast ensembling of DNNs. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Behrooz Ghorbani, Shankar Krishnan, and Ying Xiao. An investigation into neural net optimization via Hessian eigenvalue density. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2232–2241, 2019.

Çağlar Gülçehre, Marcin Moczulski, Francesco Visin, and Yoshua Bengio. Mollifying networks. In *International Conference on Learning Representations*, 2017.

Elad Hazan, Kfir Yehuda Levy, and Shai Shalev-Shwartz. On graduated optimization for stochastic non-convex problems. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 1833–1841, 2016.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (GELUs). *arXiv preprint arXiv:1606.08415*, 2016.

Elad Hoffer, Berry Weinstein, Itay Hubara, Tal Ben-Nun, Torsten Hoefler, and Daniel Soudry. Mix & Match: Training Convnets with mixed image sizes for improved accuracy, speed and scale resiliency. *arXiv preprint arXiv:1908.08986*, 2019.

Michael F. Hutchinson. A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. *Communications in Statistics—Simulation and Computation*, 18(3):1059–1076, 1989.

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.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

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

Jason Kuen, Xiangfei Kong, Zhe Lin, Gang Wang, Jianxiong Yin, Simon See, and Yap-Peng Tan. Stochastic downsampling for cost-adjustable inference and improved regularization in convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7929–7938, 2018.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2980–2988, 2017.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. In *International Conference on Learning Representations*, 2020.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Hagay Michaeli, Tomer Michaeli, and Daniel Soudry. Alias-free Convnets: Fractional shift invariance via polynomial activations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16333–16342, 2023.

Hossein Mobahi and John W. Fisher, III. A theoretical analysis of optimization by Gaussian continuation. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pp. 1205–1211, 2015.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nathan Srebro. Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Barak A. Pearlmutter. Fast exact multiplication by the Hessian. *Neural Computation*, 6(1):147–160, 1994.

Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5301–5310, 2019.

Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

Samarth Sinha, Animesh Garg, and Hugo Larochelle. Curriculum by smoothing. In *Advances in Neural Information Processing Systems*, volume 33, pp. 21653–21664, 2020.

Alexandros Stergiou, Ronald Poppe, and Grigorios Kalliatakis. Refining activation downsampling with SoftPool. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10357–10366, 2021.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the Inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826, 2016.

Mingxing Tan and Quoc Le. EfficientNetV2: Smaller models and faster training. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 10096–10106, 2021.

Cristina Vasconcelos, Hugo Larochelle, Vincent Dumoulin, Rob Romijnders, Nicolas Le Roux, and Ross Goroshin. Impact of aliasing on generalization in deep convolutional networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10529–10538, 2021.

Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European Conference on Computer Vision*, pp. 3–19, 2018.

Richard Zhang. Making convolutional networks shift-invariant again. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 7324–7334, 2019.

A REFERENCE PROTOCOL AND METHOD DEFINITIONS

The comparisons that support our final selection use the protocol below. The three retained methods and the plain control are defined as they were executed in the final selection batch (Appendix B). Insertion sites, schedules and the final bypass are specified below. Some generated tables write R, G, RG and Plain as Resolution, Gaussian, Combined and plain.

A.1 TRAINING RECIPE

Table 6 gives the reference training recipe.

Table 6: Reference training recipe.

Data	CIFAR-10, all 50,000 training and 10,000 test images; no augmentation; per-channel normalization with the statistics of the training images.
Model	ResNet-20 with BatchNorm, widths 16/32/64, option-A shortcuts (subsampling and zero-padding), 269,722 parameters; pinned initial weights per seed.
Optimizer	SGD, learning rate $\eta = 0.005$, momentum 0.9, weight decay 5×10^{-4} on all parameters; $\eta\,(u+1)/60$ for the first 60 updates, then cosine decay to zero over the global update index u.
Batches	A pinned permutation per epoch; batches of 128 images (the last one of 80), each split into microbatches of 32 whose losses are weighted by their number of examples. BatchNorm therefore normalizes over microbatches.
Budget	30 epochs of 391 updates, 11,730 updates in total.
Evaluation	Before training and after every epoch, on the test set and on a fixed probe of 500 training images, in eval mode with BatchNorm running statistics.

A.2 CONVENTIONS

Statistics and pairing. We report the record after epoch 30, without best-epoch selection, as the mean and sample standard deviation (SD) over seeds; a configuration with a single valid seed has no SD. A batch is a set of runs launched together; each of its configurations is an arm. Within a batch, runs with the same seed share their initial weights, BatchNorm buffers and data order, and we compute gains as means of per-seed paired differences. We do not pair runs across batches: plain runs with identical pinned initial weights and data order finished up to 0.73 points apart in different batches, for reasons we have not established. Bars and bands in the figures show ± 1 sample SD; they are not confidence intervals.

Evaluation paths. The *current path* evaluates the network with the intervention of the epoch just completed. The *target path* evaluates the same weights at the native resolution without added filters, which is the network kept at the end. After epoch 30 the two paths coincide for every retained method. Before that, the target path uses BatchNorm running statistics accumulated on the current path, so it combines the effect of removing the intervention with a mismatch of these statistics.

Training loss. In the selection batch, training cross-entropy is evaluated on a fixed probe of 500 training images. It is not the full-training-set CE. Full-training-set CE was evaluated later for the twenty models of the five-seed landscape study (Appendix F). The online mean loss during an epoch is another quantity: it is computed while the weights are changing.

Supplementary material. The supplementary material contains the anonymized code for training, evaluation and analysis, the run configurations, the per-run numerical exports and the scripts that generate every table and figure. It also contains the complete exploration catalogue, which retains all 165 configurations, 377 runs and 15 experiments, including failures and separate launches. The results needed for the conclusions of this paper are reported in these appendices.

A.3 INSERTION SITES AND THE REDUCTION POINT

Residual blocks are indexed from 0, so "after block 1" denotes the complete output of the second residual block, after its residual addition and final ReLU. The adaptive max-pooling is applied to this tensor as it enters the third block, so both branches of every later block, including the shortcut, see the reduced grid. For an input side n and output side r, the one-dimensional adaptive pooling window of output index $i \in \{0, \dots, r-1\}$ is $[\lfloor i\, n/r \rfloor, \lceil (i+1)n/r \rceil)$; the two-dimensional window is its Cartesian product across axes. Table 7 lists the sites in forward order. Table 8 gives the exact reference transitions.

Table 7: Insertion sites of the ResNet-20 reference model. Each residual block has two convolutions, and stride-2 convolutions open blocks 3 and 6. Map sides are given for $r = 16/24/32$ at the reduction point; the Gaussian-only method never reduces, so its maps are those of the $r = 32$ column. Post-ReLU sites are the stem ReLU output (p_0) and the output of each block (p_1–p_9).

Stage	Map side, $r = 16/24/32$	Conv outputs (COMBINED)	Post-ReLU (GAUSSIAN)
Stem	32 / 32 / 32	c_0	p_0
Blocks 0–1	32 / 32 / 32	c_1–c_4	p_1, p_2
Reduction (input of block 2)	16 / 24 / 32	—	—
Block 2	16 / 24 / 32	c_5, c_6	p_3
Blocks 3–5	8 / 12 / 16	c_7–c_{12}	p_4–p_6
Blocks 6–8	4 / 6 / 8	c_{13}–c_{18}	p_7–p_9

In the combined method, each Gaussian sits between a convolution and its BatchNorm, so BatchNorm normalizes filtered activations. In the Gaussian-only method, the filtered tensor is the block output that the next block, or the pooling layer for p_9, receives on both paths; the ReLU inside each block is not a site.

A.4 SCHEDULES, OPERATORS AND BYPASS

Table 8: Executed schedules, evaluated through the controller of the frozen presets. Epochs are zero-based; a state holds for a whole epoch of 391 updates. r is the side of the map entering residual block index 2 (RESOLUTION and COMBINED), G the reference Gaussian level. Effective σ is in pixels of the feature map at the site and is the same at every site of a method.

Epochs	First update	r	G	GAUSSIAN: σ (10 sites)	COMBINED: $\sigma = \frac{r}{32}G$ (19 sites)
0–2	0	16	1.00	1.000	0.500
3–5	1173	16	0.85	0.850	0.425
6–8	2346	24	0.70	0.700	0.525
9–11	3519	24	0.60	0.600	0.450
12–14	4692	32 (bypass)	0.50	0.500	0.500
15–17	5865	32 (bypass)	0.40	0.400	0.400
18–20	7038	32 (bypass)	0.30	0.300	0.300
21–29	8211	32 (bypass)	0.00	0 (bypass)	0 (bypass)

A state holds for a whole epoch: a transition at epoch e means that update $391e$ is the first to use the new state, and the record written after k completed epochs evaluates the current path under the state of epoch $k-1$.

The Gaussian kernel uses nine positions along each axis for every nonzero σ, $w_d \propto \exp(-d^2/(2\sigma^2))$ for $d = -4, \dots, 4$, normalized to sum one. It is applied separably and channel by channel, with reflection padding that does not repeat the edge sample; on 4×4 maps, where the radius equals the map side, the same reflection is applied by explicit indexing. The scale σ is measured in pixels of the feature map at the site.

The combined method uses $\sigma = (r/32)\, G$ at all 19 sites, including c_0–c_4, which remain at 32×32 upstream of the reduction. This rescaling comes from our earlier input-resolution arms (Appendix B) and makes the effective scale non-monotone: it increases from 0.425 to 0.525 when r changes from

16 to 24 at epoch 6. We keep this executed rule rather than a local width ratio, so the combined method is not the literal composition of the two other methods.

When $G = 0$, a Gaussian hook returns its input unchanged, and when $r = 32$ the reduction hook returns its input. In the final state $(r = 32, G = 0)$, the modified network therefore computes exactly the plain ResNet-20 when given the same weights and BatchNorm buffers. The trained predictors themselves need not coincide. Neither BatchNorm statistics nor optimizer state are reset at a transition.

B CIFAR-10 EXPLORATION AND METHOD SELECTION

We used CIFAR-10 to explore interventions, to understand the first results and to select three methods for further study. This was not one homogeneous benchmark. Early pilots used image subsets, a validation split, GroupNorm or ResNet-18, and different budgets. Their numbers cannot be placed on the same ranking as the later reference experiments. The final selection rests on one paired batch under Appendix A. The same test set informed successive decisions, so the resulting scores are exploratory evidence, not an independent validation of the selected methods. The numerical comparisons below retain the evidence used for selection; the full catalogue of exploratory runs is in the supplementary material.

B.1 FROM INPUT BLUR TO FEATURE-MAP FILTERING

We first filtered the input images, using ResNet-20 with GroupNorm, 45,000 training and 5,000 validation images, and 14,040 updates. Fixed blur lowered validation accuracy: 82.55% for $\sigma = 0.5$ and 71.33% for $\sigma = 1$, against 84.11% without blur (three seeds). Warm starts that blurred the images for the first 1,500 updates before removing the blur, in one or two steps, did not recover the unfiltered result (82.93% and 83.19%).

We then moved the Gaussian to the 19 convolution outputs, following the feature-map filtering idea of Sinha et al. (2020). On a 10,000-image subset with GroupNorm, it lowered validation accuracy ($46.91 \pm 0.60\%$ against $50.36 \pm 0.52\%$, three seeds). Two one-seed pilots with BatchNorm gave the opposite sign: 62.16% against 54.26% for ResNet-18, and 59.62% against 55.36% for ResNet-20. Several conditions changed, so these pilots do not isolate a normalization effect. Under the reference protocol, the Gaussian reached $78.41 \pm 0.32\%$ with plateaus and $78.46 \pm 0.72\%$ with geometric decay, against $75.40 \pm 0.40\%$ for plain. We continued with the reference ResNet-20-BN and plateau schedule.

B.2 PROCESSING SMALLER REPRESENTATIONS

In a one-seed reference pilot, antialiased bilinear resizing of the input to $16 \to 24 \to 32$ reached 79.57%, against 75.10% for plain. Adding convolution-output filtering, with scale multiplied by $r/32$, gave 79.93%. A grid of 21 configurations with three seeds varied input versus stem reduction, bilinear versus max-pooling, schedules, and Gaussian combinations. Input reduction with the Gaussian reached $80.71 \pm 0.65\%$, compared with $78.69 \pm 0.68\%$ for the Gaussian alone and $75.62 \pm 0.49\%$ for plain. This led us to study resolution reduction as an intervention in its own right.

B.3 OPERATORS, INSERTION POINTS AND SCHEDULES

An ablation batch suggested reducing after an early residual block. The subsequent resolution-only study compared 29 targeted configurations with three seeds: seven operators at the input and after block 1 (the operators are described in Appendix B.5), max-pooling at five locations, four schedules for selected operators, and fixed-resolution controls. This was not a full factorial design. Of 87 runs, 84 were numerically valid; the perceptual operator after block 1 produced non-finite losses in all three seeds.

Max-pooling after block 1 had the highest mean among its locations ($80.60 \pm 0.20\%$). The six valid operators at that point had overlapping seed ranges and none exceeded its mean (Table 9). At the input, the ordering differed: max-pooling was below every other valid operator except MaxBlur, and

Table 9: Resolution-only batch (no additional Gaussian continuation): reduction operator at two locations, schedule $16 \to 24 \to 32$. Final test accuracy after 30 epochs, mean $\pm$ sample SD over three seeds. Plain arm of the batch: $75.53 \pm 0.36\%$. The perceptual operator after block 1 produced a non-finite loss in all three seeds.

Reduction operator	After block 1 (%)	At the input (%)
Adaptive max-pool	80.60 ± 0.20	79.23 ± 0.48
MaxBlur (max, binomial blur)	80.43 ± 0.75	79.17 ± 0.43
SoftPool	80.34 ± 0.48	79.28 ± 0.46
Bilinear (antialiased)	80.23 ± 0.38	79.56 ± 0.68
L^2 reconstruction	80.11 ± 0.52	79.64 ± 0.33
$\dot{H}^{-1}$ reconstruction	79.85 ± 0.76	79.76 ± 0.65
Perceptual (SSIM-style)	Diverged (0/3 valid)	79.98 ± 0.40

Table 10: Endpoints of the final selection batch: CIFAR-10 / ResNet-20-BN / SGD, 30 epochs. Mean $\pm$ sample SD over three seeds; gains are paired to Plain. Test and 500-image training-probe CE use saved BatchNorm statistics. Last-epoch loss is the online mean while weights are updated. All losses are in nats; full-training-set CE was not evaluated in this batch. Bold accuracy means exceed Plain.

Method	Test acc. (%)	Gain (pp)	Test CE	Probe CE	Last-epoch loss
Plain	75.43 ± 0.76	—	0.762 ± 0.012	0.211 ± 0.024	0.274 ± 0.022
Resolution	$\mathbf{80.37} \pm 0.61$	$+4.94 \pm 0.50$	0.578 ± 0.011	0.321 ± 0.011	0.358 ± 0.008
Gaussian	$\mathbf{79.91} \pm 0.27$	$+4.48 \pm 0.97$	0.593 ± 0.008	0.281 ± 0.006	0.339 ± 0.002
Combined	$\mathbf{81.51} \pm 0.22$	$+6.08 \pm 0.56$	0.542 ± 0.006	0.318 ± 0.005	0.369 ± 0.002

the perceptual and homogeneous $\dot{H}^{-1}$ reconstructions had the highest means. We do not interpret these small differences as a definitive operator ranking.

For max-pooling, the progressive $16 \to 24 \to 32$ schedule scored above the tested delayed, shorter and reversed schedules. These schedules were not exposure-matched; see Appendix G. MaxBlur behaved differently: the reversed order gave 80.46%, against 80.43% for the progressive order. Keeping the reduced grid to the end gave 76.15% at 24 and 73.13% at 16. These controls retain a different inference computation, so they do not isolate restoration of resolution at equal inference cost. We retained adaptive max-pooling after block 1 as a simple, competitive operator.

B.4 Final paired comparison

Table 10 reports the retained methods from the last selection batch, which contained 16 configurations and three seeds with matched initial weights and data order. Figure 2 shows their learning curves. The full batch compared plain, Gaussian placement, reduction locations and combinations with different per-site scale profiles. Without reduction, post-ReLU filtering scored above convolution-output filtering (79.91% versus 78.37%; paired difference 1.54 ± 0.88 points). With reduction after block 1, the order reversed: convolution-output filtering reached 81.51% against 81.00% for post-ReLU filtering, a paired difference of 0.51 ± 0.24 points. Both differences were positive in all three seeds. Per-site scale profiles did not give a consistent further benefit; six other combined arms finished within 0.6 points of the highest mean.

We selected one representative per family: max-pooling after block 1; the uniform Gaussian at ten post-ReLU sites; and max-pooling after block 1 with filtering at 19 convolution outputs. The uniform post-ReLU method was the better of the two basic Gaussian placements in this paired batch. It was not the highest mean among every historical Gaussian profile, and the selected procedures should not be called established family optima.

B.5 Other explored operators

SoftPool uses exponentially weighted local activations (Stergiou et al., 2021); MaxBlur combines max-pooling with binomial smoothing. The reconstruction operators address a different question:

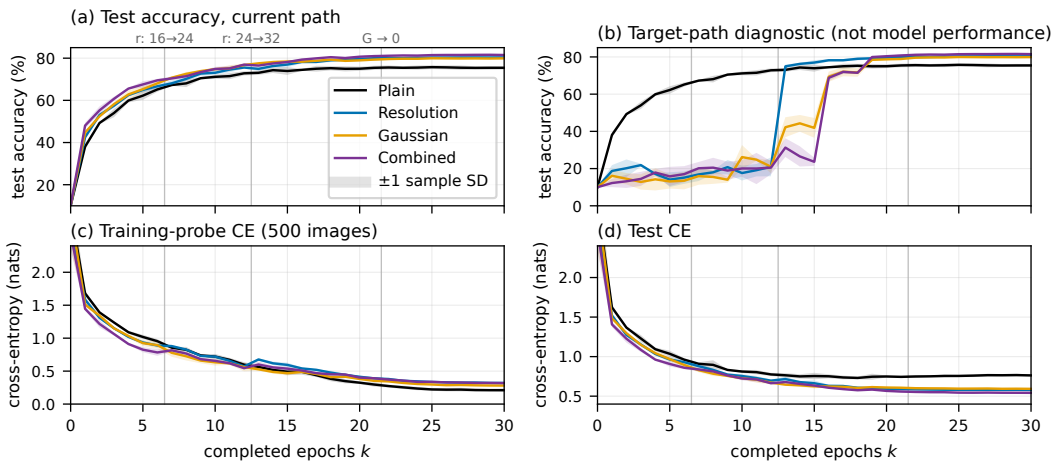


Figure 2: Selection-batch training records, means over three seeds. The current path retains the intervention of the epoch just completed; the target path removes it using saved BatchNorm statistics. Before restoration, the target diagnostic mixes the changed computation with a normalization-statistics mismatch. Training-probe CE uses 500 fixed images.

which low-resolution coefficients best reconstruct a channel h through bilinear upsampling U_r? They minimize

$$\tfrac{1}{2}(U_r z - h)^\top Q(U_r z - h), \qquad \mathbf{1}^\top (U_r z - h) = 0,$$

with $Q = I$ or the pseudoinverse of a spatial Laplacian. This is a quadratic reconstruction objective on a smaller coefficient grid. The perceptual variant uses a reconstruction criterion in the style of the structural similarity index (SSIM); its divergence inside the network is retained as a failed experiment, not removed from the catalogue.

A one-seed pilot with a Daubechies-2 (db2) wavelet reduction on a 10,000-image subset reached 56.34% validation accuracy against 55.40% for plain, at $3856\,\mathrm{s}$ versus $89\,\mathrm{s}$ per run. Finally, longer one-seed Gaussian runs illustrate budget dependence: at 120 epochs, the convolution-output plateau schedule gave 80.68% against 77.10% at the reference learning rate, and 87.16% against 86.04% at ten times that rate. These exploratory controls are not equally tuned comparisons of the final three methods.

C Transfer protocols and endpoint metrics

The transfer matrix is targeted: it does not cross every dataset with every architecture and optimizer. Within each row, all four methods use the same data, budget, initial learned parameters and sample order for each seed. Architectures with different parameter shapes require new initial weights. The twenty settings in Table 2 use cross-entropy and no augmentation; the additional objective and augmentation experiments are described separately in Appendix D.

C.1 Architecture and dataset adaptations

Full-data CIFAR-10 uses 50,000 training and 10,000 test images. The original optimizer comparison uses learning rates 0.005, 0.003, 0.02 and 0.003 for SGD, Adam, AdamW and RAdam. The numerical weight decay is 5×10^{-4}, with the algorithm-specific meaning of that parameter. Batch size is 128 accumulated over microbatches of 32, with a 60-update warm-up and cosine decay. All activation, normalization and architecture variants use SGD at 0.005 for 30 epochs.

GELU and SiLU replace the functional activations, including those at the reduction and Gaussian sites. They retain the pinned initial weights and Kaiming-ReLU initialization gain. R still follows the second block's final activation; G follows the stem and nine final block activations; RG filters the 19 convolution outputs before normalization. GroupNorm uses eight groups and new initial

weights, with the same data order and intervention sites. It has no running mean or variance to save or recalibrate.

VGG-11-BN uses eight convolutions, with G after their eight ReLU outputs and RG before their eight BatchNorm layers. R follows the first native max-pooling layer, whose usual side length is 16, and uses $8 \to 12 \to 16$. The reference side for the combined Gaussian coupling is 16, giving $\sigma = (r/16)\, G$ at its eight convolution sites, with r and G as in Appendix A.4. Later layers receive these maps directly; the four remaining native pools can bring them down to 1×1. The Gaussian is then an identity on a spatially constant single-pixel map. This changes the effective action of filtering relative to ResNet-20, and we do not interpret the architecture comparison as a controlled removal of residual connections. Its initial weights are new; data indices and ordering match the reference.

SVHN uses 50,000 of the 73,257 training images and all 26,032 test images, at 32×32. The reference schedule lasts 30 epochs, or 11,730 updates. STL-10 uses all 5,000 labelled training and 8,000 test images at 96×96. Its reference transfer lasts 60 epochs of 40 updates; the resolution schedule is $48 \to 72 \to 96$, Gaussian scales are multiplied by three, kernel support becomes 25 positions per axis, and epoch boundaries are doubled. The input-resolution scaling does not imply that the Gaussian has the same relative effect at every internal site.

The two reduced-data CIFAR-10 settings use the same 5,000 training examples and all 10,000 test images, with the ResNet-20-BN/ReLU model and SGD at 0.005. The 60-epoch setting uses 2,400 updates and the doubled schedule. The 293-epoch setting uses 11,720 updates, approximately matching the full-data reference's 11,730. Its reference boundaries are scaled by $391/40$ and rounded to whole epochs, giving transitions at updates 1,160, 2,360, 3,520, 4,680, 5,880, 7,040 and 8,200.

C.2 Additional optimizers and longer budgets

The new Adam and AdamW comparisons use learning rate 0.001, $(\beta_1, \beta_2) = (0.9, 0.999)$ and numerical $\epsilon = 10^{-8}$. Adam has zero weight decay; AdamW uses decoupled weight decay 0.01. The three datasets retain their own data and spatial adaptations above. Thus these rows are distinct recipes from the earlier Adam/AdamW study, not extra seeds for it.

CIFAR-10 at 90 epochs keeps SGD at 0.005 and stretches all intervention boundaries by three, for 35,190 updates. STL-10 at 120 epochs keeps the new Adam recipe and doubles its 60-epoch boundaries, for 4,800 updates. The learning-rate decay horizon also changes. On STL-10, G and RG improve strongly relative to their 60-epoch means, while Plain and R have lower means. Longer training is therefore not a uniform improvement of every arm, and this experiment does not retest RG's loss on STL-10 with SGD at a longer budget.

Selection and measurement. The earlier adaptive learning rates were selected using final Plain seed-0 test accuracy, with a 0.3-point near-tie preference for the lower rate. The chosen rate was shared by all methods; SGD was not retuned. These choices and the prior method selection reused the test set. Subsequent experiments are motivated follow-ups, not a fresh confirmatory sample of tasks.

Tables 11 and 12 report paired accuracy differences and endpoint CE. Training CE uses a fixed 500-image probe; for CIFAR-10 5k this probe is contained in the selected training subset. Per-channel normalization for those subsets uses the full CIFAR-10 training split. Test CE uses the complete test set. The three earlier adaptive-optimizer rows lack endpoint CE in the supplied records; later independently trained checkpoints do not fill those particular cells.

C.3 Observed computational cost

Table 13 gives total times for the transfer matrix. Its timer starts after data and model construction and includes evaluation and checkpoint writes. Evaluation frequency and the number of evaluated paths vary across campaigns. The reference used a Tesla T4, while some historical transfer records lack hardware identifiers; runs could share a device. These timings are observations of the executed recipes, not a controlled hardware benchmark.

Table 11: Within-seed accuracy gains over Plain for Table 2 (percentage points), mean ± SD. Parentheses count positive differences among three seeds. Each row uses its own matched control.

Model / norm / activation; optimizer LR; budget	R	G	RG
CIFAR-10 (50k / 10k) train / test			
ResNet-20 / BN / ReLU; SGD 0.005; 30 ep.	4.94 ± 0.50 (3/3)	4.48 ± 0.97 (3/3)	6.08 ± 0.56 (3/3)
ResNet-20 / BN / ReLU; SGD 0.005; 90 ep.	4.83 ± 0.05 (3/3)	5.43 ± 0.56 (3/3)	6.75 ± 0.27 (3/3)
ResNet-20 / BN / ReLU; Adam 0.003; 30 ep.	1.18 ± 0.40 (3/3)	-0.18 ± 0.61 (2/3)	1.15 ± 0.36 (3/3)
ResNet-20 / BN / ReLU; AdamW 0.02; 30 ep.	2.16 ± 0.68 (3/3)	-0.41 ± 0.72 (1/3)	2.09 ± 0.62 (3/3)
ResNet-20 / BN / ReLU; RAdam 0.003; 30 ep.	1.25 ± 0.53 (3/3)	0.37 ± 0.18 (3/3)	0.88 ± 0.26 (3/3)
ResNet-20 / BN / ReLU; Adam 0.001; 30 ep.	3.32 ± 0.53 (3/3)	2.53 ± 0.56 (3/3)	3.61 ± 0.28 (3/3)
ResNet-20 / BN / ReLU; AdamW 0.001; 30 ep.	3.67 ± 0.57 (3/3)	2.35 ± 0.40 (3/3)	4.23 ± 0.77 (3/3)
ResNet-20 / BN / GELU; SGD 0.005; 30 ep.	5.99 ± 0.48 (3/3)	5.48 ± 0.85 (3/3)	6.78 ± 0.48 (3/3)
ResNet-20 / BN / SiLU; SGD 0.005; 30 ep.	4.36 ± 0.96 (3/3)	3.90 ± 1.18 (3/3)	5.13 ± 0.72 (3/3)
ResNet-20 / GN / ReLU; SGD 0.005; 30 ep.	2.79 ± 0.80 (3/3)	1.16 ± 0.63 (3/3)	3.37 ± 0.67 (3/3)
VGG-11 / BN / ReLU; SGD 0.005; 30 ep.	3.10 ± 0.37 (3/3)	3.99 ± 0.63 (3/3)	3.88 ± 0.58 (3/3)
SVHN (50k / 26,032) train / test			
ResNet-20 / BN / ReLU; SGD 0.005; 30 ep.	1.16 ± 0.09 (3/3)	1.16 ± 0.12 (3/3)	1.74 ± 0.17 (3/3)
ResNet-20 / BN / ReLU; Adam 0.001; 30 ep.	0.38 ± 0.23 (3/3)	0.31 ± 0.02 (3/3)	0.51 ± 0.15 (3/3)
ResNet-20 / BN / ReLU; AdamW 0.001; 30 ep.	0.39 ± 0.24 (3/3)	0.35 ± 0.32 (2/3)	0.55 ± 0.20 (3/3)
STL-10 (5k / 8k) train / test			
ResNet-20 / BN / ReLU; SGD 0.005; 60 ep.	2.58 ± 0.13 (3/3)	1.09 ± 1.07 (3/3)	-1.29 ± 0.53 (0/3)
ResNet-20 / BN / ReLU; Adam 0.001; 60 ep.	5.92 ± 1.31 (3/3)	0.54 ± 2.22 (2/3)	-0.03 ± 1.32 (1/3)
ResNet-20 / BN / ReLU; AdamW 0.001; 60 ep.	5.65 ± 1.44 (3/3)	0.06 ± 1.68 (1/3)	-0.32 ± 1.32 (2/3)
ResNet-20 / BN / ReLU; Adam 0.001; 120 ep.	5.23 ± 2.43 (3/3)	5.30 ± 1.75 (3/3)	5.09 ± 1.58 (3/3)
CIFAR-10 (5k / 10k) train / test			
ResNet-20 / BN / ReLU; SGD 0.005; 60 ep.	6.28 ± 0.78 (3/3)	8.64 ± 1.22 (3/3)	10.62 ± 0.35 (3/3)
ResNet-20 / BN / ReLU; SGD 0.005; 293 ep.	4.23 ± 0.21 (3/3)	10.10 ± 0.87 (3/3)	11.78 ± 0.06 (3/3)

For the 160-epoch literature comparison, Table 25 separates the training-loop timer from total elapsed time and records peak allocated GPU memory. Training-loop time excludes evaluation but includes checkpoint writes and is not a pure kernel-compute measurement. The memory measure is the allocator's peak, not total device reservation. No energy consumption was measured. Only R, G, RG and the explicitly unfiltered comparator endpoints (Appendix E.3) are guaranteed to remove all added inference operations.

Timing on a dedicated device. A later campaign re-times the 160-epoch augmented SGD recipe of Appendix E.2 under controlled conditions: one Tesla T4 with no other task on the device, strictly sequential runs, three seeds, and a run order rotated by seed so that no method is favoured by its position. The timer covers the training loop alone and excludes evaluation, callbacks and the final calibration policies P0 (saved statistics), P1 and P3, which are defined in Appendix E.3. Before the runs, a pilot predicted each arm's cost per epoch and fixed an equal-time budget; the arms, their measured times and their paired differences against Plain appear in Table 14. R completes the recipe in 0.87 times Plain's loop time and is not more accurate there: -0.21 ± 0.36 points under P3, with one positive seed pair of three. SDPoint is faster by the same margin, and its ordering against Plain depends on the normalization policy, as elsewhere in this paper. P1 is about two points below P0 and P3 for every arm here, which is expected when a network trained on augmented images is recalibrated on clean ones; P3 is the policy to read in this recipe.

Table 15 decomposes the loop time by schedule phase. R's native phase costs what Plain costs, to within the spread of the timing windows, so the saving comes entirely from the epochs spent at reduced resolution. The saving available to any such schedule is therefore bounded by the fraction of the budget it spends below native resolution. The intended equal-time comparison for R did not take place. The pilot returned the same cost for all three of R's resolutions, equal to the measured cost at $r = 16$, so the rule granted the extra-budget arm 252 epochs instead of roughly 184, and it received 1.36 times Plain's time rather than the same. Its learning-rate schedule is stretched accordingly, so it is not a longer run of the same recipe either. The bias favours R, and even so its advantage is not resolved: $+0.05 \pm 0.32$ points under P3, positive in one seed pair of three. SDPoint's matched arm is unaffected, at 0.98 times Plain's time. No equal-time contrast for R can be read from these data,

Table 12: Final training-probe and test cross-entropy for the no-augmentation transfer matrix, mean ± SD over three seeds. ResNet-20-BN/ReLU is used unless the row specifies a variant. The training probe contains 500 fixed images; the test set is complete. All models use saved normalization statistics.

Recipe (model; optimizer LR; epochs)	Split	Plain	R	G	RG
CIFAR-10 (50k / 10k) train / test					
SGD 0.005; 30 ep.	Train	0.211 ± 0.024	0.321 ± 0.011	0.281 ± 0.006	0.318 ± 0.005
	Test	0.762 ± 0.012	0.578 ± 0.011	0.593 ± 0.008	0.542 ± 0.006
SGD 0.005; 90 ep.	Train	0.0059 ± 0.0082	0.0085 ± 0.0014	0.0068 ± 0.0011	0.018 ± 0.011
	Test	1.024 ± 0.015	0.726 ± 0.003	0.709 ± 0.027	0.607 ± 0.005
Adam 0.001; 30 ep.	Train	0.0049 ± 0.0004	0.047 ± 0.007	0.178 ± 0.022	0.118 ± 0.010
	Test	0.906 ± 0.010	0.593 ± 0.019	0.570 ± 0.013	0.536 ± 0.011
AdamW 0.001; 30 ep.	Train	0.0062 ± 0.0015	0.046 ± 0.008	0.162 ± 0.025	0.123 ± 0.019
	Test	0.899 ± 0.007	0.587 ± 0.016	0.574 ± 0.007	0.527 ± 0.005
ResNet-20 / BN / GELU; SGD 0.005; 30 ep.	Train	0.220 ± 0.047	0.307 ± 0.012	0.255 ± 0.021	0.327 ± 0.008
	Test	0.792 ± 0.030	0.571 ± 0.009	0.590 ± 0.010	0.539 ± 0.003
ResNet-20 / BN / SiLU; SGD 0.005; 30 ep.	Train	0.255 ± 0.010	0.322 ± 0.027	0.294 ± 0.023	0.340 ± 0.006
	Test	0.728 ± 0.028	0.557 ± 0.004	0.581 ± 0.018	0.532 ± 0.005
ResNet-20 / GN / ReLU; SGD 0.005; 30 ep.	Train	0.514 ± 0.025	0.479 ± 0.007	0.497 ± 0.009	0.472 ± 0.034
	Test	0.714 ± 0.022	0.635 ± 0.011	0.674 ± 0.012	0.614 ± 0.001
VGG-11 / BN / ReLU; SGD 0.005; 30 ep.	Train	0.000157 ± 0.000024	0.000746 ± 0.000028	0.000743 ± 0.000130	0.0021 ± 0.0003
	Test	0.881 ± 0.032	0.668 ± 0.019	0.641 ± 0.012	0.593 ± 0.009
SVHN (50k / 26,032) train / test					
SGD 0.005; 30 ep.	Train	0.034 ± 0.001	0.105 ± 0.007	0.070 ± 0.008	0.107 ± 0.011
	Test	0.248 ± 0.002	0.202 ± 0.003	0.202 ± 0.002	0.183 ± 0.006
Adam 0.001; 30 ep.	Train	0.000254 ± 0.000051	0.0028 ± 0.0003	0.0039 ± 0.0005	0.0095 ± 0.0025
	Test	0.290 ± 0.006	0.225 ± 0.004	0.233 ± 0.008	0.208 ± 0.003
AdamW 0.001; 30 ep.	Train	0.000216 ± 0.000013	0.0022 ± 0.0001	0.0036 ± 0.0004	0.0094 ± 0.0022
	Test	0.286 ± 0.013	0.226 ± 0.006	0.232 ± 0.003	0.205 ± 0.006
STL-10 (5k / 8k) train / test					
SGD 0.005; 60 ep.	Train	0.903 ± 0.019	0.832 ± 0.010	0.979 ± 0.035	1.101 ± 0.025
	Test	1.156 ± 0.014	1.084 ± 0.015	1.143 ± 0.032	1.212 ± 0.019
Adam 0.001; 60 ep.	Train	0.183 ± 0.047	0.350 ± 0.025	0.799 ± 0.025	0.855 ± 0.015
	Test	1.131 ± 0.053	0.889 ± 0.007	1.018 ± 0.019	1.035 ± 0.007
AdamW 0.001; 60 ep.	Train	0.185 ± 0.038	0.339 ± 0.060	0.793 ± 0.030	0.852 ± 0.039
	Test	1.111 ± 0.036	0.899 ± 0.028	1.019 ± 0.015	1.037 ± 0.016
Adam 0.001; 120 ep.	Train	0.0059 ± 0.0018	0.012 ± 0.003	0.477 ± 0.039	0.563 ± 0.019
	Test	1.571 ± 0.070	1.182 ± 0.080	0.923 ± 0.009	0.925 ± 0.020
CIFAR-10 (5k / 10k) train / test					
SGD 0.005; 60 ep.	Train	0.218 ± 0.070	0.432 ± 0.040	0.362 ± 0.028	0.457 ± 0.018
	Test	1.688 ± 0.094	1.329 ± 0.016	1.263 ± 0.005	1.175 ± 0.045
SGD 0.005; 293 ep.	Train	0.000589 ± 0.000036	0.000980 ± 0.000103	0.0022 ± 0.0002	0.0030 ± 0.0003
	Test	2.454 ± 0.059	2.081 ± 0.080	1.662 ± 0.016	1.538 ± 0.023

The original Adam (0.003), AdamW (0.02) and RAdam (0.003) exports contain accuracy but no endpoint CE. Their CE is not inferred from later runs; the separately trained literature-comparison cohort is reported in Appendix E.

Table 13: Measured total running time (minutes) for Table 2, mean ± SD over three seeds. Timers include evaluation and checkpoint writes; evaluation frequency and execution conditions differ across campaigns.

Recipe (model; optimizer LR; epochs)	Plain	R	G	RG
CIFAR-10 (50k / 10k) train / test				
ResNet-20 / BN / ReLU; SGD 0.005; 30 ep.	8.4 ± 0.2	8.3 ± 0.4	10.4 ± 0.1	12.2 ± 0.3
ResNet-20 / BN / ReLU; SGD 0.005; 90 ep.	22.3 ± 1.3	22.1 ± 0.9	28.2 ± 0.6	34.6 ± 3.1
ResNet-20 / BN / ReLU; Adam 0.003; 30 ep.	8.2 ± 0.1	8.3 ± 0.2	10.4 ± 0.1	12.3 ± 0.1
ResNet-20 / BN / ReLU; AdamW 0.02; 30 ep.	8.3 ± 0.1	8.2 ± 0.3	10.3 ± 0.0	12.1 ± 0.0
ResNet-20 / BN / ReLU; RAdam 0.003; 30 ep.	8.3 ± 0.3	8.2 ± 0.2	10.5 ± 0.1	12.3 ± 0.1
ResNet-20 / BN / ReLU; Adam 0.001; 30 ep.	7.7 ± 0.5	7.8 ± 0.2	10.3 ± 0.4	12.1 ± 0.6
ResNet-20 / BN / ReLU; AdamW 0.001; 30 ep.	7.5 ± 0.1	8.2 ± 0.5	9.9 ± 0.3	12.1 ± 0.2
ResNet-20 / BN / GELU; SGD 0.005; 30 ep.	7.2 ± 0.1	7.6 ± 0.1	10.1 ± 0.9	11.5 ± 0.3
ResNet-20 / BN / SiLU; SGD 0.005; 30 ep.	7.8 ± 0.7	8.0 ± 0.5	9.8 ± 0.3	11.7 ± 0.4
ResNet-20 / GN / ReLU; SGD 0.005; 30 ep.	8.0 ± 0.6	7.7 ± 0.1	10.3 ± 0.4	11.8 ± 0.3
VGG-11 / BN / ReLU; SGD 0.005; 30 ep.	10.1 ± 0.7	9.8 ± 0.1	14.1 ± 0.6	13.2 ± 0.4
SVHN (50k / 26,032) train / test				
ResNet-20 / BN / ReLU; SGD 0.005; 30 ep.	8.2 ± 0.3	8.3 ± 0.1	11.2 ± 0.1	12.6 ± 0.2
ResNet-20 / BN / ReLU; Adam 0.001; 30 ep.	8.1 ± 0.1	8.3 ± 0.1	11.4 ± 0.1	12.6 ± 0.3
ResNet-20 / BN / ReLU; AdamW 0.001; 30 ep.	8.2 ± 0.4	8.3 ± 0.1	11.2 ± 0.2	13.0 ± 0.6
STL-10 (5k / 8k) train / test				
ResNet-20 / BN / ReLU; SGD 0.005; 60 ep.	10.4 ± 0.4	10.4 ± 0.2	17.4 ± 0.9	18.6 ± 0.3
ResNet-20 / BN / ReLU; Adam 0.001; 60 ep.	10.6 ± 0.6	10.5 ± 0.6	17.6 ± 0.3	18.7 ± 0.4
ResNet-20 / BN / ReLU; AdamW 0.001; 60 ep.	10.7 ± 0.2	10.5 ± 0.6	18.2 ± 0.1	18.4 ± 0.5
ResNet-20 / BN / ReLU; Adam 0.001; 120 ep.	20.1 ± 0.3	18.7 ± 0.1	32.3 ± 1.7	35.0 ± 1.2
CIFAR-10 (5k / 10k) train / test				
ResNet-20 / BN / ReLU; SGD 0.005; 60 ep.	1.6 ± 0.0	1.8 ± 0.0	2.3 ± 0.1	2.7 ± 0.1
ResNet-20 / BN / ReLU; SGD 0.005; 293 ep.	7.4 ± 0.2	7.5 ± 0.1	10.0 ± 0.6	11.7 ± 0.8

Table 14: Training time and paired accuracy differences against Plain in the 160-epoch augmented SGD recipe, three seeds on one dedicated Tesla T4 with sequential runs. Loop time excludes evaluation. Superscripts count positive seed pairs. The extra-time row received 1.36 times Plain's time, not the intended match; see the text.

Arm	Epochs	Loop (min)	×Plain	P0 − Plain	P1 − Plain	P3 − Plain
Plain	160	38.9	1.00	—	—	—
R	160	33.8	0.87	$-0.23 \pm 0.31^{\,0/3}$	$-0.40 \pm 0.59^{\,1/3}$	$-0.21 \pm 0.36^{\,1/3}$
R, extra time	252	53.1	1.36	$+0.14 \pm 0.31^{\,2/3}$	$-0.55 \pm 1.25^{\,1/3}$	$+0.05 \pm 0.32^{\,1/3}$
SDPoint	160	34.0	0.87	$-2.14 \pm 1.22^{\,0/3}$	$-1.49 \pm 1.23^{\,0/3}$	$+0.15 \pm 0.48^{\,2/3}$
SDPoint, matched	179	38.1	0.98	$-0.96 \pm 0.45^{\,0/3}$	$-1.56 \pm 0.81^{\,0/3}$	$+0.16 \pm 0.32^{\,2/3}$

Table 15: Measured seconds per epoch by schedule phase in the same runs, over the stated number of timing windows. R's native phase costs what Plain costs, so its saving comes from the reduced-resolution epochs alone.

Arm	Phase	Seconds per epoch	Windows
Plain	whole run	14.60 ± 0.08	117
R	$r = 16$, first 20%	8.70 ± 0.18	21
R	$r = 24$, next 20%	10.66 ± 0.07	24
R	native, last 60%	14.65 ± 0.07	72
SDPoint	whole run	12.76 ± 0.13	117

and none is claimed. The timings of Table 4 come from a separate campaign and are not replaced by these; the two agree on the ratio, 0.86 and 0.87 of Plain's loop time for R in this recipe.

D ADDITIONAL OBJECTIVES, AUGMENTATION AND INPUT RESIZING

These studies retain their own matched Plain controls and three seeds. They address specific omissions in the earlier matrix; their different schedules and optimizer recipes should not be combined into one universal ranking.

D.1 TRAINING OBJECTIVES

All objective experiments use CIFAR-10 (50k training / 10k test), ResNet-20-BN/ReLU, no augmentation and 30 epochs of the reference SGD recipe. Label smoothing uses smoothing weight 0.1 (Szegedy et al., 2016); focal loss uses exponent $\gamma = 2$ (Lin et al., 2017). The squared-logit objective is

$$\ell(z, y) = \frac{1}{10} \sum_{c=1}^{10} (z_c - \mathbf{1}\{c = y\})^2,$$

where z contains logits. This is not the squared error of softmax probabilities. Table 16 retains all six squared-loss learning rates, including poorly performing and adverse comparisons.

Table 16: Changing the training objective: CIFAR-10 (50k / 10k), ResNet-20-BN/ReLU, SGD, 30 epochs, no augmentation, saved BatchNorm. Test accuracy (%), mean ± SD over three seeds. Bold means exceed Plain at the same learning rate.

Training objective	LR	Plain	R	G	RG
Label smoothing 0.1	0.005	74.84 ± 0.97	$\mathbf{79.03} \pm 0.55$	$\mathbf{79.00} \pm 0.39$	$\mathbf{80.19} \pm 0.29$
Focal ($\gamma = 2$)	0.005	72.86 ± 0.55	$\mathbf{77.74} \pm 0.52$	$\mathbf{77.58} \pm 0.93$	$\mathbf{79.24} \pm 0.35$
Squared logits	0.0025	41.11 ± 1.11	$\mathbf{41.22} \pm 0.41$	$\mathbf{41.79} \pm 0.95$	$\mathbf{41.83} \pm 0.96$
Squared logits	0.005	46.98 ± 0.41	$\mathbf{47.71} \pm 0.53$	$\mathbf{48.86} \pm 0.94$	$\mathbf{48.72} \pm 0.58$
Squared logits	0.01	54.99 ± 0.60	$\mathbf{55.83} \pm 0.68$	$\mathbf{59.18} \pm 0.74$	$\mathbf{58.04} \pm 0.28$
Squared logits	0.05	80.15 ± 0.31	$\mathbf{82.75} \pm 0.86$	$\mathbf{82.39} \pm 0.34$	$\mathbf{83.46} \pm 0.14$
Squared logits	0.1	85.38 ± 0.35	$\mathbf{86.09} \pm 0.30$	$\mathbf{86.51} \pm 0.26$	$\mathbf{86.53} \pm 0.18$
Squared logits	0.5	86.20 ± 0.49	86.13 ± 0.19	85.55 ± 0.20	86.16 ± 0.18

All three interventions help with label smoothing and focal loss. With squared logits, gains remain at learning rates 0.05 and 0.1 but disappear at 0.5. At small rates, Plain's low accuracy indicates an unfavorable recipe; it is not evidence that squared-error training intrinsically fails. Cross-entropy evaluated through softmax is a common diagnostic in Table 19, but it is not the loss optimized in these squared-logit runs. These observations do not isolate the training objective from learning-rate suitability.

D.2 AUGMENTATION WITH MATCHED CONTROLS

Augmentation zero-pads four pixels, draws a 32×32 crop, then horizontally flips with probability $1/2$, before normalization. Evaluation uses clean images. All runs retain physical microbatches of 32, effective batches of 128, momentum 0.9 and weight decay 5×10^{-4}, with the 60-update warm-up and cosine decay spanning their declared budget.

Table 17 includes both 30- and 60-epoch budgets at learning rates 0.005 and 0.01, plus a 60-epoch no-augmentation control at 0.01. Crucially, this study keeps R's restoration at epoch 12 and G's removal at epoch 21 even in the 60-epoch runs. It therefore differs from the fractionally stretched schedules of the 90- and 120-epoch studies (Appendix C) and of the 160-epoch comparison (Appendix E.2). Paired gains appear in Table 18; common endpoint CEs are in Table 19.

The decrease in gain with augmentation is visible against the matching no-augmentation control, not only against a separately tuned historical baseline. The small positive means for R and RG at 0.01 do not persist with the same sign in every seed. This limits a claim of additional regularization benefit, without establishing which interaction with cropping, optimization or the schedule is responsible.

Table 17: Augmentation and its matched 60-epoch control: CIFAR-10 (50k / 10k), ResNet-20-BN/ReLU, SGD and cross-entropy. Final test accuracy (%), mean ± SD over three seeds, with saved BatchNorm. Bold means exceed Plain. Intervention boundaries remain at their original absolute epochs in both budgets.

Augmentation	Epochs	LR	Plain	R	G	RG
Crop + flip	30	0.005	79.78 ± 0.13	$\mathbf{80.20} \pm 0.18$	$\mathbf{80.64} \pm 0.18$	$\mathbf{81.10} \pm 0.42$
Crop + flip	30	0.01	84.70 ± 0.10	$\mathbf{84.78} \pm 0.38$	83.74 ± 0.29	$\mathbf{84.95} \pm 0.36$
Crop + flip	60	0.005	85.51 ± 0.51	$\mathbf{85.70} \pm 0.21$	85.42 ± 0.36	$\mathbf{86.04} \pm 0.09$
Crop + flip	60	0.01	88.27 ± 0.24	$\mathbf{88.45} \pm 0.18$	87.83 ± 0.27	$\mathbf{88.39} \pm 0.13$
None	60	0.01	80.27 ± 0.27	$\mathbf{83.52} \pm 0.20$	$\mathbf{82.84} \pm 0.37$	$\mathbf{84.51} \pm 0.35$

Table 18: Paired accuracy gains for Table 17, in percentage points. Mean ± SD and positive seed counts.

Recipe	R	G	RG
Crop + flip; 30 ep.; LR 0.005	0.42 ± 0.14 (3/3)	0.86 ± 0.30 (3/3)	1.32 ± 0.33 (3/3)
Crop + flip; 30 ep.; LR 0.01	0.09 ± 0.36 (2/3)	-0.96 ± 0.20 (0/3)	0.25 ± 0.43 (2/3)
Crop + flip; 60 ep.; LR 0.005	0.19 ± 0.45 (2/3)	-0.09 ± 0.63 (1/3)	0.52 ± 0.53 (3/3)
Crop + flip; 60 ep.; LR 0.01	0.19 ± 0.39 (2/3)	-0.44 ± 0.35 (0/3)	0.12 ± 0.12 (2/3)
None; 60 ep.; LR 0.01	3.25 ± 0.33 (3/3)	2.57 ± 0.29 (3/3)	4.24 ± 0.25 (3/3)

Table 19: Common cross-entropy measurements for Tables 16 and 17, mean ± SD over three seeds. Train uses 500 clean images; Test uses all 10,000. For squared-logit training, CE is evaluated through softmax and is not the optimized loss.

Training recipe	Split	Plain	R	G	RG
Label smoothing 0.1, LR 0.005	Train	0.449 ± 0.026	0.516 ± 0.012	0.451 ± 0.002	0.489 ± 0.003
	Test	0.771 ± 0.018	0.664 ± 0.010	0.664 ± 0.007	0.640 ± 0.005
Focal ($\gamma = 2$), LR 0.005	Train	0.437 ± 0.015	0.509 ± 0.014	0.470 ± 0.021	0.496 ± 0.003
	Test	0.779 ± 0.017	0.665 ± 0.009	0.670 ± 0.016	0.639 ± 0.005
Squared logits, LR 0.0025	Train	2.170 ± 0.008	2.168 ± 0.006	2.162 ± 0.007	2.165 ± 0.008
	Test	2.164 ± 0.007	2.161 ± 0.003	2.159 ± 0.006	2.160 ± 0.003
Squared logits, LR 0.005	Train	2.118 ± 0.004	2.116 ± 0.002	2.093 ± 0.006	2.106 ± 0.011
	Test	2.113 ± 0.003	2.108 ± 0.003	2.089 ± 0.008	2.095 ± 0.002
Squared logits, LR 0.01	Train	2.038 ± 0.005	2.037 ± 0.006	1.986 ± 0.010	2.011 ± 0.010
	Test	2.035 ± 0.004	2.027 ± 0.004	1.983 ± 0.005	1.998 ± 0.002
Squared logits, LR 0.05	Train	1.552 ± 0.009	1.564 ± 0.006	1.559 ± 0.005	1.567 ± 0.004
	Test	1.699 ± 0.006	1.670 ± 0.007	1.676 ± 0.006	1.664 ± 0.003
Squared logits, LR 0.1	Train	1.478 ± 0.011	1.488 ± 0.005	1.484 ± 0.006	1.500 ± 0.007
	Test	1.632 ± 0.006	1.626 ± 0.002	1.620 ± 0.002	1.626 ± 0.003
Squared logits, LR 0.5	Train	1.487 ± 0.006	1.501 ± 0.012	1.506 ± 0.010	1.503 ± 0.008
	Test	1.626 ± 0.003	1.629 ± 0.007	1.635 ± 0.006	1.628 ± 0.006
Crop + flip, 30 ep., LR 0.005	Train	0.550 ± 0.021	0.532 ± 0.002	0.504 ± 0.006	0.512 ± 0.020
	Test	0.591 ± 0.007	0.573 ± 0.010	0.570 ± 0.014	0.554 ± 0.009
Crop + flip, 30 ep., LR 0.01	Train	0.367 ± 0.017	0.368 ± 0.026	0.365 ± 0.024	0.355 ± 0.007
	Test	0.451 ± 0.007	0.454 ± 0.011	0.476 ± 0.003	0.445 ± 0.005
Crop + flip, 60 ep., LR 0.005	Train	0.300 ± 0.011	0.308 ± 0.014	0.272 ± 0.009	0.269 ± 0.023
	Test	0.436 ± 0.009	0.426 ± 0.006	0.439 ± 0.010	0.422 ± 0.003
Crop + flip, 60 ep., LR 0.01	Train	0.156 ± 0.010	0.164 ± 0.008	0.177 ± 0.020	0.159 ± 0.006
	Test	0.361 ± 0.007	0.357 ± 0.002	0.369 ± 0.003	0.354 ± 0.004
None, 60 ep., LR 0.01	Train	0.0013 ± 0.0001	0.0019 ± 0.0002	0.0025 ± 0.0004	0.0023 ± 0.0000
	Test	0.830 ± 0.013	0.664 ± 0.003	0.688 ± 0.016	0.615 ± 0.017

D.3 Fixed input-resolution schedules on STL-10

Table 20 reports a separate study of antialiased bilinear resizing of the input images. It uses STL-10 (5k / 8k), ResNet-20-BN/ReLU, SGD at 0.005 and no augmentation. These fixed schedules are not the internal max-pooling method R. The three-level 60-epoch schedule uses side lengths 48 for 12 epochs, 72 for 12, then 96. The four-stage version uses 48, 60, 72 and 84 for six epochs each before restoring 96. A linear version increases through even sizes during the first 24 epochs.

At 60 epochs, the progressive and four-stage means exceed the native-resolution control by 1.10 and 1.21 points while taking less training time. Longer progressive runs reach about 60.3–60.5%, but

Table 20: Fixed input-resolution schedules on STL-10 (5k / 8k), ResNet-20-BN/ReLU, SGD 0.005 without augmentation, saved BatchNorm. Mean ± SD over three seeds. These runs resize images with antialiased bilinear interpolation; they do not use the internal max-pooling procedure R.

Input schedule	Epochs	Test acc. (%)	Train CE	Test CE	Train (min)
Native 96	60	57.67 ± 1.09	0.968 ± 0.043	1.144 ± 0.029	9.04 ± 0.33
48 / 72 / 96	60	58.77 ± 0.78	0.974 ± 0.011	1.123 ± 0.009	7.08 ± 0.28
48 / 60 / 72 / 84 / 96	60	58.88 ± 0.48	0.986 ± 0.025	1.127 ± 0.008	7.55 ± 0.30
Linear ramp to 96	60	58.05 ± 0.81	1.013 ± 0.042	1.138 ± 0.025	7.66 ± 0.30
Native 96	72	59.52 ± 1.10	0.815 ± 0.043	1.098 ± 0.027	11.24 ± 0.05
Linear ramp to 96	72	60.25 ± 0.51	0.871 ± 0.020	1.085 ± 0.011	9.69 ± 0.38
48 / 72 / 96	74	60.48 ± 0.46	0.813 ± 0.018	1.085 ± 0.014	9.55 ± 0.05

Train CE uses 500 images. The longer progressive runs have 72 or 74 epochs and are not exact equal-time controls. Their within-campaign native-resolution controls are included. No adaptive schedule is included.

the native-resolution 72-epoch control already reaches 59.52%. The longer schedules use different update counts and slightly more measured time than the 60-epoch Plain run; they are not exact equal-time comparisons. They show that input resizing is also useful in this recipe, without establishing its superiority to internal reduction. The initial weights, data order and trainer differ from the main STL-10 transfer study, so those scores are not paired across campaigns.

E Comparison with CBS and SDPoint

The comparison uses CIFAR-10 (50,000 training and 10,000 test images), ResNet-20-BN/ReLU and seeds 0, 1 and 2. Each recipe contains Plain, R, G, RG, two CBS configurations and SDPoint. Initial learned parameters, sample order and augmentation draws are paired within recipe and seed. SDPoint uses a separate random stream for its intervention choices. The final scheduled checkpoint is evaluated; no best-test-epoch selection is used.

E.1 Operators and implementation provenance

CBS (Sinha et al., 2020) filters convolution outputs while decreasing Gaussian scale. Our port follows the authors' ResNet implementation: a normalized 3×3 depthwise Gaussian with zero padding, inserted after the stem convolution and the two main-branch convolutions in each block, before BatchNorm. Shortcut projections are not filtering sites. This gives 19 sites on ResNet-20. G instead uses a separable 9×9 Gaussian with reflection boundaries at ten post-activation sites, and a different schedule. It is consequently a CBS-inspired procedure, not CBS itself.

CBS-P, which keeps the published schedule, starts with $\sigma = 1$ and multiplies it by 0.9 every five actual epochs, so $\sigma(e) = 0.9^{\lfloor e/5 \rfloor}$ for zero-based epoch e. At the 30-epoch endpoint, its native inference still uses $\sigma = 0.59049$. At 160 epochs, $\sigma = 0.9^{31}$ and the float32 kernel is numerically an identity; the extra convolutions are nevertheless executed. CBS-B, adapted to the budget, keeps the same operator and sites but uses the predetermined budget adaptation

$$u_{\text{off}} = \lfloor 0.7U \rfloor, \qquad \sigma(u) = 0.9^{\lfloor 22u/u_{\text{off}} \rfloor} \ (u < u_{\text{off}}),$$

then bypasses filtering exactly. Here U is the total update count and $22 = \lceil \log(0.1)/\log(0.9) \rceil$. CBS-B is our adaptation, not a published CBS schedule.

SDPoint (Kuen et al., 2018) draws uniformly from ten choices: no extra downsampling, or one of the nine residual-block boundaries. It independently draws ratio 0.5 or 0.75. At the selected boundary, adaptive average pooling follows residual addition and precedes the final ReLU. The same configuration is used for every image and microbatch of a logical optimizer update. The output side is rounded from the current side times the ratio, matching the released implementation; all products here are integers. Inference always uses the full-resolution instance, chosen in advance.

The source audit used the CBS author repository at commit `5f62e7d` and the SDPoint author repository at `0013c5d`[1]. The SDPoint port corrects two code defects: the released dispatcher does not activate BasicBlocks, and a truth-value expression loses a sampled zero index. Our explicit block selection follows the paper's CIFAR construction. The supplied implementation checks compare CBS kernels and forwards with the author code, verify pooling placement, exercise every SDPoint instance, and test exact identity bypass, valid gradients, independent random streams and restart behaviour. These are adaptations to our backbone and trainer; the authors' original architectures, training recipes and model-selection rules are not reproduced.

E.2 Five recipes and two historical cohorts

The short comparison has two 30-epoch recipes without augmentation: reference SGD at 0.005, and AdamW at 0.02 with weight decay 5×10^{-4}. They use effective batch size 128, physical microbatches of 32, a 60-update warm-up and cosine decay. Twelve SGD runs for Plain/R/G/RG reuse seeds 0–2 of the five-seed landscape cohort of Appendix F; they are not the original selection-table runs. The other short-comparison runs form their own paired cohort.

The longer comparison contains 63 new runs: seven methods, three seeds and three 160-epoch recipes (62,560 updates each). SGD with crop/flip augmentation uses rate 0.1, momentum 0.9, weight decay 10^{-4}, physical batch size 128, no warm-up, and tenfold rate reductions at epochs 80 and 120. It adapts the CIFAR recipe of He et al. (2016), with a different exact update budget, initialization, normalization and decay scope; it is not an exact reproduction. AdamW keeps rate 0.02, weight decay 5×10^{-4}, betas $(0.9, 0.999)$, numerical $\epsilon = 10^{-8}$, effective/physical batches 128/32 and the 60-update warm-up followed by cosine decay. It is run both with and without crop/flip augmentation.

R uses sides 16, 24 and 32 with transitions at epochs 32 and 64. G uses 16-epoch scale plateaus and is removed at epoch 112. RG preserves its original coupling. CBS-P retains its five-epoch decay, whereas CBS-B removes filtering at update 43,792. SDPoint continues sampling throughout training. These rules were fixed before the new runs, but their choice followed earlier test results; this does not preregister the overall research process retrospectively.

E.3 Four endpoint BatchNorm conventions

Every evaluation begins from the original checkpoint and preserves its learned weights and BatchNorm affine parameters. All methods receive every policy, using training images only for recalibration:

P0: saved. Evaluate with the checkpoint's running means and variances. For SDPoint these mix training instances, so P0 alone is not its intended instance-specific evaluation.

P1: clean recalibration. Reset the buffers and counters, then make one pass through all 50,000 clean training images in fixed file order, with batches of 500 and cumulative averaging. Freeze the resulting statistics for evaluation. This is the primary policy fixed for the initial comparison.

P2: smaller calibration batches. Repeat P1 with batches of 32. The final shorter batch is retained. This measures dependence on the calibration batch size, not a new training batch size.

P3: training-loader recalibration. Starting from the saved buffers without resetting them, make one pass through a fixed seed-specific training order, with momentum 0.1 and the physical training batch size. Use crop/flip only when the training recipe uses it. This follows the logic of the SDPoint authors' evaluation pass, adapted to our data and loader.

These passes update only running statistics; subsequent evaluation is in inference mode. Cumulative averaging of batch variances is a defined calibration procedure, not an exact reconstruction of population variance. P3 changes reset, averaging, order, batch size and, for augmented training, image transformations together. A P1/P3 difference cannot be attributed solely to augmentation or to one normalization parameter.

[1]Author code: `https://github.com/pairlab/CBS` and `https://github.com/xternalz/SDPoint`. Full commit identifiers and file hashes are in the supplementary material.

Table 21 reports every policy. Table 22 gives full-training and full-test CE under P1 and P3. Tables 23 and 24 give the seed-paired contrasts used in the text. P2 remains close to P1 (at most 0.17 points between policy means for a method and recipe), whereas P3 can change accuracy by several points. Changing only the recalibration batch size therefore does not reproduce the full P1/P3 effect.

Table 21: All BatchNorm endpoint conventions for the literature comparison: CIFAR-10 / ResNet-20-BN/ReLU. Test accuracy (%), mean ± SD over three seeds. P0: saved; P1: clean cumulative, batch 500; P2: clean cumulative, batch 32; P3: training-loader EMA. Every column uses the same checkpoints.

Method	P0	P1	P2	P3
SGD 0.005, 30 ep., no aug.				
Plain	75.47 ± 0.87	75.46 ± 0.85	75.46 ± 0.88	75.43 ± 0.96
R	80.38 ± 0.40	80.46 ± 0.51	80.44 ± 0.52	80.28 ± 0.48
G	79.93 ± 0.13	79.94 ± 0.03	79.96 ± 0.01	80.01 ± 0.15
RG	81.48 ± 0.34	81.63 ± 0.27	81.65 ± 0.29	81.53 ± 0.31
CBS-P	77.92 ± 0.60	78.01 ± 0.61	77.98 ± 0.62	77.86 ± 0.70
CBS-B	78.61 ± 0.90	78.71 ± 0.90	78.70 ± 0.92	78.59 ± 0.79
SDPoint	77.92 ± 1.17	79.83 ± 0.34	79.85 ± 0.32	79.84 ± 0.38
AdamW 0.02, 30 ep., no aug.				
Plain	84.25 ± 0.18	84.28 ± 0.20	84.28 ± 0.19	84.34 ± 0.20
R	86.61 ± 0.20	86.65 ± 0.14	86.64 ± 0.14	86.58 ± 0.16
G	83.61 ± 0.32	83.75 ± 0.23	83.70 ± 0.24	83.64 ± 0.20
RG	86.15 ± 0.28	86.24 ± 0.36	86.23 ± 0.38	86.06 ± 0.22
CBS-P	80.39 ± 0.36	80.56 ± 0.24	80.57 ± 0.27	80.42 ± 0.12
CBS-B	83.20 ± 0.17	83.23 ± 0.38	83.22 ± 0.33	83.20 ± 0.37
SDPoint	85.95 ± 0.69	87.57 ± 0.59	87.59 ± 0.59	87.50 ± 0.61
SGD 0.1, 160 ep., crop + flip				
Plain	91.49 ± 0.34	89.66 ± 0.17	89.62 ± 0.14	91.53 ± 0.35
R	91.64 ± 0.15	88.56 ± 1.28	88.53 ± 1.30	91.66 ± 0.15
G	90.43 ± 0.33	85.67 ± 1.63	85.57 ± 1.59	90.46 ± 0.39
RG	90.66 ± 0.32	87.17 ± 1.13	87.16 ± 1.20	90.70 ± 0.41
CBS-P	91.31 ± 0.32	88.85 ± 0.31	88.78 ± 0.27	91.36 ± 0.34
CBS-B	91.51 ± 0.24	89.16 ± 0.33	89.13 ± 0.31	91.54 ± 0.21
SDPoint	90.23 ± 0.21	88.43 ± 0.43	88.35 ± 0.41	91.90 ± 0.31
AdamW 0.02, 160 ep., no aug.				
Plain	84.91 ± 0.12	84.95 ± 0.26	84.91 ± 0.21	84.86 ± 0.25
R	85.75 ± 0.64	85.83 ± 0.65	85.82 ± 0.62	85.72 ± 0.64
G	84.75 ± 0.86	84.74 ± 0.83	84.74 ± 0.89	84.63 ± 0.84
RG	86.46 ± 0.23	86.55 ± 0.32	86.53 ± 0.32	86.42 ± 0.29
CBS-P	83.35 ± 0.37	83.33 ± 0.47	83.33 ± 0.47	83.27 ± 0.30
CBS-B	83.70 ± 0.33	83.71 ± 0.31	83.72 ± 0.35	83.71 ± 0.26
SDPoint	85.71 ± 1.07	88.94 ± 0.12	88.92 ± 0.14	88.84 ± 0.12
AdamW 0.02, 160 ep., crop + flip				
Plain	90.84 ± 0.39	88.30 ± 0.65	88.28 ± 0.63	90.84 ± 0.36
R	90.50 ± 0.26	88.46 ± 0.44	88.37 ± 0.42	90.53 ± 0.17
G	88.45 ± 0.22	81.18 ± 1.87	81.01 ± 1.85	88.52 ± 0.31
RG	89.42 ± 0.12	85.65 ± 0.71	85.57 ± 0.80	89.44 ± 0.15
CBS-P	89.47 ± 0.29	86.48 ± 0.67	86.46 ± 0.70	89.52 ± 0.15
CBS-B	89.18 ± 0.05	86.40 ± 0.49	86.32 ± 0.48	89.25 ± 0.07
SDPoint	88.59 ± 0.55	88.18 ± 0.46	88.12 ± 0.40	91.18 ± 0.12

Figure 3 makes the policy dependence visible without choosing the most favourable policy for each method. The no-augmentation AdamW advantage of SDPoint persists across P1–P3. The augmented settings allow closer orderings to reverse, including R versus SDPoint. This is why both specified recalibration policies appear in Table 3.

Native and forced-unfiltered CBS. Forcing the 30-epoch CBS-P checkpoint to remove its remaining filter gives 44.34% under SGD and 72.61% under AdamW after clean recalibration, versus

Table 22: Full-training-set (50,000 images) and test (10,000) CE for the comparison cohort, mean $\pm$ SD over three seeds. P1 and P3 are defined in Appendix E. All evaluations use clean images after calibration; only the calibration procedure differs.

	P1		P3	
Method	Train CE	Test CE	Train CE	Test CE
SGD 0.005, 30 ep., no aug.				
Plain	0.204 ± 0.021	0.753 ± 0.015	0.206 ± 0.020	0.760 ± 0.016
R	0.299 ± 0.010	0.575 ± 0.009	0.301 ± 0.010	0.579 ± 0.010
G	0.274 ± 0.005	0.589 ± 0.001	0.276 ± 0.004	0.594 ± 0.002
RG	0.314 ± 0.004	0.542 ± 0.005	0.315 ± 0.003	0.546 ± 0.005
CBS-P	0.370 ± 0.010	0.643 ± 0.014	0.371 ± 0.010	0.649 ± 0.014
CBS-B	0.192 ± 0.012	0.657 ± 0.018	0.194 ± 0.012	0.664 ± 0.019
SDPoint	0.477 ± 0.010	0.588 ± 0.010	0.477 ± 0.010	0.589 ± 0.010
AdamW 0.02, 30 ep., no aug.				
Plain	0.000341 ± 0.000117	0.953 ± 0.007	0.000376 ± 0.000125	0.967 ± 0.011
R	0.0012 ± 0.0001	0.696 ± 0.012	0.0012 ± 0.0001	0.707 ± 0.006
G	0.024 ± 0.004	0.683 ± 0.012	0.025 ± 0.004	0.691 ± 0.012
RG	0.0059 ± 0.0011	0.640 ± 0.005	0.0061 ± 0.0013	0.652 ± 0.006
CBS-P	0.127 ± 0.011	0.685 ± 0.018	0.129 ± 0.011	0.694 ± 0.020
CBS-B	0.0024 ± 0.0004	0.880 ± 0.014	0.0025 ± 0.0003	0.893 ± 0.013
SDPoint	0.111 ± 0.008	0.386 ± 0.007	0.113 ± 0.008	0.390 ± 0.007
SGD 0.1, 160 ep., crop + flip				
Plain	0.076 ± 0.020	0.413 ± 0.014	0.013 ± 0.001	0.366 ± 0.018
R	0.129 ± 0.045	0.431 ± 0.047	0.017 ± 0.000	0.350 ± 0.008
G	0.233 ± 0.071	0.531 ± 0.054	0.034 ± 0.003	0.372 ± 0.011
RG	0.189 ± 0.043	0.489 ± 0.058	0.030 ± 0.001	0.376 ± 0.022
CBS-P	0.113 ± 0.004	0.440 ± 0.015	0.015 ± 0.001	0.371 ± 0.013
CBS-B	0.105 ± 0.010	0.425 ± 0.004	0.015 ± 0.000	0.357 ± 0.009
SDPoint	0.214 ± 0.011	0.353 ± 0.016	0.087 ± 0.001	0.252 ± 0.008
AdamW 0.02, 160 ep., no aug.				
Plain	0.000002 ± 0.000000	1.328 ± 0.026	0.000002 ± 0.000000	1.339 ± 0.022
R	0.000073 ± 0.000116	1.063 ± 0.019	0.000097 ± 0.000156	1.069 ± 0.021
G	0.000057 ± 0.000074	1.170 ± 0.061	0.000053 ± 0.000065	1.192 ± 0.062
RG	0.000158 ± 0.000221	0.995 ± 0.030	0.000175 ± 0.000260	1.005 ± 0.034
CBS-P	0.000060 ± 0.000065	1.316 ± 0.047	0.000066 ± 0.000076	1.328 ± 0.057
CBS-B	0.000042 ± 0.000064	1.295 ± 0.020	0.000052 ± 0.000078	1.313 ± 0.024
SDPoint	0.0011 ± 0.0001	0.630 ± 0.019	0.0012 ± 0.0001	0.636 ± 0.019
AdamW 0.02, 160 ep., crop + flip				
Plain	0.098 ± 0.027	0.634 ± 0.052	0.0045 ± 0.0004	0.573 ± 0.031
R	0.096 ± 0.028	0.571 ± 0.012	0.0095 ± 0.0004	0.516 ± 0.009
G	0.400 ± 0.107	0.826 ± 0.111	0.031 ± 0.004	0.551 ± 0.015
RG	0.214 ± 0.055	0.629 ± 0.019	0.024 ± 0.002	0.515 ± 0.013
CBS-P	0.142 ± 0.042	0.636 ± 0.042	0.013 ± 0.003	0.558 ± 0.007
CBS-B	0.135 ± 0.017	0.637 ± 0.025	0.015 ± 0.003	0.571 ± 0.015
SDPoint	0.190 ± 0.021	0.397 ± 0.014	0.069 ± 0.001	0.311 ± 0.002

Table 23: Paired accuracy differences for the 30-epoch recipes (percentage points), mean ± SD and positive seed counts. Each contrast pairs the same seed, campaign and BatchNorm policy.

Contrast	P1	P3
SGD 0.005, 30 ep., no aug.		
R − Plain	5.00 ± 0.76 (3/3)	4.85 ± 0.86 (3/3)
G − Plain	4.48 ± 0.88 (3/3)	4.58 ± 0.83 (3/3)
RG − Plain	6.17 ± 0.76 (3/3)	6.10 ± 0.74 (3/3)
SDPoint − Plain	4.37 ± 0.70 (3/3)	4.40 ± 0.96 (3/3)
R − SDPoint	0.63 ± 0.19 (3/3)	0.45 ± 0.16 (3/3)
RG − SDPoint	1.80 ± 0.08 (3/3)	1.69 ± 0.25 (3/3)
G − CBS-P	1.93 ± 0.63 (3/3)	2.15 ± 0.64 (3/3)
G − CBS-B	1.23 ± 0.91 (3/3)	1.42 ± 0.75 (3/3)
AdamW 0.02, 30 ep., no aug.		
R − Plain	2.37 ± 0.29 (3/3)	2.24 ± 0.29 (3/3)
G − Plain	-0.53 ± 0.31 (0/3)	-0.70 ± 0.20 (0/3)
RG − Plain	1.96 ± 0.41 (3/3)	1.72 ± 0.26 (3/3)
SDPoint − Plain	3.29 ± 0.51 (3/3)	3.16 ± 0.48 (3/3)
R − SDPoint	-0.92 ± 0.53 (0/3)	-0.92 ± 0.58 (0/3)
RG − SDPoint	-1.33 ± 0.91 (0/3)	-1.44 ± 0.74 (0/3)
G − CBS-P	3.20 ± 0.17 (3/3)	3.22 ± 0.08 (3/3)
G − CBS-B	0.53 ± 0.23 (3/3)	0.43 ± 0.28 (3/3)

Table 24: Paired accuracy differences for the 160-epoch recipes (percentage points), mean ± SD and positive seed counts. Each contrast pairs the same seed, campaign and BatchNorm policy.

Contrast	P1	P3
SGD 0.1, 160 ep., crop + flip		
R − Plain	-1.10 ± 1.39 (0/3)	0.13 ± 0.20 (2/3)
G − Plain	-4.00 ± 1.62 (0/3)	-1.08 ± 0.41 (0/3)
RG − Plain	-2.50 ± 1.11 (0/3)	-0.84 ± 0.42 (0/3)
SDPoint − Plain	-1.23 ± 0.28 (0/3)	0.37 ± 0.50 (2/3)
R − SDPoint	0.13 ± 1.49 (2/3)	-0.24 ± 0.38 (1/3)
RG − SDPoint	-1.26 ± 1.27 (0/3)	-1.20 ± 0.72 (0/3)
G − CBS-P	-3.18 ± 1.93 (0/3)	-0.90 ± 0.30 (0/3)
G − CBS-B	-3.50 ± 1.93 (0/3)	-1.09 ± 0.53 (0/3)
AdamW 0.02, 160 ep., no aug.		
R − Plain	0.88 ± 0.44 (3/3)	0.86 ± 0.39 (3/3)
G − Plain	-0.21 ± 0.59 (1/3)	-0.23 ± 0.62 (1/3)
RG − Plain	1.60 ± 0.32 (3/3)	1.56 ± 0.28 (3/3)
SDPoint − Plain	3.99 ± 0.14 (3/3)	3.99 ± 0.18 (3/3)
R − SDPoint	-3.11 ± 0.55 (0/3)	-3.13 ± 0.57 (0/3)
RG − SDPoint	-2.39 ± 0.30 (0/3)	-2.42 ± 0.34 (0/3)
G − CBS-P	1.41 ± 0.90 (3/3)	1.36 ± 0.90 (3/3)
G − CBS-B	1.03 ± 1.05 (2/3)	0.92 ± 1.03 (2/3)
AdamW 0.02, 160 ep., crop + flip		
R − Plain	0.16 ± 1.01 (2/3)	-0.31 ± 0.52 (1/3)
G − Plain	-7.12 ± 1.25 (0/3)	-2.32 ± 0.33 (0/3)
RG − Plain	-2.65 ± 1.33 (0/3)	-1.40 ± 0.32 (0/3)
SDPoint − Plain	-0.13 ± 0.94 (2/3)	0.34 ± 0.31 (2/3)
R − SDPoint	0.28 ± 0.16 (3/3)	-0.65 ± 0.27 (0/3)
RG − SDPoint	-2.52 ± 0.84 (0/3)	-1.74 ± 0.04 (0/3)
G − CBS-P	-5.30 ± 2.46 (0/3)	-1.00 ± 0.18 (0/3)
G − CBS-B	-5.22 ± 2.29 (0/3)	-0.74 ± 0.32 (0/3)

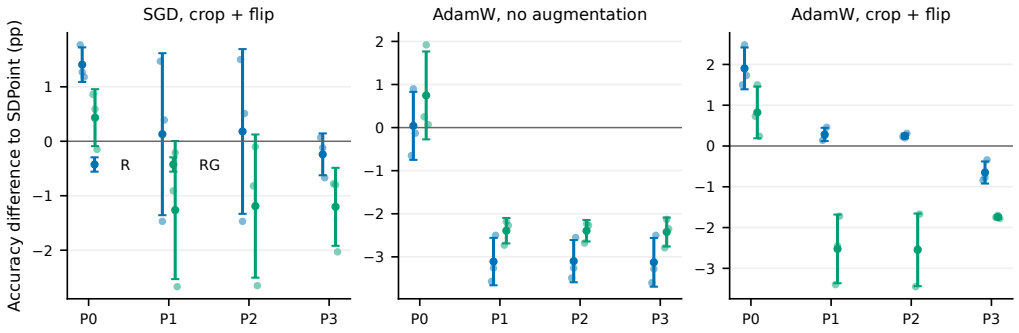


Figure 3: R and RG relative to SDPoint after 160 epochs under all four endpoint BatchNorm policies. Dots include the three seed-paired differences; larger markers and bars show their mean and sample SD. Positive means higher accuracy than SDPoint. P0 is a saved-statistics diagnostic, not a substitute for SDPoint's instance-specific evaluation.

native means 78.01% and 80.56%. This is a diagnostic of changing the inference network, not the appropriate score for ranking CBS-P. All main comparisons retain its native filter. At 160 epochs that filter is numerically an identity, although it still incurs the executed convolution cost.

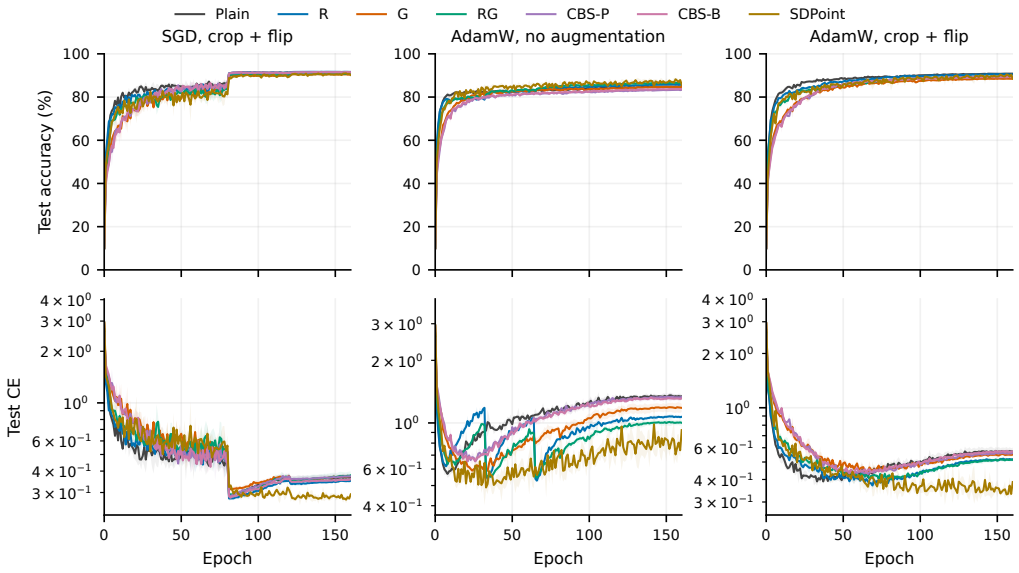


Figure 4: The 160-epoch learning curves: mean and sample SD over three seeds; CE uses a logarithmic axis. Each epoch evaluates the current intervention state with saved BatchNorm statistics (P0). In particular, SDPoint's saved buffers mix training instances. These curves describe the recorded training process; the recalibrated final comparisons are in Table 3.

Figure 4 retains the available per-epoch observations. Changes near schedule transitions combine changes of weights, interventions and saved normalization statistics. They must not be read as P1 or P3 learning curves. The recorded P1 intermediate checks at epochs 40, 80, 120 and 160 are in the supplementary material.

Table 25 reports the observed cost of every longer run group. The 63 new and 42 historical checkpoints have complete four-policy endpoint exports. Their repeated evaluations do not count as new training replications. The evidence supports comparisons of these explicit recipes and full-resolution inference, not dominance over SDPoint's entire cost–accuracy trade-off or over all CBS schedules.

Table 25: Cost of the 160-epoch comparison, mean ± SD over three seeds. Total includes periodic evaluation, curve diagnostics and final calibration. Peak memory is the maximum PyTorch-allocated GPU memory, not total device occupancy. One training run per T4; sessions were shared.

Method	Train loop (min)	Total (min)	Peak allocated (MiB)
SGD 0.1, 160 ep., crop + flip			
Plain	37.2 ± 0.8	40.7 ± 0.9	540 ± 0
R	32.0 ± 0.4	35.3 ± 0.4	540 ± 0
G	47.7 ± 1.8	52.4 ± 2.0	630 ± 0
RG	48.9 ± 1.6	54.3 ± 1.8	834 ± 0
CBS-P	41.8 ± 0.6	46.0 ± 0.7	541 ± 0
CBS-B	41.6 ± 1.1	45.6 ± 1.3	540 ± 0
SDPoint	33.9 ± 0.7	37.6 ± 0.8	540 ± 0
AdamW 0.02, 160 ep., no aug.			
Plain	41.3 ± 0.9	45.0 ± 0.9	541 ± 1
R	43.4 ± 3.3	46.8 ± 3.2	540 ± 0
G	53.0 ± 0.7	57.7 ± 0.5	610 ± 0
RG	66.1 ± 1.7	71.2 ± 1.7	610 ± 0
CBS-P	48.8 ± 0.8	53.2 ± 0.9	541 ± 1
CBS-B	47.6 ± 2.7	51.7 ± 2.6	540 ± 1
SDPoint	42.8 ± 0.6	46.5 ± 0.5	540 ± 0
AdamW 0.02, 160 ep., crop + flip			
Plain	42.8 ± 1.4	46.4 ± 1.2	541 ± 0
R	43.5 ± 2.3	47.1 ± 2.3	541 ± 1
G	53.7 ± 0.2	58.4 ± 0.4	612 ± 1
RG	65.1 ± 3.1	70.1 ± 3.2	612 ± 0
CBS-P	49.6 ± 1.9	53.9 ± 1.9	541 ± 0
CBS-B	48.1 ± 0.8	52.4 ± 0.9	541 ± 1
SDPoint	43.7 ± 1.2	47.4 ± 1.2	542 ± 0

E.4 Stopping SDPoint at the return to native resolution

A subsequent control stops SDPoint's random downsampling at the exact update where R returns permanently to native resolution: after epoch 12 in the 30-epoch recipe or epoch 64 in the 160-epoch recipes. Before stopping, its sampled choices match standard SDPoint; optimizer state and BatchNorm buffers are not reset. Nine new runs cover the three recipes below, paired with standard SDPoint at seeds 0–2. At evaluation both use the full-resolution network.

Table 26: Stopped minus standard SDPoint: paired full-test accuracy differences in points (mean ± SD, three seeds). P0 uses saved BatchNorm statistics; P1 and P3 recalibrate according to the policies above.

Recipe	P0	P1	P3
SGD 30, no augmentation	$+1.77 \pm 1.72$	-0.14 ± 0.40	-0.06 ± 0.30
AdamW 160, no augmentation	$+3.40 \pm 0.84$	$+0.16 \pm 0.17$	$+0.27 \pm 0.18$
AdamW 160, crop + flip	$+2.20 \pm 0.47$	-0.00 ± 0.27	-0.44 ± 0.34

Under the primary P1 policy, stopping changes mean accuracy by at most 0.16 points (Table 26). In long AdamW without augmentation, stopped SDPoint still exceeds Plain by 4.16 ± 0.03 points. The large saved-statistics differences shrink after recalibration; stopping also gives the running statistics a long native-only training phase. These results do not resolve a substantial accuracy cost of stopping under P1, but neither establish equivalence nor make the result independent of the BatchNorm policy. The standard SDPoint results remain the literature comparator.

F WEIGHT-SPACE DIAGNOSTICS AND NUMERICAL CHECKS

The geometry study uses twenty separately trained reference models: plain, R, G and RG at seeds 0–4 (Table 27). Initial weights and data order are paired within each seed. The finite-sensitivity, Hessian, trace and intermediate-state evaluations below reuse those checkpoints; no additional optimization is needed for these diagnostics. The five-seed batch is separate from the three-seed selection batch, so its repeated seed labels do not justify pooling the runs.

Table 27: Endpoints of the separate five-seed landscape batch: CIFAR-10 / ResNet-20-BN / SGD, 30 epochs, saved BatchNorm. Accuracy uses all 10k test images; CE uses all 50k training or 10k test images. Mean ± sample SD. Bold accuracy means exceed Plain; this batch is not pooled with Table 2.

Method	Test accuracy (%)	Train CE (nats)	Test CE (nats)
Plain	75.65 ± 0.78	0.197 ± 0.019	0.759 ± 0.010
Resolution	$\mathbf{80.21} \pm 0.36$	0.303 ± 0.008	0.584 ± 0.010
Gaussian	$\mathbf{80.12} \pm 0.49$	0.273 ± 0.006	0.587 ± 0.021
Combined	$\mathbf{81.58} \pm 0.30$	0.312 ± 0.006	0.542 ± 0.005

F.1 EVALUATION SETS AND NORMALIZATION

The training and test probes each contain 1,000 fixed, class-balanced examples. Calibration uses a fixed 2,000-image training subset, disjoint from the training probe, in four batches of 500. Larger finite-sensitivity checks use 10,000 training images and the full test set; the larger training subset includes the training probe but is disjoint from calibration. Calibration resets running statistics and accumulates their cumulative batch averages without gradient updates. Evaluation then uses these buffers in eval mode. No test examples calibrate BatchNorm, and previous evaluation points do not leave their buffers in the next evaluation.

Writing b for the BatchNorm buffers and $\widehat{b}(\boldsymbol{\theta})$ for this calibration procedure clarifies the three functions measured:

$$L_{\text{saved}}(\boldsymbol{\theta}) = L(\boldsymbol{\theta}; b_{\text{checkpoint}}), \qquad L_{\text{centre}}(\boldsymbol{\theta}) = L(\boldsymbol{\theta}; \widehat{b}(\boldsymbol{\theta}_0)), \qquad L_{\text{point}}(\boldsymbol{\theta}) = L(\boldsymbol{\theta}; \widehat{b}(\boldsymbol{\theta})).$$

The calibration batch size, order and procedure are part of the definition of $\widehat{b}$; it is not an exact population statistic. In centre-frozen evaluations, $\boldsymbol{\theta}_0$ is the checkpoint about which the plot or derivative is computed. Saved and centre-frozen policies keep the buffers constant under weight perturbations. Pointwise recalibration includes their change. At intermediate checkpoints (Appendix F.5), the intervention state is also specified independently of the buffers.

F.2 DIRECTION SAMPLING, AGGREGATION AND AMPLITUDES

The perturbed vector contains 268,336 convolution and classifier weights, arranged in 698 output-filter or classifier-row blocks. Biases and BatchNorm affine parameters are fixed. Every measured block has a nonzero norm. Gaussian draws are generated deterministically and shared across methods within a seed. Each convolutional filter or classifier-row block is scaled as in Li et al. (2018):

$$\boldsymbol{d}_j = \|\boldsymbol{\theta}_j\|_2 \frac{\boldsymbol{u}_j}{\|\boldsymbol{u}_j\|_2}, \qquad \boldsymbol{u}_j \sim \mathcal{N}(0, \boldsymbol{I}). \tag{1}$$

For a given probe and BatchNorm convention, write the evaluation loss as L. With $K = 20$ directions, the finite sensitivity is

$$S_{\boldsymbol{\theta}}(\varepsilon) = \frac{1}{K} \sum_{k=1}^{K} \left[\frac{L(\boldsymbol{\theta} + \varepsilon \boldsymbol{d}_k) + L(\boldsymbol{\theta} - \varepsilon \boldsymbol{d}_k)}{2} - L(\boldsymbol{\theta}) \right]. \tag{2}$$

Each checkpoint's central loss is subtracted and negative values are retained. One useful displacement scale, for a weight displacement δ with blocks δ_j, is

$$\rho(\delta) = \left(\frac{1}{698} \sum_j \frac{\|\delta_j\|_2^2}{\|\boldsymbol{\theta}_j\|_2^2} \right)^{1/2}.$$

For the random directions, $\rho(\varepsilon \boldsymbol{d}) = |\varepsilon|$. Directions normalized using different checkpoints need not have the same physical length or span the same plane.

The amplitudes are $0.005, 0.01, 0.02, 0.025, 0.05, 0.1, 0.2, 0.25, 0.35, 0.5$. Twenty directions per model give $20 \times 20 \times 3 \times 2 \times 10 = 24{,}000$ directional sensitivity values across networks, policies, probes and amplitudes. We first average over directions within a checkpoint, then pair each method with plain in that seed, and only then average across the five seeds. Repeated directions and amplitudes do not increase the number of independent training replications. Sample SDs over seeds are descriptive spreads, not confidence intervals or hypothesis tests.

For a twice differentiable fixed function with Hessian $\boldsymbol{H} = \nabla^2_{\boldsymbol{\theta}} L$ (Appendix F.4) and a sufficiently small displacement,

$$\frac{2S_{\boldsymbol{\theta}}(\varepsilon)}{\varepsilon^2} \longrightarrow \frac{1}{K} \sum_k \boldsymbol{d}_k^\top \boldsymbol{H} \boldsymbol{d}_k.$$

The symmetric evaluation cancels the first-order term. At finite amplitudes, ReLU activation changes and nonquadratic terms can invalidate that approximation; pointwise recalibration also changes the function. We therefore call S a finite sensitivity, not a Hessian eigenvalue or a universal flatness measure. Its expected-loss-increase interpretation has an antecedent in expected sharpness (Neyshabur et al., 2017); the symmetric evaluations and block-normalized distribution specify our estimator. Tables 28 and 29 report the larger-set sensitivities and their paired differences.

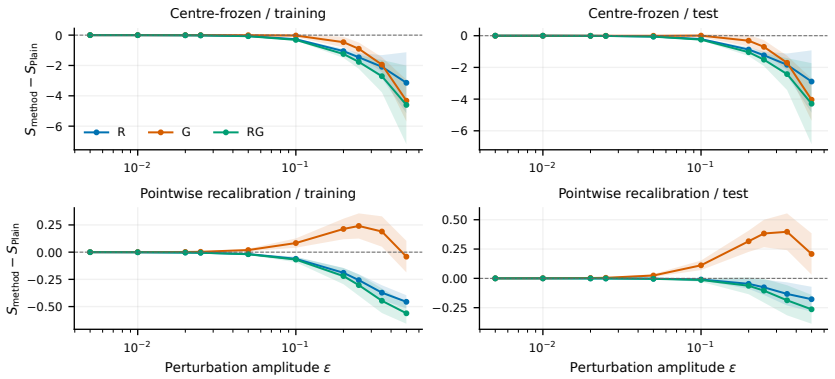


Figure 5: Finite sensitivity relative to Plain in the five-seed reference study. Points average paired seed differences after averaging 20 directions per checkpoint; bands show ± 1 sample SD. Negative means a smaller CE increase. Rows use centre-frozen BatchNorm or pointwise recalibration; columns use the training or test probe. Vertical scales differ. Saved-statistics and larger-set checks are in this appendix.

F.3 What the surfaces show

A random plane evaluates $L(\boldsymbol{\theta}_0 + a\boldsymbol{d}_1 + b\boldsymbol{d}_2)$ on a grid. The axes are two chosen directions in parameter space; they are not coordinates of input images. An absolute surface displays the CE itself. A centred surface displays $L(\boldsymbol{\theta}_0 + a\boldsymbol{d}_1 + b\boldsymbol{d}_2) - L(\boldsymbol{\theta}_0)$, so every centre has height zero even though the centre CEs differ across models. This removes the vertical offset, not the shape. The smooth bowl-like appearance of a sampled plane does not establish convexity in all parameter directions. Figure 6 shows a central window of the final seed-0 centre-frozen planes. Figure 7 retains the full measured range under centre-frozen and pointwise policies, with common scales across methods and policies. The large corner increase for R under centre-frozen statistics illustrates why a central slice is not a global ranking of surfaces. The chosen seed and two underlying random directions are unchanged; the central window does not select another checkpoint or a more favourable plane.

F.4 Hessian definitions and convergence

The Hessian is that of mean CE, excluding weight decay, on the selected weight subset at the final checkpoint. The ordinary operator is $\boldsymbol{H} = \nabla^2_{\boldsymbol{\theta}} L$. For block-relative coordinates, $\boldsymbol{\theta} = \boldsymbol{\theta}_0 + \boldsymbol{A}\boldsymbol{q}$ with

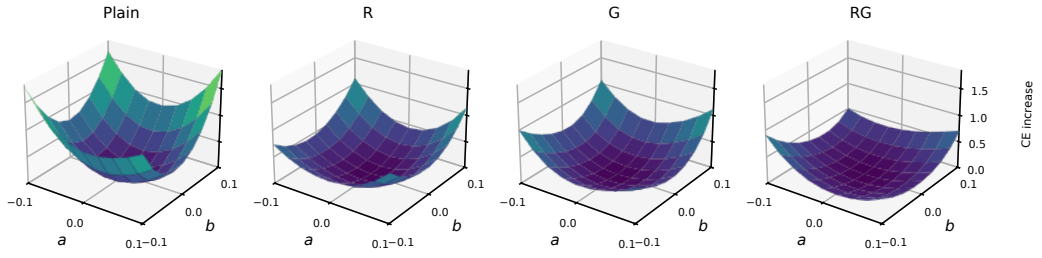


Figure 6: Centre-frozen BatchNorm loss surfaces for the four final seed-0 models: CIFAR-10 / ResNet-20-BN / SGD, 30 epochs, 1k-image training probe. Height is $L(\boldsymbol{\theta}_0 + a\boldsymbol{d}_1 + b\boldsymbol{d}_2) - L(\boldsymbol{\theta}_0)$. The same view and axis/colour scales are used throughout. This central window, $a, b \in [-0.1, 0.1]$, joins the existing grid vertices without extra smoothing. The seed and underlying direction draws are shared across methods.

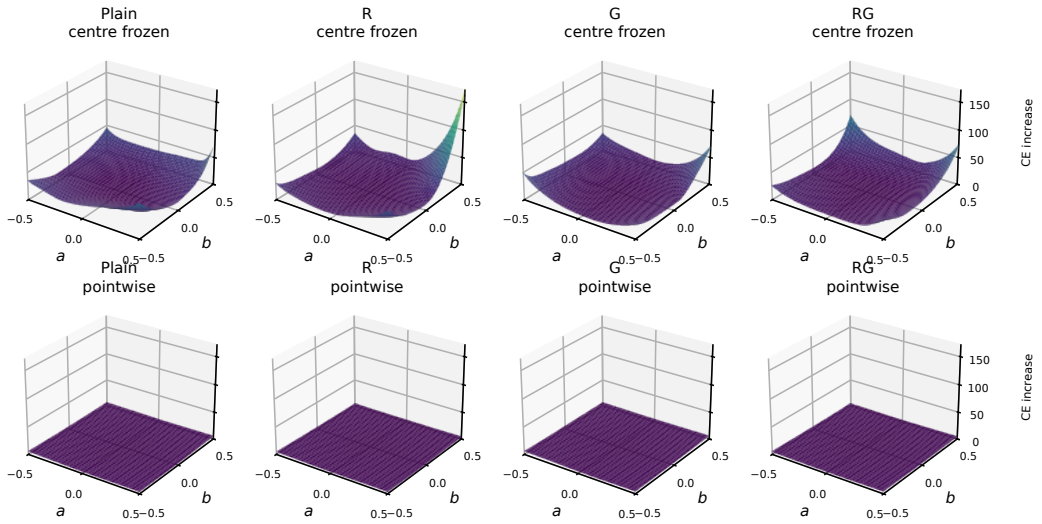


Figure 7: Full seed-0 random-plane training-CE surfaces, $a, b \in [-0.5, 0.5]$, on the 1k-image probe. Top: centre-frozen BatchNorm; bottom: pointwise recalibration. All panels share axis limits, colour scale and view. Heights are measured CE increases over the appropriate centre; the 41-by-41 grid is joined without extra smoothing. The common scale makes the much smaller recalibrated surfaces appear nearly flat.

Table 28: Finite sensitivity S (nats) on larger evaluation sets. CIFAR-10, ResNet-20-BN/ReLU, SGD, final 30-epoch checkpoints of the five-seed landscape batch. Entries are means ± sample SD across seeds, each averaging 20 directions. S is the centre-subtracted CE increase in Equation (2), not the unperturbed loss.

Evaluation set	ε	Plain	R	G	RG
Saved BatchNorm statistics					
Train subset (10k)	0.1	0.732 ± 0.060	0.468 ± 0.028	0.726 ± 0.086	0.431 ± 0.044
Train subset (10k)	0.25	5.269 ± 0.356	3.815 ± 0.294	4.423 ± 0.184	3.528 ± 0.319
Full test set (10k)	0.1	0.625 ± 0.054	0.407 ± 0.027	0.633 ± 0.082	0.386 ± 0.039
Full test set (10k)	0.25	4.866 ± 0.360	3.616 ± 0.283	4.170 ± 0.199	3.370 ± 0.312
BatchNorm recalibrated at each point					
Train subset (10k)	0.1	0.192 ± 0.015	0.135 ± 0.005	0.283 ± 0.048	0.128 ± 0.025
Train subset (10k)	0.25	1.020 ± 0.067	0.769 ± 0.022	1.283 ± 0.120	0.729 ± 0.098
Full test set (10k)	0.1	0.102 ± 0.009	0.089 ± 0.004	0.213 ± 0.043	0.092 ± 0.023
Full test set (10k)	0.25	0.700 ± 0.061	0.616 ± 0.022	1.079 ± 0.119	0.607 ± 0.096

Table 29: Paired differences $S_{\text{method}} - S_{\text{plain}}$ (nats), mean ± sample SD over five seeds. Superscripts count negative seed pairs. The populations and protocols are those of Table 28.

Evaluation set	ε	ΔS: R	ΔS: G	ΔS: RG
Saved BatchNorm statistics				
Train subset (10k)	0.1	$-0.264 \pm 0.045^{\,5/5}$	$-0.006 \pm 0.111^{\,3/5}$	$-0.301 \pm 0.045^{\,5/5}$
Train subset (10k)	0.25	$-1.454 \pm 0.237^{\,5/5}$	$-0.846 \pm 0.404^{\,5/5}$	$-1.741 \pm 0.355^{\,5/5}$
Full test set (10k)	0.1	$-0.218 \pm 0.041^{\,5/5}$	$+0.008 \pm 0.108^{\,2/5}$	$-0.239 \pm 0.043^{\,5/5}$
Full test set (10k)	0.25	$-1.250 \pm 0.246^{\,5/5}$	$-0.696 \pm 0.412^{\,5/5}$	$-1.496 \pm 0.355^{\,5/5}$
BatchNorm recalibrated at each point				
Train subset (10k)	0.1	$-0.057 \pm 0.014^{\,5/5}$	$+0.091 \pm 0.047^{\,0/5}$	$-0.064 \pm 0.024^{\,5/5}$
Train subset (10k)	0.25	$-0.250 \pm 0.059^{\,5/5}$	$+0.264 \pm 0.126^{\,0/5}$	$-0.291 \pm 0.099^{\,5/5}$
Full test set (10k)	0.1	$-0.013 \pm 0.008^{\,5/5}$	$+0.110 \pm 0.043^{\,0/5}$	$-0.011 \pm 0.021^{\,4/5}$
Full test set (10k)	0.25	$-0.084 \pm 0.057^{\,5/5}$	$+0.379 \pm 0.130^{\,0/5}$	$-0.093 \pm 0.102^{\,4/5}$

$\boldsymbol{A}_j = \|\boldsymbol{\theta}_{0,j}\|_2 \boldsymbol{I}$, we use $\boldsymbol{H}_{\text{rel}} = \boldsymbol{A}^\top \boldsymbol{H} \boldsymbol{A}$ and keep $\boldsymbol{A}$ fixed during differentiation. Normalizing by filter norms reduces an obvious scale mismatch but does not give invariance to arbitrary reparameterizations (Dinh et al., 2017). It also does not make the random directions isotropic in relative coordinates: their block covariance includes a factor inverse to block dimension. Random-direction averages and extreme eigenvalues answer different questions.

Hessian–vector products (HVPs) are computed by differentiating the loss gradient, accumulated with correct sample weights. Lanczos uses full reorthogonalization and two deterministic starts for each of 160 problems (twenty networks, two probes, two fixed-statistics policies and two coordinate choices). Each requested eigenpair must satisfy an explicit residual criterion of the form

$$\|\boldsymbol{H}v - \lambda v\|_2 \le 10^{-3} \max\{|\lambda|, 10^{-3}|\lambda_{\max}|\}, \qquad \|v\|_2 = 1,$$

with $\boldsymbol{H}$ replaced by $\boldsymbol{H}_{\text{rel}}$ when appropriate. All 160 problems met the criterion for the two largest and the smallest algebraic eigenvalue, from both starts; they required 40–110 iterations. The largest recorded relative residual is 6.4×10^{-7} for the two leading eigenpairs and 9.95×10^{-4} for the minimum eigenpair. Residuals assess the numerical eigenproblem, not whether the checkpoint is a local optimum. Table 30 reports the centre-frozen spectrum and trace summaries in unscaled units; the direction-weighted trace $\text{tr}(\boldsymbol{HC})$ is defined in the next paragraph. The original 160 eigenproblems additionally cover saved statistics and relative coordinates; all three methods have smaller leading eigenvalues than Plain in each of the five seeds under each of those conditions.

Trace estimates and the perturbation covariance. A later trace study adds traces on the forty centre-frozen objectives (twenty checkpoints, two probes). For a Rademacher vector z (Hutchinson, 1989), a single HVP gives per-block contributions $q_j = z_j^\top (\boldsymbol{H}z)_j$. Summing q_j, $\|\boldsymbol{\theta}_j\|^2 q_j$ or

Table 30: Centre-frozen Hessian measurements in their original units, mean ± SD across five seeds. Mean CE without weight decay, 1k-image probes and 268,336 convolution/classifier weights. The traces are signed; the presence of negative eigenvalues matters when interpreting them.

Method	$\lambda_{\max}(H)$	$\lambda_{\min}(H)$	$\mathrm{tr}(H)$	$\lambda_{\max}(H_{\mathrm{rel}})$	$\mathrm{tr}(H_{\mathrm{rel}})$	$\mathrm{tr}(HC)$
Training probe						
Plain	2354 ± 160	-4.29 ± 1.33	14976 ± 621	3775 ± 260	24092 ± 961	104.7 ± 6.5
R	1438 ± 107	-2.95 ± 0.40	9966 ± 304	2297 ± 167	16002 ± 454	73.5 ± 1.6
G	1275 ± 128	-3.07 ± 0.45	9844 ± 729	2090 ± 215	16624 ± 1205	99.1 ± 5.6
RG	1135 ± 70	-2.55 ± 0.40	8535 ± 513	1838 ± 107	13957 ± 781	66.3 ± 2.8
Test probe						
Plain	3230 ± 296	-9.57 ± 0.59	21408 ± 1198	5194 ± 480	34463 ± 1812	150.3 ± 6.1
R	1664 ± 70	-4.72 ± 0.91	11505 ± 365	2654 ± 126	18485 ± 570	85.3 ± 3.1
G	1482 ± 104	-5.09 ± 0.80	11591 ± 625	2430 ± 179	19580 ± 1047	117.1 ± 7.6
RG	1268 ± 69	-4.39 ± 0.71	9662 ± 528	2052 ± 107	15806 ± 786	75.1 ± 2.1

$(\|\boldsymbol{\theta}_j\|^2/p_j)q_j$ estimates $\mathrm{tr}(\boldsymbol{H})$, $\mathrm{tr}(\boldsymbol{H}_{\mathrm{rel}})$ or $\mathrm{tr}(\boldsymbol{HC})$, respectively, where p_j is the block dimension and $\boldsymbol{C}$ is the block-diagonal covariance of the random directions of Equation (1):

$$\boldsymbol{C}_{jj} = \frac{\|\boldsymbol{\theta}_j\|_2^2}{p_j}\boldsymbol{I}_{p_j}, \qquad \boldsymbol{C}_{jk} = \mathbf{0} \quad (j \neq k).$$

The factor $1/p_j$ is essential: the random directions have fixed norm per block, not independent unit-variance entries in relative coordinates. All three are signed traces, not sums of positive eigenvalues.

The rule starts with 64 draws and doubles a whole method quartet if any trace's Monte Carlo standard error exceeds 5% of its absolute estimate, up to a cap of 256. Every objective reached the threshold at 128 draws, for 5,120 HVPs in total. The largest relative Monte Carlo standard error is 4.81%. Shared draws allow paired Monte Carlo errors; these are distinct from SD across training seeds. Ordinary and relative traces are lower than Plain in all thirty method–seed–probe contrasts. For the direction-weighted training trace, G's mean ratio to Plain is about 0.95, with mixed signs across seeds, compared with 0.71 for R and 0.64 for RG. On the test probe the mean ratios are about 0.78, 0.57 and 0.50 for G, R and RG.

The checks of this trace study reproduce each centre loss and gradient norm, preserve parameter and buffer hashes, and compare the estimator with a 528-parameter dense Hessian. The algebraic block formulas match to about 9×10^{-14}. This validates the recorded estimator, not the use of a quadratic approximation at every plotted amplitude. Per unit of ρ^2, the squared root-mean-square relative displacement, the leading relative eigenvalue implies curvature about 13,000–27,000 times larger than the direction-averaged curvature. The stored extreme directions devote only about 0.4–1.0% of their squared relative displacement to rescaling entire filters or rows; their curvature is therefore not explained by pure filter rescaling. Along the leading block-relative eigendirection on the training probe, a relative displacement of 0.001 produces a median CE increase of 0.94 nats with centre-frozen statistics and 0.005 with pointwise recalibration. This comparison holds the weights and direction fixed and changes only the evaluation function.

Along leading eigendirections, local finite differences agree over very small relative displacements (roughly 10^{-5}–3×10^{-4}). Along random directions, discrepancies of about 7–32% remain at amplitudes 0.005–0.05 and are not removed by float64 evaluation. Activation-boundary crossings are a possible contributor, not a demonstrated unique cause.

The evaluations use float32 with TF32 disabled and deterministic cuDNN settings. On 960 small-amplitude comparisons, float64 recomputation changes S by at most 4.0×10^{-7} nats. Repeated HVPs were identical in the recorded checks; the relative symmetry discrepancy was at most 5.7×10^{-6}, and float32–float64 HVP differences at most 4.1×10^{-4}. Dense pilot checks and classifier-only central finite differences support the implementation. Finite differences through all perturbed ReLU weights do *not* show uniform convergence over the tested steps. We therefore interpret the HVP as the automatic-differentiation local Hessian, not as a quadratic model valid over every plotted interval.

The smallest eigenvalue is negative in all 160 cases, and the corresponding checks include float64 Rayleigh quotients. Yet the CE gradient norms range from 1.88 to 7.37 on the selected parameters: the checkpoints are not stationary points of these evaluation losses. These facts are compatible with visually bowl-shaped random slices. Neither a positive random slice nor a finite straight-line barrier certifies a local minimum or its basin.

F.5 Intermediate states, segments and PCA

Ten directions are evaluated at epochs 3, 6, 9, 12, 18, 21 and 30, at amplitudes 0.1 and 0.25, under both the state that produced the checkpoint and the final state. The recorded transitions additionally compare before/after states at fixed weights. R already has lower sensitivity in the first measured produced-state checkpoints; this does not establish what happened before epoch 3. For G, the saved-statistics ordering changes as filtering is removed, whereas pointwise-recalibrated sensitivity need not follow it. A lower loss under the state that produced the weights is common, but not universal: at G's epoch-3 transition with recalibration, the next state has slightly lower loss in 1 of the five training-probe seeds and 3 of the five test-probe seeds. We avoid a blanket claim that the training state always minimizes fixed-weight loss.

Straight-segment evaluations use

$$\boldsymbol{\theta}(\alpha) = (1-\alpha)\boldsymbol{\theta}_A + \alpha\boldsymbol{\theta}_B, \qquad B_{A,B} = \max_{\alpha\in\mathcal{G}} L(\boldsymbol{\theta}(\alpha)) - \max\{L(\boldsymbol{\theta}_A), L(\boldsymbol{\theta}_B)\},$$

where $\mathcal{G}$ is a 51-point uniform grid in $[0, 1]$. They use the final intervention state and pointwise-recalibrated BatchNorm. The comparisons are plain–R, plain–G, plain–RG and R–RG, each at five seeds. Figure 8 shows these segments. The barriers are positive on both probes, but the R–RG barrier is not always higher than plain–R: it is lower in 3 of the five seeds on the training probe (seeds 1, 2 and 4) and in 1 on the test probe (seed 1). These are barriers on sampled straight segments in the given parameterization. They do not exclude low-loss curved paths, as found by mode-connectivity methods (Garipov et al., 2018); no optimized connecting path was computed here.

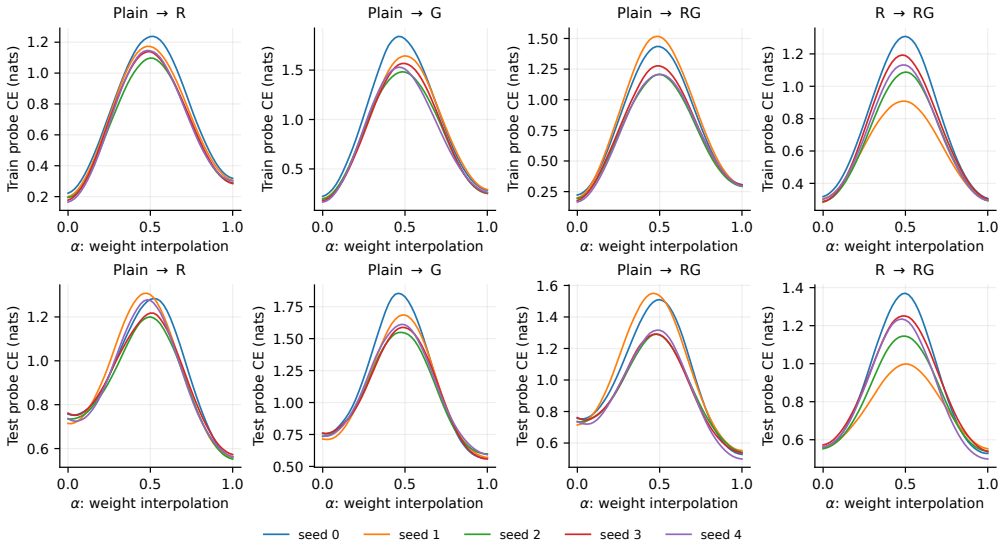


Figure 8: Measured straight-segment losses on the training and test probes, with recalibration at every point. Each thin line is one seed. Endpoints correspond to separately trained models, and the horizontal coordinate is interpolation in weights, not training time. Barriers do not establish disconnected basins.

Principal component analysis (PCA) of the recorded weight trajectory gives an alternative to random directions (Li et al., 2018). In the shared four-method analysis, the first two components explain only 56.6–59.0% of the centred checkpoint variance across seeds; four explain 93.2–93.4%. Large individual projection errors remain in two dimensions, so a checkpoint's projection onto the first two

components is not the checkpoint itself, and the loss on that plane at the projected location is not its measured loss. Connecting recorded checkpoints also does not recover the unobserved optimization path between them. We therefore do not show trajectory planes and do not use these projections as evidence for a geometric mechanism.

F.6 Follow-up geometry: learning rate and SDPoint

The follow-up uses the same diagnostic definitions on further learning rates and recipes. Its cohorts are kept separate from the twenty-model reference study. Its numerical exports are in the supplementary material.

Protocol and learning-rate check. The matched geometry evaluation uses final native ResNet-20-BN checkpoints, mean CE without weight decay, 1,000 fixed class-balanced training or test images, and the 268,336 convolution/classifier weights in 698 blocks. BatchNorm is recalibrated once at each checkpoint on 2,000 training images and held fixed for Hessian calculations. Saved-statistics and pointwise-recalibrated finite sensitivity are separate policies; neither is a Hessian that differentiates through minibatch statistics. Lanczos converged for all 82 leading-eigenvalue problems under the stated 10^{-3} relative-residual criterion: 41 checkpoints on two probes, namely Plain and R at $\eta = 0.005$ and 0.01 with five seeds each, SDPoint in the short SGD recipe, and Plain, R and SDPoint in both long AdamW recipes, with three seeds each. The traces use 128 Rademacher draws per objective, with relative Monte Carlo standard error at most 4.90% across the three centre-frozen trace estimators in the exports; this error is distinct from seed SD. The same 20 filter-normalized direction draws are used within paired comparisons at amplitudes $0.01, 0.05, 0.1, 0.25$. A repeated reference checkpoint agreed with the earlier geometry pipeline to relative error at most 1.1×10^{-5} across 16 numerical outputs.

Table 31: Paired R−Plain differences after 30 SGD epochs, five seeds (mean ± SD). Accuracy uses full-test P1 (50k calibration); Hessian traces use 1k-image probes with centre-frozen BatchNorm (2k calibration). Every cell has its mean's sign in all five pairs.

SGD rate	Δ test acc. (pp)	$\Delta\mathrm{tr}(H_{\mathrm{train}})/10^3$	$\Delta\mathrm{tr}(H_{\mathrm{test}})/10^3$
0.005	$+4.66 \pm 0.83$	-5.01 ± 0.77	-9.90 ± 1.18
0.010	$+4.20 \pm 0.56$	$+2.96 \pm 0.30$	-8.52 ± 1.65

Table 31 uses this harmonized protocol. At $\eta = 0.01$, the R–Plain training-probe trace difference is $+2{,}957 \pm 296$ (five positive differences), while the test-probe difference is $-8{,}523 \pm 1{,}646$ (five negative differences). The learning-rate-0.005 training and test differences are $-5{,}010 \pm 774$ and $-9{,}903 \pm 1{,}178$. The direction-weighted trace $\mathrm{tr}(\boldsymbol{HC})$ likewise changes sign on the training probe (-31.2 to $+17.1$), whereas on the test probe it remains negative (-65.0 to -43.2). The leading eigenvalue follows the same train/test sign pattern at the higher rate. These are two different empirical objectives, so the contrast should not be called an inversion of a single notion of model sharpness.

SDPoint across recipes. Table 32 uses only the same three seeds per method and recipe, including R and Plain in the short SGD rows. The five-seed reference summaries elsewhere in the paper use two additional runs. Accuracy is full-test P1 (calibration on all 50,000 training images); Hessian quantities use the smaller probe and 2,000-image centre calibration defined above. They should not be interpreted as the same evaluation function. The traces are signed, although all displayed means are positive.

For the short SGD recipe, SDPoint minus R is $-5{,}597 \pm 295$ for test-probe $\mathrm{tr}(\boldsymbol{H})$ and -28.10 ± 1.79 for $\mathrm{tr}(\boldsymbol{HC})$, with three negative pairs for both, whereas its full-test accuracy difference is -0.63 ± 0.19 points. Under AdamW without augmentation, R minus Plain is $+3.54 \pm 1.55$ for $\mathrm{tr}(\boldsymbol{H})$, and SDPoint minus Plain is $+1.28 \pm 0.25$ (three positive pairs each); the respective P1 accuracy gains are $+0.88 \pm 0.44$ and $+3.99 \pm 0.14$ points. With crop and flip, R minus Plain has trace $+1.62 \pm 0.59$ (three positive pairs), while its P1 accuracy gain is $+0.16 \pm 1.01$ points. The leading eigenvalue is less uniform for R in the unaugmented AdamW recipe (two of three differences positive). The finite

Table 32: Three matched seeds per recipe: mean full-test accuracy under P1, and centre-frozen test-probe geometry in the original numerical units. The geometry is evaluated on 1,000 test images with its own 2,000-image centre calibration. Absolute curvatures should be compared only within each recipe.

Recipe	Method	Acc. (%)	$\lambda_{\max}$	$\mathrm{tr}(\boldsymbol{H})$	$\mathrm{tr}(\boldsymbol{HC})$
SGD, 30 ep.	Plain	75.46	3108.94	20771.93	149.73
	R	80.46	1622.15	11413.99	84.20
	SDPoint	79.83	937.86	5816.73	56.10
AdamW 160, no aug.	Plain	84.95	0.507	6.306	37.32
	R	85.83	0.879	9.851	60.23
	SDPoint	88.94	0.693	7.583	42.80
AdamW 160, aug.	Plain	88.30	0.752	6.943	43.02
	R	88.46	0.872	8.560	52.56
	SDPoint	88.18	1.508	8.824	47.77

sensitivity at $\varepsilon = 0.1$ also changes sign between short SGD and both AdamW recipes with centre-frozen statistics on the test probe. These are small targeted cohorts, and cross-recipe differences combine optimizer, budget and augmentation.

G ADDITIONAL CONTROLS ON ORDER AND MODEL SELECTION

These controls were carried out after the original transfer and literature comparisons. They do not enlarge the twenty-setting transfer matrix or remove the earlier use of CIFAR-10 test results for exploratory choices. Runs are paired by initialization and data order within each recipe and seed. Reported SDs are sample spreads of paired training-run differences. Frozen numerical exports are in the supplementary material.

G.1 ORDER AT MATCHED RESOLUTION EXPOSURE

The standard R schedule uses sides $16 \to 24 \to 32$, with six epochs at each reduced side in the 30-epoch recipe and 32 at each reduced side in the 160-epoch recipe. The reversed control swaps the first two phases; the mixed control permutes their epochs. Both preserve the number of updates at each resolution, the start and duration of the final native phase, the global learning-rate schedule, and optimizer and BatchNorm states across transitions. Reordering changes which learning rates coincide with each resolution. Mixing additionally changes transition frequency and the history of BatchNorm buffers.

Table 33: Final P1 test-accuracy differences in points relative to standard R (mean $\pm$ SD over paired seeds). All variants use the same exposure to each resolution and the same final native phase.

Recipe	Seeds	Reversed − R	Mixed − R
SGD 30, no augmentation	5	-0.25 ± 0.28	$+0.06 \pm 0.31$
AdamW 160, no augmentation	3	$+0.13 \pm 0.72$	$+1.33 \pm 0.19$

Table 33 reports the differences. In short SGD, standard, reversed and mixed R gain 4.66, 4.41 and 4.72 points over Plain, respectively; every method–Plain pair is positive. In long AdamW without augmentation the gains are 0.88, 1.02 and 2.21 points. The mixed control exceeds standard R in each of the three AdamW seeds. These comparisons do not establish an advantage for monotone ordering in the tested recipes, nor equivalence of the schedules. The learning-rate interaction and, for mixing, the extra transitions and BatchNorm history remain part of the intervention.

G.2 A SINGLE REDUCED RESOLUTION

Two further controls keep the reduced phase of R but use a single side throughout it: 16 (R_{16}) or 24 (R_{24}), for the first 12 of 30 epochs in the SGD recipe and the first 64 of 160 in the AdamW recipe. They preserve the number of updates, the start and duration of the final native phase, the global learning-rate schedule and the paired initializations and data orders. Compute is not matched: a side of 16 costs less, and a side of 24 more, than the two reduced phases of R.

Table 34: Final P1 test-accuracy differences in points for a single reduced side held during the whole reduced phase (mean $\pm$ SD over paired seeds).

Recipe	Seeds	$R_{16}-R$	$R_{24}-R$	$R_{16}-$Plain	$R_{24}-$Plain
SGD 30, no augmentation	5	-1.11 ± 0.14	-0.34 ± 0.22	$+3.55 \pm 0.79$	$+4.32 \pm 0.82$
AdamW 160, no augmentation	3	-1.65 ± 0.49	-0.06 ± 0.52	-0.77 ± 0.24	$+0.83 \pm 0.13$

In short SGD (Table 34), R_{24} retains 4.32 of the 4.66 points gained by R over Plain, and R_{16} retains 3.55; both remain below R in every seed. The remaining difference between R and R_{24} is of the same size as the reversed-order difference in Table 33. In long AdamW without augmentation, R_{24} and R differ by 0.06 points on average, with one of three paired differences positive, whereas R_{16} falls below Plain in every seed. Under P0 and P3, R_{16} remains below R in every pair; $R_{24}-R$ is -0.28 and -0.24 points in short SGD and -0.01 and $+0.09$ points in long AdamW. Most of the gain is thus available from a temporary exposure to a single reduced side of 24, and a long phase at 16 is costly in both recipes. These controls do not separate the duration of the reduced phase from its resolution, and three seeds do not resolve differences of a few tenths of a point.

G.3 VALIDATION-BASED SELECTION OF THE LEARNING RATE AND STOPPING EPOCH

A fixed stratified split of CIFAR-10 gives 45,000 fitting examples and 5,000 validation examples (500 per class). Validation images enter neither gradient updates, input normalization nor BatchNorm calibration. Plain and R each use rates $\{0.0025, 0.005, 0.01, 0.02, 0.04\}$ and 30 training epochs, with seeds 10–12 for selection. The mean validation accuracy over these seeds (Table 35) selects a learning rate and epoch for each method. Plain permits epochs 1–30; R permits epochs 13–30, after the native network is restored. Selection uses saved BatchNorm statistics on the native network and never uses test scores. Ties prefer the later epoch, then the smaller learning rate. The choices are fixed before six confirmation runs on new seeds 13–15.

Table 35: Validation selection over three seeds. Entries are mean validation accuracies at the best admissible epoch for each method and learning rate. Both selections reach the upper edge of the grid.

Learning rate	Plain (%)	R (%)
0.0025	70.84	73.25
0.0050	74.69	79.75
0.0100	77.74	82.81
0.0200	81.19	84.77
0.0400	83.82	86.53

The selected settings are rate 0.04 and epoch 28 for Plain, and rate 0.04 and epoch 29 for R. On the three new seeds, full-test P1 accuracies average 83.46% and 85.95%: the paired gain is 2.49 ± 0.24 points, positive in all three pairs. P1 calibrates on the 45,000 fitting images only. The final-epoch gain is 2.52 ± 0.25 points, so selecting an earlier epoch changes little here. At the selected checkpoints the P0 and P3 gains are 2.30 ± 0.39 and 2.59 ± 0.28 points. Selection and confirmation together comprise 36 runs and about 4.2 GPU-hours of measured training-loop time.

The 4.66-point contrast of the five-seed reference cohort under P1 (Table 31) uses a different learning rate, training cohort and 50,000-image training set. Their difference cannot be assigned wholly to tuning Plain. The 2.49-point estimate is a result for this finite search, not an upper bound on R's gain or a claim that either method is optimally tuned. A wider grid could change either selected setting. The later validation control does not undo the project's earlier exposure to CIFAR-10 test results.