STRUCTURAL RE-PARAMETERIZATION IS NOT MAGIC

Anonymous authors
Paper under double-blind review

ABSTRACT

Some vision models are trained with several parallel kernels per layer, each with its own BatchNorm, summed during training and folded into a single kernel at inference, a technique known as structural re-parameterization. It has been reported to give a modest accuracy benefit, explained so far by implicit ensembles, diversity and "training-time overparameterization". We study a simple instance, two identical branches, and take it apart. We attribute the accuracy benefit to two simple mechanisms, previously unidentified. The mechanisms are optimization effects: a coordinate specific learning rate boost at the last layer and a directional preconditioning effect at the first layer. The boost acts on the scale coordinate and helps to grow the logit scale at the end of training. The preconditioning cools the learning rate of the highest variance mode of the input, making the system better conditioned. We show this by dissecting the model one layer at a time, deriving the modified step in closed form, and seeing that applying the corresponding optimization modifications to a single branch reproduces the observed benefit. We conduct our study in a small VGG like network on CIFAR and then show that the two mechanisms carry over to production architectures on ImageNet.

1 INTRODUCTION

A foldable block is a layer trained as a wider structure and folded at inference into a plain one. Each branch is a convolution followed by its own BatchNorm and the branch outputs are summed. Because BatchNorm divides by the batch standard deviation, each branch is non-linear in its kernel during training and the sum is not a single convolution. At inference the batch statistics are replaced by constants and the structure can be folded into one kernel.

We analyze the simplest form of it, two branches of the same shape, each a 3×3 convolution with its own BatchNorm, summed before the ReLU. The two branches share one BatchNorm scale:

$$\gamma \left(\frac{w_1 * x - \mu_1}{\sigma_1} + \frac{w_2 * x - \mu_2}{\sigma_2} \right) + \beta_1 + \beta_2, \tag{1}$$

where μ_i and σ_i are the batch mean and standard deviation of $w_i * x$ over the batch and the spatial positions, γ is the BatchNorm scale the two branches share and β_1, β_2 their shifts. At inference the block is the single kernel $\gamma\,(w_1/\sigma_1 + w_2/\sigma_2)$, and computes exactly what one branch can compute. Thus the redundant branch coordinate can be invisible to the represented function while remaining visible to the optimization dynamics. The block is a version of the deployed "trivial over-parameterization" of MobileOne (Vasu et al., 2023b), which gives each kernel its own scale, $\gamma_1 w_1/\sigma_1 + \gamma_2 w_2/\sigma_2$. We chose this over the independent scale formulation because it is simpler to analyze, and because we found that independent scales end up with $\gamma_1 \approx \gamma_2$ and give results very similar to those of the shared scale (Appendix A.4). We emphasize that both branches have the same shape, 3×3, so multiscale structure such as RepVGG's 1×1 and identity branches (Ding et al., 2021b) is out of scope. In the controlled CIFAR-100 cells we use, the two branches gain one to three points of training accuracy over a single branch, a benefit of the size the literature in this field reports.

Figure 1 and Table 1 capture the main results of the paper. The benefit of this structure lives in the first and last layer and can be recovered by a (different) learning rate intervention in those layers.

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

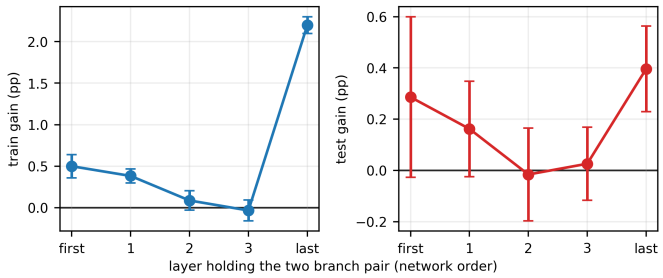

Figure 1: Where the gain lives. CIFAR-100, depth 5, the default recipe of §3.2. The horizontal axis is the layer holding the two branch pair, ordered by depth, with every other layer single. The vertical axis is the paired per seed gain in accuracy, the pair arm minus the single, in percentage points, train in the left pane and test in the right. The single branch network reaches 81.82 $\pm$ 0.08 percent train and 66.62 $\pm$ 0.10 percent test accuracy.

Table 1: The benefit lives in the first and last layer and can be emulated by a single branch with a specific learning rate intervention at the two ends. The second row is one network having a pair at both layers, not the sum of two arms. Arms are given in full in Appendix A.1 (Table 12). The intervention at the last layer is a boost on the scale coordinate and bias of that layer, and the intervention on the first is a reduction of the learning rate along the top eigenvector of the input covariance.

arm	train gain
pair at every layer	+2.69 $\pm$ 0.09
first and last paired	+2.77 $\pm$ 0.11
single branch reproduction	+2.65 $\pm$ 0.11

Contributions.

- **Two learning rate effects account for the benefit.** The observed accuracy benefit is accounted for, within experimental resolution, by two optimization effects. Placing the pair at one layer at a time shows that the gain lives at the first and the last layer and the two sum to the benefit of the whole network. A single branch network with the corresponding learning rate change at each end reproduces the benefit of re-parameterization (Figure 1 and Table 1).
- **The last layer's effect is a learning rate boost on the scale coordinate.** Boosting the single reproduces the benefit of the pair at the last layer. The boost pays at the last layer because in common recipes the classifier is decayed and the scale coordinate of the last layer is free (§6).
- **The first layer's effect is a directional preconditioning.** The branches cause a cooling of the learning rate on the maximum eigenvalue direction of the covariance. When this eigenvalue is large with respect to the others, this can make learning ill conditioned. Cooling that direction results in an optimization benefit (§7).
- **The benefit is an optimization benefit: additional representational capacity is not required.** It survives in the full batch limit, where the block spans exactly the functions a single branch spans at every step of training (§4).
- **The mechanisms transfer across model classes and scales.** They survive in a fully connected model with one normalized layer (§7.3) and scale up to production architectures on ImageNet (§8).

2 Related work

Structural re-parameterization. DiracNets (Zagoruyko & Komodakis, 2018) are the precursor to this field: instead of training the model with an explicit skip connection, the skip connection is re-parameterized and folded into the kernel. ExpandNets (Guo et al., 2020) is the same idea, each convolution expanded into a chain of wider linear layers and contracted at inference. ACNet (Ding et al., 2019) is the first block that adds the non-linear training dynamics, a 3×3 with a 1×3 and a

3×1 beside it, each with its own BatchNorm, and explains the gain partly by "a strengthened kernel skeleton". RepVGG (Ding et al., 2021b) named the technique, summed 1×1 and identity branches, and explained the gain as an implicit ensemble of shallower models. Diverse Branch Block (Ding et al., 2021a) attributed the gain to "diverse connections and the training-time nonlinearity brought by batch normalization" and tested the block we study: two duplicate 3×3 branches. RepGhost (Chen et al., 2024) found that adding a BatchNorm on top of a skip connection beats a plain skip and attributed it "to the training-time non-linearity of BN". MobileOne (Vasu et al., 2023b) adopted the technique and added extra identical branches, which it calls trivial over-parameterization. Other related works, from quantization to vision transformers and segmentation, are collected in Appendix A.3.

To our knowledge, results in this literature are single run test accuracies without error bars, and training accuracy is not reported. All works report a small but consistent gain of a few tenths to about three points of top-1 accuracy and that large models benefit less, which is also what we observe in our study. This paper therefore provides, to our knowledge, the first multi seed measurement of the effect and the first report of its training side results.

Re-parameterization as optimization. That a re-parameterization can act as an optimizer goes back to Arora et al. (2018), who showed that gradient descent on a linear layer written as a product of matrices is a preconditioned gradient descent on the folded weight, and that this preconditioning can accelerate training. For the block itself, a proposed explanation has treated it as a change of optimizer. OREPA (Hu et al., 2022) replaced each branch's BatchNorm by a learnable scale, kept the benefit, and concluded that "it is the scaling factors in the norm layers that counts most". RepOptimizer (Ding et al., 2023) proved that a block of parallel linear branches with constant scales and "no training-time nonlinearity like BN or dropout" is a single kernel trained with a constant per channel multiplier on its gradient, and built a method from it by replacing the BatchNorms with searched constants. This line of work attributes the main benefit of the technique to a change in the learning rate, establishing that structural re-parameterization can behave like a modified optimizer. Our work identifies the structure of that optimizer in the non-linear BatchNorm case, finds that it is localized to two layers, and demonstrates that the two resulting interventions are sufficient to reproduce the observed optimization benefit.

3 SETUP

3.1 The two branch block and its fold The block we study is a conv–BatchNorm–ReLU block each of whose output channels has two 3×3 kernels instead of one, each followed by its own BatchNorm, summed before the ReLU, as in (1). For the equations we assume x centered and drop the μ_i terms. Each branch is normalized by its own statistic. Our object throughout is the block with one BatchNorm scale shared by the two branches, equation (2). Every derivation and every measurement in this paper uses that form.

$$\gamma \left(\frac{w_1 * x}{\sigma_1} + \frac{w_2 * x}{\sigma_2} \right) + \beta_1 + \beta_2, \qquad \sigma_i = \sqrt{w_i^\top \Sigma w_i}, \tag{2}$$

where Σ is the covariance of the layer's input. We find it is natural to change coordinates to a **whitened** system. Writing $z := \Sigma^{-1/2} x$, $y_i := \Sigma^{1/2} w_i$ and $p_i := y_i / \|y_i\|$, we have $\sigma_i = \|y_i\|$ and the block computes $v^\top z$ with

$$p_+ := \tfrac{1}{2}(p_1 + p_2), \qquad p_- := \tfrac{1}{2}(p_1 - p_2), \qquad v := \gamma\,(p_1 + p_2) = 2\gamma \cos\frac{\theta}{2}\,\hat{p}_+, \tag{3}$$

where θ, with $\cos\theta := p_1^\top p_2 = w_1^\top \Sigma w_2 / (\sigma_1 \sigma_2)$, is the angle between the branches in the whitened coordinates. The output direction $\hat{p}_+$, the unit vector along p_+, is the only direction that appears in the output. The **separation axis** $\hat{p}_-$, the unit vector along p_-, does not. However we find that it is important for the training dynamics. A closed pair, $\theta = 0$, has no separation axis and computes what a single branch computes, so it receives the same loss gradient, although the effective learning rate can be different (§5).

We write $v_{\max}$ for the top eigenvector of Σ and $\lambda_{\max}$, λ_{next} for its two largest eigenvalues. A channel is *open* when its whitened branch angle is at least 10° and *parked* when the alignment $|\hat{p}_- \cdot v_{\max}|$ is at least 0.95.

3.2 Recipe, cells and the primary quantity The default cell is CIFAR-100 with a conv–BatchNorm–ReLU stack of depth 5: three stages of 32, 64 and 128 channels holding one, two and two layers, then global average pooling and one linear classifier. Depth 3 is one layer per stage. Kernels are Kaiming uniform with the two branches drawn independently. Training is SGD with momentum 0.9, a cosine schedule from learning rate 0.1, batch 128, 100 epochs, random crop and flip, and weight decay 5e-4 on the conv and linear weight tensors only, with BatchNorm γ and β and every bias undecayed. Five seeds unless otherwise said. Every run deviates from this recipe only on the axis it exists to study.

Most ablations compare the single arm against the pair, by which we mean the two branch block of (2). Layers are indexed from 0, and "first" and "last" mean layer 0 and the final conv layer, not the classifier.

The primary quantity is train accuracy at the end of training, reported as the paired per seed difference of the re-parameterized arm minus its own single at the same seed, mean $\pm$ standard error over seeds, in percentage points. We call this difference the **gain**, and it is the vertical axis of many figures in this work. Train accuracy is primary because the effect is an optimization effect (§4.2). We show that the block changes optimization, and no intrinsic improvement in generalization is claimed. Test accuracy is reported beside it where relevant, and in the appendix for the main sweeps.

Some per channel quantities such as the angle between the branches are displayed as medians instead of means. A mean over all of a layer's channels mixes the number of channels taking part with the size of the quantity itself, which makes it incomparable across arms. When medians are displayed, the median is formed within each seed over its open channels and averaged over seeds, and the 25th and 75th percentiles of those channels pooled across seeds follow it in brackets, as median (25th–75th). Σ, the input covariance, is calculated per epoch from 20 batches.

4 The Effect and Its Regime

Parts of the literature attribute the gain to over-parameterization, and the same literature reports that it decreases with the size of the model (§2). "Over-parameterization" can be read in several ways, and additional representational capacity turns out not to be required for the observed gain.

4.1 The regime The effect matches the evidence in the literature. It is small and it decays with model size.

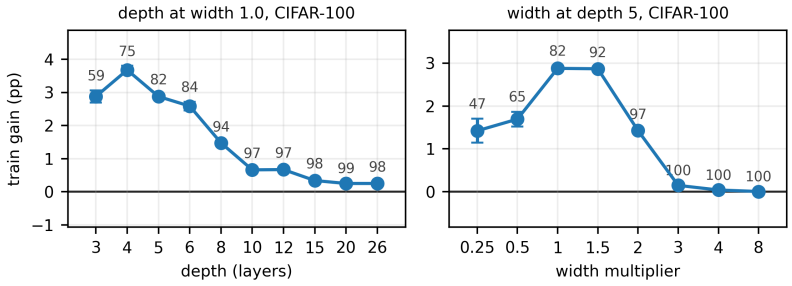

Figure 2: The regime. The paired per seed train accuracy gain of pair over single, in percentage points, on CIFAR-100. Left: depth in conv–BatchNorm–ReLU layers. Right: the width multiplier of the 32/64/128 stack at depth 5. The number over each point is the single branch train accuracy at that depth or width. The same two sweeps on CIFAR-10, and the test gains, which follow the train gains at a smaller size, are in Appendix A.5.

4.2 Capacity The question is whether the two branches can fit something a single branch cannot. At inference they cannot, as they fold, but during training it is in principle possible. The batch statistics σ_i move with the samples drawn, and the input of the interior layers moves with the weights. Due to these effects two normalizers could in principle compute something a single branch cannot.

The argument that capacity is not at play runs as follows. In the full batch limit with augmentation off, the batch is the training set and σ^2 becomes $w^\top \Sigma w$ as in (2). Each branch is then affine in its input, so the two branch block is just a different parameterization of a single branch. An effect that

builds on extra representational capacity should therefore show decreased gain with batch size. It does not (Table 2). The benefit is also kept when the pair runs at full batch on the first layer alone, where the input also does not move. That shows the benefit does not require additional capacity. We run the full batch in a smaller cell because it is more affordable: depth 3 with 16/32/64 channels, no augmentation and a constant learning rate of 0.05, for 5,000 epochs of one optimizer step each. There the single reaches 49.02 percent train accuracy. At the end of the run the pair at the first layer alone gains +0.98 $\pm$ 0.22.

Table 2: The batch size sweep in the default cell. The batch sizes are step matched at the 39,100 optimizer steps of the default cell, 39,200 at batch 256 to 1,024 and 40,000 at 2,048, so batch 128 is the default cell. Two arms run at each batch size, the pair at the first layer alone and the pair at every layer. There are two cells for each batch size at the end of the run and at matched fit, meaning the epoch at which the single at that batch size first reaches 70 percent train accuracy.

batch	epochs	single, train	pair at the first layer, matched fit	pair at every layer, matched fit	pair at the first layer, end of run	pair at every layer, end of run
128	100	81.82 %	+0.72 $\pm$ 0.16	+1.64 $\pm$ 0.10	+0.50 $\pm$ 0.14	+2.69 $\pm$ 0.09
256	200	86.59 %	+0.48 $\pm$ 0.12	+1.56 $\pm$ 0.08	+0.51 $\pm$ 0.12	+3.21 $\pm$ 0.19
512	400	90.43 %	+0.33 $\pm$ 0.16	+1.83 $\pm$ 0.14	+0.65 $\pm$ 0.09	+3.47 $\pm$ 0.09
1,024	800	93.24 %	+0.46 $\pm$ 0.13	+2.29 $\pm$ 0.14	+0.45 $\pm$ 0.08	+3.24 $\pm$ 0.09
2,048	1,600	94.86 %	+0.80 $\pm$ 0.13	+3.32 $\pm$ 0.14	+0.45 $\pm$ 0.08	+3.08 $\pm$ 0.07

5 THE BLOCK AS A PRECONDITIONER

We claim that the effect can be studied by inspecting the preconditioner the paired layer induces in the gradient flow limit, and that the system has two regimes with different effect and benefit: one when the angle between the branches closes, which is what the last layer does, and one when the angle between them is non-zero and their separation axis aligns with $v_{\max}$, which is what the first layer does. The Jacobian induced preconditioner is exact for gradient flow, while the experiments are discrete with momentum. What the equations predict is therefore a direction and a size of the step, not a trajectory. Appendix B.6 checks that against the realized steps and shows the approximation is reasonable, and Appendix B gives the general expression.

We work with the shared scale block (2) in the coordinates of §3.1. The output depends on the two biases only through their sum, so each receives the same gradient, $\partial L/\partial\beta_1 = \partial L/\partial\beta_2 = \partial L/\partial(\beta_1 + \beta_2)$. The bias therefore receives a plain $\times 2$ boost. The interesting part of the dynamics lives in the kernel weights, which is what we study in this section.

We treat Σ as fixed, which is exact at layer 0 at full batch without augmentation. Let $r := \partial L/\partial v$. Gradient flow on the underlying parameters, in the whitened coordinates, is

$$\dot{\gamma} = -(p_1 + p_2)^\top r, \qquad \dot{p}_i = -\frac{\gamma}{\sigma_i^2} P_i \Sigma P_i \, r, \qquad P_i := I - p_i p_i^\top, \tag{4}$$

with P_i the projector onto the tangent plane at p_i. Appendix B derives (4) from Euclidean gradient flow on the kernels, where Σ survives the change of coordinates because the flow is Euclidean in the kernel and not in its whitened image. Write J for the Jacobian of the output with respect to the trainable parameters, $J := \partial v/\partial(\gamma, w)$ for a single branch and $J := \partial v/\partial(\gamma, w_1, w_2)$ for the pair. Then (4) is $\dot{v} = -JJ^\top r$, so $JJ^\top$ is the preconditioner the block applies to the loss gradient. For a single branch, $v = \gamma p$ and $\sigma^2 = w^\top \Sigma w$, this is

$$JJ^\top = pp^\top + \frac{\gamma^2}{\sigma^2} P\Sigma P, \qquad P := I - pp^\top. \tag{5}$$

The first term moves the scale of v and the second its direction, and for an eigenvector of Σ orthogonal to p the second term is $\gamma^2\lambda_i/\sigma^2$: the effective learning rate along each eigenvector of Σ is proportional to its eigenvalue. On CIFAR-100 and ImageNet the top eigenvalue is an order of magnitude above the second, 9.6 on CIFAR-100 (Table 27) and 10.1 on ImageNet at 224 pixels (Table 31), so the step along $v_{\max}$ is an order of magnitude hotter than along any other direction. For a linear layer the loss Hessian is the input covariance, whose largest eigenvalue sets the largest stable

learning rate (LeCun et al., 1992; 1998). Through (5) that eigenvalue is also the hottest effective rate here, which means $v_{\max}$ caps the step.

5.1 Two regimes: the branches open at the first layer and close at the last Measuring the angle we see that the first and last layers, where the effect lives, exhibit the opposite behavior. In the first layer most channels stay open and the angle between the branches is large, while in the last layer they all close quickly. Measuring the separation axis, we empirically find that on the open channels it aligns with the maximum eigenvalue direction $v_{\max}$ of the covariance matrix Σ.

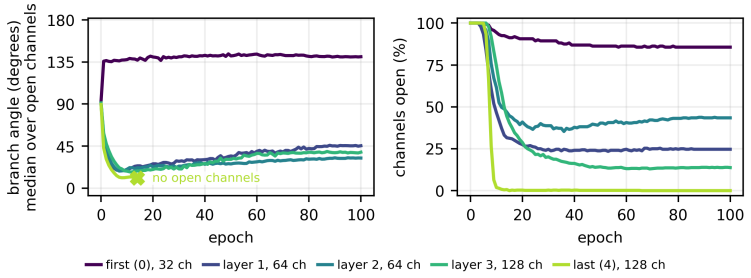

Figure 3: The whitened angle between the branches in different layers in the default cell read at every epoch. Left: the whitened branch angle, the median over that layer's open channels (angle at least 10°), with the total channel count in the legend. On the right the percentage of open channels in each layer. The first layer opens past 90° and stays open, the last layer closes over the first ten epochs, and the interior layers open much less and keep a minority of channels open.

Table 3: The geometry of the first layer over training. Open channels are channels with a whitened angle of at least 10°, counted per seed out of the layer's 32. The angle is the whitened branch angle and the alignment is $|\hat{p}_- \cdot v_{\max}|$, where $v_{\max}$ is the maximum eigenvalue direction of the data covariance Σ. The angle is the mean over seeds of each seed's median over its open channels, with the 25th and 75th percentiles of the same channels pooled across seeds in brackets. The alignment is read the same way. The 0.543 at initialization is the tilt whitening gives a Kaiming draw and not alignment. An isotropic draw in the whitened frame reaches the same state (Table 18).

epoch	channels open	whitened angle	alignment with $v_{\max}$
0, initialization	32.0 of 32	93° (65–121)	0.543 (0.276–0.822)
1	32.0 of 32	136° (121–145)	0.962 (0.914–0.978)
100	27.4 of 32	141° (130–150)	0.998 (0.988–1.000)

The rest of the section treats these two regimes. At the last layer the branches are closed, $\theta = 0$, and there is no separation axis. At the first layer they are open and the axis is parked on $v_{\max}$, which at an alignment of 0.998 is the statement that $\hat{p}_-$ is an eigenvector of Σ.

5.2 The closed pair: a hotter scale Write $\Pi_+ = \hat{p}_+\hat{p}_+^\top$ and $\Pi_- = \hat{p}_-\hat{p}_-^\top$ for the projectors onto the output direction $\hat{p}_+$ and the separation axis $\hat{p}_-$, and $Q := I - \Pi_+ - \Pi_-$ for the orthogonal projection onto the rest of the space. Closed means $p_1 = p_2 = \hat{p}_+$, so $P_1 = P_2 = I - \Pi_+$ and the general operator (10) of Appendix B.2 gives

$$JJ^\top = 4\Pi_+ + \frac{2\gamma^2}{\sigma^2}\,(I - \Pi_+)\,\Sigma\,(I - \Pi_+). \tag{6}$$

Against (5) this is a $\times 4$ boost on the scale. The tangential part is misleading, as it looks as if it has $\times\frac{1}{2}$ learning rate under the same function as a single branch, or $\times 2$ for the same γ of the single. Under the decay–norm balance of Appendix C, the pair and a single branch representing the same function have equal tangential coefficients, leaving a $\times 4$ boost on the scale and a $\times 2$ boost on the bias. Section 6 shows that applying these multipliers to a single branch reproduces the benefit, consistent with this account.

5.3 The open pair: cooling along $v_{\max}$ Define the input variance along the output direction and along the separation axis, $\lambda_+ := \hat{p}_+^\top\Sigma\hat{p}_+$ and $\lambda_- := \hat{p}_-^\top\Sigma\hat{p}_-$. When $\hat{p}_-$, the separation axis, aligns with $v_{\max}$, $\lambda_- = \lambda_{\max}$. Then $\Sigma\hat{p}_- = \lambda_{\max}\hat{p}_-$. We also take the approximation $\sigma_1 = \sigma_2 = \sigma$ (justified in B.6.1), and define $B := \lambda_+ \sin^2\frac{\theta}{2} + \lambda_- \cos^2\frac{\theta}{2}$. $\Sigma\hat{p}_- = \lambda_-\hat{p}_-$ puts $\hat{p}_-$ orthogonal

to $\Sigma\hat{p}_+$ and to ΣQ, so the two terms coupling them vanish. Then the preconditioner for the pair becomes

$$JJ^\top = 4\cos^2\tfrac{\theta}{2}\,\Pi_+ + \frac{2\gamma^2}{\sigma^2}\Big[B\sin^2\tfrac{\theta}{2}\,\Pi_+ + B\cos^2\tfrac{\theta}{2}\,\Pi_- + \sin^2\tfrac{\theta}{2}\,(\Pi_+\Sigma Q + Q\Sigma\Pi_+) + Q\Sigma Q\Big]. \quad (7)$$

The same orthogonality makes $\hat{p}_-$ an eigenvector of the operator,

$$JJ^\top\hat{p}_- = \frac{2\gamma^2}{\sigma^2}\,B\cos^2\tfrac{\theta}{2}\,\hat{p}_-, \qquad \lambda_+ \le B \le \lambda_- \ \text{ when } \ \lambda_- \ge \lambda_+. \quad (8)$$

B is the convex combination $\lambda_+\sin^2\frac{\theta}{2} + \lambda_-\cos^2\frac{\theta}{2}$ of the two variances, so it lies between them whenever $\lambda_- > \lambda_+$, which is the case we observe in the system. So on the $\hat{p}_-$ direction λ_- is replaced by $B\cos^2\frac{\theta}{2}$, and against a single branch computing the same function the decay–norm balance of Appendix C turns that into $B\cos\frac{\theta}{2}/\lambda_-$, equation (19) of Appendix B.5, so this direction is cooled. The large covariance eigenvalue produces a large effective learning rate along its direction relative to other directions, a potential source of ill-conditioned learning dynamics. When the separation axis aligns with $v_{\max}$, the pair suppresses this amplification. The cooling and whitening experiments in §7 support this account of the first layer's benefit.

6 The boost on the scale coordinate at the last layer

6.1 The multiplier reproduces the pair In Table 4 below it can be seen that in both depth settings the boost is within error of the pair, meaning the boost captures the effect. For more detailed ablations including the balance between scale and bias refer to Appendix A.9.

Table 4: The last layer boost in the default cell and at depth 3 at 32/64/128. The whole network multiplier arms are in Appendix A.9.

arm	depth 5	depth 3
pair at the last layer	+2.20 ± 0.10	+1.53 ± 0.10
$\gamma\times4$ $\beta\times2$ on the last layer, no second branch	+2.12 ± 0.12	+1.44 ± 0.06

6.2 Why the last layer In a normalized network the last layer before the classifier is the only one whose output is not scale invariant. Every earlier layer's output is normalized by the layer after it. A network that is mostly correct can lower the loss by growing the logit scale. The same growth has been observed in normalized language models (Loshchilov & Ginsburg, 2026). In the common recipe the classifier's weights are decayed while the BatchNorm scales are not. The last layer's scale is therefore free to increase while the classifier's scale is penalized. To check this we compare the norm of the classifier, the last layer's scale, and the gain under the default recipe and under a recipe with the classifier undecayed. When the classifier is decayed its norm plateaus early and the last convolution's scale carries the growth, rising by 59 percent from epoch 25 to 100 against 45 percent when the classifier is undecayed and its norm keeps growing. The gain from having the pair on the last convolution is also diminished when the classifier is undecayed (Table 5). Moving the classifier weight decay alone shows the same thing. We run decay from 0 to 2e-3 on the classifier weights only, with every other kernel at the 5e-4 of the recipe, and the gain of the pair rises monotonically with it, from +0.97 ± 0.15 at 0 decay to +2.51 ± 0.06 at 2e-3 (Table 24).

Table 5: Each entry gives the quantities at epoch 25 and at epoch 100 (end of training), the latter in bold. The "classifier decayed" column is the recipe of §3.2. The "classifier undecayed" column removes that decay from the classifier's weight matrix alone and changes nothing else. $\|W_{\mathrm{fc}}\|_F$ is the Frobenius norm of the classifier weight matrix and the last layer γ is the mean over that layer's 128 channels of its BatchNorm scale, both taken from a single branch network. The same table at depth 3 is in A.10.

quantity	classifier decayed	classifier undecayed
$\|W_{\mathrm{fc}}\|_F$ at epoch 25 / 100	24.54 ± 0.03 / **23.40 ± 0.02**	55.81 ± 0.08 / **73.71 ± 0.07**
last layer γ at epoch 25 / 100	3.404 ± 0.009 / **5.423 ± 0.008**	2.676 ± 0.007 / **3.878 ± 0.004**
gain of the pair at the last layer, at epoch 25 / 100	+0.77 ± 0.22 / **+2.20 ± 0.10**	+0.46 ± 0.20 / **+0.95 ± 0.15**

The pair at the last layer also helps more only once a certain level of accuracy is reached, the gain first peaking as capacity increases and then declining as the model closes on interpolation (Appendix

A.1). Since increasing the margins should be beneficial only once the network is right often enough, the pattern is expected. The effect is also worth more under scheduling than at a constant learning rate, which fits the demand for a larger scale in mid to late training (Appendix A.9). The two mechanisms also act on different timescales: the first layer gain is stable from early in training, between +0.50 and +0.95 from epoch 25 on, whereas the last layer gain grows strongly late in training, from +0.69 $\pm$ 0.07 at epoch 60 to +2.20 $\pm$ 0.10 at epoch 100 (Table 21).

7 DIRECTIONAL PRECONDITIONING AT THE FIRST LAYER

7.1 Cooling by hand Cooling the single branch by hand along $v_{\max}$ reproduces the gain of the pair. Reproduction by itself does not prove causality, since scaling one direction, for example, also shortens the step. In Appendix A.11 we run controls for this and show among other things that cooling this direction and keeping the gradient norm constant keeps the benefit.

Table 6: Cooling by hand reproduces the first layer's pair in the default cell and at depth 3. Each entry is the gain in train accuracy. The cooling factor is the measured effective learning rate along $v_{\max}$ of the pair divided by that of the single, at the end of training, taken as the median over the open channels and averaged over seeds. It is 0.048 at depth 5 and 0.026 at depth 3. Appendix A.11 runs the controls such as cooling the eigenvector of the next biggest eigenvalue and a random direction (Table 26).

arm	depth 5	depth 3
pair at the first layer	+0.50 $\pm$ 0.14	+1.21 $\pm$ 0.16
the gradient cooled along $v_{\max}$, no second branch	+0.51 $\pm$ 0.21	+1.28 $\pm$ 0.22

7.2 Whitening removes the gain If the effect of the first layer builds on $\lambda_{\max}$ being substantially bigger than the other eigenvalues, it should disappear under ZCA whitening, the images multiplied by $(\Sigma + \varepsilon I)^{\alpha}$ with $\alpha = -\frac{1}{2}$, where the covariance in the transform is that of the whole 32 by 32 by 3 image, not the patch covariance Σ of §3.1. It does, and sharpening the spectrum increases it. The gain is not a result of the residual getting smaller either, as the train accuracy of the single does not substantially change across the input spectrum ladder. In contrast, the larger gain under sharpening might be explained by the reduced accuracy of the single, although the number of open channels leads us to believe that the cooling mechanism is also at play.

Table 7: The input spectrum ladder at the first layer, depth 3. This is displayed at depth 3 because this is where the gain is the largest. Rows are doses of the one transform $(\Sigma + \varepsilon I)^{\alpha}$, the covariance being that of the whole image, the pair against the single. The ZCA rows have $\alpha = -\frac{1}{2}$. The angle and the alignment are those of the training time probe, the mean over seeds of each seed's median over its open channels, with the 25th and 75th percentiles of the same channels pooled across seeds in brackets. The open column is the mean number of open channels per seed, of 32. The same ladder at depth 5 is Table 30. A.12 changes the dataset instead, and shows that the geometry in that case is also determined by the spectrum. Bold marks a gain more than two standard errors above zero.

dose	single, train	pair gain	angle	open, of 32	$v_{\max}$ alignment
sharpened, α = 0.30, ε = 0.01	47.68 %	**+3.99 $\pm$ 0.42**	146° (138–162)	29.0 $\pm$ 0.6	0.998 (0.994–1.000)
unwhitened	58.77 %	**+1.21 $\pm$ 0.16**	146° (132–152)	26.3 $\pm$ 0.9	0.999 (0.992–0.999)
ZCA, ε = 10	58.94 %	**+0.54 $\pm$ 0.13**	126° (103–134)	24.3 $\pm$ 1.3	0.995 (0.986–0.998)
ZCA, ε = 1	58.88 %	+0.11 $\pm$ 0.19	110° (85–116)	23.0 $\pm$ 0.0	0.986 (0.980–0.991)
ZCA, ε = 0.1	58.13 %	+0.15 $\pm$ 0.20	86° (71–98)	16.3 $\pm$ 0.3	0.265 (0.203–0.347)

7.3 A synthetic model with one normalized layer To study how the system behaves at different limits of the empirical spectrum, we build a synthetic model whose spectrum we can set freely: one normalized layer of 64 channels, ReLU then a linear readout of 10 outputs, on a fixed sample of 4,096 points in 27 dimensions whose empirical covariance is Σ. It is trained by full batch descent under mean squared error against a fixed two layer ReLU teacher of 128 hidden units and 10 outputs, drawn once (the full setup is in Appendix D.1). Here the gain is a reduction in training loss, not an accuracy difference. The deterministic model reproduces the spectrum dependent benefit, the parking of the separation axis on $v_{\max}$ and the ordering of the resting angle by the spectrum (Appendix D.2), although its angle rests 11 to 29 degrees more open. What we learn from this is that the phenomena discussed are not specific to convolutions. The gain also disappears when $\lambda_{\max} = \lambda_{\text{next}}$, although the separation axis has a large component in the plane spanned by the directions corresponding to $\lambda_{\max}$ and λ_{next} (Appendix D.3, Table 36).

8 SCALING UP

We now repeat the argument on ImageNet at full resolution on five networks. We train five arms: the single, the pair at the first layer alone, at the last alone and at both, and the single branch reproduction, a single branch network carrying the two learning rate effects by hand. The pair at both ends and its single branch reproduction survive at this scale on every network. The first layer's term alone is small, and on MobileNetV4 and MobileOne-S0 it sits close to its own standard error.

The recipe: ImageNet at 224 pixels, a 40 epoch cosine from learning rate 0.1 at batch 256, SGD with momentum 0.9, kernel only weight decay 1e-4, random resized crop and flip, three seeds. Validation is the standard resize to 256 and center crop to 224. Everything else follows §3.2.

The networks: we experiment on five models, chosen for being small enough to train in this cell and for admitting both mechanisms as built: a dense convolution on raw pixels at the stem, and a final convolution with its own normalization whose scale reaches the logits. MobileOne-S0 is trained in its deployed form, with each of its multi branch blocks replaced by the single convolution, BatchNorm and ReLU it folds into, so the pair is the only branching in it.

In Table 8 below we can see that the mechanisms transfer.

Table 8: The pair on ImageNet, three seeds. Each entry is the paired per seed gain in train accuracy over the single, in percentage points, and the two numbers after the name of each network are the train and test accuracy of the single in percent. The first three columns place the two branch block at the first convolution alone, at the last alone, and at both. The reproduction is a single branch network with the first layer's gradient cooled along $v_{\max}$ and $\gamma\times4$ $\beta\times2$ on the last convolution, the multiplier of §6.1. The cooling factor is the measured effective learning rate of the pair along $v_{\max}$ over the single's at the end of training. Median over the open channels, mean over seeds: 0.016 ResNet-18, 0.078 ShuffleNetV2, 0.012 MobileNetV3-Small, 0.087 MobileNetV4, 0.21 MobileOne-S0.

network, single train / test	first layer alone	last layer alone	pair, both	reproduction
ResNet-18, 68.5 / 69.2	+0.20 $\pm$ 0.02	+0.36 $\pm$ 0.02	+0.50 $\pm$ 0.04	+0.39 $\pm$ 0.01
ShuffleNetV2 $\times$1.0, 64.4 / 66.4	+0.18 $\pm$ 0.08	+0.99 $\pm$ 0.13	+1.02 $\pm$ 0.04	+0.92 $\pm$ 0.04
MobileNetV3-Small, 59.3 / 62.7	+0.22 $\pm$ 0.02	+0.88 $\pm$ 0.40	+0.80 $\pm$ 0.21	+1.04 $\pm$ 0.20
MobileNetV4-Conv-Small, 70.7 / 69.0	+0.21 $\pm$ 0.14	+0.17 $\pm$ 0.06	+0.52 $\pm$ 0.11	+0.53 $\pm$ 0.13
MobileOne-S0, 61.9 / 64.7	+0.35 $\pm$ 0.31	+1.01 $\pm$ 0.36	+0.98 $\pm$ 0.08	+1.40 $\pm$ 0.31

9 LIMITATIONS

1. **One block shape.** We study two identical 3×3 branches with per branch BatchNorm and not RepVGG's 3×3 with a 1×1 and an identity, or MobileOne's four branches. The account is one instance taken apart, not a general theory.

2. **The theory is local.** §5 reads the preconditioner of the block at a point and its two limits. It does not prove convergence to each state.

3. **One plain stack and one recipe.** The controlled study was conducted on a conv–BatchNorm–ReLU stack with ReLU as the only activation. We do show the benefit transfers to AdamW (Appendix A.7) but have not tested other optimizers, and other recipes might behave differently.

4. **The interior layers.** The two ends account for most of the gain. We do not claim that the interior contributes nothing: we simply did not see a benefit we could resolve in our setup.

5. **The test side.** Test gains appear where there is still optimization error to digest. We do not claim anything about generalization, and for all we know this technique can overfit just the same.

6. **We did not tune the structure, and do not claim that the setting of the paper is the best one (Appendix A.9, Table 22).** We did not look for the recipe that extracts the most benefit, or the best way to use the branched block in a deployed model. We hope this will be explored in future work.

Reproducibility statement

Every number in this paper comes from a training run or a measurement that can be repeated. Section 3.2 gives the full recipe and every figure states exactly how it deviates from it. Appendix A carries every sweep the main text summarizes in full, Appendix D carries the complete setup of the synthetic model, and §8 gives the ImageNet cell with Appendix A.14 carrying its five networks in full. The derivation of the preconditioner and of the two limits the main text reads is in Appendix B, with every assumption named where it is made and the measurement that supports it in B.6, and the decay–norm balance it relies on in Appendix C.

AI use statement

In this work we used generative AI tools (ChatGPT, Claude Code, Codex and Gemini) to assist with the mathematical derivations of Appendix B and to check its mathematical claims, to provide feedback on the experimental methodology, to assist with the implementation of the experiments and the measurements, and to generate the figures in this paper. We also used them to draft and edit parts of this paper for readability, to summarize and search the literature, and to format references. We have reviewed all AI-assisted work. Every derivation in Appendix B was checked by hand by the authors, and its assumptions are tested against measurements listed in B.6. Every experiment was specified by the authors. AI-generated code was reviewed by the authors and covered by tests where it forms a measurement. The related work section was checked against the cited papers 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.

References

Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the Optimization of Deep Networks: Implicit Acceleration by Overparameterization. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 244–253. PMLR, 10–15 Jul 2018.

Chengpeng Chen, Zichao Guo, Haien Zeng, Pengfei Xiong, and Jian Dong. RepGhost: A Hardware-Efficient Ghost Module via Re-parameterization. arXiv preprint arXiv:2211.06088, 2024.

Xiangxiang Chu, Liang Li, and Bo Zhang. Make RepVGG Greater Again: A Quantization-Aware Approach. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(10):11624–11632, March 2024.

Xiaohan Ding, Yuchen Guo, Guiguang Ding, and Jungong Han. ACNet: Strengthening the Kernel Skeletons for Powerful CNN via Asymmetric Convolution Blocks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Diverse Branch Block: Building a Convolution as an Inception-Like Unit. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10886–10895, June 2021a.

Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. RepVGG: Making VGG-Style ConvNets Great Again. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13733–13742, June 2021b.

Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Scaling Up Your Kernels to 31x31: Revisiting Large Kernel Design in CNNs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11963–11975, June 2022.

Xiaohan Ding, Honghao Chen, Xiangyu Zhang, Kaiqi Huang, Jungong Han, and Guiguang Ding. Re-parameterizing Your Optimizers rather than Architectures. In *The Eleventh International Conference on Learning Representations*, 2023.

Shuxuan Guo, Jose M. Alvarez, and Mathieu Salzmann. ExpandNets: Linear Over-parameterization to Train Compact Convolutional Networks. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1298–1310. Curran Associates, Inc., 2020.

Mu Hu, Junyi Feng, Jiashen Hua, Baisheng Lai, Jianqiang Huang, Xiaojin Gong, and Xian-Sheng Hua. Online Convolutional Re-Parameterization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 568–577, June 2022.

Atli Kosson, Bettina Messmer, and Martin Jaggi. Rotational Equilibrium: How Weight Decay Balances Learning Across Neural Networks. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 25333–25369. PMLR, 21–27 Jul 2024.

Yann LeCun, Patrice Simard, and Barak Pearlmutter. Automatic Learning Rate Maximization by On-Line Estimation of the Hessian's Eigenvectors. In S. Hanson, J. Cowan, and C. Giles (eds.), *Advances in Neural Information Processing Systems*, volume 5. Morgan-Kaufmann, 1992.

Yann LeCun, Leon Bottou, Genevieve B. Orr, and Klaus-Robert Müller. Efficient BackProp. In Genevieve B. Orr and Klaus-Robert Müller (eds.), *Neural Networks: Tricks of the Trade*, pp. 9–50. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

Ilya Loshchilov and Boris Ginsburg. Training nGPT. arXiv preprint arXiv:2608.01284, 2026.

Pavan Kumar Anasosalu Vasu, James Gabriel, Jeff Zhu, Oncel Tuzel, and Anurag Ranjan. FastViT: A Fast Hybrid Vision Transformer Using Structural Reparameterization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5785–5795, October 2023a.

Pavan Kumar Anasosalu Vasu, James Gabriel, Jeff Zhu, Oncel Tuzel, and Anurag Ranjan. MobileOne: An Improved One Millisecond Mobile Backbone. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7907–7917, June 2023b.

Ao Wang, Hui Chen, Zijia Lin, Jungong Han, and Guiguang Ding. RepViT: Revisiting Mobile CNN From ViT Perspective. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15909–15920, June 2024.

Guoyu Yang, Yuan Wang, Daming Shi, and Yanzhong Wang. Golden Cudgel Network for Real-Time Semantic Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 25367–25376, June 2025.

Sergey Zagoruyko and Nikos Komodakis. DiracNets: Training Very Deep Neural Networks Without Skip-Connections. arXiv preprint arXiv:1706.00388, 2018.

Mingyang Zhang, Xinyi Yu, Jingtao Rong, and Linlin Ou. RepNAS: Searching for Efficient Re-parameterizing Blocks. arXiv preprint arXiv:2109.03508, 2022.

Mingshu Zhao, Yi Luo, and Yong Ouyang. RepNeXt: A Fast Multi-Scale CNN using Structural Reparameterization. arXiv preprint arXiv:2406.16004, 2024.

A CONTROLS AND LONG FORM RESULTS

This appendix covers the full form of every sweep the body quotes and every control the body summarizes.

Conventions are those of §3.1 and §3.2 and are not restated per caption: the open threshold of 10°, the parked threshold of 0.95, the paired per seed difference against its own single at the same seed, and the median over open channels with its 25th and 75th percentiles in brackets. In the appendix each result uses three seeds unless its caption says otherwise, against five in the main text.

A.1 THE PLACEMENT SWEEP IN FULL

Main text Figure 1 draws one column of this table. Here are all six, three depths under the cosine and the same three at a constant learning rate of 0.1, which is what lets each layer's gain be checked for its dependence on the schedule.

Table 9: The placement sweep, CIFAR-100 at 32/64/128, the two branch block at one layer with every other layer single. The rows give the index of the layer in which the pair is placed while all other layers carry the single only. The largest entry of each column is bold. The constant rate columns hold the learning rate at 0.1 for all 100 epochs and change nothing else. Five seeds in the depth 5 cosine column, in the first and last layers of the depth 3 cosine column and in the first layer of the depth 5 constant rate column, three seeds elsewhere.

paired layer	depth 3, cosine	depth 3, constant rate	depth 5, cosine	depth 5, constant rate	depth 8, cosine	depth 8, constant rate
0, the first	$+1.21 \pm 0.16$	$\mathbf{+1.89 \pm 0.20}$	$+0.50 \pm 0.14$	$\mathbf{+0.44 \pm 0.07}$	$+0.50 \pm 0.06$	$\mathbf{+0.52 \pm 0.14}$
1	-0.02 ± 0.05	$+0.79 \pm 0.28$	$+0.38 \pm 0.08$	-0.25 ± 0.30	$+0.32 \pm 0.10$	$+0.44 \pm 0.09$
2	$\mathbf{+1.53 \pm 0.10}$	$+0.81 \pm 0.24$	$+0.09 \pm 0.12$	-0.56 ± 0.32	-0.05 ± 0.10	$+0.27 \pm 0.15$
3	—	—	-0.03 ± 0.12	-0.48 ± 0.17	$+0.04 \pm 0.12$	-0.00 ± 0.09
4	—	—	$\mathbf{+2.20 \pm 0.10}$	-0.02 ± 0.20	-0.05 ± 0.17	$+0.25 \pm 0.11$
5	—	—	—	—	$+0.04 \pm 0.09$	-0.11 ± 0.17
6	—	—	—	—	-0.04 ± 0.07	-0.13 ± 0.13
7	—	—	—	—	$\mathbf{+1.05 \pm 0.16}$	$+0.37 \pm 0.16$

Removing the scheduling leaves the first layer's gain largely intact and cuts the last layer's, which is what A.9 reads as the boost being a late training term whose size the anneal sets.

Additivity. If the gain is a first layer term and a last layer term the two should sum to the whole network, and they do, along both capacity axes.

Table 10: The two terms along depth, CIFAR-100 at 32/64/128 under the default recipe. Five seeds at depth 4, in the depth 5 row, in depth 8's pair at every layer and in the first layer and last layer terms of depth 3, three seeds elsewhere.

depth at width 1.0	single, train, percent	pair at the first layer	pair at the last layer	pair at every layer
3	59.06 ± 0.05	$+1.21 \pm 0.16$	$+1.53 \pm 0.10$	$+3.10 \pm 0.20$
4	75.35 ± 0.08	$+1.45 \pm 0.15$	$+2.45 \pm 0.09$	$+3.68 \pm 0.13$
5	81.82 ± 0.08	$+0.50 \pm 0.14$	$+2.20 \pm 0.10$	$+2.69 \pm 0.09$
6	83.94 ± 0.05	$+0.81 \pm 0.06$	$+2.17 \pm 0.02$	$+2.59 \pm 0.13$
8	93.50 ± 0.12	$+0.50 \pm 0.06$	$+1.05 \pm 0.16$	$+1.48 \pm 0.04$
10	96.95 ± 0.06	$+0.15 \pm 0.08$	$+0.52 \pm 0.09$	$+0.66 \pm 0.05$
12	97.31 ± 0.08	$+0.11 \pm 0.07$	$+0.41 \pm 0.08$	$+0.67 \pm 0.05$
15	98.29 ± 0.03	$+0.13 \pm 0.04$	$+0.21 \pm 0.07$	$+0.34 \pm 0.06$

The two terms behave differently. The last layer's term rises to a peak and falls, +1.05, +1.11, +2.20, +1.24 and +0.01 across the width axis and +1.53, +2.45, +2.20, +2.17, +1.05, +0.52, +0.41 and +0.21 across depth, which is the shape the margin reading predicts: growing the logit scale pays only once the network is right often enough, and stops paying once there is no residual left. The first layer's term has no such window. It is +0.42 to +1.28 at every width from 47 to 97 percent train accuracy and reaches zero only at interpolation.

Table 12 gives the three arms of Table 1 with their test gains.

Table 11: The two terms along width at depth 5, columns as Table 10, with the $\gamma\times4$ $\beta\times2$ multiplier on the last layer of a single branch network beside the last layer's pair. The width column states the multiplier on the 32/64/128 default with its actual width given beside it. Five seeds in the last layer column at every width and in the width 1.0 row, three seeds elsewhere.

width at depth 5	single, train, percent	pair at the first layer	pair at the last layer	$\gamma\times4$ $\beta\times2$ on the last layer	pair at every layer
0.25 (8/16/32)	46.87 ± 0.16	+1.28 ± 0.10	+1.05 ± 0.27	+0.89 ± 0.43	+1.42 ± 0.28
0.5 (16/32/64)	64.66 ± 0.19	+0.42 ± 0.22	+1.11 ± 0.15	+1.30 ± 0.06	+1.69 ± 0.17
1.0 (32/64/128)	81.82 ± 0.08	+0.50 ± 0.14	+2.20 ± 0.10	+2.12 ± 0.12	+2.69 ± 0.09
2.0 (64/128/256)	97.36 ± 0.06	+0.45 ± 0.07	+1.24 ± 0.08	+1.17 ± 0.06	+1.43 ± 0.04
4.0 (128/256/512)	99.94 ± 0.01	−0.01 ± 0.00	+0.01 ± 0.01	+0.01 ± 0.00	+0.04 ± 0.00

Table 12: The arms of Table 1, in the default cell, each the paired per seed gain in train and in test accuracy against the single in percentage points. That single reaches 81.82 ± 0.08 percent train and 66.62 ± 0.10 percent test accuracy. Five seeds.

arm	train gain	test gain
pair at every layer	+2.69 ± 0.09	+0.33 ± 0.29
pair at the first and last layers	+2.77 ± 0.11	+0.55 ± 0.18
single branch reproduction	+2.65 ± 0.11	+0.29 ± 0.15

A.2 THE LEARNING RATE AND THE WEIGHT DECAY

The learning rate sweep here rules out a global learning rate change, and the decay axis rules out the reading that two branches are simply two kernels regularized differently. The sweep also serves as a control for the cooling effect. Global cooling still keeps the two branches ahead. We also compare the branches at constant learning rate to show that the benefit is not a by product of the scheduling.

The learning rate sweep. Six values over a factor of 32, from 0.0125 to 0.4, on CIFAR-100 and CIFAR-10 under the cosine and on CIFAR-100 at a rate held constant for all 100 epochs (Figures 4 to 6), with the two arms' absolute accuracies drawn beside the gains (Figures 7 to 9). If the pair were the single at a different rate, its best rate would sit below the single's and past some rate it would fall below the single, because a step the single already finds too hot is hotter still in the pair. Neither happens. Under the cosine both arms peak at 0.1 on both datasets and fall away from it together, the pair simply better at every rate rather than shifted. At a constant rate the axis never reaches a peak and the pair is always better than the single.

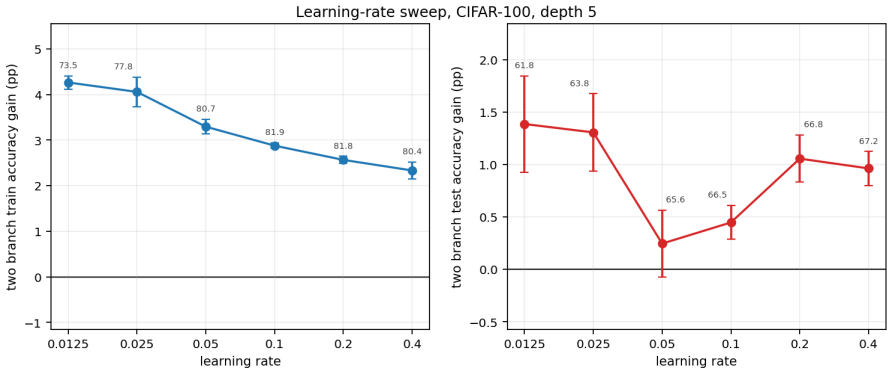

Figure 4: The learning rate sweep at depth 5 on CIFAR-100 under the cosine. The horizontal axis is the initial rate, six values from 0.0125 to 0.4. Three seeds at every rate except 0.1, which is the default cell at five.

The decay sweep. The kernel only decay runs from zero to 2e-3 (Figure 10). The gain of the two branches (at every layer) is present at zero decay, +2.42 ± 0.27, and rises monotonically with the decay to +3.44 ± 0.11 at 2e-3. Decay is therefore not responsible for the gain, but it does set its size, and in the direction §6.2 predicts: kernel only decay reaches the classifier's weight matrix. This sweep moves the regularization over all weight kernels and not just that of the classifier. The sweep

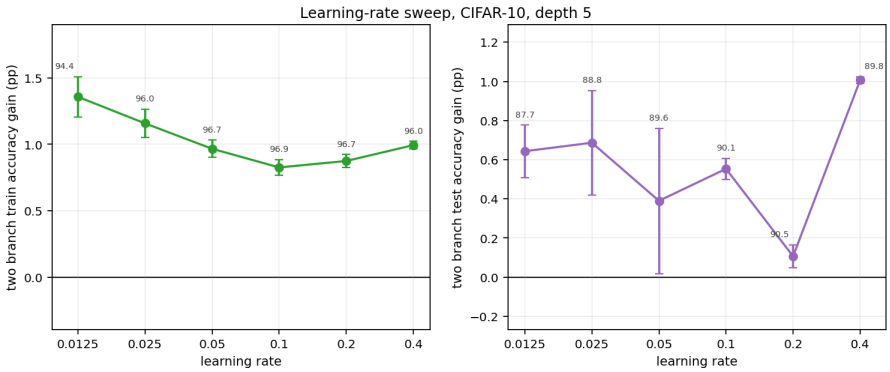

Figure 5: The same sweep on CIFAR-10.

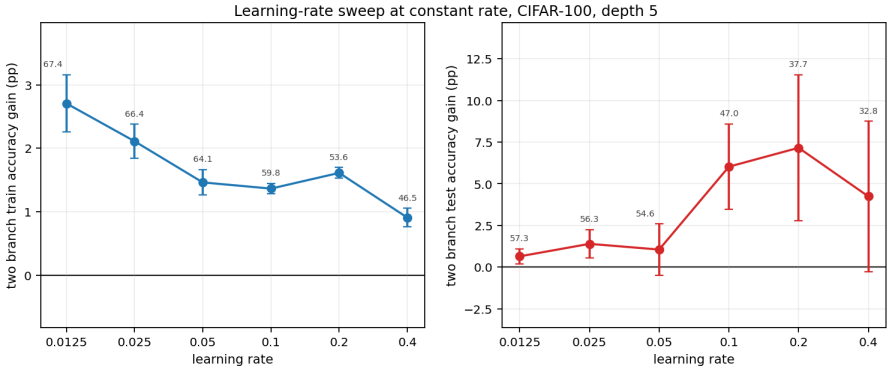

Figure 6: The same sweep on CIFAR-100 with the rate held constant for all 100 epochs instead of annealed.

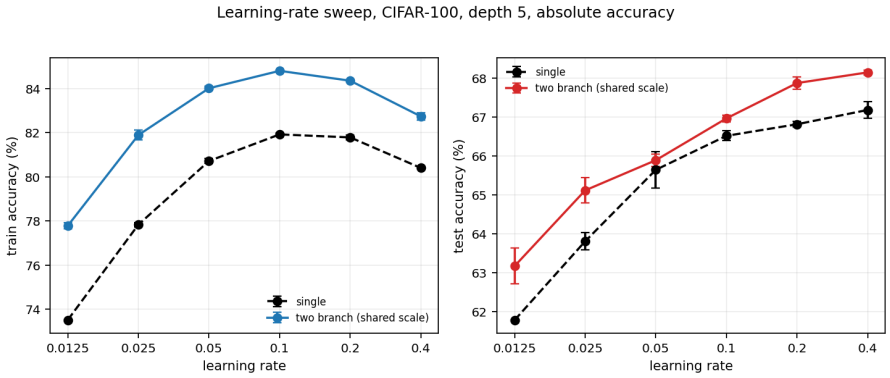

Figure 7: The runs of Figure 4 read as absolute accuracy rather than as a gain. The single is the dashed black line and the two branch arm the colored one, train left and test right, bars one standard error over seeds. In train both arms peak at 0.1 and neither peak sits at a different rate from the other.

that moves the classifier's decay alone, and reads it against the pair and the boost, is Table 24 of this appendix, whose classifier norms and last layer scales are Table 25.

A.3 Additional related work

QARepVGG (Chu et al., 2024) removed the per branch BatchNorms to make the block quantizable. RepLKNet (Ding et al., 2022) used a foldable small kernel branch beside 31×31 kernels as an inductive bias for small data, and RepNAS (Zhang et al., 2022) found that too many branches overfit.

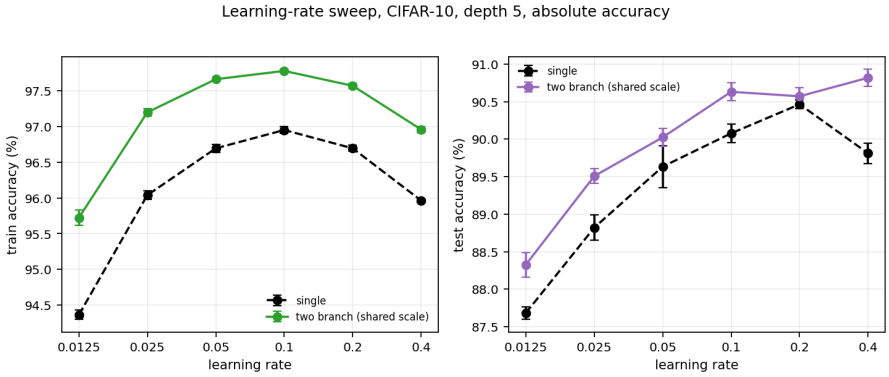

Figure 8: The same reading on CIFAR-10, the runs of Figure 5.

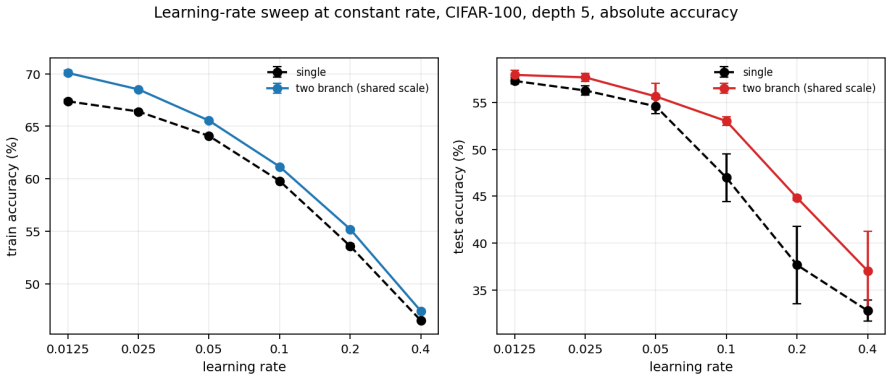

Figure 9: The same reading at a constant rate, the runs of Figure 6. Neither arm reaches a peak: both are best at the coldest rate tested and fall from there, the pair ahead throughout.

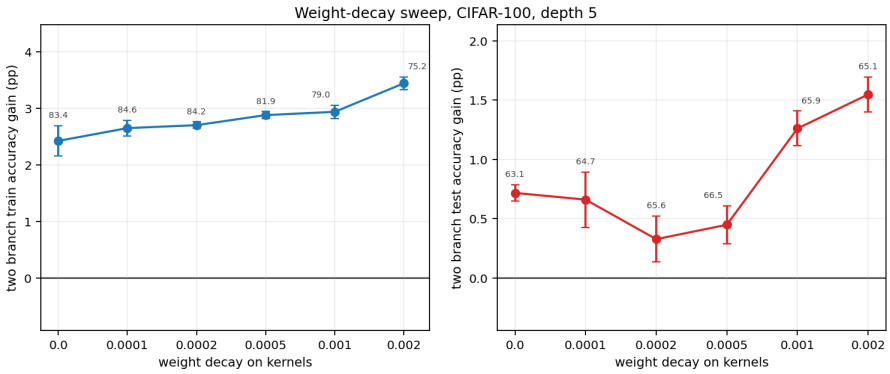

Figure 10: The kernel only weight decay sweep at depth 5 on CIFAR-100. The horizontal axis is the coefficient applied to the convolution and Linear weight tensors alone, with BatchNorm scales and shifts and every bias undecayed, six values from zero to 2e-3. Three seeds at every value except 5e-4, which is the default cell at five.

FastViT, RepViT and RepNeXt (Vasu et al., 2023a; Wang et al., 2024; Zhao et al., 2024) carried the block into more modern models. Golden Cudgel Network (Yang et al., 2025) folds vertical chains of convolutions and horizontal parallel paths into one kernel for real-time segmentation, and attributes the training time gain to "avoiding issues of gradient vanishing and explosion" and to added capacity.

A.4 THE SHARED SCALE

The object of this paper is the block with one BatchNorm scale shared by the two branches, equation (2). The similar block that has been deployed gives each branch its own. This entry holds the three measurements that let the one stand for the other: the effect is present under both conventions, a channel's two scales run together, and the first layer's geometry is the same. The shared scale matches or beats the deployed block on train accuracy at every depth and has similar test accuracy.

The effect under both conventions.

Table 13: Shared scale vs independent scale, CIFAR-100 at 32/64/128 under the default recipe, every layer paired. Rows are the depth of the conv–BatchNorm–ReLU stack. "Independent scale" gives each branch its own BatchNorm scale and bias, and "shared scale" is one scale shared by the two branches, which is the form (2) writes and the paper uses throughout. Five seeds, except three in the shared scale columns at depth 3.

depth	independent scale, train gain	shared scale, train gain	independent scale, test gain	shared scale, test gain
3	+2.15 $\pm$ 0.07	+3.10 $\pm$ 0.20	+1.53 $\pm$ 0.19	+1.96 $\pm$ 0.30
5	+1.97 $\pm$ 0.07	+2.69 $\pm$ 0.09	+0.95 $\pm$ 0.27	+0.33 $\pm$ 0.29
8	+1.10 $\pm$ 0.11	+1.48 $\pm$ 0.04	+0.14 $\pm$ 0.12	+0.34 $\pm$ 0.19

The two scales of a channel. We show that in our experiment the independent scales have on average very close values, strengthening the claim that the shared scale runs in a similar regime to the independent scales. Because the branch labels are arbitrary, the quantity reported is the larger of a channel's two scales over the smaller, which is at least one and does not change if the branches are relabeled. The table shows the statistics of the ratio $\gamma_{\text{larger}}/\gamma_{\text{smaller}}$, not the ratio of the statistics, and its median is very close to 1 at every layer.

Table 14: The ratio of a channel's two branch scales at the end of training for independent scales, in the default depth 5 cell. The channels of all seeds are pooled into one set here rather than summarized per seed as §3.2 asks, deliberately, because the claim is about the distribution over channels and not about a per seed center. Five seeds.

layer	channels in layer	median	25th–75th percentile
first (0)	32	1.053	1.018–1.133
1	64	1.067	1.034–1.121
2	64	1.073	1.037–1.138
3	128	1.076	1.033–1.136
last (4)	128	1.011	1.005–1.019

At the last layer, where the closed limit of §5.2 is read, the two scales agree to about one percent in the median. At the first layer, where the open and parked limit of §5.3 is read, the median is 1.053. Nowhere in the network does the typical channel run its two branches at appreciably different scales.

The deployed block. The two figures below are main text Figures 1 and 3 remade with independent scales, five seeds, and the block behaves and looks the same. Their captions there apply unchanged.

A.5 THE REGIME IN FULL

Main text Figure 2 draws the train panes of the two CIFAR-100 sweeps. Here all four sweeps are drawn in full, train beside test, on both CIFAR-100 and CIFAR-10.

Each figure is two panes, train on the left and test on the right. The "gain" vertical axis is the paired per seed accuracy difference of the arm paired at every layer minus the single at the same depth or width and seed, in percentage points, and the bars are one standard error over seeds. The number printed over each point is the single branch's accuracy in that pane, so a point whose annotation is near 100 has no accuracy residual left.

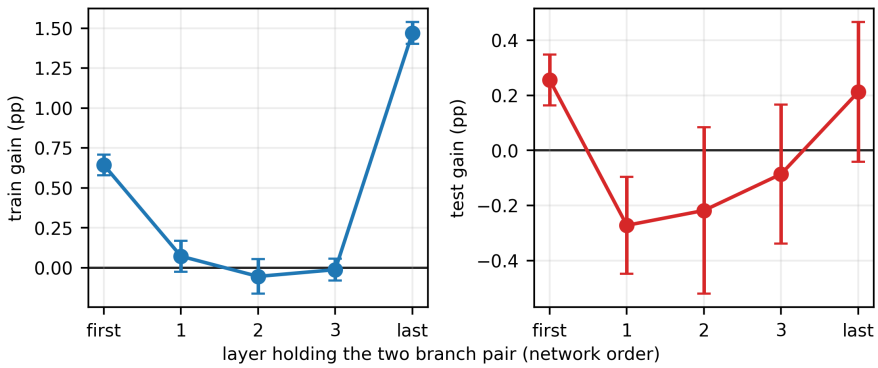

Figure 11: Figure 1 with independent scales: the same ends only shape, with a flatter interior. The single branch network reaches 81.90 ± 0.06 percent train and 66.64 ± 0.11 percent test accuracy.

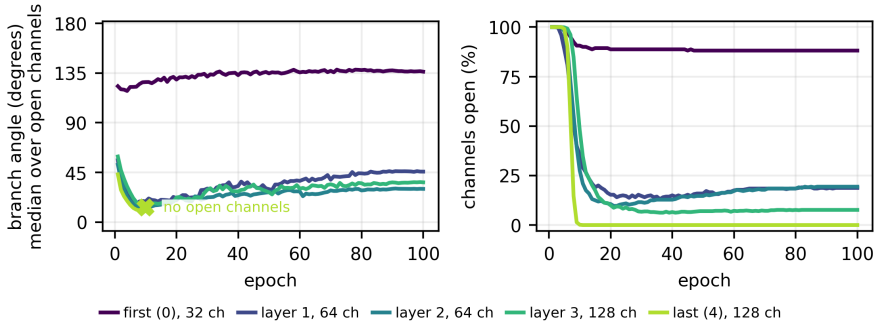

Figure 12: Figure 3 with independent scales: the same geometry, the first layer open past 90° throughout and the last closed by epoch 9.

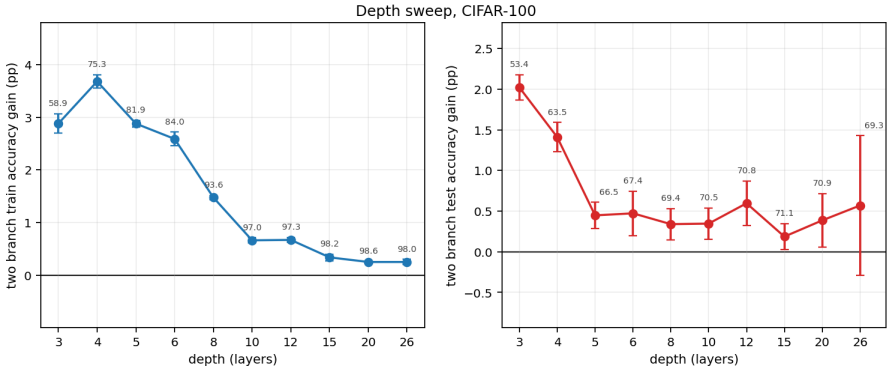

Figure 13: Depth on CIFAR-100 at 32/64/128 under the default recipe. The horizontal axis is the depth of the conv–BatchNorm–ReLU stack. Five seeds at depths 3, 4, 5 and 8, three elsewhere.

The effect behaves as the literature reports. It is small, it decays with depth, and it is gone at interpolation. The gain along the width axis, which this paper adds, also decays when the model size is large.

A.6 The batch sweep in full

Main text Table 2 gives the batch sweep of the default cell and §4.2 the full batch limit. Table 15 gives both in full, with their test gains. The full batch row is CIFAR-100 at depth 3 with 16, 32 and

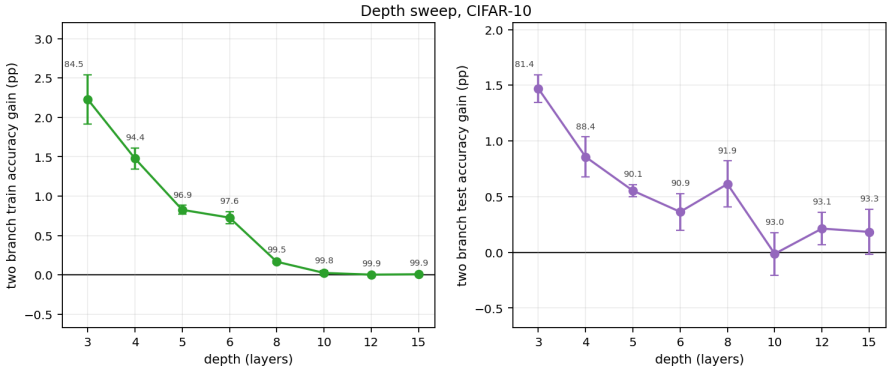

Figure 14: Depth on CIFAR-10. The CIFAR-10 figures draw train in green and test in purple.

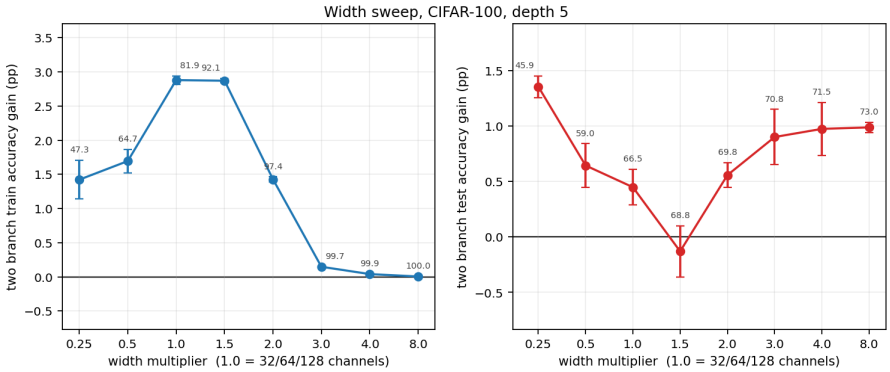

Figure 15: Width at depth 5 on CIFAR-100. The horizontal axis is the multiplier on the 32/64/128 channel stack. Five seeds at width 1.0, three elsewhere.

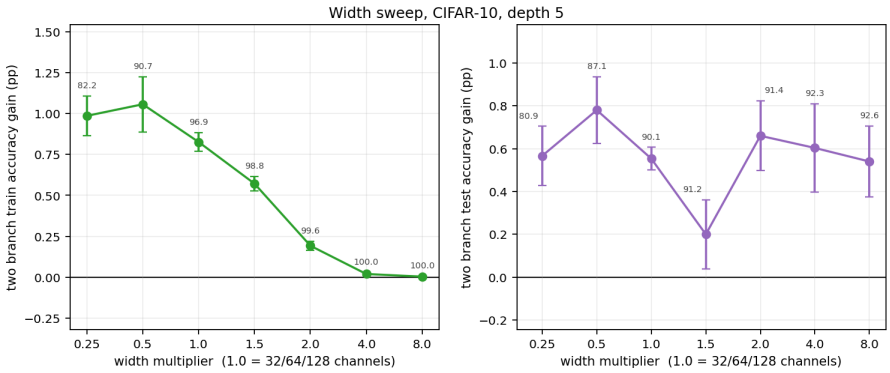

Figure 16: Width at depth 5 on CIFAR-10.

64 channels, no augmentation, kernel only decay 5e-4 and the learning rate held at 0.05, for 5,000 epochs of one optimizer step each.

A.7 An adaptive optimizer

The mechanisms of §5 are effects on particular coordinates of an SGD step. An adaptive optimizer divides every coordinate by the square root of its own second moment, so the question is whether the mechanisms of this paper transfer. They do. The AdamW arms keep the default cell and change only the optimizer: AdamW with its default moment decay rates of 0.9 and 0.999 and stability constant 1e-8, a cosine from learning rate 3e-3, and decoupled weight decay 5e-4 on the kernels only.

Table 15: The batch size sweep in the default cell, and the full batch limit at depth 3 at 16/32/64 without augmentation at a constant learning rate of 0.05. Rows are batch sizes. 50,000 is the whole training split, so one optimizer step per epoch. "pair at the first layer" is the two branch block at layer 0 alone with every other layer single and "pair at every layer" is every layer paired. The matched fit readings at the default cell's batch sizes are in Table 2. At full batch, read at the first epoch at which the single reaches 35 percent train accuracy, the pair at the first layer gains +0.87 $\pm$ 0.43 and the pair at every layer +5.68 $\pm$ 0.25. Five seeds throughout.

batch	epochs	single, train	pair at the first layer, end of run	pair at every layer, end of run	pair at the first layer, test	pair at every layer, test
the default cell, depth 5						
128	100	81.82 %	+0.50 $\pm$ 0.14	+2.69 $\pm$ 0.09	+0.36 $\pm$ 0.11	+0.33 $\pm$ 0.29
256	200	86.59 %	+0.51 $\pm$ 0.12	+3.21 $\pm$ 0.19	+0.06 $\pm$ 0.19	+0.12 $\pm$ 0.28
512	400	90.43 %	+0.65 $\pm$ 0.09	+3.47 $\pm$ 0.09	+0.35 $\pm$ 0.16	$-$0.02 $\pm$ 0.20
1,024	800	93.24 %	+0.45 $\pm$ 0.08	+3.24 $\pm$ 0.09	+0.09 $\pm$ 0.10	$-$0.22 $\pm$ 0.16
2,048	1,600	94.86 %	+0.45 $\pm$ 0.08	+3.08 $\pm$ 0.07	+0.21 $\pm$ 0.20	+0.17 $\pm$ 0.20
depth 3 at 16/32/64, full batch						
50,000	5,000	49.02 %	+0.98 $\pm$ 0.22	+6.75 $\pm$ 0.53	$-$0.83 $\pm$ 0.58	$-$0.15 $\pm$ 1.16

Under Adam the closed pair's multiplier is not the $\times 4$ of §5.2. That $\times 4$ is two factors of two, both on the scale coordinate: the gradient on the shared γ is $(p_1 + p_2)^\top r$, twice a single branch's, and the output moves by $(p_1 + p_2)$ per unit change of γ, again twice. Adam normalizes each coordinate's step by its own gradient scale, so the first factor is divided out and only the second survives. The two shifts are two parameters that each take their own normalized step, so their $\times 2$ stays. A closed pair under Adam should therefore behave as $\gamma\times 2$ $\beta\times 2$.

Table 16: The two placements and the last layer's multiplier under AdamW, beside the same arms under SGD. Both mechanisms transfer. The geometry column is the training time probe read as §3.2 says, the open count and the angle each formed per seed and then averaged over seeds.

arm	gain under AdamW	the same arm under SGD	test gain under AdamW	geometry at the paired layer, epoch 100
pair at every layer	+2.33 $\pm$ 0.14	+2.69 $\pm$ 0.09	+0.52 $\pm$ 0.22	layer 0: 31.7 of 32 open, 139°, on $v_{\max}$ 0.988, last layer closed
pair at the first layer	+0.53 $\pm$ 0.08	+0.50 $\pm$ 0.14	+0.15 $\pm$ 0.22	31.7 of 32 open, 142°, on $v_{\max}$ 0.988
pair at the last layer	+1.34 $\pm$ 0.30	+2.20 $\pm$ 0.10	+0.20 $\pm$ 0.10	closed, 0 of 128 open, cosine between the two raw kernels 0.9916
$\gamma\times 2$ $\beta\times 2$ on the last layer, no second branch	+1.14 $\pm$ 0.11	not run	$-$0.02 $\pm$ 0.04	—

A.8 The geometry over training

Main text Figure 3 draws the angle and open fraction per layer and Table 3 gives the first layer at three epochs. Here is the geometry end state, the angle and the alignment. For the last layer all channels but a single straggler are closed by epoch 10.

Table 17: Branch angle against placement gain, default cell. Rows are layers from 0 at the input. The angle is the Σ-whitened branch angle of §3.1, the median over the layer's open channels and then the mean over seeds, as in Table 3. The alignment of the last layer is undefined as there are no open channels. Five seeds.

layer	gain when the layer alone is paired	angle, epoch 100	channels open, epoch 100	alignment with $v_{\max}$, epoch 100
0, the first	+0.50 $\pm$ 0.14	141°	27.4 of 32 (86 %)	0.998
1	+0.38 $\pm$ 0.08	45°	15.8 of 64 (25 %)	0.361
2	+0.09 $\pm$ 0.12	32°	27.8 of 64 (43 %)	0.227
3	$-$0.03 $\pm$ 0.12	38°	17.6 of 128 (14 %)	0.843
4, the last	+2.20 $\pm$ 0.10	closed, 0°	0.0 of 128	undefined

A Kaiming draw is isotropic in kernel space, not in the whitened frame, so at layer 0 the separation axis starts tilted toward $v_{\max}$ before any training (Table 3, Table 18). The isotropic arm removes that tilt: every branch kernel of layer 0 is $\Sigma^{-1/2}$ applied to a standard normal draw, rescaled to the Kaiming kernel's response variance $w^\top \Sigma w$, so the axis starts at the random direction overlap of about 0.15 with everything else unchanged. Both arms park within two epochs.

Table 18: The initial tilt removed, layer 0 of the default cell but with constant learning rate, three seeds an arm. Rows are the initialization and the ends of epochs 1 to 10. "overlap" is $|\hat{p}_- \cdot v_{\max}|$, the median over the layer's open channels and then the mean over seeds $\pm$ the standard error over seeds. "parked" is the count of those channels at an overlap of at least 0.95, out of 32, the mean over seeds. "angle" is the median whitened branch angle over the same channels and then the mean over seeds. Both arms are the pair at every layer at a learning rate held at 0.1 for ten epochs. The standard columns are the Kaiming draw of §3.2 and the isotropic columns the whitened isotropic draw described above.

epoch	overlap, standard	overlap, isotropic	parked of 32, standard	parked of 32, isotropic	angle, standard	angle, isotropic
0, initialization	0.566 ± 0.060	0.151 ± 0.009	3.0	0.0	99°	90°
1	0.969 ± 0.003	0.899 ± 0.012	22.0	3.0	139°	122°
2	0.987 ± 0.001	0.962 ± 0.003	26.3	20.0	137°	136°
3	0.992 ± 0.001	0.973 ± 0.003	28.0	23.7	138°	140°
4	0.995 ± 0.000	0.982 ± 0.001	28.3	27.0	138°	144°
5	0.995 ± 0.000	0.985 ± 0.002	28.3	28.3	135°	142°
6	0.996 ± 0.000	0.989 ± 0.001	28.7	30.0	135°	145°
7	0.996 ± 0.000	0.993 ± 0.001	29.0	29.3	136°	147°
8	0.997 ± 0.001	0.994 ± 0.001	28.3	29.3	137°	146°
9	0.997 ± 0.000	0.996 ± 0.001	29.0	29.3	137°	148°
10	0.997 ± 0.000	0.995 ± 0.001	28.3	29.3	137°	145°

A.9 THE LAST LAYER'S ARMS IN FULL

Table 4 gives two rows of this table. Here is the argument in full: the pair, its reproduction, each half of the multiplier alone, the same multiplier moved elsewhere, and a plain hotter global step.

Table 19: The last layer's boost arms in two cells, ordered as the argument runs. γ and β are the BatchNorm scale and shift of the named layer, and "$\gamma\times4$ $\beta\times2$" means those two parameter groups step at four and two times the network's learning rate with nothing else changed.

arm	default cell	depth 3 at 32/64/128
the pair to explain		
the pair at the last layer	$+2.20 \pm 0.10$	$+1.53 \pm 0.10$
the reproduction, no second branch		
$\gamma\times4$ $\beta\times2$ on the last layer	$+2.12 \pm 0.12$	$+1.44 \pm 0.06$
$\gamma\times4$ alone on the last layer	$+1.66 \pm 0.07$	$+1.17 \pm 0.19$
$\beta\times2$ alone on the last layer	$+0.42 \pm 0.13$	$+0.29 \pm 0.14$
the same multiplier placed elsewhere		
$\gamma\times4$ $\beta\times2$ on the first layer	$+0.16 \pm 0.12$	-0.29 ± 0.08
$\gamma\times4$ $\beta\times2$ on every layer	$+2.29 \pm 0.12$	$+1.35 \pm 0.04$
$\gamma\times4$ alone on every layer	$+1.44 \pm 0.06$	$+1.16 \pm 0.03$
$\beta\times2$ alone on every layer	$+0.45 \pm 0.10$	$+0.37 \pm 0.07$
a plain hotter step		
single at global learning rate 0.2	-0.28 ± 0.02	$+0.07 \pm 0.12$
single at global learning rate 0.4	-1.67 ± 0.07	-0.89 ± 0.07

The multiplier placed at the first layer gives no significant benefit, and a plain hotter global step is the same or worse than the single, so the boost is specific to the last layer's two coordinates and is not a global rate change.

The network only starts paying to grow the logit scale once it is mostly right on the training set. The cosine is what makes the demand expensive to meet, because it cools the whole network over exactly the period in which the demand arrives. A hotter step on that coordinate should therefore be worth most under the anneal and worth less at a rate held constant, where the coordinate was never cooled. That is what both arms do, the pair and its multiplier alike (Table 20).

Table 20: The schedule contrast at the last layer. Multiplier means the $\gamma\times4$ $\beta\times2$ multiplier on a single branch network. The cosine is the default schedule and constant learning rate is 0.1.

cell	pair gain, cosine	multiplier gain, cosine	pair gain, constant rate	multiplier gain, constant rate
depth 3	$+1.53 \pm 0.10$	$+1.44 \pm 0.06$	$+0.81 \pm 0.24$	$+0.59 \pm 0.18$
depth 5	$+2.20 \pm 0.10$	$+2.12 \pm 0.12$	-0.02 ± 0.20	$+0.08 \pm 0.12$

As §6.2 writes, we expect the benefit from this mechanism to arrive in mid to late training, and it does (Table 21).

Table 21: When each term is paid, in the default cell. Rows are epochs of the 100 epoch cosine. The second column is the reference single's own train accuracy at that epoch, which says how far the network has fitted. The last three columns are the paired per seed train accuracy gain of each arm over that single at the same seed, in percentage points. The arms are the pair at the last layer, the $\gamma\times4$ $\beta\times2$ multiplier on that layer with no second branch anywhere, and the pair at the first layer. Five seeds.

epoch	single, train accuracy	pair at the last layer	$\gamma\times4$ $\beta\times2$ multiplier	pair at the first layer
2	20.4 %	$+0.82 \pm 0.17$	$+0.46 \pm 0.14$	$+1.65 \pm 0.24$
10	47.2 %	$+1.19 \pm 0.42$	$+0.76 \pm 0.16$	$+1.14 \pm 0.19$
25	56.0 %	$+0.77 \pm 0.22$	$+0.72 \pm 0.18$	$+0.75 \pm 0.16$
40	60.2 %	$+0.86 \pm 0.15$	$+0.77 \pm 0.22$	$+0.95 \pm 0.09$
50	63.2 %	$+0.69 \pm 0.20$	$+0.67 \pm 0.14$	$+0.60 \pm 0.11$
60	66.6 %	$+0.69 \pm 0.07$	$+0.81 \pm 0.18$	$+0.70 \pm 0.09$
70	70.5 %	$+1.14 \pm 0.08$	$+0.98 \pm 0.10$	$+0.78 \pm 0.14$
80	75.2 %	$+1.35 \pm 0.11$	$+1.36 \pm 0.05$	$+0.58 \pm 0.13$
90	79.8 %	$+1.84 \pm 0.10$	$+1.88 \pm 0.17$	$+0.67 \pm 0.07$
100	81.8 %	$+2.20 \pm 0.10$	$+2.12 \pm 0.12$	$+0.50 \pm 0.14$

The pair supplies a factor of four on the scale and two on the shift because that is what a closed pair happens to supply, not because four is optimal. Pushing past it keeps paying.

Table 22: The last layer's learning rate multiplier beyond the one the pair supplies, depth 3 at 32/64/128. Every multiplier arm carries no second branch anywhere. The single reaches 59.06 percent train accuracy with a last layer γ of 5.729. The pair's γ is the shared scale of (2), and since its last layer is closed its folded scale is $2\gamma = 8.290$, the value of the $\gamma\times4$ arm. Maximal values in bold.

multiplier	train gain	last layer γ	test gain
the last layer pair	$+1.53 \pm 0.10$	4.145	$+1.41 \pm 0.16$
$\gamma\times2$ $\beta\times1$	$+0.82 \pm 0.24$	6.559	$+0.25 \pm 0.25$
$\gamma\times4$ $\beta\times2$, the pair's own multiplier	$+1.44 \pm 0.06$	8.285	$+0.96 \pm 0.21$
$\gamma\times8$ $\beta\times4$	$+2.66 \pm 0.24$	10.373	$+1.63 \pm 0.18$
$\gamma\times16$ $\beta\times8$	$+3.34 \pm 0.08$	13.112	$\mathbf{+1.90 \pm 0.19}$
$\gamma\times32$ $\beta\times16$	$\mathbf{+3.47 \pm 0.10}$	16.222	$+1.86 \pm 0.25$

The gain rises to $\gamma\times32$, so in this cell the pair's own multiplier is not the best one. This paper does not claim to have found the optimum. What the sweep establishes is that the mechanism is a learning rate on those two coordinates and that more of it buys more, up to at least eight times what the pair supplies.

A.10 The scale of the last layer

Main text Table 5 gives this measurement in the default cell at two epochs. Table 23 is the depth 3 version at four epochs, which is where it was first made and which carries the first layer's scale as a control.

A.11 The cooling controls

Scaling one direction of the first layer's kernel gradient does two things at once. It turns the step away from that direction, and it shortens it, because the scaled direction carries part of the gradient's

Table 23: The classifier cap and the last layer's scale, depth 3 at 32/64/128. The first six rows are read from two single branch arms. $\|W_{\text{fc}}\|_F$ is the Frobenius norm of the classifier's weight matrix and γ is the BatchNorm scale, the mean over that layer's channels. Five seeds in the classifier decayed rows, three in the classifier undecayed rows.

quantity	arm	epoch 5	epoch 25	epoch 50	epoch 100
$\|W_{\text{fc}}\|_F$	single	21.10 ± 0.05	26.63 ± 0.01	26.40 ± 0.02	25.85 ± 0.02
$\|W_{\text{fc}}\|_F$	single, classifier undecayed	28.78 ± 0.09	59.91 ± 0.11	73.12 ± 0.08	78.23 ± 0.07
last layer γ	single	1.727 ± 0.013	3.538 ± 0.019	4.730 ± 0.016	5.729 ± 0.011
last layer γ	single, classifier undecayed	1.612 ± 0.004	2.838 ± 0.031	3.437 ± 0.021	3.964 ± 0.023
first layer γ	single	1.086 ± 0.009	1.883 ± 0.068	2.144 ± 0.101	2.148 ± 0.054
first layer γ	single, classifier undecayed	1.087 ± 0.043	1.894 ± 0.048	2.107 ± 0.078	2.127 ± 0.109
gain of the last layer pair	classifier decayed	$+1.00 \pm 0.14$	$+1.15 \pm 0.26$	$+1.04 \pm 0.13$	$+1.53 \pm 0.10$
gain of the last layer pair	classifier undecayed	$+0.94 \pm 0.30$	$+0.66 \pm 0.04$	$+0.21 \pm 0.12$	$+0.62 \pm 0.04$

Table 24: The last layer's pair and boost, $\times 2$ on the bias and $\times 4$ on the scale, against the weight decay on the classifier's weight matrix. The decay on the rest of the kernels stays at the default 5e-4. Five seeds. §6.2 quotes the two ends of the pair's sweep, which are in bold, and Table 25 reads the classifier norm and last layer scale at the same decay values.

decay on the classifier's weight matrix	single, train	pair train gain	boost train gain	single, test	pair test gain	boost test gain
0	84.91 ± 0.14	$\mathbf{+0.97 \pm 0.15}$	$+1.16 \pm 0.16$	66.44 ± 0.12	-0.11 ± 0.15	-0.09 ± 0.23
1e-4	84.26 ± 0.03	$+1.27 \pm 0.13$	$+1.33 \pm 0.08$	66.53 ± 0.12	$+0.00 \pm 0.32$	$+0.13 \pm 0.15$
2e-4	83.44 ± 0.07	$+1.62 \pm 0.10$	$+1.64 \pm 0.04$	66.76 ± 0.05	-0.18 ± 0.18	-0.25 ± 0.16
5e-4, the recipe	81.95 ± 0.06	$+2.02 \pm 0.08$	$+2.02 \pm 0.06$	66.41 ± 0.07	$+0.56 \pm 0.18$	$+0.04 \pm 0.15$
1e-3	80.29 ± 0.01	$+2.44 \pm 0.04$	$+2.34 \pm 0.07$	66.11 ± 0.11	$+0.71 \pm 0.12$	$+0.54 \pm 0.15$
2e-3	78.45 ± 0.08	$\mathbf{+2.51 \pm 0.06}$	$+2.47 \pm 0.07$	65.85 ± 0.14	$+0.89 \pm 0.24$	$+0.81 \pm 0.16$

Table 25: The two quantities of Table 5 along the sweep of Table 24, read from the single branch arm at each decay value. $\|W_{\text{fc}}\|_F$ is the Frobenius norm of the classifier's weight matrix and the last layer γ is the mean over that layer's 128 BatchNorm scales, at epoch 25 and at the end of training. Five seeds.

decay on the classifier's weight matrix	$\|W_{\text{fc}}\|_F$, epoch 25	last layer γ, epoch 25	$\|W_{\text{fc}}\|_F$, epoch 100	last layer γ, epoch 100
0	55.81 ± 0.08	2.676 ± 0.007	73.71 ± 0.07	3.878 ± 0.004
1e-4	40.84 ± 0.05	2.971 ± 0.006	42.70 ± 0.01	4.682 ± 0.003
2e-4	33.58 ± 0.02	3.151 ± 0.005	33.36 ± 0.03	5.013 ± 0.005
5e-4, the recipe	24.54 ± 0.03	3.404 ± 0.009	23.40 ± 0.02	5.423 ± 0.008
1e-3	18.94 ± 0.04	3.559 ± 0.005	17.57 ± 0.02	5.675 ± 0.008
2e-3	14.59 ± 0.02	3.713 ± 0.004	13.07 ± 0.01	5.893 ± 0.005

length. The arm of §7.1 does both, so on its own it cannot say which of the two pays. Two arms here separate them. Each keeps one half of the intervention and drops the other, taking the size of the effect it keeps from the intervention itself at every step, so the two are exact complements rather than matched by hand.

The arm that holds the step length restores each channel's gradient norm after the direction has been scaled, and so turns the step without shortening it. The arm that holds the direction shrinks each channel's whole gradient by the factor the directional scaling would have produced at that step, and so shortens the step without turning it.

Table 26: The cooling controls in the default cell. Every arm is a single branch network except the first, identical to the others except for what is done to the first layer's kernel gradient. The factor is 0.048 except in the rows that say otherwise, and it is 20.83, the reciprocal of 0.048, in the two heated rows, which turn the step toward $v_{\max}$ instead of away from it. Length held means that each channel's gradient norm is restored after the cooling to what it was before, so the step is turned and not shortened.

arm	gain
pair at the first layer	$+0.50 \pm 0.14$
cooled along $v_{\max}$, the arm of §7.1	$+0.51 \pm 0.21$
the same shortening, direction held	-0.06 ± 0.15
frozen along $v_{\max}$, length held (factor 0)	$+0.24 \pm 0.10$
cooled at 0.024, length held	$+0.47 \pm 0.12$
cooled at 0.048, length held	$+0.47 \pm 0.11$
cooled at 0.096, length held	$+0.38 \pm 0.15$
cooled at 0.192, length held	$+0.35 \pm 0.11$
cooled on the eigenvector of λ_{next}	$+0.18 \pm 0.11$
cooled on a random orthogonal direction	-0.33 ± 0.08
heated along $v_{\max}$	-3.67 ± 0.20
heated along $v_{\max}$, length held	-1.90 ± 0.08

A.12 Choosing a predictor

The theory of Appendix B says that the first layer's dynamics are driven by the spectrum of the input covariance, and the input spectrum ladder of §7.2 confirms it. This motivates us to ask whether a quantity can be computed from the data alone that will predict the dynamics of the pair: the value of the angle, the number of open channels and the time to alignment with $v_{\max}$. The gain itself is not compared across datasets, since four of the five reach 98 to 100 percent train accuracy, where §4 shows the pair has nothing left to gain.

We try this using the input spectrum ladder and four other datasets apart from CIFAR-100. All five run in the default cell, depth 5 at 32/64/128 under the cosine from 0.1 with kernel only weight decay 5e-4, at the image scale of CIFAR, each at its own step matched budget.

Table 27: The first layer's mechanism on five datasets in the default cell, the pair at the first layer alone. Rows are sorted by $\mu_{\text{eff}} = (\lambda_{\max} - \lambda_{\text{next}})/\text{tr}\,\Sigma$.

dataset	μ_{eff}	$\lambda_{\max}/\lambda_{\text{next}}$	single, train	angle	open, of 32	alignment
GTSRB	0.75	14.4	100.00 %	152° (147–156)	31.0	0.998
SVHN	0.71	10.3	98.45 %	143° (134–146)	27.0	0.998
EuroSAT	0.68	10.5	99.86 %	143° (131–153)	26.0	0.997
CIFAR-100	0.63	9.6	81.82 %	143° (132–151)	28.3	0.999
Flowers-102	0.55	6.7	99.94 %	131° (120–144)	25.7	0.995

We check four predictors against three quantities. The alignment time is the optimizer step at which half of the channels open at that step have parked, alignment at least 0.95, the median over seeds. The angle is the whitened branch angle over open channels at the final epoch, and the open channels are those past 10° (§3.1), out of 32.

Table 28: Four candidate predictors against the dynamics they are meant to order, Spearman rank correlation. "Alignment time" is the optimizer step at which half of the first layer's channels open at that step have parked, alignment at least 0.95. Its sign is negative because a larger predictor means it aligns earlier. The alignment time column runs over the eight settings in which half of the open channels park. The flattest whitening dose, in which they never do, is left out of that column.

predictor	alignment time	angle	open channels
$\mu_{\text{eff}} = (\lambda_{\max} - \lambda_{\text{next}})/\text{tr}\,\Sigma$	-0.85	$+0.97$	$+0.81$
$\lambda_{\max}/(\text{tr}\,\Sigma - \lambda_{\max})$	-0.85	$+0.97$	$+0.81$
$\lambda_{\max}/\lambda_{\text{next}}$	-0.85	$+0.93$	$+0.79$
$\lambda_{\max} - \lambda_{\text{next}}$	-0.90	$+0.75$	$+0.50$

The first three candidates order the angle and the open count about equally well; $\lambda_{\max} - \lambda_{\text{next}}$ is best on the alignment time and worst on the other two. μ_{eff} is chosen over the other candidates due

to our conclusions in Appendix D.3. In the synthetic model we see that when $\lambda_{\max}$ and λ_{next} are held and the remaining eigenvalues are lifted (Appendix D.3, Table 37), the ratio does not move while μ_{eff} and the gain both fall. It is chosen over $\lambda_{\max}/(\text{tr}\,\Sigma - \lambda_{\max})$ because that predictor does not see λ_{next}: in the synthetic model the gain vanishes when $\lambda_{\max} = \lambda_{\text{next}}$ (Table 36), where μ_{eff} reads zero and that predictor does not.

Figures 17 to 20 draw the alignment, the open count and the resting angle over the datasets and the input spectrum ladder, and the alignment time over training.

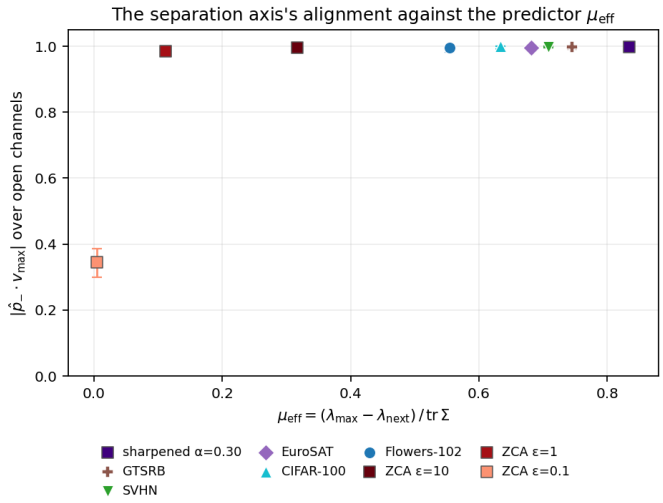

Figure 17: The alignment of the separation axis with $v_{\max}$ against the predictor, nine settings in the default cell. The horizontal axis is $\mu_{\text{eff}} = (\lambda_{\max} - \lambda_{\text{next}})/\text{tr}\,\Sigma$, computed from the patch covariance of the first layer input in that setting. The vertical axis is $|\hat{p}_- \cdot v_{\max}|$ at the final epoch. The marker is the mean over seeds of the median over the open channels and the bar the 25th and 75th percentiles of those channels pooled across seeds. Five seeds in the ZCA and sharpened settings, three in the five datasets, here and in Figures 18 to 20.

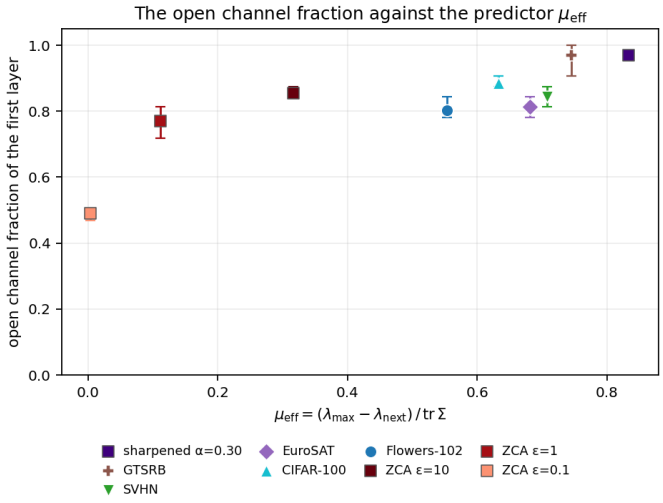

Figure 18: The open channel fraction against the predictor, the same nine settings, axes and markers as Figure 17.

Table 29 gives the realized angle of each setting beside its spectrum.

Main text Table 7 gives this ladder at depth 3. Table 30 gives it in the default cell, which is the cell the nine settings above are read in, and it has the same shape: the gain falls with the spectrum, and past the mild doses the pair pays nothing.

Table 29: The resting angle of the nine settings in the default cell, sorted by μ_{eff}. μ_{eff} is $(\lambda_{\max} - \lambda_{\text{next}})/\text{tr}\,\Sigma$ and $\lambda_{\max}/\lambda_{\text{next}}$ the ratio of the two largest eigenvalues. Five seeds in the ZCA and sharpened rows, three in the dataset rows.

setting	μ_{eff}	$\lambda_{\max}/\lambda_{\text{next}}$	angle
sharpened CIFAR-100, $\alpha = 0.30$	0.83	21.4	151°
GTSRB	0.75	14.4	152°
SVHN	0.71	10.3	143°
EuroSAT	0.68	10.5	143°
CIFAR-100	0.63	9.6	143°
Flowers-102	0.55	6.7	131°
ZCA, $\varepsilon = 10$	0.32	3.3	127°
ZCA, $\varepsilon = 1$	0.11	1.7	101°
ZCA, $\varepsilon = 0.1$	0.004	1.03	73°

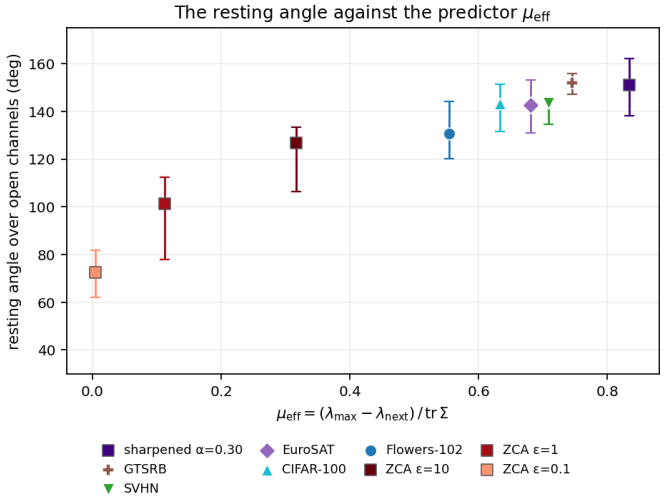

Figure 19: The resting angle against the predictor μ_{eff}, the nine settings of Figure 17 in the default cell. The horizontal axis is $\mu_{\text{eff}} = (\lambda_{\max} - \lambda_{\text{next}})/\text{tr}\,\Sigma$, the vertical axis the whitened branch angle at the final epoch, marker at the mean over seeds of each seed's median over its open channels and bar the 25th and 75th percentiles.

Table 30: The input spectrum ladder at the first layer in the default cell, depth 5 at 32/64/128 under the cosine, the pair at the first layer alone against the single of the same dose and seed. Rows are doses of the transform $(\Sigma + \varepsilon I)^{\alpha}$ applied to the images, with $\alpha = -\frac{1}{2}$ on the ZCA rows. The depth 3 form of this ladder is main text Table 7. Five seeds. The unwhitened row is a separate run from the three seed CIFAR-100 point of Figures 17 to 20. Bold marks a gain more than two standard errors above zero.

dose	single, train	pair gain
sharpened, $\alpha = 0.30$, $\varepsilon = 0.01$	72.88 %	**+2.57 ± 0.14**
unwhitened	81.91 %	**+0.63 ± 0.04**
ZCA, $\varepsilon = 10$	81.83 %	**+0.40 ± 0.11**
ZCA, $\varepsilon = 1$	82.14 %	+0.03 ± 0.08
ZCA, $\varepsilon = 0.1$	81.80 %	−0.05 ± 0.10

A.13 THE INPUT SPECTRUM

Every claim §5 and §7 make about the first layer is a claim about the spectrum of the covariance of the patches its kernel sees. This entry gives that spectrum for ImageNet at 224 pixels, the resolution §8 uses, and for the five datasets of Appendix A.12.

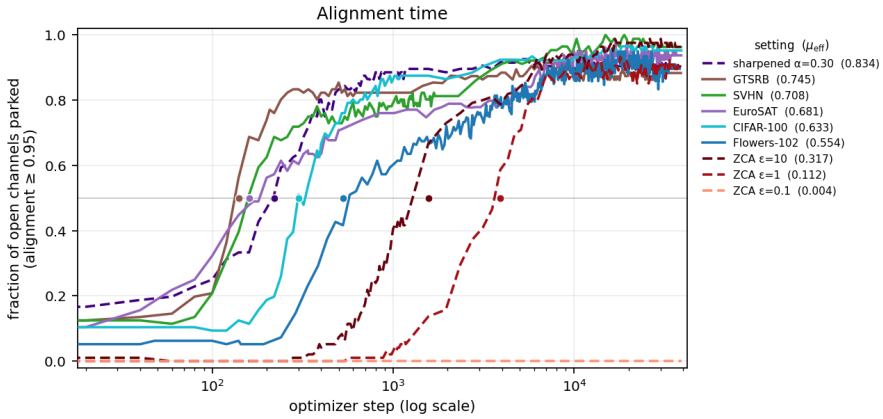

Figure 20: Occupation of the top eigendirection over training, the same nine settings in the default cell. The horizontal axis is the optimizer step on a log scale and the vertical axis the fraction of the currently open first layer channels whose alignment with $v_{\max}$ is at least 0.95, read only where at least eight channels are open. The line is the mean over the seeds and the dot the alignment time, the median over seeds of the step at which half of the channels open at that step have parked. Records are every 20 steps for the first three epochs and once an epoch after. Colors and markers are those of Figure 17, the spectrum doses dashed, the legend ordered by μ_{eff}.

Table 31: The first layer's input spectrum, sorted by μ_{eff}. $\lambda_{\max}/\lambda_{\text{next}}$ is the ratio of the two largest eigenvalues and μ_{eff} is $(\lambda_{\max} - \lambda_{\text{next}})/\text{tr}\,\Sigma$.

setting	$\lambda_{\max}/\lambda_{\text{next}}$	μ_{eff}
ImageNet, 224 pixels	10.1	0.767
GTSRB	14.4	0.745
SVHN	10.3	0.708
EuroSAT	10.5	0.681
CIFAR-100	9.6	0.633
Flowers-102	6.7	0.554

A.14 THE FIVE NETWORKS OF §8 IN FULL

This is the full version of Table 8, split into two: Table 32 shows the gain and Table 33 the geometry. Generally we see that all the phenomena we saw in the smaller scale model appear here. The last layer of MobileNetV4 is the exception: 212 of its 1280 channels are still past 10° in the whitened frame at epoch 40, but the count is falling, from about 270 at epoch 35. The gain pattern holds regardless.

Table 32: Table 8 in full. The five networks of §8 with their parameter count, their published top-1 accuracy (for MobileOne-S0, that of the network trained with its own branches), their own single's train and test accuracy in this cell, and the gains of the four arms, three seeds. The arms are those of Table 8: first alone and last alone place the two branch block at the first or the last convolution only, pair places it at both, and reproduction is the single branch reproduction of Table 8. The gains are Table 8's train gains with their test gains beside them, the paired per seed difference against the single at the same seed in percentage points; the test gain is read from the final epoch validation accuracy, one snapshot per seed. The geometry at the paired positions is in Table 33.

network	parameters	published top-1	single train / test	arm	train gain	test gain
ResNet-18	11.69M	69.76	68.5 / 69.2	first alone	+0.20 ± 0.02	+0.12 ± 0.09
				last alone	+0.36 ± 0.02	+0.00 ± 0.13
				pair	+0.50 ± 0.04	+0.08 ± 0.12
				reproduction	+0.39 ± 0.01	+0.04 ± 0.11
ShuffleNetV2 ×1.0	2.28M	69.36	64.4 / 66.4	first alone	+0.18 ± 0.08	−0.18 ± 0.05
				last alone	+0.99 ± 0.13	+0.27 ± 0.07
				pair	+1.02 ± 0.04	+0.33 ± 0.13
				reproduction	+0.92 ± 0.04	+0.23 ± 0.05
MobileNetV3-Small	2.54M	67.67	59.3 / 62.7	first alone	+0.22 ± 0.02	+0.18 ± 0.02
				last alone	+0.88 ± 0.40	+0.75 ± 0.44
				pair	+0.80 ± 0.21	+0.57 ± 0.30
				reproduction	+1.04 ± 0.20	+0.94 ± 0.21
MobileNetV4-Conv-Small	3.77M	73.8	70.7 / 69.0	first alone	+0.21 ± 0.14	+0.29 ± 0.19
				last alone	+0.17 ± 0.06	−0.46 ± 0.05
				pair	+0.52 ± 0.11	−0.01 ± 0.15
				reproduction	+0.53 ± 0.13	+0.02 ± 0.10
MobileOne-S0	2.09M	71.4	61.9 / 64.7	first alone	+0.35 ± 0.31	+0.32 ± 0.45
				last alone	+1.01 ± 0.36	+0.58 ± 0.42
				pair	+0.98 ± 0.08	+0.64 ± 0.14
				reproduction	+1.40 ± 0.31	+1.04 ± 0.37

Table 33: The geometry at the paired positions of the §8 networks at the end of training, three seeds, for the arms of Table 32 that carry a pair. The geometry column reads the stem's open channels, whitened branch angle and alignment with $v_{\max}$ as §3.1 defines them, and the state of the last layer. Every number is the mean over the three seeds, and each angle and alignment is first the median over that seed's open channels, as in Table 3. The angle of the last layer is given only where it keeps open channels.

network	arm	geometry at the paired position
ResNet-18	first alone	stem 51.0 of 64 open (80 %), 159°, on $v_{\max}$ 1.000
	last alone	last layer kernel cosine 1.0000, 1.3 of 512 open (<1 %)
	pair	stem 50.3 of 64 open (79 %), 159°, on $v_{\max}$ 1.000; last layer kernel cosine 1.0000, 1.0 of 512 open (<1 %)
ShuffleNetV2 ×1.0	first alone	stem 18.0 of 24 open (75 %), 150°, on $v_{\max}$ 0.999
	last alone	last layer kernel cosine 1.0000, 0.0 of 1024 open (0 %)
	pair	stem 17.3 of 24 open (72 %), 149°, on $v_{\max}$ 0.999; last layer kernel cosine 1.0000, 0.3 of 1024 open (<1 %)
MobileNetV3-Small	first alone	stem 12.3 of 16 open (77 %), 152°, on $v_{\max}$ 0.998
	last alone	last layer kernel cosine 1.0000, 1.0 of 576 open (<1 %)
	pair	stem 12.0 of 16 open (75 %), 148°, on $v_{\max}$ 0.999; last layer kernel cosine 1.0000, 1.7 of 576 open (<1 %)
MobileNetV4-Conv-Small	first alone	stem 22.0 of 32 open (69 %), 149°, on $v_{\max}$ 0.999
	last alone	last layer kernel cosine 0.9999, 213.0 of 1280 open (17 %), 20.8°
	pair	stem 23.3 of 32 open (73 %), 150°, on $v_{\max}$ 0.998; last layer kernel cosine 0.9999, 212.3 of 1280 open (17 %), 20.4°
MobileOne-S0	first alone	stem 32.3 of 48 open (67 %), 148°, on $v_{\max}$ 0.999
	last alone	last layer kernel cosine 1.0000, 0.0 of 1024 open (0 %)
	pair	stem 32.0 of 48 open (67 %), 139°, on $v_{\max}$ 0.995; last layer kernel cosine 1.0000, 0.0 of 1024 open (0 %)

B THE PRECONDITIONER IN FULL

Section 5 prints the block's preconditioner twice, once for each of the two states §5.1 measures. This appendix shows the derivation and offers a motivation as to why this phenomenon occurs by taking several approximations, measurements and assumptions about the loss. The full dynamics of a two branch block form a coupled nonlinear system, and we do not prove that training converges to what §5.1 measures. Every statement in this appendix is local and per channel. The per step measurements here estimate Σ per step rather than per epoch as in the main paper.

B.1 THE FLOW

The block's trainable parameters are the shared scale γ, the two kernels w_1 and w_2, and the two shifts, which §5 disposes of.

In the coordinates of §3.1 each branch is a point on the sphere,

$$p_i = \frac{\Sigma^{1/2} w_i}{\sigma_i}, \qquad \sigma_i = \left\| \Sigma^{1/2} w_i \right\|, \qquad \frac{\partial p_i}{\partial w_i} = \frac{P_i \Sigma^{1/2}}{\sigma_i}, \qquad P_i := I - p_i p_i^\top .$$

With $\partial L / \partial p_i = \gamma r$ the chain rule gives the two gradients the block is trained on,

$$\frac{\partial L}{\partial w_i} = \frac{\gamma}{\sigma_i} \, \Sigma^{1/2} P_i \, r, \qquad \frac{\partial L}{\partial \gamma} = (p_1 + p_2)^\top r. \tag{9}$$

Gradient flow on the kernels then moves the two branch directions and the scale as

$$\dot{w}_i = -\frac{\partial L}{\partial w_i}, \qquad \dot{p}_i = \frac{P_i \Sigma^{1/2} \dot{w}_i}{\sigma_i} = -\frac{\gamma}{\sigma_i^2} \, P_i \Sigma P_i \, r, \qquad \dot{\gamma} = -(p_1 + p_2)^\top r,$$

which is (4) of the body. The covariance survives the change of coordinates because the flow is Euclidean in the kernel and not in its whitened image.

B.2 THE OPERATOR IN GENERAL

The network sees only the effective weight, so the flow on the parameters is a flow on that weight through an operator,

$$\dot{v} = \dot{\gamma} \, (p_1 + p_2) + \gamma \, (\dot{p}_1 + \dot{p}_2) = -J J^\top r,$$

with

$$J J^\top = (p_1 + p_2)(p_1 + p_2)^\top + \gamma^2 \left(\frac{P_1 \Sigma P_1}{\sigma_1^2} + \frac{P_2 \Sigma P_2}{\sigma_2^2} \right). \tag{10}$$

This assumes nothing about the angle or the axis.

The first term already carries the whole boost on the scale coordinate. Since $p_1 + p_2 = 2 \cos \frac{\theta}{2} \, \hat{p}_+$ it is

$$(p_1 + p_2)(p_1 + p_2)^\top = 4 \cos^2 \tfrac{\theta}{2} \, \Pi_+,$$

and at $\theta = 0$ that is the $\times 4$ of §5.2 against the $pp^\top$ of (5).

For the rest, take $\sigma_1 = \sigma_2 = \sigma$ (B.6.1) and write the two branches in the frame of §3.1,

$$p_1 = \cos \tfrac{\theta}{2} \, \hat{p}_+ + \sin \tfrac{\theta}{2} \, \hat{p}_-, \qquad p_2 = \cos \tfrac{\theta}{2} \, \hat{p}_+ - \sin \tfrac{\theta}{2} \, \hat{p}_-.$$

Resolving (10) in Π_+, Π_- and Q, and with B as in §5.3, the operator falls into six families:

$$\begin{aligned} JJ^\top = &\left(4\cos^2\tfrac{\theta}{2} + \frac{2\gamma^2}{\sigma^2} B \sin^2\tfrac{\theta}{2}\right)\Pi_+ \\ &+ \frac{2\gamma^2}{\sigma^2} B\cos^2\tfrac{\theta}{2}\,\Pi_- \\ &+ \frac{2\gamma^2}{\sigma^2}\sin^2\tfrac{\theta}{2}\,(\Pi_+\Sigma Q + Q\Sigma\Pi_+) \\ &+ \frac{2\gamma^2}{\sigma^2} Q\Sigma Q \\ &+ \frac{2\gamma^2}{\sigma^2}\cos^2\tfrac{\theta}{2}\,(\Pi_-\Sigma Q + Q\Sigma\Pi_-) \\ &+ \frac{4\gamma^2}{\sigma^2}\sin^2\tfrac{\theta}{2}\cos^2\tfrac{\theta}{2}\,(\Pi_+\Sigma\Pi_- + \Pi_-\Sigma\Pi_+)\,. \end{aligned} \tag{11}$$

The first four families are (7), the form §5.3 quotes. The last two are the coupling of the separation axis to the Q subspace and its coupling to the output direction, and they are what the parked state removes.

B.3 The two states of §5

Closed. At $\theta = 0$ every term carrying $\sin^2\frac{\theta}{2}$ vanishes and $B \to \lambda_-$. Since $\Pi_- + Q = I - \Pi_+$, the three families that remain combine into one,

$$\lambda_-\Pi_- + (\Pi_-\Sigma Q + Q\Sigma\Pi_-) + Q\Sigma Q = (I-\Pi_+)\,\Sigma\,(I-\Pi_+),$$

so with the family that moves the scale the operator is

$$JJ^\top = 4\Pi_+ + \frac{2\gamma^2}{\sigma^2}\,(I-\Pi_+)\,\Sigma\,(I-\Pi_+),$$

which is (6). The family coupling the separation axis to the Q subspace is load-bearing here: without it the closed limit would read

$$4\Pi_+ + \frac{2\gamma^2}{\sigma^2}\,(\lambda_-\Pi_- + Q\Sigma Q)\,,$$

which is not (6). At $\theta = 0$ the separation axis is not defined, and $\hat{p}_-$ can be any unit direction orthogonal to $\hat{p}_+$. The combined form does not depend on which.

Open and parked. When the axis is an eigenvector of the covariance, $\Sigma\hat{p}_- = \lambda_-\hat{p}_-$, two orthogonality identities follow,

$$\hat{p}_+^\top\Sigma\hat{p}_- = \lambda_-\,\hat{p}_+^\top\hat{p}_- = 0, \qquad \hat{p}_-^\top\Sigma q = \lambda_-\,\hat{p}_-^\top q = 0 \quad \text{for every } q \text{ in } Q.$$

The last two families of (11) are therefore both identically zero, and the four that remain are

$$JJ^\top = 4\cos^2\tfrac{\theta}{2}\,\Pi_+ + \frac{2\gamma^2}{\sigma^2}\Big[B\sin^2\tfrac{\theta}{2}\,\Pi_+ + B\cos^2\tfrac{\theta}{2}\,\Pi_- + \sin^2\tfrac{\theta}{2}\,(\Pi_+\Sigma Q + Q\Sigma\Pi_+) + Q\Sigma Q\Big],$$

which is (7). The same two identities make $\hat{p}_-$ an eigenvector of the operator,

$$JJ^\top\hat{p}_- = \frac{2\gamma^2}{\sigma^2}\,B\cos^2\tfrac{\theta}{2}\,\hat{p}_-,$$

which is (8).

B.4 THE SEPARATION AXIS PARKS ON $v_{\max}$

Section 5.1 measures that the first layer's separation axis holds $v_{\max}$ within one epoch and stays there. What follows is the account of how it gets there.

Two assumptions. The first is about the loss: that it is pushing the $v_{\max}$ component out of this channel's output, so that the two components along $v_{\max}$ share a sign,

$$\left(\hat{p}_+^\top v_{\max}\right)\left(v_{\max}^\top r\right) > 0.$$

We call it the sign assumption. It does not follow from anything above, and B.6.3 measures it where the result is claimed. Some assumption about the loss is unavoidable, since the loss is what drives the system, and this is the relevant one: $\hat{p}_+$ and $\hat{p}_-$ are orthogonal, so the axis can align with $v_{\max}$ only if the output direction gives it up. This assumption is required per channel, only for the channels that open, not for the whole layer. The layer can still carry $v_{\max}$ in its other channels.

The second assumption is about the input: that Σ has a top eigenvalue $\lambda_{\max}$ well separated from λ_{next}, so that $\lambda_{\max} - \lambda_{\text{next}}$ is a gap. We call it the gap assumption, and it holds in every setting we test (Appendix A.13).

Beyond these two, the equations depend on quantities that evolve during training, such as the growth of γ. We measure those in the default cell, assume they hold generally, and record them in B.6.

The axis equation. Subtracting the two direction equations of (4), at equal branch σ (B.6.1, and B.6.7 for what the mismatch does),

$$\dot{p}_- = -\frac{\gamma}{2\sigma^2}\left(P_1\Sigma P_1 - P_2\Sigma P_2\right) r. \tag{12}$$

Expanding the projectors in $\hat{p}_+$ and $\hat{p}_-$, normalizing, and projecting out the radial direction gives the motion of the unit axis. It splits exactly into two parts:

- terms carrying $\hat{p}_+^\top r$, the gradient of the scale, and
- terms carrying $\hat{p}_-^\top r$ and $\hat{p}_-^\top \Sigma r$, the gradient along the separation axis.

Empirically the first is small and fluctuating (B.6.5), while the second is larger and sign coherent once the axis has a substantial overlap with $v_{\max}$. We treat them separately below, and show that the first produces fluctuations of the axis along $v_{\max}$ which are enough to give the second something to work with, after which the second takes over.

The scale gradient part. Keeping only the terms in $\hat{p}_+^\top r$ and substituting $\dot{\gamma}$ for it,

$$\begin{aligned} \dot{\hat{p}}_- &= -\frac{\gamma\dot{\gamma}}{2\sigma^2}\left(I - \hat{p}_-\hat{p}_-^\top\right) M\,\hat{p}_-, \\ M &:= \Sigma - 2\cos^2\tfrac{\theta}{2}\left(\hat{p}_+\hat{p}_+^\top\Sigma + \Sigma\hat{p}_+\hat{p}_+^\top\right), \end{aligned} \tag{13}$$

which is the normalized form of a linear flow under the input covariance with a rank two correction along the output direction, that is, Oja's rule for the top eigenvector of M.

We define

$$s(t) := \int_0^t \frac{\gamma\dot{\gamma}}{2\sigma^2}\, dt'.$$

Dropping the correction, so that M is replaced by Σ held fixed, the flow can be solved in the eigenbasis of Σ. Writing $a_i(t)$ for the weight of the axis on the i-th eigenvector of Σ and $a_{\max}(t)$ for its weight on $v_{\max}$,

$$\frac{a_i(t)}{a_{\max}(t)} = \frac{a_i(0)}{a_{\max}(0)}\, e^{\,s(t)\,(\lambda_{\max}-\lambda_i)}, \tag{14}$$

so for $s < 0$ the weight converges exponentially onto $v_{\max}$ and for $s > 0$ it escapes at the same rate.

The correction to Σ in M lies along the output direction, and it matters only through the weight $\hat{p}_+^\top v_{\max}$ that direction carries on $v_{\max}$, which the sign assumption shrinks with training. The top

direction of M is therefore $v_{\max}$ to the extent that the output has emptied of it. Empirically s fluctuates in both directions throughout training with a small positive drift (B.6.5). On its own this part therefore causes no alignment, only fluctuations of the axis along $v_{\max}$.

The axis gradient part. Keep instead the terms in $\hat{p}_-^\top r$ and $\hat{p}_-^\top \Sigma r$, dot with $v_{\max}$, and approximate the two projections by their components along $v_{\max}$,

$$\hat{p}_-^\top r \approx \left(\hat{p}_-^\top v_{\max}\right)\left(v_{\max}^\top r\right), \qquad \hat{p}_+^\top \Sigma \hat{p}_- \approx \lambda_{\max}\left(\hat{p}_+^\top v_{\max}\right)\left(\hat{p}_-^\top v_{\max}\right),$$

which B.6.8 and B.6.10 measure. With λ_+ and λ_- as §5.3 defines them, the overlap of the axis with $v_{\max}$ obeys

$$\begin{aligned}\frac{d}{dt}\left(\hat{p}_-^\top v_{\max}\right) \approx \frac{\gamma}{\sigma^2}\cos\tfrac{\theta}{2}\,\left(v_{\max}^\top r\right)\left(\hat{p}_+^\top v_{\max}\right)\left(\hat{p}_-^\top v_{\max}\right) \\ \times\left[\,2\lambda_{\max} - \lambda_{\max}\left(\hat{p}_-^\top v_{\max}\right)^2 - \lambda_+\cos^2\tfrac{\theta}{2} - \lambda_-\sin^2\tfrac{\theta}{2}\,\right].\end{aligned} \tag{15}$$

Neither Rayleigh quotient can exceed $\lambda_{\max}$, so the bracket is bounded below by

$$\lambda_{\max}\left(1 - \left(\hat{p}_-^\top v_{\max}\right)^2\right),$$

so the bracket is positive throughout the climb. Under the sign assumption everything multiplying the overlap is positive, so the alignment grows in whichever direction it already points, at a rate proportional to the overlap itself.

The full mechanism combines the two parts. When a fluctuation of $\dot{\gamma}$, that is of the $\hat{p}_+^\top r$ term, throws the overlap up, the push of (15) grows and the alignment is kept. The push is also proportional to $\hat{p}_+^\top v_{\max}$, so it lasts as long as the output direction still holds weight on $v_{\max}$ to give, and it stops when the axis is $v_{\max}$: the exact rate vanishes there because $\hat{p}_-$ is a unit vector, while (15), being an approximation along $v_{\max}$, does not, and the remainder terms of B.6.8 and B.6.10 carry the difference.

B.5 The angle

The length of the separation vector sets the angle, $\|p_-\| = \sin\frac{\theta}{2}$, so the rate of the angle follows from $\hat{p}_-^\top \dot{p}_-$.

At equal branch σ (B.6.1, and B.6.7 for what the mismatch does), and setting aside for the moment the one term that carries $\hat{p}_+^\top \Sigma \hat{p}_-$, which is small once the axis is close to $v_{\max}$ (B.6.11), the angle obeys:

$$\begin{aligned}\dot{\theta} &= \frac{2\gamma}{\sigma^2}\tan\tfrac{\theta}{2}\left(-\frac{\dot{\gamma}}{2}\left(\lambda_+\sin^2\tfrac{\theta}{2} + \lambda_-\cos^2\tfrac{\theta}{2}\right) + \cos\tfrac{\theta}{2}\,K\right), \\ K &:= \hat{p}_+^\top \Sigma\left(I - \hat{p}_+\hat{p}_+^\top\right) r.\end{aligned} \tag{16}$$

The bracket carries B of §5.3, where λ_- is the top eigenvalue (B.6.2).

The two terms compete. The first closes the angle while γ grows. The K term is the pull of the loss on the input variance the output direction sits at. Since $\lambda_+ = \hat{p}_+^\top \Sigma \hat{p}_+$, its gradient on the sphere is

$$\nabla_{\hat{p}_+}\lambda_+ = 2\left(I - \hat{p}_+\hat{p}_+^\top\right)\Sigma\hat{p}_+, \qquad K = \tfrac{1}{2}\,\nabla_{\hat{p}_+}\lambda_+ \cdot r,$$

and under gradient descent on the loss, λ_+ changes as

$$\dot{\lambda}_+ \;\propto\; -\nabla_{\hat{p}_+}\lambda_+ \cdot r \;=\; -2K.$$

So $K > 0$ means the loss is pushing the output direction toward lower input variance, and that push opens the angle. When $\lambda_{\max}$ stands well above the rest of the spectrum, the gap assumption of B.4, the variance λ_+ along the output direction is dominated by the share of $v_{\max}$ in that direction, so $K > 0$ is the statement that the loss pushes $v_{\max}$ out of the output direction, which is the sign assumption of B.4.

Three cases. With $\sin^2\frac{\theta}{2} = 1 - \cos^2\frac{\theta}{2}$ the bracket of (16) is a quadratic in $\cos\frac{\theta}{2}$, with

$$\text{leading coefficient} = -\frac{\dot\gamma}{2}\left(\lambda_- - \lambda_+\right), \qquad \text{discriminant} = K^2 - \dot\gamma^2\lambda_+\left(\lambda_- - \lambda_+\right).$$

The scale shrinking. When $\dot\gamma < 0$ both terms of (16) open the angle, and it keeps opening with no root to rest at.

The scale growing, the pull too weak. When $\dot\gamma > 0$ and $K/\dot\gamma$ is below $\sqrt{\lambda_+(\lambda_- - \lambda_+)}$ the quadratic has no real root. So the bracket is negative at every angle and the angle closes to zero. In this regime the channel closes.

The scale growing, the pull strong enough. When $\dot\gamma > 0$ and $K/\dot\gamma$ is at least $\sqrt{\lambda_+(\lambda_- - \lambda_+)}$ there are two roots, both at positive $\cos\frac{\theta}{2}$, since

$$\cos\tfrac{\theta^*_{\text{closed}}}{2}\cos\tfrac{\theta^*_{\text{open}}}{2} = \frac{\lambda_+}{\lambda_- - \lambda_+}, \qquad \cos\tfrac{\theta^*_{\text{closed}}}{2} + \cos\tfrac{\theta^*_{\text{open}}}{2} = \frac{2K}{\dot\gamma\left(\lambda_- - \lambda_+\right)}.$$

Writing θ^*_{closed} and θ^*_{open} for the two roots, (16) reads

$$\begin{aligned}\dot\theta = \frac{2\gamma}{\sigma^2}\tan\tfrac{\theta}{2}\,\frac{\dot\gamma}{2}\left(\lambda_- - \lambda_+\right)\\ \times\left(\cos\tfrac{\theta}{2} - \cos\tfrac{\theta^*_{\text{closed}}}{2}\right)\left(\cos\tfrac{\theta^*_{\text{open}}}{2} - \cos\tfrac{\theta}{2}\right).\end{aligned} \tag{17}$$

The angle then has two basins. Below θ^*_{closed} it falls to zero, the closed pair of §5.2. Above it, it moves toward θ^*_{open} from either side.

The boundary. Linearizing at the open root with K and $\dot\gamma$ held fixed (B.6.4), the root is stable exactly when

$$\cos^2\tfrac{\theta}{2} < \frac{\lambda_+}{\lambda_- - \lambda_+}, \tag{18}$$

and we call the equality case the boundary. The open root always satisfies (18) and the closed root never does: the condition selects which of the two equilibria is the stable one rather than constraining where the system can sit. The boundary is therefore

$$\cos\tfrac{\theta}{2} = \sqrt{\frac{\lambda_+}{\lambda_- - \lambda_+}} = \sqrt{\cos\tfrac{\theta^*_{\text{closed}}}{2}\,\cos\tfrac{\theta^*_{\text{open}}}{2}}, \qquad \cos\tfrac{\theta^*_{\text{open}}}{2} < \cos\tfrac{\theta}{2} < \cos\tfrac{\theta^*_{\text{closed}}}{2}.$$

Empirically we see that the system violates the bound and sits on the closed side. Several factors account for it: first, the neglected terms exert a force that closes the angle, moving it away from the root (B.6.11). Second, the system is not at equilibrium and the angle keeps fluctuating. What we do see is that the value the angle fluctuates around moves with the input spectrum when testing different datasets and spectra, and that the layer is bimodal: either an angle open around a value or a closed angle, not a continuum.

How far the axis is cooled now follows. Since

$$B = \lambda_+ + \left(\lambda_- - \lambda_+\right)\cos^2\tfrac{\theta}{2},$$

at the boundary (18) the two parts of B are equal, $B = 2\lambda_+$, and the coefficient on the axis is $2\lambda_+^2/(\lambda_- - \lambda_+)$, a reduction against $\lambda_{\max}$ by about $2(\lambda_+/\lambda_{\max})^2$. In the measurements taken at every optimizer step the medians satisfy (B.6.6):

$$\frac{\lambda_+}{\lambda_- - \lambda_+} \lesssim \cos^2\tfrac{\theta}{2} \lesssim \frac{2\lambda_+}{\lambda_- - \lambda_+}$$

An open pair presents the same function as a single branch whose scale, which we write γ_{single}, is $2\gamma\cos\frac{\theta}{2}$, with γ the pair's shared scale, and at the balance the tangential coefficient is proportional to the scale, so the pair's is $1/\cos\frac{\theta}{2}$ times the single's. Using (8), and assuming the decay–norm balance of Appendix C holds, we get the cooling factor.

$$\frac{\text{the pair's coefficient along } \hat p_-}{\text{the single's along the same direction}} = \frac{B\cos\frac{\theta}{2}}{\lambda_-} \tag{19}$$

B.6 MEASURED SUPPORT FOR THE ASSUMPTIONS

Each step of B.4 and B.5, and each restriction B.2 and B.3 make, leans on a stated regime. This section records the measurement behind each one. Unless an entry says otherwise, every number is the first layer of the default cell, read from measurements taken at every optimizer step in three seeds and formed per channel over its open steps and then across channels, and "the climb" is a channel's final ascent of its alignment with $v_{\max}$ from 0.5 to 0.9. Only a channel that starts below 0.5 has one, and about half start above it because of the Kaiming tilt (Table 3), so the climbing channels are 17, 11 and 12 in the three seeds, 40 in all. Percentages "of channels" in B.6.8 and B.6.10 are of these 40.

B.6.1 — $\sigma_1 = \sigma_2$ (§5.3, B.2). The ratio of the larger branch σ to the smaller starts at a median of 1.3 to 1.5 at initialization and settles at 1.07 to 1.09, quartiles 1.04 and 1.17, over epochs 5 to 100. B.6.7 measures what the mismatch does to the axis and the angle. It shows that the system reaches the same state even in channels where there is mismatch, which strengthens the approximation that drops this term.

B.6.2 — λ_- is the top eigenvalue (B.5). At epoch 100 the per channel median of $\lambda_-/\lambda_{\max}$ over open channels is 0.994 to 0.998 across seeds, with 25th percentile 0.978 to 0.994.

B.6.3 — the sign assumption, $(\hat{p}_+^\top v_{\max})(v_{\max}^\top r) > 0$, and K with it (B.4, B.5). The product is not positive at every step but has a positive net, and the same holds of K, which B.5 defines and which carries the product: through the climb K is positive on 54 to 59 percent of steps, with a positive net in all channels.

B.6.4 — K over the linearization (B.5). Once the angle has settled, from epoch 2 on, the per channel mean of K changes by 3 to 26 percent between consecutive epochs and stays within a factor of two in 78 to 100 percent of channels.

B.6.5 — the $\dot{\gamma}$ term and the Oja clock (B.4). The terms of the axis equation that carry $\hat{p}_+^\top r$ are proportional to $\dot{\gamma}$, whose per step sign is almost a coin flip with a small positive drift: $\dot{\gamma} > 0$ on 51 to 53 percent of the steps of the first five epochs. $\gamma\dot{\gamma}/2\sigma^2$ fluctuates hard and its slow accumulation shows only over epochs. Through the drift these terms push the overlap a little away from alignment. The push is small, a quarter to two fifths of the push of the $\hat{p}_-^\top r$ terms.

B.6.6 — B against its boundary value (§5.3, B.5). At the boundary (18) the two parts of B, λ_+ and $(\lambda_- - \lambda_+)\cos^2\frac{\theta}{2}$, are equal and $B = 2\lambda_+$, so the distance of B from the boundary is the per channel ratio $B/(2\lambda_+)$. At epoch 100 its median over open channels is 1.4 to 1.7 across the three seeds, with pooled quartiles of 1.3 and 3.2, and 26 or 27 of each seed's 27 to 29 open channels sit above 1.

B.6.7 — the branch-σ mismatch (B.4, B.5). The ratio of the larger branch σ to the smaller is 1.07 to 1.09 from epoch 5 on, quartiles 1.04 and 1.17 (B.6.1). Setting $\sigma_1 = \sigma_2$ drops one term from the axis equation, the one proportional to $(\sigma_2^2 - \sigma_1^2)/(\sigma_1^2 + \sigma_2^2)$, which vanishes at the parked state and moves no fixed point. Over the 40 climbing channels it is 0.25 of the retained term in the median channel, quartiles 0.09 and 0.50, and pushes the same way in 37 of the 40.

B.6.8 — the remainder of the $\hat{p}_-^\top r$ term (B.4). Over each channel's climb the $v_{\max}$ component of r carries the alignment, with a positive net in every climbing channel, while the remainder that the $v_{\max}$ approximation behind (15) drops pushes the other way in 82 percent of channels at about a quarter of the drive along $v_{\max}$, and the $v_{\max}$ share is the larger on 70 to 80 percent of climb steps.

B.6.9 — the flow limit (B.5). The first order change of the applied step accounts for each step's realized change of the angle to 2 percent. Per step the residual is therefore negligible. But it has the same sign and exerts a net closing force, so it does not average out. Measured against the K term it is a closing force of about 20 percent of the opening force over the period in which the angle opens.

B.6.10 — the cross term (B.4). Over each channel's climb the cross term $\hat{p}_+^\top \Sigma \hat{p}_-$ is 0.92 to 0.94 of the $v_{\max}$ approximation used in (15), with the same sign on every step. The error is 4 percent of the first term of the bracket, the exact bracket is 1.06 times the approximated bracket of (15), and it has the sign of $\hat{p}_+^\top v_{\max}$ on every climb step of every climbing channel.

B.6.11 — what closes the angle (B.5). The open root of (16) is where the two terms the equation keeps come into balance. The realized angle rests closed of it, and this entry collects every force

that pushes it there. Each is given as a share of the K term, the one that opens the angle, together with the channels it acts in.

The $\dot{\gamma}$ term of (16) exerts a closing force. In the applied step, momentum included, it closes the angle at about three quarters of the opening of the K term in the median channel over epochs 2 to 5, a per channel ratio of 0.66 to 0.81, and the K term still wins in two thirds to nine tenths of the channels.

Written out at equal σ, the first order angle rate carries one more term inside the bracket of (16), $\cos\frac{\theta}{2}\cos\theta\,(\hat{p}_+^\top \Sigma \hat{p}_-)(\hat{p}_-^\top r)$, which (16) sets aside because the cross quotient vanishes once the axis is $v_{\max}$. In these measurements it does not vanish. Over the channels that stay open it is -0.1 to -0.2, closing in 60 to 97 percent of those channels.

The second order part of the discrete step is the last of the three. Per step it is negligible, the 2 percent of the first order change that B.6.9 reports, but it is one-signed and closing, so it does not average out. Measured against the K term it is a closing force of about 20 percent of the opening force over the epochs in which the angle opens.

C THE DECAY–NORM BALANCE

This appendix carries the claim §5 makes when referring to the decay–norm balance. The kernel of a normalized branch is scale-free: the loss depends on w only through its direction, so by (9) the kernel gradient is $(\gamma/\sigma)\,\Sigma^{1/2}Pr$, and it is orthogonal to w because $w^\top\Sigma^{1/2}Pr = \sigma\, p^\top Pr = 0$. In this appendix λ without a subscript is the weight decay coefficient, the recipe's 5e-4 on the kernels, and nothing else. The eigenvalues of Σ, which carry a subscript everywhere else in the paper ($\lambda_{\max}$, λ_+, λ_-), do not enter this derivation. One step of stochastic gradient descent with decay λ is $w' = (1-\eta\lambda)\,w - \eta\,(\gamma/\sigma)\,\Sigma^{1/2}Pr$, and that orthogonality kills the cross term in the norm,

$$\|w'\|^2 = (1-\eta\lambda)^2\|w\|^2 + \eta^2\frac{\gamma^2}{\sigma^2}\left\|\Sigma^{1/2}Pr\right\|^2. \tag{20}$$

The norm therefore has a stationary point where the decay balances the gradient's effect on the norm, which is the standard balance for a scale-invariant weight (Kosson et al., 2024). At stationarity, with $\eta\lambda$ small,

$$\|w\|^2 = \frac{\eta\,\gamma^2}{2\lambda\,\sigma^2}\,\mathbb{E}\left\|\Sigma^{1/2}Pr\right\|^2. \tag{21}$$

At a fixed direction $\sigma^2 = w^\top\Sigma w$ is proportional to $\|w\|^2$, so (21) becomes

$$\sigma^4 \;\propto\; \frac{\eta\,\gamma^2}{2\lambda}\,\mathbb{E}\left\|\Sigma^{1/2}Pr\right\|^2, \tag{22}$$

the constant of proportionality depending on the direction alone. Two things follow. At the balance, $\sigma^2 \propto \gamma$: the branch's norm follows its scale. The scale itself keeps moving through training, and at the last layer it grows to the end (Table 5), so the balance is read quasi-statically. The norm tracks the scale as it drifts. And the tangential coefficient of (5) is then proportional to γ.

Now we compare a single and a pair at $\theta = 0$ representing the same function and at the same residual. Write γ_{single} and σ_{single} for the scale and the branch standard deviation of the single, and γ_{pair} and σ_{pair} for the shared scale and the branch standard deviation of the pair. To represent the same function a closed pair must have its scale at half the single's, $\gamma_{\text{pair}} = \gamma_{\text{single}}/2$. Everything else is the same. So the $\sigma^2 \propto \gamma$ balance fixes the ratio, $\sigma^2_{\text{single}}/\sigma^2_{\text{pair}} = \gamma_{\text{single}}/\gamma_{\text{pair}} = 2$. This means that the tangential coefficients for the single and the pair are the same,

$$\frac{\gamma^2_{\text{single}}}{\sigma^2_{\text{single}}} = \frac{4\gamma^2_{\text{pair}}}{2\sigma^2_{\text{pair}}} = 2\frac{\gamma^2_{\text{pair}}}{\sigma^2_{\text{pair}}},$$

so a closed pair has no boost in the tangential direction. The scale parts of (5) and (6) carry no γ^2/σ^2 at all and keep their $\times 4$, and the bias keeps the $\times 2$ of §5.

This derivation works exactly for the closed pair. For the open pair we assume the same proportionality rule holds. The other experiments in this paper (the cooling factor of §7.1) are consistent with this assumption.

D THE SYNTHETIC MODEL

Section 7.3 summarizes this model in a paragraph. This appendix is the whole of it: the setup, the checks that it behaves as the network does, and the three limits it can reach that no dataset can. Its value is that the target is known, so the input spectrum and the thing the task wants can be moved separately.

D.1 SETUP

Input. A fixed sample of 4,096 centered points in 27 dimensions, the dimension of the network's 3×3 color patch, drawn so that the empirical covariance is exactly a chosen diagonal Σ.

Model. One normalized linear layer of 64 channels, a ReLU, and a linear readout of output dimension 10, which is the network's conv–BatchNorm–ReLU block with the convolution replaced by a dense layer. Per channel the single computes $\gamma\, w^\top x/\sigma+\beta$ and the pair $\gamma\,(w_1^\top x/\sigma_1+w_2^\top x/\sigma_2)+\beta$, with one shared γ and one β, each σ being the square root of that branch's Rayleigh quotient against the fixed sample covariance. The ReLU is applied to that quantity and the readout maps the 64 rectified channels to the 10 outputs. The loss is the mean squared error against the teacher's outputs, and training is full batch gradient descent, so there is no batch noise anywhere in this model.

Target. A fixed two layer ReLU teacher with 128 hidden units and 10 outputs, drawn once, whose outputs are standardized to zero mean and unit variance per output. Its hidden directions are drawn isotropic **in the whitened frame**. An isotropic draw in the raw frame, which is what Kaiming initialization gives, would give the teacher a large overlap with $v_{\max}$, against the assumption that the loss has to push that direction down, so the whitened draw is the default. D.3 studies what happens when the teacher is given more of $v_{\max}$ on purpose. "The target's share on $v_{\max}$" scales the teacher's $v_{\max}$ coordinate so that a chosen fraction of its whitened energy lies there. The untouched draw sits at 1/27, about 0.037.

Arms. The single, the pair, "cooled", the single with its kernel gradient along $v_{\max}$ multiplied by a constant before every step, which is the intervention of §7.1, "boosted", the single with the scale learning rate at four times and the shift learning rate at twice, which is the closed pair multiplier of §6.1. The cooled arm's constant is the same quantity the network's cooling uses, the coefficient of the preconditioner along $v_{\max}$ at the end of training in the pair over the single, the median over open channels, measured in the same cell and seed, so each seed applies its own factor.

Recipe. Full batch descent at learning rate 0.1 with momentum 0.9 for 10,000 steps, kernel only weight decay 5e-4, ten seeds that vary only the student's initialization.

Reporting. Accuracy does not exist in this model, so the primary quantity is the paired per seed base ten logarithm of the arm's training loss over the single's, both averaged over the final 2,000 steps, mean $\pm$ standard error over seeds. A negative number is an improvement, and the antilog is given beside it where it helps. Angles are medians over open channels on the 10° threshold of §3.1, "parked" keeps the meaning §3.1 gives it, an open channel whose separation axis overlaps $v_{\max}$ above 0.95, and every geometric quantity is averaged per channel over the final 2,000 steps before the median is taken, because the loss and the angle oscillate under full batch descent.

D.2 THE MODEL BEHAVES AS THE NETWORK DOES

Two checks, each the analog of a measurement in the network.

The direction controls. On the network's CIFAR-100 spectrum, with a ratio of the two largest eigenvalues of 9.6, every arm other than the pair is a single branch identical to the others except for which direction of its kernel gradient is scaled and by what factor (Table 34). As in the network (Table 26), only cooling along $v_{\max}$ reproduces the pair, cooling any other direction gives no benefit, and heating along $v_{\max}$ hurts.

The five datasets.

The models do not match completely, but μ_{eff} roughly orders the dynamics: over the five datasets (Table 35) its Spearman rank correlation, the statistic of Table 28, with the angle is +0.90 in the network and +0.70 in the synthetic model, and with the alignment +0.30 and +0.70.

Table 34: Scaling one direction of the layer's kernel gradient in the synthetic model, ten seeds. "paired log loss ratio" is the arm's final band mean loss over the single's, base ten. Negative is better and "loss ratio" is its antilog. The second column names the direction and gives the input variance along it, the Rayleigh quotient of Σ there. The factor is 0.091, the mean over seeds. Heating applies its reciprocal. The random direction is a fixed unit vector orthogonal to $v_{\max}$, drawn once per seed. Every row here is paired against the same single runs as the CIFAR-100 rows of Tables 36 and 38, so the pair and the cooled arm carry the same values there.

arm	direction scaled, and the input variance along it	factor	paired log loss ratio	loss ratio
the two branch pair	—	—	-0.255 ± 0.003	0.556
cooled along $v_{\max}$	$v_{\max}$, 21.8	0.091	-0.247 ± 0.018	0.566
cooled along the eigenvector of λ_{next}	the second eigenvector, 2.28	0.091	-0.004 ± 0.005	0.991
cooled along a random direction	random, orthogonal to $v_{\max}$, 0.26	0.091	$+0.015 \pm 0.007$	1.036
heated along $v_{\max}$	$v_{\max}$, 21.8	11.0	$+0.339 \pm 0.064$	2.183

Table 35: The synthetic model against the network on the same five spectra, sorted by μ_{eff}. The unqualified columns are the first layer of the network in Table 27, in the default cell. The "synthetic model" columns are this model at ten seeds. "angle" is the whitened branch angle, in both columns the mean over seeds of each run's median over its open channels. "alignment" is the median overlap of the separation axis with $v_{\max}$ over open channels.

dataset	μ_{eff}	angle	angle, synthetic model	alignment	alignment, synthetic model
GTSRB	0.75	152°	163°	0.998	0.993
SVHN	0.71	143°	172°	0.998	0.997
EuroSAT	0.68	143°	168°	0.997	0.994
CIFAR-100	0.63	143°	161°	0.999	0.991
Flowers-102	0.55	131°	149°	0.995	0.987

The same geometry drawn over the nine settings of Figures 17 to 20 shows us that the synthetic model's geometry is also ordered by the predictor.

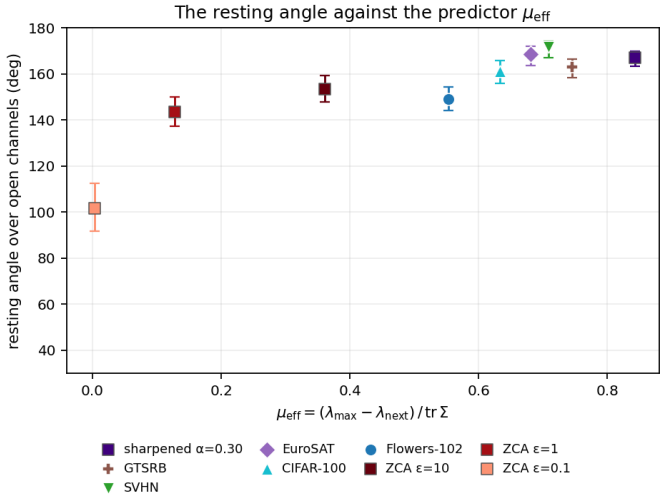

Figure 21: Figure 19 for the synthetic model.

D.3 THE THREE LIMITS

μ_{eff} bundles two things, the gap between the two largest eigenvalues and how far the rest of the spectrum sits below them, and it does not see the output direction at all. Three limits separate them.

First: the two largest eigenvalues made exactly equal. There is then no single top eigenvector to select.

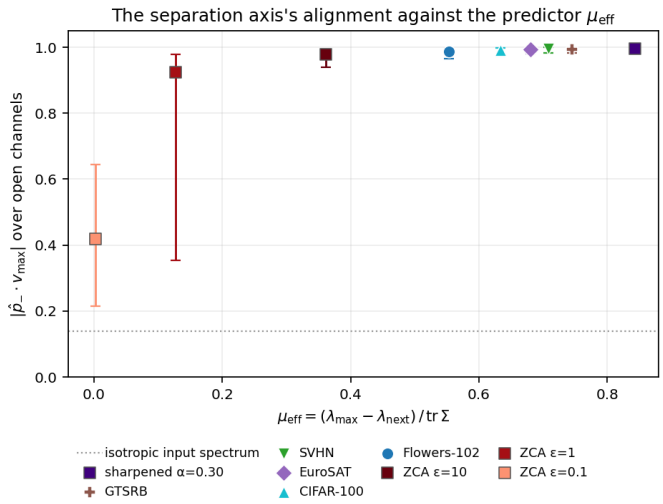

Figure 22: Figure 17 for the synthetic model. Isotropic input spectrum stands for the initial alignment of weights drawn isotropically in the whitened frame.

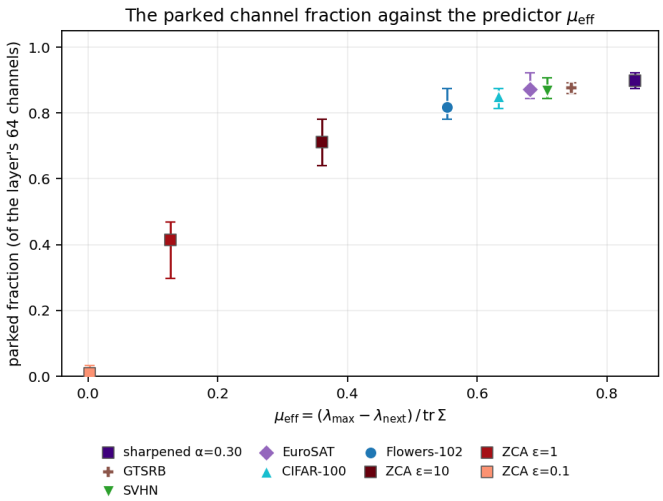

Figure 23: The parked channel fraction against the predictor for the synthetic model, in place of Figure 18's open fraction: the model keeps every channel open in all nine settings (at least 98 percent in every seed), so the open fraction is flat at one and the parked fraction carries the information.

With no gap the pair still ends up with significant overlap with the top plane but it is worth no gain. Cooling the whole plane does provide benefit. The reason is that the large effective learning rate along the top two eigenvectors caps the step size, and a separation axis that cancels only one direction of the top plane leaves the other with its full variance, so the step is no better conditioned and the pair gains nothing.

Second: the top two held, the rest of the spectrum moves.

Moving μ_{eff} while keeping the ratio constant diminishes the gain and the angle but not the number of parked channels. The single's own loss also falls along this sweep, from 0.209 to 0.056, so part of the shrinking gain may be the task getting easier rather than the spectrum alone, the same caution §7.2 raises for the sharpened dose. The ratio $\lambda_{\max}/\lambda_{\text{next}}$, which ranked the nine settings almost as well as μ_{eff} in Table 28, does not move here at all, so μ_{eff} is the better of the two. Our choice of μ_{eff} as the predictor in Appendix A.12 rests on this comparison.

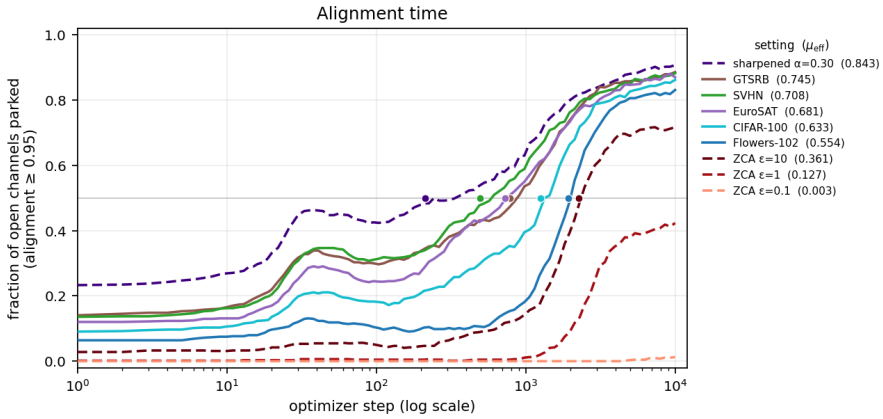

Figure 24: Figure 20 for the synthetic model.

Table 36: The top plane is the plane of the two eigenvectors whose eigenvalues are made equal. In the first row the top eigenvalue stands alone, so the columns that refer to the top eigenvalues or the top plane are read on $v_{\max}$ alone. The CIFAR tail with the top eigenvalue standing alone against the same tail with the two largest made equal, ten seeds a row. μ_{eff} is 0.633 in the first row and exactly 0 in the others. The pair, cooled and boosted columns are paired log loss ratios as in Table 34, the cooled single's factor applied to both top eigenvectors in the rows where the top two are equal.

spectrum	share of the trace in the top eigenvalues	pair	cooled over the top plane	boosted	angle	parked in the top plane, of 64	overlap with the top plane
CIFAR-100, top alone at 21.79	0.71	-0.255 ± 0.003	-0.247 ± 0.018	$+0.008 \pm 0.005$	161°	54.4	0.991
top two equal at 21.79	0.87	-0.027 ± 0.007	-0.219 ± 0.030	$+0.019 \pm 0.003$	167°	11.1	0.701
top two equal at 10	0.75	-0.002 ± 0.007	-0.235 ± 0.003	$+0.007 \pm 0.002$	165°	10.6	0.685
top two equal at 2.276	0.40	-0.001 ± 0.002	-0.006 ± 0.002	$+0.005 \pm 0.002$	145°	1.0	0.419

Table 37: $\lambda_{\max}$ held at 21.79 and λ_{next} at 2.276 while the 25 remaining eigenvalues of the CIFAR spectrum are all set to one level, given in the first column. Ten seeds a row. "trace" is the resulting tr Σ and μ_{eff} is $(\lambda_{\max} - \lambda_{\text{next}})$/tr Σ. "single's loss" is the single's own final band mean squared error, an absolute number and not a ratio.

level the rest is lifted to	trace	μ_{eff}	share of the trace below λ_{next}	single's loss	pair	angle	parked of 64
0.01	24.3	0.803	0.010	0.209	-0.412 ± 0.005	161°	59.3
0.1	26.6	0.735	0.094	0.070	-0.086 ± 0.004	141°	59.3
0.3	31.6	0.618	0.238	0.061	-0.049 ± 0.006	129°	55.5
0.8	44.1	0.443	0.454	0.057	-0.021 ± 0.005	115°	56.4
1.5	61.6	0.317	0.609	0.056	-0.013 ± 0.006	106°	57.5
2.27	80.8	0.241	0.702	0.056	-0.015 ± 0.005	101°	57.7

Third: the share of $v_{\max}$ in the target. Appendix B.4 assumes the target does not need $v_{\max}$: the cooling pays because that direction carries most of the input variance and little of the target. Here that share can be set directly. The gain falls as the target takes $v_{\max}$, but it does not vanish. With 70 percent of the target's energy on $v_{\max}$ the pair keeps 42 percent of its gain and 27 channels parked. What falls is the number of parked channels, from 64 to 27, while the gain per parked channel holds between -0.0040 and -0.0058 along the whole curve. So the system specializes and the mechanism runs in fewer channels. The result shows that the sign assumption of Appendix B.4 is needed per channel and not for the whole layer and that a layer can have a significant output along the $v_{\max}$ direction but still use the cooling mechanism.

Table 38: The CIFAR spectrum held fixed while the teacher's $v_{\max}$ row is rescaled so that the stated fraction of its whitened energy lies on $v_{\max}$. The same teacher draw throughout.

the target's share on $v_{\max}$	pair gain	cooled gain	boosted gain	parked (of 64)	angle	gain per parked channel
0	-0.257 ± 0.003	-0.286 ± 0.003	$+0.015 \pm 0.004$	63.7	164°	-0.0040
0.02	-0.258 ± 0.007	-0.273 ± 0.007	$+0.012 \pm 0.009$	58.8	161°	-0.0044
0.037, untouched	-0.255 ± 0.003	-0.247 ± 0.018	$+0.008 \pm 0.005$	54.4	161°	-0.0047
0.07	-0.246 ± 0.002	-0.252 ± 0.011	$+0.006 \pm 0.002$	50.8	161°	-0.0048
0.12	-0.237 ± 0.004	-0.204 ± 0.019	$+0.003 \pm 0.008$	46.4	160°	-0.0051
0.2	-0.223 ± 0.007	-0.059 ± 0.025	-0.011 ± 0.007	40.3	159°	-0.0055
0.3	-0.215 ± 0.007	-0.054 ± 0.027	-0.010 ± 0.005	37.0	158°	-0.0058
0.5	-0.162 ± 0.004	-0.016 ± 0.026	$+0.004 \pm 0.004$	30.1	154°	-0.0054
0.7	-0.108 ± 0.005	$+0.010 \pm 0.019$	$+0.003 \pm 0.005$	26.5	142°	-0.0041

The cooled single tracks the pair only up to a share of about 0.1 and falls away past it, -0.059 against -0.223 at 0.2 and nothing at 0.7. The cooling arm scales the gradient along $v_{\max}$ in every channel by one factor, while the pair cools only the channels that park. As the target takes $v_{\max}$ the parked count falls from 64 to 27 and the other channels keep $v_{\max}$ in their output direction, which the cooled single denies them.