COME ON AGENT, LET'S STAY PLASTIC: WELL-CONDITIONED OPTIMISATION FOR MULTI-TASK AND META-RL

Anonymous authors
Paper under double-blind review

ABSTRACT

Meta-learning with gradient adaptation assumes that pretraining on many tasks leaves a network that learns new tasks faster than a random one. We show that this can fail in deep reinforcement learning: networks pretrained on many tasks in parallel with Adam or Muon adapt to held-out tasks more slowly than a freshly initialised network. We ask whether optimisers that keep the weights close to orthogonal, which preserve plasticity when tasks arrive in sequence, also preserve adaptability after parallel pretraining. We study AdamO and MuonO, which add a decoupled soft orthogonality regulariser to Adam and Muon, and Parseval regularisation, and relate few-step adaptation to the neural tangent kernel of the pretrained network. In pre-registered experiments in two gridworld families, the regularised optimisers preserve the adaptability of deep networks and, at moderate width, adapt better than a fresh network, and they match or beat plasticity baselines and entropy controls. In the settings we tested outside gridworlds, pretraining did not reduce adaptability. The choice of pretraining optimiser therefore matters for gradient adaptation where pretraining reduces adaptability.

1 INTRODUCTION

Meta-learning trains one network on a distribution of tasks so that it can learn new tasks from the same distribution quickly (Finn et al., 2017; Duan et al., 2016). In meta-learning with gradient adaptation, the network adapts to a new task by a few gradient steps on its weights, as in the inner loop of model-agnostic meta-learning (Finn et al., 2017, MAML) or in fine-tuning after multi-task pretraining. In reinforcement learning (RL) the pretraining often covers many tasks in parallel, and the pretrained network is judged by how fast it learns a held-out task, not by its return before any update. The pretrained weights must therefore fit the training tasks and remain a good starting point for later gradient steps.

The second requirement can fail. Deep RL networks lose plasticity, the ability to keep learning, as training goes on (Lyle et al., 2022; 2023; Nikishin et al., 2022; Sokar et al., 2023; Dohare et al., 2024), and a network warm-started from earlier training can generalise worse than a fresh one (Ash & Adams, 2020). We find that, in two gridworld families, deep networks pretrained in parallel on many tasks with Adam or Muon adapt to held-out tasks more slowly than a random initialisation. One family of methods keeps the weights well conditioned. Parseval regularisation (Chung et al., 2024) and AdamO, Adam with a decoupled soft orthogonality regulariser (Rosseau et al., 2026), preserve plasticity in continual learning, where tasks arrive one at a time and only the next task counts. We ask whether they also preserve gradient adaptability after parallel pretraining, and whether the pretrained network then transfers better than a fresh one.

We study AdamO, MuonO, which adds the same regulariser to Muon (Jordan et al., 2024), and Parseval regularisation as pretraining optimisers for meta-learning with gradient adaptation. We analyse linearised gradient adaptation (Section 4.2). For the linearised network, Proposition 3 gives the loss after a fixed number of adaptation steps from the new task's initial residual and empirical neural tangent kernel (Jacot et al., 2018; Zhou et al., 2021). Our layerwise and task-block bounds then specify how weight defects, realised features and coupling between training tasks affect those steps (Propositions 4, 6 and 8). The analysis permits different adaptation rates at equal initial outputs.

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.

We test these predictions with PPO agents pretrained in parallel on many goals of a nine-room gridworld and adapted to held-out goals with the same plain Adam inner loop for every arm. In a pre-registered study of deep networks, pretraining with Adam or Muon leaves a network that adapts far worse than a fresh one, while AdamO, MuonO and Parseval regularisation keep the ability to adapt and, at width 64, exceed the fresh network. They also beat shrink-and-perturb, entropy-matched Muon and approximately entropy-matched Adam, and match or beat L2-init. The result replicates in a MiniGrid DoorKey variant whose held-out tasks conflict with pretraining, and a regularised arm is the best or tied-best arm at every depth we tested. The regularised networks keep their feature rank, their Jacobian conditioning and nearly equal kernel scales across training tasks, while the loss of adaptability under the base optimisers goes with an overconfident policy on the new tasks.

Our contributions are the following.

- We show in pre-registered studies on fresh seeds that deep networks pretrained in parallel on many tasks with Adam or Muon lose the ability to adapt to new tasks by gradient steps, and that AdamO, MuonO and Parseval regularisation preserve it and, at width 64, give positive transfer over a fresh network, in two gridworld families and against plasticity baselines and entropy controls (Sections 6.1 and 6.2).
- We give linearised kernel results for finite-step adaptation, realised features and per-task training rates (Sections A.2, A.3 and 4.2).
- We measure what the regularised networks preserve: feature rank and Jacobian conditioning through depth, and equal per-task kernel scales during parallel pretraining, and report which kernel predictions fail and where policy entropy explains more (Sections 6.3 to 6.5).

2 Related work

Meta-learning with gradient adaptation. MAML learns an initialisation from which a few gradient steps solve a new task (Finn et al., 2017), and its success has been attributed largely to the reuse of features learned in the outer loop (Raghu et al., 2020). Other methods learn how the inner loop steps, with per-parameter learning rates (Li et al., 2017) or with warp layers that precondition the inner-loop gradient (Flennerhag et al., 2020), and MetaNTK replaces the inner loop by its kernel form in function space (Zhou et al., 2021). We use no meta-gradient: the network is pretrained on many tasks in parallel, and a plain Adam inner loop adapts it, so the only difference between our arms is the geometry of the pretrained weights. In-context meta-RL, in which a recurrent network adapts through its state with fixed weights (Duan et al., 2016; Nikulin et al., 2024), is a different mechanism that we do not address.

Plasticity loss. Deep RL networks lose plasticity through capacity loss, a primacy bias, dormant neurons, implicit under-parameterisation and continual training (Lyle et al., 2022; Nikishin et al., 2022; Sokar et al., 2023; Kumar et al., 2021; Dohare et al., 2024), and warm-started networks can generalise worse than fresh ones (Ash & Adams, 2020). Proposed methods reset the network, shrink and perturb its weights (Ash & Adams, 2020), or pull them towards their initial values (Kumar et al., 2025, L2-init), and such regenerative methods consistently mitigate plasticity loss in on-policy RL (Juliani & Ash, 2024). These works study one task or a task sequence, while we study adaptation after parallel pretraining and use L2-init, shrink-and-perturb and a reset as baselines.

Orthogonality, isometry and Muon. Prior analyses of deep network training use dynamical isometry of the input-output Jacobian (Saxe et al., 2014; Pennington et al., 2017; Xiao et al., 2018); Section 3 defines the term. In continual RL, Parseval regularisation keeps networks plastic (Chung et al., 2024), spectral regularisation keeps the largest singular value of each layer near one (Lewandowski et al., 2025), and AdamO, motivated by dynamical isometry, preserves plasticity when tasks arrive one at a time (Rosseau et al., 2026). Muon applies an orthogonalised momentum as the update of each hidden matrix (Jordan et al., 2024), which does not keep the accumulated weights orthogonal. Our increment is to test these optimisers as pretraining optimisers for gradient adaptation to new tasks, to add the regulariser to Muon, and to derive when the constraint helps.

Multi-task kernels and shared representations. Multi-task methods reweight, combine or project task gradients (Yu et al., 2020), although tuned unitary scalarisation often matches them

(Kurin et al., 2022), and NTKMTL balances the convergence speeds of tasks with the spectra of their per-task kernel blocks (Qin et al., 2025). We do not reweight tasks, and instead measure how the pretraining optimiser changes the per-task kernel blocks. Linear shared-representation theory gives guarantees for few-shot learning that do not depend on the basis of the learned span (Du et al., 2021; Tripuraneni et al., 2021), while we analyse finite-step adaptation through the empirical kernel.

3 BACKGROUND

3.1 PROBLEM SETTING

A task distribution $p(\tau)$ is a distribution over related Markov decision processes $\mathcal{M}_\tau = (\mathcal{S}, \mathcal{A}, P_\tau, R_\tau, \gamma)$ with shared state and action spaces, for example the goals of one maze. In *parallel pretraining* one network with parameters θ trains on a set of training tasks drawn from $p(\tau)$ at the same time, here by drawing a task for every episode. In *gradient adaptation* the pretrained parameters θ_0 are then trained on held-out tasks from the same distribution by a fixed procedure, the inner loop, with a fixed budget of environment transitions. This is the inner loop of MAML (Finn et al., 2017) without the outer meta-gradient: the pretraining optimiser, not a meta-objective, decides where the inner loop starts. We judge θ_0 by the mean success on the held-out tasks over the whole adaptation budget, the area under the adaptation curve (AUC), and we also report the success before any update (zero-shot). A fresh network adapted in the same way is the reference: a pretrained network that adapts better shows positive transfer, and one that adapts worse has lost plasticity.

3.2 PLASTICITY, ISOMETRY AND OPTIMISERS

Plasticity is the ability of a network to reduce the loss on new targets (Lyle et al., 2023). It is commonly measured by the fraction of dormant units (Sokar et al., 2023) or the effective rank of the features, the exponential of the entropy of the normalised singular values (Roy & Vetterli, 2007). Dynamical isometry keeps all singular values of the end-to-end Jacobian near one, so signals neither vanish nor explode with depth (Saxe et al., 2014; Pennington et al., 2017; Xiao et al., 2018). We call $\|W^\top W - \alpha I\|$ the *Gram defect* of W for the target scale $\alpha > 0$. Our agents use proximal policy optimisation (Schulman et al., 2017, PPO). For a momentum matrix with compact singular value decomposition $M = U\Sigma V^\top$, Muon uses an approximation to $UV^\top$ for its matrix step (Jordan et al., 2024). We compute it by Newton–Schulz iteration (Section B). AdamO (Rosseau et al., 2026) is Adam with a decoupled soft orthogonality regulariser.

4 ORTHOGONALITY-REGULARISED PRETRAINING FOR GRADIENT ADAPTATION

We use the soft orthogonality regulariser of AdamO (Rosseau et al., 2026) during parallel pretraining. The adaptation inner loop never uses it, so it acts only through the pretrained weights.

4.1 THE REGULARISER, ADAMO AND MUONO

For a hidden weight matrix $W \in \mathbb{R}^{n_{\text{out}} \times n_{\text{in}}}$ we penalise the smaller of its two Gram matrices,

$$R(W) = \tfrac{1}{2}\|WW^\top - I\|_F^2 \text{ if } n_{\text{out}} \le n_{\text{in}}, \qquad R(W) = \tfrac{1}{2}\|W^\top W - \tfrac{n_{\text{out}}}{n_{\text{in}}} I\|_F^2 \text{ otherwise.} \quad (1)$$

$R(W) = 0$ exactly when W has full rank $\min(n_{\text{out}}, n_{\text{in}})$ and all its singular values equal $\sqrt{\max(1, n_{\text{out}}/n_{\text{in}})}$, the scale of the reference Muon update (Section B). The regulariser is not added to the loss. After each optimiser step we apply the decoupled step

$$W \leftarrow W - \eta\, c\, \nabla_W R(W), \quad (2)$$

where η is the current learning rate and c the regulariser coefficient. We regularise every hidden matrix of the network and exclude output heads, biases and normalisation parameters, so the heads stay free to represent task-specific values and actions. Regularised matrices start from an orthogonal initialisation at the target scale, so $R = 0$ at the start. In our experiments we use the target $2I$ and the initialisation gain $\sqrt{2}$ of the released Parseval code (Chung et al., 2024) (Section B).

AdamO applies Adam to all parameters and then the decoupled step to the hidden matrices. MuonO applies Muon to the hidden matrices and an Adam-family update to all other parameters, and then the decoupled step (Section B). Muon orthogonalises the proposed matrix step. MuonO additionally penalises the Gram defect of the resulting weights, which the Muon step does not constrain. If the penalty were added to the loss, as in Parseval regularisation, Adam would rescale it per coordinate and Muon would orthogonalise it with the task gradient, so its contribution would no longer scale with the defect. The strength of the regulariser is set by the product ηc: near the target one decoupled step multiplies the defect $E = WW^\top - I$ by $1-4\eta c$ up to higher-order terms, so the same coefficient acts more weakly at a smaller learning rate, and we check engagement by the measured Gram defect.

4.2 Why conditioning should help gradient adaptation

The kernel view of the inner loop. For all parameters of a network, a few small gradient steps from θ_0, as in a MAML inner loop, are approximated by the linearised model, in which the residual on the new task evolves as $r_m = (I-\eta\mathcal{K})^m r_0$ for the empirical neural tangent kernel $\mathcal{K} = JJ^\top$ at θ_0 (Proposition 3; Jacot et al., 2018; Zhou et al., 2021). For residual component q, Proposition 3 gives the factor $(1-\eta\lambda_q)^m$ after m steps. Thus two networks with equal outputs before adaptation can incur different losses within the same step budget when the new task has residual mass in directions with different kernel eigenvalues (Corollary 1). Proposition 4 bounds the kernel using the realised feature Grams and backward Jacobians of each layer. For a linear network with Gram defects at most ε, its condition number is at most that of the input Gram times $((1+\varepsilon)/(1-\varepsilon))^{L-1}$, but the bound does not extend to ReLU networks: with orthogonal weights, a ReLU network can have a kernel condition number that grows linearly with depth (Proposition 5).

Several tasks in parallel. During parallel pretraining the kernel of all training inputs splits into blocks $\mathcal{K}_{ij} = J_i J_j^\top$ (Section A.3). The diagonal blocks set per-task rates: in the linearised model with decoupled tasks, a task whose largest block eigenvalue lies below the smallest block eigenvalue of another task is fitted later at every tolerance, an implicit curriculum set by the kernel (Proposition 7). The off-diagonal blocks carry first-order transfer and interference, which is the layerwise factorisation of interference summed over layers (Proposition 6). In a linear network whose weights have Gram defects at most ε, the mean eigenvalue of every diagonal block lies between $L(1-\varepsilon)^{L-1}$ and $L(1+\varepsilon)^{L-1}$ times the mean squared input norm of that task, so weights with small Gram defects keep the per-task scales as equal as the inputs allow (Proposition 8). For a held-out task u, the Rayleigh quotient q_u of its kernel block at its initial residual bounds the first linearised step from both sides and every later step from below (Proposition 9).

The theory motivates three empirical checks for meta-learning with gradient adaptation. First, the regulariser should not raise the fit to the pretraining tasks. Second, it should give a small difference before adaptation and a larger one after a few adaptation steps. Third, the per-task kernel scales should stay more nearly equal under the regulariser, and held-out goals should have a larger q_u. The theory has no policy and says nothing about exploration, so we test separately whether policy entropy explains a gain, and it concerns weights, so we measure realised features and kernels.

5 Experimental setup

Pretraining and adaptation. An agent in a 15×15 gridworld of nine rooms (Chung et al., 2024) observes its goal, and each network is pretrained on 20 goals in parallel, with a goal drawn for every episode (Section B). The agent is PPO (Schulman et al., 2017) with separate actor and critic networks of width 64. After pretraining we reset the optimiser state and adapt the network to six disjoint test goals for $262{,}144$ transitions, which are 128 PPO rollouts of $2{,}048$ transitions. Every arm adapts with the same plain Adam inner loop without the regulariser, selected once on validation goals, so only the pretrained weights differ. The primary endpoint is the full AUC, the mean success on the test goals over the whole adaptation, and we also report the zero-shot success. The lead experiment uses a plain tanh network with 16 hidden layers, whose Jacobian at initialisation already has singular values from 3×10^{-5} to 30, and 4.19M pretraining transitions. It was pre-registered with its primary contrast, with 64 fresh seeds and test goals not used in any earlier selection (Section D). We repeat it at depths 2, 4, 8 and 32 and after 16.8M transitions, and the shallow experiment uses the two tanh layers of Chung et al. (2024) with three pretraining lengths and 64 other seeds (Section C

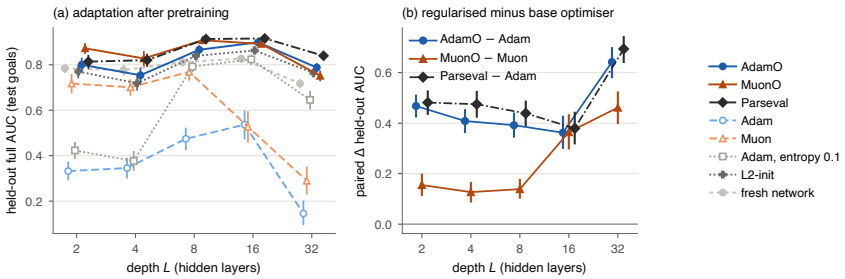

Figure 1: Held-out adaptation against the depth L of a plain tanh network of width 64, after 4.19M transitions of parallel pretraining on 20 goals, 64 paired seeds, test goals. (a) Full AUC of each arm. (b) Paired differences between each regularised arm and its base optimiser. 95% bootstrap intervals.

and Table 3). The second family is MiniGrid DoorKey from XLand-MiniGrid (Nikulin et al., 2024) with the same architecture, budgets and procedure, per-seed training tasks and its own adaptation learning rate (Sections F and 6.2).

Arms and baselines. We compare Adam, AdamO, Muon, MuonO and Parseval regularisation, which adds the penalty to the loss with coefficient 10^{-3}, a value in the published hyperparameter search and used in our runs (Chung et al., 2024). The regulariser coefficient is $c = 0.1$. Paired arms start from identical parameters, and every regularised matrix starts at the regulariser's target, so the arms differ only in their updates. We add three Adam-pretrained controls and a fresh-network reference: L2-init, which pulls the hidden weight matrices towards their initial values in our implementation (coefficient 10^{-3}) (Kumar et al., 2025); shrink-and-perturb, our once-before-adaptation, hidden-matrix variant of Ash & Adams (2020), $\theta \leftarrow 0.8\,\theta + 0.01\,\theta_{\text{fresh}}$ with our settings 0.8 and 0.01; Adam with entropy coefficient 0.1 instead of 0.01; and a fresh network with no pretraining. Entropy-matched arms adjust the entropy coefficient of Adam or Muon by feedback so that their policy entropy follows the pretraining trajectory of AdamO or MuonO with the same seed (Section E). We also test a residual LayerNorm network under Muon (Section D).

Statistics. The seed is the unit of inference, and paired arms share the initialisation and the tasks. We report paired differences with 95% percentile intervals from at least $10{,}000$ bootstrap resamples of whole seeds, not adjusted for multiple comparisons, and the number of seeds in which the first arm is better. Pretraining learning rates were selected per optimiser family at depth 2 and reused in the NineRooms depth sweep and first width-256 run: Adam, AdamO and the Adam-based baselines use 10^{-4}, and Muon, MuonO and Parseval use 2.5×10^{-4} (Section C). The pre-registered primary contrast at depth 16 compares Parseval at 2.5×10^{-4} with L2-init at 10^{-4}; we also report L2-init at Parseval's rate. The width-256 follow-up and DoorKey select their own rates (Sections E and F).

6 Results

6.1 Orthogonality preserves adaptability in deep networks

In a depth-16 network, pretraining with Adam or Muon removes much of the ability to adapt to held-out goals, and the orthogonality-regularised optimisers keep it and, at width 64, exceed a fresh network (Table 1). Every arm fits the 20 training goals (training success 0.95 to 1.00), so the arms differ in adaptation and not in training fit. After Adam or Muon pretraining the full AUC is 0.537 or 0.527, far below a fresh network (0.827). AdamO, MuonO and Parseval reach 0.899, 0.892 and 0.916, so the regulariser raises the full AUC by $+0.362$ $[0.303, 0.426]$ over Adam (63 of 64 seeds) and by $+0.365$ $[0.300, 0.431]$ over Muon (61 of 64). Parseval beats a fresh network by $+0.089$ $[0.074, 0.105]$ (61 of 64 seeds), AdamO by $+0.072$ and MuonO by $+0.065$, so at width 64 pretraining with the regulariser gives positive transfer to new goals.

Plasticity baselines and entropy. The regularised optimisers beat every plasticity baseline at the pre-registered learning rates. Shrink-and-perturb, applied once before adaptation, does not recover the adaptability of the Adam network (0.562). L2-init keeps much of it (0.863). The pre-registered

Table 1: Held-out adaptation of depth-16 networks after 4.19M transitions of parallel pretraining on 20 goals, pre-registered, 64 fresh paired seeds, six untouched test goals, the same fresh Adam inner loop for every arm. Left: full AUC (mean success over the 262,144 adaptation transitions), zero-shot success and policy entropy at the end of pretraining. Right: paired full-AUC differences with 95% bootstrap intervals and the number of seeds in which the first arm is better. Ent. Adam: pretraining entropy coefficient 0.1; Pre-LN: residual network with layer normalisation. * pre-registered primary contrast; † exploratory.

Arm	Full	Zero-shot	Entropy	Contrast	Full AUC	Seeds
Adam	0.537	0.040	0.354	Parseval − L2-init*	$+0.053$ $[0.037, 0.070]$	49
Muon	0.527	0.062	0.287	AdamO − Adam	$+0.362$ $[0.303, 0.426]$	63
AdamO	0.899	0.094	0.488	MuonO − Muon	$+0.365$ $[0.300, 0.431]$	61
MuonO	0.892	0.105	0.527	Parseval − Adam	$+0.379$ $[0.320, 0.441]$	63
Parseval	0.916	0.113	0.650	AdamO − Ent. Adam	$+0.075$ $[0.053, 0.100]$	55
L2-init	0.863	0.084	0.499	MuonO − Ent. Adam	$+0.068$ $[0.048, 0.090]$	56
Shrink & perturb	0.562	0.044	0.354	Parseval − Ent. Adam	$+0.092$ $[0.074, 0.113]$	61
Ent. Adam (0.1)	0.824	0.067	0.438	Parseval − Fresh	$+0.089$ $[0.074, 0.105]$	61
Fresh	0.827	0.036		AdamO − L2-init†	$+0.036$ $[0.019, 0.052]$	43
Pre-LN Muon	0.724	0.042	0.294	MuonO − L2-init†	$+0.029$ $[0.012, 0.046]$	42
				Pre-LN Muon − Fresh†	-0.102 $[-0.144, -0.063]$	17

primary contrast, Parseval against L2-init, is $+0.053$ $[0.037, 0.070]$ (49 of 64 seeds), with Parseval at a larger learning rate. With both at 2.5×10^{-4}, L2-init reaches 0.890, Parseval still beats it by $+0.026$ $[0.015, 0.037]$, and L2-init ties AdamO ($+0.008$ $[-0.005, 0.021]$ for AdamO; Section E). The regulariser also keeps the policy entropy higher (Table 1), so we test whether entropy alone explains the gain. Adam pretrained with the entropy coefficient 0.1 recovers the adaptability of a fresh network (0.824) but not more, and AdamO beats it by $+0.075$ $[0.053, 0.100]$. Muon pretrained with its entropy matched to that of MuonO (64 of 64 seeds within 0.03 nats) also reaches only about the level of a fresh network (0.846), and MuonO beats it by $+0.046$ $[0.030, 0.063]$. Adam cannot be matched in the same way: early in pretraining its entropy stays 0.1 to 0.3 nats above that of AdamO even with no entropy bonus. Adam matched as closely as the controller allows (2 of 64 seeds within 0.03 nats) reaches 0.814, and AdamO beats it by $+0.084$ $[0.060, 0.113]$ (Section E). A residual network with layer normalisation trained with Muon keeps more adaptability than plain Muon (0.724) but stays below a fresh network (-0.102 $[-0.144, -0.063]$; Section D).

Depth and pretraining length. In a pre-registered sweep over $L \in \{2, 4, 8, 16, 32\}$ (Figure 1, Section E), a regularised arm is the best or tied-best arm, and every regularised arm beats Adam with entropy coefficient 0.1, at every depth. Our pre-registered prediction that the AdamO gain over Adam grows with depth failed: the gain is large at every depth and largest at depths 2 and 32 ($+0.468$ and $+0.642$). With depth the base optimisers lose most of their adaptability (Muon from depth 16, Adam at depth 32), and from depth 8 on the regularised arms exceed a fresh network on average, while at depths 2 and 4 AdamO only matches it. After 16.8M pretraining transitions the regularised arms still beat a fresh network (AdamO $+0.064$ $[0.050, 0.079]$) but roughly tie L2-init (Section E). At width 256 AdamO still beats Adam ($+0.178$ $[0.127, 0.230]$ with retuned learning rates), but no regularised arm beats a fresh network, with or without retuning (Section E).

6.2 The result replicates in a DoorKey variant

The depth-16 result replicates in MiniGrid DoorKey, in a variant in which the held-out tasks conflict with pretraining (64 fresh seeds, pre-registered, Section F). A task is a 5×5 DoorKey layout of XLand-MiniGrid (Nikulin et al., 2024) with its own permutation of the six actions and a one-hot task identifier in the input, and the architecture, arms and procedure are those of NineRooms, with 20 training layouts drawn per seed, six test layouts and an adaptation learning rate of 2.5×10^{-4} selected on validation layouts. We chose this variant with a pre-registered gate: of four candidate cells, taken in a fixed order, it was the first in which Adam fits the training tasks at depth 16, a fresh network learns the held-out tasks, and Adam adapts at least 0.05 worse than a fresh network. The two cells with plain layouts failed the gate. In the final run, Adam and Muon adapt worse than a fresh network (full AUC 0.313 and 0.219 against 0.441), and AdamO, MuonO and Parseval adapt

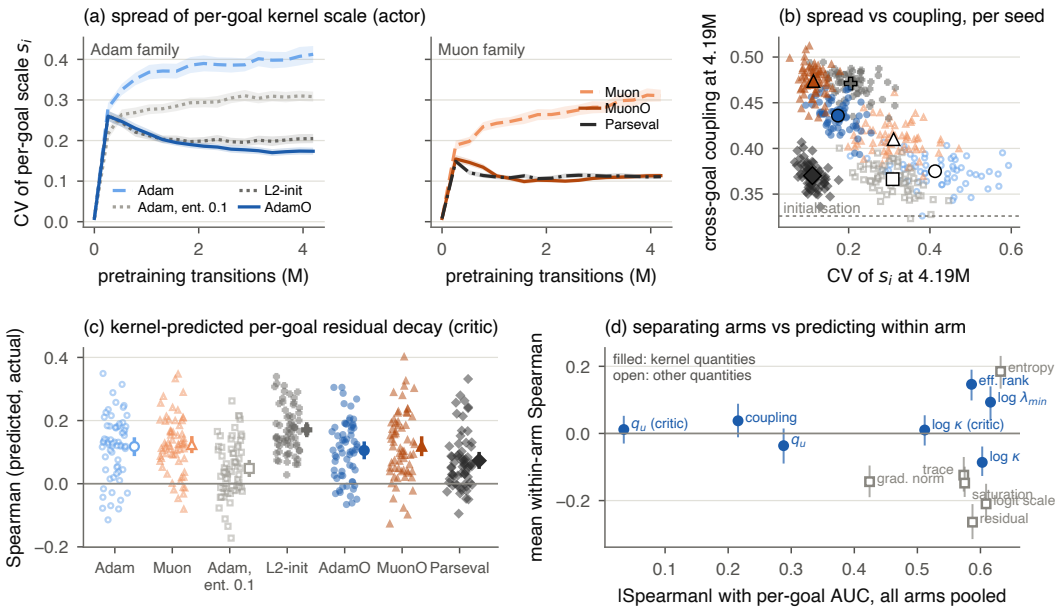

Figure 2: The multi-task kernel during depth-16 parallel pretraining in NineRooms, 64 seeds. (a) Coefficient of variation (CV) of the per-goal actor kernel scale s_i over the 20 training goals (mean, 95% band). (b) Per seed at the end of pretraining: CV of s_i against the mean normalised coupling between the kernel blocks of training goals (actor); large markers: arm means. (c) Per-seed Spearman correlation between the kernel-predicted and measured per-goal residual decay (critic). (d) Each quantity by how well it separates arms (absolute pooled Spearman correlation with per-goal held-out AUC) and predicts within arms (mean within-arm Spearman, 95% interval); filled: kernel quantities.

better (0.588, 0.662, 0.659). Both primary contrasts succeed: AdamO against Adam is $+0.275$ $[0.205, 0.345]$, and Parseval, the pilot-selected regularised arm, against the fresh network is $+0.218$ $[0.172, 0.262]$. The regularised arms also beat L2-init and Adam with entropy coefficient 0.1 by 0.14 to 0.24, which only reach the level of a fresh network. Here the gain is not an entropy effect: at the end of pretraining the regularised arms have lower policy entropy than Adam (AdamO 0.11 against 0.21 nats). The result holds in a constructed variant, not in plain DoorKey, and at depth 2, where pretraining is far from saturated, entropy-tuned Adam and L2-init beat AdamO and Parseval.

6.3 Lost adaptability goes with an overconfident policy

In NineRooms at depth 16, the loss of adaptability under Adam and Muon goes with an overconfident policy on the held-out goals, and the regularised arms in addition keep well-conditioned hidden layers (Section G, Figure 4). Our pre-registered hypothesis, saturation of the deep tanh units, passes its test, but controls show that it is not the cause: Adam with entropy coefficient 0.1 keeps most of the saturation and adapts like a fresh network, Adam at a larger learning rate saturates more and adapts better, and at depth 2 Adam loses adaptability without saturation. The held-out policy entropy and the logit scale track adaptation best within arms (mean within-arm Spearman correlations $+0.29$ and -0.30 over 12 arms). Only the regularised arms keep their feature rank through depth: against Adam matched as closely as possible to AdamO's entropy, AdamO has a deep feature rank of 37 against 18, a Gram defect of 0.050 against 0.233 and an end-to-end Jacobian $\log_{10} \kappa$ of 5.9 against 9.5. This evidence is correlational, and AdamO also has a higher held-out entropy than this Adam (0.66 against 0.46), so our data cannot separate the two.

6.4 The regulariser keeps per-task kernel scales nearly equal

We measured the empirical neural tangent kernel of actor and critic on 17 snapshots of the depth-16 runs (seven arms, 64 seeds), with five pre-registered predictions (Section H, Figure 2).

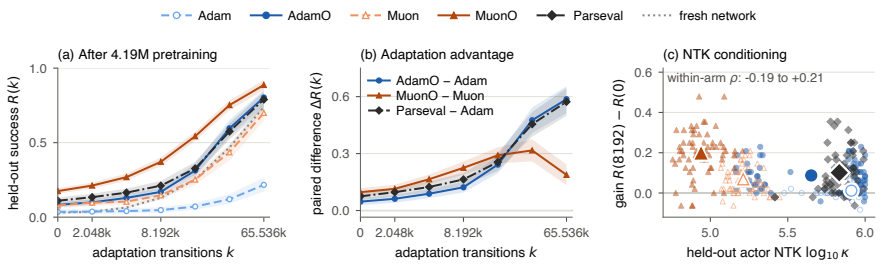

Figure 3: Shallow (depth-2) networks after 4.19M transitions of parallel pretraining, 64 seeds per arm. (a) Success on the held-out goals during gradient adaptation (grey dotted: fresh network). (b) Paired differences to the base optimiser with 95% intervals: small before adaptation, larger after a few updates. (c) Adaptation gain against the condition number of the held-out empirical NTK per seed; the arms separate, but within an arm the kernel does not predict the gain.

Per-task scales. The regularised optimisers keep the per-task kernel scales nearly equal, as the layerwise bounds suggest. At the end of pretraining the coefficient of variation of the actor's per-goal scale over the 20 training goals is 0.413 for Adam and 0.174 for AdamO, a paired difference of -0.239 $[-0.260, -0.218]$, and MuonO against Muon and Parseval against Adam give -0.197 $[-0.213, -0.181]$ and -0.302 $[-0.324, -0.280]$, likewise for the critic. After an early jump in every arm, the spread grows under Adam and Muon and shrinks under the regularised arms (Figure 2a).

Coupling and held-out rates. The prediction that the kernel blocks of the training goals merge under Adam and Muon failed: the actor coupling between goals ends higher under the regularised arms (AdamO 0.44, MuonO 0.47) than under Adam (0.38) and Muon (0.41), and no arm approaches full coupling. Held-out goals have a larger rate q_u under the regularised arms, but within an arm q_u hardly predicts per-goal adaptation. Within arms the policy entropy and the logit scale, declared as controls, rank per-goal adaptation better than every kernel quantity. The kernel therefore tracks the optimiser but does not by itself explain which seed or goal adapts fastest. In two further pre-registered experiments (Section H), when the reward of half the training goals is multiplied by 0.1, their actor kernel scale falls to 0.73 of the others under Adam but only to 0.85 to 0.91 under the regularised arms, so the regularised arms keep the scales more nearly equal also for these low-reward goals. The prediction that the regularised arms learn the low-reward goals earlier is falsified: AdamO and MuonO learn them as late as their base optimisers, and lowering the reward raises the held-out AUC of Adam and Muon by 0.22. The predicted growth of the gain with the number of goals also fails: AdamO beats Adam by $+0.60$ after pretraining on 5 goals and by $+0.11$ on 60, so the gain is largest with few tasks.

6.5 In shallow networks the gain is mostly an entropy effect

With the depth-2 network of Chung et al. (2024) the regulariser also preserves adaptability, but the gain over the base optimiser is mostly an entropy effect (64 paired seeds, Table 3). After 4.19M pretraining transitions AdamO beats Adam by $+0.478$ $[0.429, 0.523]$ and MuonO beats Muon by $+0.128$ $[0.096, 0.160]$. The difference between AdamO and Adam is small before adaptation ($+0.047$ $[0.028, 0.066]$ zero-shot) and large over the first $65{,}536$ adaptation transitions ($+0.398$ $[0.348, 0.447]$), consistent with the conditional signature (Corollary 1, Figure 3). Entropy matching recovers most of this gain ($+0.411$ of $+0.478$ for Adam, all of it for Muon), only MuonO exceeds a fresh network, and after 16.8M transitions no arm does (Table 3).

6.6 Limitations

The positive evidence comes from two gridworld families, PPO and width 64, and in DoorKey from a constructed variant; at width 256 no regularised arm beats a fresh network, with or without retuned learning rates. In NineRooms the learning rates were selected at depth 2, baselines use default settings, and L2-init at the larger learning rate of Parseval ties AdamO. The mechanism and kernel evidence is correlational, and the theory concerns linear models and linearised networks. With

in-context adaptation, in RL^2 agents on XLand-MiniGrid and on a recurrent MetaMaze family, the regulariser gives no gain (Table 3). Loss of adaptability, and the gain from the regulariser, appeared only in the two gridworld families, whose held-out tasks conflict with pretraining: in CartPole, HalfCheetah and Ant Adam-pretrained networks adapt far better than fresh ones, and in MinAtar pretraining was too weak for a test and a pilot showed no gain (Table 3). The regulariser does not improve training fit in small settings (Table 3). On the continual stream of Chung et al. (2024) its gains are mostly an entropy effect (Table 3), and on a PointRobot stream it increases forgetting of earlier goals (Table 3). In a pilot, meta-training with Reptile avoids the loss of adaptability (Table 3).

7 Conclusion

Meta-learning with gradient adaptation assumes that pretraining on many tasks leaves a network that learns a new task faster than a random one. We showed that this assumption can fail for deep RL networks pretrained in parallel: at depth 16, in two gridworld families, pretraining with Adam or Muon leaves a network that adapts to held-out tasks worse than a fresh network, so pretraining reduces the adaptability it is meant to provide.

Optimisers that keep the weights close to orthogonal prevent most of this loss. AdamO, MuonO and Parseval regularisation preserve adaptability and, at width 64, exceed a fresh network. The gain holds against shrink-and-perturb, a larger entropy bonus and, for MuonO, entropy-matched Muon, while L2-init at the learning rate of Parseval ties AdamO. Our linearised kernel results relate few-step progress on a new task to its kernel and initial residual, and per-task training rates to the kernel blocks. The regularised networks keep their feature rank, their conditioning through depth and nearly equal per-task kernel scales, although within an arm policy entropy predicts adaptation better than the kernel and feature measures.

For practitioners who pretrain an agent on a task distribution and fine-tune it later, these results mean that the pretrained network should be compared with a fresh one on held-out tasks, and that a regulariser developed for continual learning also preserves adaptability in this setting. For research on plasticity, they show that the loss of plasticity is not limited to tasks that arrive in sequence.

The evidence has clear limits. The loss of adaptability appeared only in the two gridworld families: in CartPole, HalfCheetah and Ant pretraining did not reduce adaptability, in MinAtar no loss qualified for a test, and Reptile meta-training avoided it. At width 256 no regularised arm exceeds a fresh network, at depth 2 most of the gain is an entropy effect, and the mechanism evidence is correlational. Future work should identify which task distributions cause the loss at larger scale, test the regulariser with meta-training objectives whose inner loop is long, and replace the uniform orthogonality target with one that uses knowledge of the task distribution.

AI use statement

We used generative AI tools to write and test code, run and analyse experiments, search the literature, analyse and derive mathmatical properties and draft and reformulate text. All AI-generated code was checked by tests and by reproduction gates against published implementations, and all cited works were checked against the original papers. The authors set the research questions and made the key decisions. LLM-generated code was checked with unit tests and independent LLM code reviews. Reported numbers were cross-checked against the run archives by independent LLM audits, and every citation was checked for existence and for support of the cited claim by an automated pipeline that reads the cited papers. The authors take full responsibility for the content.

Reproducibility statement

Section 4 and Section B define the regulariser, the two optimisers and the controls exactly. Section 5 and Section B give the environments, the learning-rate selection for each arm, the seeds and the compute. Every reported number is a paired comparison over the stated seeds, computed from result files that record the configuration and code version of each run. Proofs appear in Sections A, A.2 and A.3. Code and result files will be made available upon publication.

Ethics statement

This work uses simulated environments only and involves no human subjects and no personal data. We see no direct negative societal impact beyond that of general progress in reinforcement learning.

References

Jordan T. Ash and Ryan P. Adams. On warm-starting neural network training. In *Advances in Neural Information Processing Systems*, volume 33, 2020. arXiv:1910.08475. 1, 2, 5

Wesley Chung, Lynn Cherif, David Meger, and Doina Precup. Parseval regularization for continual reinforcement learning. In *Advances in Neural Information Processing Systems*, 2024. 1, 2, 3, 4, 5, 8, 9, 18, 20

Shibhansh Dohare, J. Fernando Hernandez-Garcia, Qingfeng Lan, Parash Rahman, A. Rupam Mahmood, and Richard S. Sutton. Loss of plasticity in deep continual learning. *Nature*, 632:768–774, 2024. 1, 2

Simon S. Du, Wei Hu, Sham M. Kakade, Jason D. Lee, and Qi Lei. Few-shot learning via learning the representation, provably. In *International Conference on Learning Representations*, 2021. arXiv:2002.09434. 3

Yan Duan, John Schulman, Xi Chen, Peter L. Bartlett, Ilya Sutskever, and Pieter Abbeel. RL^2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016. 1, 2

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1126–1135, 2017. arXiv:1703.03400. 1, 2, 3, 12

Sebastian Flennerhag, Andrei A. Rusu, Razvan Pascanu, Francesco Visin, Hujun Yin, and Raia Hadsell. Meta-learning with warped gradient descent. In *International Conference on Learning Representations*, 2020. 2

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, volume 31, pp. 8571–8580, 2018. arXiv:1806.07572. 1, 4, 12, 13

Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks. `https://kellerjordan.github.io/posts/muon/`, 2024. 1, 2, 3

Arthur Juliani and Jordan T. Ash. A study of plasticity loss in on-policy deep reinforcement learning. In *Advances in Neural Information Processing Systems*, 2024. arXiv:2405.19153. 2

Aviral Kumar, Rishabh Agarwal, Dibya Ghosh, and Sergey Levine. Implicit under-parameterization inhibits data-efficient deep reinforcement learning. In *International Conference on Learning Representations*, 2021. 2

Saurabh Kumar, Henrik Marklund, and Benjamin Van Roy. Maintaining plasticity in continual learning via regenerative regularization. In *Proceedings of the 3rd Conference on Lifelong Learning Agents*, volume 274 of *Proceedings of Machine Learning Research*, pp. 410–430, 2025. arXiv:2308.11958. 2, 5

Vitaly Kurin, Alessandro De Palma, Ilya Kostrikov, Shimon Whiteson, and M. Pawan Kumar. In defense of the unitary scalarization for deep multi-task learning. In *Advances in Neural Information Processing Systems*, 2022. 3

Alex Lewandowski, Michał Bortkiewicz, Saurabh Kumar, András György, Dale Schuurmans, Mateusz Ostaszewski, and Marlos C. Machado. Learning continually by spectral regularization. In *International Conference on Learning Representations*, 2025. arXiv:2406.06811. 2

Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li. Meta-SGD: Learning to learn quickly for few-shot learning. *arXiv preprint arXiv:1707.09835*, 2017. 2

Clare Lyle, Mark Rowland, and Will Dabney. Understanding and preventing capacity loss in reinforcement learning. In *International Conference on Learning Representations*, 2022. 1, 2

Clare Lyle, Zeyu Zheng, Evgenii Nikishin, Bernardo Avila Pires, Razvan Pascanu, and Will Dabney. Understanding plasticity in neural networks. In *International Conference on Machine Learning*, 2023. 1, 3

Evgenii Nikishin, Max Schwarzer, Pierluca D'Oro, Pierre-Luc Bacon, and Aaron Courville. The primacy bias in deep reinforcement learning. In *International Conference on Machine Learning*, 2022. 1, 2

Alexander Nikulin, Vladislav Kurenkov, Ilya Zisman, Artem Agarkov, Viacheslav Sinii, and Sergey Kolesnikov. XLand-MiniGrid: Scalable meta-reinforcement learning environments in JAX. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. 2, 5, 6, 19

Jeffrey Pennington, Samuel S. Schoenholz, and Surya Ganguli. Resurrecting the sigmoid in deep learning through dynamical isometry: Theory and practice. In *Advances in Neural Information Processing Systems*, 2017. 2, 3

Xiaohan Qin, Xiaoxing Wang, Ning Liao, and Junchi Yan. NTKMTL: Mitigating task imbalance in multi-task learning from neural tangent kernel perspective. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2510.18258. arXiv:2510.18258. 3, 16

Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. Rapid learning or feature reuse? Towards understanding the effectiveness of MAML. In *International Conference on Learning Representations*, 2020. 2

Andries Rosseau, Robert Müller, and Ann Nowé. Preserving plasticity in continual learning via dynamical isometry. In *Forty-Third International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2606.09762. arXiv:2606.09762. 1, 2, 3

Olivier Roy and Martin Vetterli. The effective rank: A measure of effective dimensionality. In *European Signal Processing Conference*, 2007. 3

Andrew M. Saxe, James L. McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *International Conference on Learning Representations*, 2014. 2, 3

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 3, 4

Ghada Sokar, Rishabh Agarwal, Pablo Samuel Castro, and Utku Evci. The dormant neuron phenomenon in deep reinforcement learning. In *International Conference on Machine Learning*, 2023. 1, 2, 3

Nilesh Tripuraneni, Chi Jin, and Michael I. Jordan. Provable meta-learning of linear representations. In *International Conference on Machine Learning*, 2021. arXiv:2002.11684. 3

Lechao Xiao, Yasaman Bahri, Jascha Sohl-Dickstein, Samuel S. Schoenholz, and Jeffrey Pennington. Dynamical isometry and a mean field theory of CNNs: How to train 10,000-layer vanilla convolutional neural networks. In *International Conference on Machine Learning*, 2018. 2, 3

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems*, 2020. 2

Yufan Zhou, Zhenyi Wang, Jiayi Xian, Changyou Chen, and Jinhui Xu. Meta-learning with neural tangent kernels. In *International Conference on Learning Representations*, 2021. arXiv:2102.03909. 1, 2, 4, 13

A WHY WELL-CONDITIONED WEIGHTS HELP A SHARED NETWORK

Roadmap. This appendix proves the statements of Section 4.2; the exact results concern linearised networks. Section A.1 gives two facts about linear layers used later, Section A.2 treats adaptation to one task and Section A.3 treats several tasks trained in parallel.

Notation. $\|\cdot\|_2$ is the spectral norm, $\|\cdot\|_F$ the Frobenius norm and $\langle X, Y\rangle = \mathrm{tr}(X^\top Y)$. X^+ is the Moore–Penrose pseudoinverse and $O(k)$ the orthogonal group. $\sigma_q(X)$ is the q-th largest singular value and $\lambda_{\max}, \lambda_{\min}$ are extreme eigenvalues of a symmetric matrix. For a matrix X of rank ρ, $\kappa(X) = \sigma_1(X)/\sigma_\rho(X)$; for a positive-definite matrix this is $\lambda_{\max}/\lambda_{\min}$. The index m counts gradient steps, q indexes spectral modes, and a bare $\delta \in (0, 1)$ is a target reduction factor (the backpropagated errors $\delta_{t,a}$ carry task and sample indices). The regulariser coefficient is written ω in this appendix.

A.1 ONE LAYER AND A LINEAR STACK

Let $z = Wh$ with $W \in \mathbb{R}^{n_{\mathrm{out}} \times n_{\mathrm{in}}}$. Task t has per-sample losses $\ell_{t,a}$ on inputs $h_{t,a}$ and total loss $L_t = \sum_a \ell_{t,a}$. Let H_t have rows $h_{t,a}^\top$ and Δ_t have rows $\delta_{t,a}^\top = (\partial \ell_{t,a}/\partial z)^\top$, so that $G_t = \nabla_W L_t = \Delta_t^\top H_t$. For $c > 0$ define the Gram defect $\varepsilon_c(W) = \|W^\top W - cI\|_2$. Since $\|X\|_2 \le \|X\|_F$, the Frobenius defect penalised by the regulariser upper-bounds ε_c; a finite coefficient does not enforce a prescribed value of it.

Proposition 1 (First-order interference factorises)**.** *If L_j is twice continuously differentiable near W, a step $W^+ = W - \eta G_i$ gives*

$$L_j(W^+) - L_j(W) = -\eta \langle G_i, G_j\rangle + O(\eta^2 \|G_i\|_F^2), \tag{3}$$

$$\langle G_i, G_j\rangle = \sum_{a,b} \langle \delta_{i,a}, \delta_{j,b}\rangle \langle h_{i,a}, h_{j,b}\rangle = \langle \Delta_i \Delta_j^\top, H_i H_j^\top\rangle, \tag{4}$$

$$|\langle G_i, G_j\rangle| \le \|\Delta_i \Delta_j^\top\|_F \|H_i H_j^\top\|_F. \tag{5}$$

Thus this layer has zero first-order cross-task interference if either overlap matrix is zero, for example if every input of task i is orthogonal to every input of task j.

Proof. Taylor's theorem with the Hessian bounded on a neighbourhood of W gives the first line, since $\nabla_W L_j = G_j$. Expanding $G_t = \sum_a \delta_{t,a} h_{t,a}^\top$ and $\mathrm{tr}(h\delta^\top \delta' h'^\top) = \langle \delta, \delta'\rangle \langle h, h'\rangle$ gives equation 4. Frobenius Cauchy–Schwarz gives the bound. □

Proposition 2 (Accumulation through linear depth)**.** *Let $W_{L:1} = W_L \cdots W_1$ with compatible shapes, and suppose $0 \le \varepsilon < 1$ and $\|W_l^\top W_l - I\|_2 \le \varepsilon$ for all l. Then*

$$\|W_{L:1}^\top W_{L:1} - I\|_2 \le (1 + \varepsilon)^L - 1, \tag{6}$$

and the same constant bounds $|\langle W_{L:1} h, W_{L:1} h'\rangle - \langle h, h'\rangle|/(\|h\|\|h'\|)$. The backward analogue, for $W_{L:1}^\top$ acting on output errors, holds if instead $\|W_l W_l^\top - I\|_2 \le \varepsilon$ for all l. Because $\varepsilon < 1$, the forward condition forces every layer to be tall or square and the backward condition forces it to be wide or square, so both hold together only for square layers.

Proof. Every eigenvalue of $W_l^\top W_l$ lies in $[1 - \varepsilon, 1 + \varepsilon]$, so $(1 - \varepsilon)\|x\|^2 \le \|W_l x\|^2 \le (1 + \varepsilon)\|x\|^2$ for all x. Applying this layer by layer gives $(1 - \varepsilon)^L I \preceq W_{L:1}^\top W_{L:1} \preceq (1 + \varepsilon)^L I$. By convexity, $(1 + \varepsilon)^L + (1 - \varepsilon)^L \ge 2$, so the larger deviation from one is $(1 + \varepsilon)^L - 1$. Since $\langle W_{L:1} h, W_{L:1} h'\rangle - \langle h, h'\rangle = h^\top (W_{L:1}^\top W_{L:1} - I) h'$, the operator-norm inequality gives the inner-product bound. Replacing every layer by its transpose proves the backward case. A positive-definite $W_l^\top W_l$ requires full column rank, hence $n_{\mathrm{out}} \ge n_{\mathrm{in}}$, and symmetrically for $W_l W_l^\top$. □

A.2 FUNCTION-SPACE VIEW: THE NEURAL TANGENT KERNEL

This subsection states the few-step adaptation calculation for all parameters of a network, linearised at its current parameters, as an approximation to a few small fine-tuning or MAML inner-loop steps (Finn et al., 2017). The kernel that appears is the empirical neural tangent kernel (Jacot et al., 2018);

Zhou et al. (2021) build meta-learning algorithms on this function-space view of MAML. We use only the elementary identities below. All statements concern the *linearised* model; we do not bound its error for a finite network. In the infinite-width regime analysed by Jacot et al. (2018), the limiting kernel stays constant during training under their assumptions.

Setting. Let $f_\theta(x) \in \mathbb{R}^{n_L}$ with parameters $\theta \in \mathbb{R}^p$, differentiable in θ at θ_0 on every support and query input used below. For support inputs $x_1, \dots, x_n$ stack the outputs sample by sample into $f(\theta) \in \mathbb{R}^{nn_L}$, and let $J = \partial f(\theta_0)/\partial\theta \in \mathbb{R}^{nn_L \times p}$. The empirical kernel is $\mathcal{K} = JJ^\top$, with eigenpairs (λ_q, χ_q), $\lambda_q \geq 0$ and orthonormal χ_q. For query inputs with stacked outputs $f_{\mathrm{qu}}(\theta)$ and Jacobian J_{qu}, let $\mathcal{K}_{\mathrm{qs}} = J_{\mathrm{qu}}J^\top$. Let Π_0 be the orthogonal projector onto $\ker\mathcal{K} = \ker J^\top = \mathrm{range}(J)^\perp$. The linearised model is $f^{\mathrm{lin}}(\theta) = f(\theta_0) + J(\theta - \theta_0)$, and similarly for $f^{\mathrm{lin}}_{\mathrm{qu}}$. A new task has targets $y \in \mathbb{R}^{nn_L}$ and loss $\ell(\theta) = \frac{1}{2}\|f(\theta) - y\|^2$. Linearised gradient descent is $\theta_{m+1} = \theta_m - \eta J^\top(f^{\mathrm{lin}}(\theta_m) - y)$ from θ_0 with $\eta > 0$, with residual $r_m = f^{\mathrm{lin}}(\theta_m) - y$ and loss $\ell_m = \frac{1}{2}\|r_m\|^2$.

Proposition 3 (Linearised adaptation is kernel spectral filtering)**.** *In this setting, with* $\ell_\infty = \frac{1}{2}\|\Pi_0 r_0\|^2$*:*

(i) $r_m = (I - \eta\mathcal{K})^m r_0$ *with* $r_0 = f(\theta_0) - y$*, and*

$$\ell_m = \frac{1}{2}\sum_q (1 - \eta\lambda_q)^{2m}(\chi_q^\top r_0)^2, \qquad \ell_m - \ell_\infty = \frac{1}{2}\sum_{q:\lambda_q>0} (1 - \eta\lambda_q)^{2m}(\chi_q^\top r_0)^2. \quad (7)$$

(ii) The query predictions after m *steps, which the outer loop of MAML evaluates, are*

$$f^{\mathrm{lin}}_{\mathrm{qu}}(\theta_m) = f_{\mathrm{qu}}(\theta_0) - \mathcal{K}_{\mathrm{qs}}\mathcal{K}^+\big[I - (I - \eta\mathcal{K})^m\big]r_0. \quad (8)$$

(iii) If $\lambda_{\max}(\mathcal{K}) > 0$ *and* $0 < \eta < 2/\lambda_{\max}(\mathcal{K})$*, then* ℓ_m *is non-increasing,* $r_m \to \Pi_0 r_0$ *and* $\ell_m \to \ell_\infty$*. If* $\lambda_{\max}(\mathcal{K}) > 0$*,* $\eta > 2/\lambda_{\max}(\mathcal{K})$ *and* $\chi_q^\top r_0 \neq 0$ *for some* $\lambda_q > 2/\eta$*, then* $\ell_m \to \infty$*. At* $\eta = 2/\lambda_{\max}(\mathcal{K})$ *with* $\lambda_{\max}(\mathcal{K}) > 0$*, the component of* r_0 *on the top eigenspace is not damped. If* $\mathcal{K} = 0$*, then* $r_m = r_0$ *for every* η *and* m*.*

(iv) Suppose $(I - \Pi_0)r_0 \neq 0$*, and let* $a \leq b$ *be the smallest and largest positive eigenvalues of* $\mathcal{K}$ *whose eigenspaces contain a non-zero component of* r_0 *(for a repeated eigenvalue, the spectral projector applied to* r_0 *is non-zero). Let* $\kappa_r = b/a$*, the condition number of* $\mathcal{K}$ *restricted to the smallest invariant subspace that contains* $(I - \Pi_0)r_0$*. For every* $\eta > 0$ *and* $m \geq 0$*,*

$$\ell_m - \ell_\infty \leq \rho(\eta)^{2m}(\ell_0 - \ell_\infty), \qquad \rho(\eta) = \max\{|1 - \eta a|, |1 - \eta b|\}, \quad (9)$$

where $\rho^0 = 1$ *also when* $\rho = 0$*. With* $\eta = 1/\lambda_{\max}(\mathcal{K})$*,* $\rho = 1 - a/\lambda_{\max}(\mathcal{K})$*; with* $\eta = 1/b$*,* $\rho = 1 - 1/\kappa_r$*, so* $\ell_m - \ell_\infty \leq e^{-2m/\kappa_r}(\ell_0 - \ell_\infty)$ *and any integer* $m \geq \lceil(\kappa_r/2)\log(1/\delta)\rceil$ *reduces the reducible loss by at least the factor* $\delta \in (0, 1)$*; with* $\eta = 2/(a + b)$*,* $\rho = (\kappa_r - 1)/(\kappa_r + 1)$*. Conversely, for every* $\eta > 0$*, with* P_a *the spectral projector of* $\mathcal{K}$ *for eigenvalue* a*,* $\ell_m - \ell_\infty \geq \frac{1}{2}(1 - \eta a)^{2m}\|P_a r_0\|^2$*.*

Proof. (i) $f^{\mathrm{lin}}(\theta_{m+1}) - f^{\mathrm{lin}}(\theta_m) = J(\theta_{m+1} - \theta_m) = -\eta\mathcal{K}r_m$, so $r_{m+1} = (I - \eta\mathcal{K})r_m$. Expanding in the eigenbasis gives the first sum; the terms with $\lambda_q = 0$ are constant and sum to ℓ_∞. (ii) $\theta_m - \theta_0 = -\eta J^\top \sum_{j<m}(I - \eta\mathcal{K})^j r_0$. On an eigenvector with $\lambda_q > 0$, $\eta\sum_{j<m}(1 - \eta\lambda_q)^j = [1 - (1 - \eta\lambda_q)^m]/\lambda_q$, which is the action of $\mathcal{K}^+[I - (I - \eta\mathcal{K})^m]$. On $\ker\mathcal{K} = \ker J^\top$ both $J^\top$ and $\mathcal{K}_{\mathrm{qs}} = J_{\mathrm{qu}}J^\top$ vanish. Apply J_{qu}. (iii) For $0 < \eta < 2/\lambda_{\max}$, $|1 - \eta\lambda_q| < 1$ for every $\lambda_q > 0$, so every term of equation 7 with $\lambda_q > 0$ is non-increasing in m and tends to zero. If $\lambda_q > 2/\eta$, then $|1 - \eta\lambda_q| > 1$ and that term diverges. At $\eta\lambda_{\max} = 2$ the top factor is $(-1)^m$. (iv) Only eigenvalues in $\{\lambda_q > 0 : \chi_q^\top r_0 \neq 0\} \subseteq [a, b]$ enter the second sum in equation 7, and $|1 - \eta\lambda| \leq \rho(\eta)$ on $[a, b]$ because $|1 - \eta\lambda|$ is convex in λ. The three step sizes give the three values of ρ, and $1 - 1/\kappa_r \leq e^{-1/\kappa_r}$. Every term of equation 7 is non-negative, so dropping all terms except those of eigenvalue a gives the lower bound. □

Layerwise structure of the kernel. The finite-step expression in Proposition 3 makes the kernel at the start of adaptation relevant only through directions that carry the new task's residual. We now bound that kernel in terms of quantities that can be measured at the pretrained weights. The linear case gives a weight-defect bound; for nonlinear networks the bound also requires the realised features and activation derivatives. Let $L \geq 2$, $h_0 = x$, $z_l = W_l h_{l-1}$, $h_l = \sigma(z_l)$ for $1 \leq l \leq L-1$ and $f = z_L = W_L h_{L-1}$, with $W_l \in \mathbb{R}^{n_l \times n_{l-1}}$, no biases and $\theta = (W_1, \ldots, W_L)$. The coordinatewise σ is differentiable at every realised pre-activation, or σ' is given a fixed value at its kinks, as automatic differentiation does; at a kink this defines a selected linear surrogate rather than a classical derivative of f. Let $D_l(x) = \operatorname{diag} \sigma'(z_l(x))$, $\Delta_L = I_{n_L}$ and $\Delta_l(x) = \Delta_{l+1}(x) W_{l+1} D_l(x) \in \mathbb{R}^{n_L \times n_l}$, the derivative $\partial f(x)/\partial z_l$ away from kinks. Let H_{l-1} have rows $h_{l-1}(x_i)^\top$ and $\mathcal{G}_{l-1} = H_{l-1} H_{l-1}^\top$ be the realised feature Gram on the support inputs, with $\mathcal{G}_0 = XX^\top$ for the input matrix $X = H_0$. We write $\sigma_{n_L}(\cdot)$ for the n_L-th singular value, which is zero if $n_l < n_L$.

Proposition 4 (Isometry and realised features bound the kernel's conditioning)**.** *In this setting:*

(a) $\mathcal{K} = \sum_{l=1}^{L} \mathcal{K}_l$*, where the* (i,j) *block of* $\mathcal{K}_l$ *is* $[\mathcal{G}_{l-1}]_{ij}\, \Delta_l(x_i) \Delta_l(x_j)^\top$.

(b) For any σ*,*

$$\begin{aligned} \lambda_{\min}(\mathcal{K}) &\geq \textstyle\sum_l \lambda_{\min}(\mathcal{G}_{l-1}) \min_i \sigma_{n_L}(\Delta_l(x_i))^2, \\ \lambda_{\max}(\mathcal{K}) &\leq \textstyle\sum_l \lambda_{\max}(\mathcal{G}_{l-1}) \max_i \|\Delta_l(x_i)\|_2^2. \end{aligned} \tag{10}$$

In particular $\mathcal{K} \succeq \mathcal{K}_L = \mathcal{G}_{L-1} \otimes I_{n_L}$*, so* $\lambda_{\min}(\mathcal{K}) \geq \lambda_{\min}(\mathcal{G}_{L-1})$*, the smallest eigenvalue of the realised penultimate-feature Gram. The inequality* $\mathcal{K} \succeq \mathcal{G}_{L-1} \otimes I_{n_L}$ *holds for any architecture whose output is a trained linear read-out* W_L *of features* h_{L-1} *that do not depend on* W_L*, with* W_L *an independent parameter block.*

(c) (Linear.) Let σ *be the identity and* $0 \leq \varepsilon < 1$*. Assume* $\|W_l^\top W_l - I\|_2 \leq \varepsilon$ *for* $1 \leq l \leq L-1$ *and* $\|W_l W_l^\top - I\|_2 \leq \varepsilon$ *for* $2 \leq l \leq L$*; by Proposition 2 this makes* W_1 *tall or square,* W_L *wide or square and every hidden layer square. Then* $\mathcal{K}_l = \mathcal{G}_{l-1} \otimes W_{L:l+1} W_{L:l+1}^\top$ *(with* $W_{L:L+1} = I$*),*

$$L(1-\varepsilon)^{L-1}(XX^\top \otimes I_{n_L}) \preceq \mathcal{K} \preceq L(1+\varepsilon)^{L-1}(XX^\top \otimes I_{n_L}), \tag{11}$$

$\mathcal{K}$ *and* $XX^\top \otimes I_{n_L}$ *have the same kernel; if* $X \neq 0$*, then* $\kappa(\mathcal{K}) \leq \kappa(XX^\top)\big((1+\varepsilon)/(1-\varepsilon)\big)^{L-1}$*. For* $\varepsilon = 0$*,* $\mathcal{K} = L\, XX^\top \otimes I_{n_L}$ *at every depth.*

Proof. (a) The derivative of $f(x)$ with respect to the entry $(W_l)_{ab}$ is $\Delta_l(x)_{:,a} h_{l-1}(x)_b$, so the contribution of W_l to the (i,j) block of $JJ^\top$ is $\sum_{a,b} \Delta_l(x_i)_{:,a} \Delta_l(x_j)_{:,a}^\top h_{l-1}(x_i)_b h_{l-1}(x_j)_b$, which is the stated product. (b) Let $\mathcal{D}_l = \operatorname{blockdiag}_i \Delta_l(x_i)$. Then $\mathcal{K}_l = \mathcal{D}_l(\mathcal{G}_{l-1} \otimes I_{n_l})\mathcal{D}_l^\top$, so for v with blocks v_i, $v^\top \mathcal{K}_l v \in [\lambda_{\min}(\mathcal{G}_{l-1}), \lambda_{\max}(\mathcal{G}_{l-1})] \cdot \|\mathcal{D}_l^\top v\|^2$ and $\|\mathcal{D}_l^\top v\|^2 = \sum_i \|\Delta_l(x_i)^\top v_i\|^2 \in [\min_i \sigma_{n_L}(\Delta_l(x_i))^2, \max_i \|\Delta_l(x_i)\|_2^2] \cdot \|v\|^2$. Sum over l and use Courant–Fischer. $\Delta_L = I$ gives the head term, and every $\mathcal{K}_l \succeq 0$. The read-out argument uses only that $\partial f/\partial W_L$ is $I \otimes h_{L-1}^\top$ and that the other parameter blocks add positive semidefinite terms. (c) With σ the identity, $\Delta_l = W_{L:l+1}$ does not depend on x and $\mathcal{G}_{l-1} = X W_{l-1:1}^\top W_{l-1:1} X^\top$. The proof of Proposition 2 gives $(1 \mp \varepsilon)^{l-1} I$ as Loewner bounds for $W_{l-1:1}^\top W_{l-1:1}$ and, by its backward case, $(1 \mp \varepsilon)^{L-l} I$ for $W_{L:l+1} W_{L:l+1}^\top$. For positive semidefinite $0 \preceq A_1 \preceq A_2$ and $0 \preceq B_1 \preceq B_2$, $A_2 \otimes B_2 - A_1 \otimes B_1 = (A_2 - A_1) \otimes B_2 + A_1 \otimes (B_2 - B_1) \succeq 0$. Hence every $\mathcal{K}_l$ lies between $(1 \mp \varepsilon)^{L-1} XX^\top \otimes I$; summing L terms gives equation 11. Two positive semidefinite matrices sandwiched between positive multiples of each other have the same kernel, and Courant–Fischer on its orthogonal complement bounds the extreme non-zero eigenvalues; $\kappa(XX^\top \otimes I) = \kappa(XX^\top)$. □

Proposition 5 (Gates can make the kernel ill-conditioned with orthogonal weights)**.** *Let* $\sigma = \max(\cdot, 0)$*,* $L \geq 2$*,* $n_0 = \cdots = n_{L-1} = n \geq 3$ *and* $n_L = 1$*. Let* W_1 *be orthogonal and* $W_2, \ldots, W_{L-1}$ *permutation matrices. Let* $z \in \mathbb{R}^n$ *have no zero entries, with non-empty positive set* P *and a negative set* N *on which* z *is not constant, and let* z' *equal* z *except that its entries on* N *are permuted so that* $z' \neq z$*. Let* P_{L-1} *be the image of* P *under the permutation* $W_{L-1} \cdots W_2$*, and let* $W_L = w$ *be a unit row vector with support in* P_{L-1}*. Put* $x = W_1^\top z$ *and* $x' = W_1^\top z'$*. Then, for any value assigned to* $\sigma'(0)$*:*

(i) $W_l^\top W_l = I$ *for* $l < L$ *and* $ww^\top = 1$*, so the* weight-isometry *hypotheses of Proposition 4(c) hold with* $\varepsilon = 0$*; also* $\|x\| = \|x'\|$ *and* $\kappa(\mathcal{G}_0) = \|x + x'\|^2/\|x - x'\|^2 < \infty$*;*

(ii) $h_l(x) = h_l(x')$ *for* $1 \le l \le L - 1$*, so every realised Gram* $\mathcal{G}_l$*,* $l \ge 1$*, of the pair has rank one;*

(iii) on the pair, $\mathcal{K} = (L-1)\|z_+\|^2 \mathbf{1}\mathbf{1}^\top + \mathcal{G}_0$*, and the kernel of the parameters of layers* $2, \dots, L$ *alone is* $(L-1)\|z_+\|^2 \mathbf{1}\mathbf{1}^\top$*, of rank one;*

(iv) $\kappa(\mathcal{K}) = \kappa(\mathcal{G}_0) + 4(L-1)\|z_+\|^2/\|x - x'\|^2$*, which grows linearly in L, whereas the linear network with the same weights has* $\mathcal{K} = L\mathcal{G}_0$ *and* $\kappa(\mathcal{K}) = \kappa(\mathcal{G}_0)$ *at every depth. For any support set that contains* x *and* x'*,* $\lambda_{\max}(\mathcal{K})/\lambda_{\min}(\mathcal{K})$ *is at least this value (and infinite if* $\mathcal{K}$ *is singular).*

Proof. (i) Permutations and orthogonal matrices have orthonormal columns, W_1 preserves norms and $\|z\| = \|z'\|$. With $r^2 = \|x\|^2$ and $g = \langle x, x' \rangle$, $\mathcal{G}_0$ has eigenvectors $(1, \pm 1)$ and eigenvalues $r^2 \pm g = \|x \pm x'\|^2/2$; $x \ne x'$ because $z \ne z'$, and $g = \|z_+\|^2 + \langle z_N, z'_N \rangle > 0$ because both negative parts are entrywise negative. (ii) $z_1(x) = z$ and $z_1(x') = z'$ agree on P and are negative on N, so $h_1 = z_+$ for both and D_1 is the indicator of P for both. For $l \ge 2$, $W_l h_{l-1}$ is a permutation of a non-negative vector, so $h_l = W_l h_{l-1}$, identical for both inputs, with $\|h_l\| = \|z_+\|$ and positive entries exactly on the image P_l of P; the pre-activations and hence D_l agree, whatever value $\sigma'(0)$ takes. (iii) Backwards, $\Delta_{L-1} = wD_{L-1} = w$ since w is supported where the pre-activations are positive. If Δ_{l+1} is a unit row vector supported in P_{l+1}, then $\Delta_{l+1} W_{l+1}$ is a unit vector supported in P_l and D_l is the identity there, so $\Delta_l = \Delta_{l+1} W_{l+1}$, again unit and identical for both inputs. (For $L = 2$, $\Delta_1 = wD_1 = w$ directly.) By Proposition 4(a), $\mathcal{K}_1 = \mathcal{G}_0$ and $\mathcal{K}_l = \|z_+\|^2 \mathbf{1}\mathbf{1}^\top$ for $2 \le l \le L$. (iv) $\mathcal{K}$ has the eigenvectors $(1, \pm 1)$ of $\mathcal{G}_0$, with eigenvalues $2(L-1)\|z_+\|^2 + \|x + x'\|^2/2$ and $\|x - x'\|^2/2$; divide. In the linear network $\Delta_l = wW_{L-1} \cdots W_{l+1}$ is a unit vector and $\mathcal{G}_{l-1} = \mathcal{G}_0$, so $\mathcal{K} = L\mathcal{G}_0$. Kernel entries depend only on the two inputs involved, so the pair's kernel is a principal submatrix of the kernel of any larger support set, whose extreme eigenvalues enclose those of the submatrix (Cauchy interlacing). □

Corollary 1 (Equal zero-shot loss, different few-step loss)**.** *Let two models* (f, θ_0) *and* (f', θ'_0) *have equal outputs on the support inputs,* $f(\theta_0) = f'(\theta'_0)$*, and non-zero kernels* $\mathcal{K}, \mathcal{K}'$ *there. For the same targets, run linearised gradient descent (Proposition 3) with* $\eta = 1/\lambda_{\max}(\mathcal{K})$ *and* $\eta' = 1/\lambda_{\max}(\mathcal{K}')$ *respectively. Then* $r_0 = r'_0$ *and* $\ell_0 = \ell'_0$*. Moreover:*

(i) If $\mathcal{K}, \mathcal{K}' \succ 0$ *and* $m \ge 1$*, then* $\ell'_m \le (1 - 1/\kappa(\mathcal{K}'))^{2m} \ell_0$ *and* $\ell_m \ge \frac{1}{8} \sum_{q : \lambda_q \le \lambda_{\max}(\mathcal{K})/(2m)} (\chi_q^\top r_0)^2$*. Hence* $\ell'_m < \ell_m$ *whenever* $(1 - 1/\kappa(\mathcal{K}'))^{2m} \|r_0\|^2$ *is smaller than one quarter of the mass of* r_0 *on the eigenvectors of* $\mathcal{K}$ *with* $\lambda_q \le \lambda_{\max}(\mathcal{K})/(2m)$*.*

(ii) If $\mathcal{K}$ *and* $\mathcal{K}'$ *have common eigenvectors* χ_q *and* $\lambda_q/\lambda_{\max}(\mathcal{K}) \le \lambda'_q/\lambda_{\max}(\mathcal{K}')$ *for every* q*, then* $\ell'_m \le \ell_m$ *for all* m*, with strict inequality for every* $m \ge 1$ *iff* $\chi_q^\top r_0 \ne 0$ *for some* q *at which the inequality between the normalised eigenvalues is strict.*

(iii) For $0 < \eta < 2/\lambda_{\max}(\mathcal{K})$ *the linearised iterates converge to* $\theta_\infty = \theta_0 - J^+ r_0$*, so the limiting support loss* $\frac{1}{2}\|\Pi_0 r_0\|^2$ *depends only on* $\mathrm{range}(J)$*; two models with the same range (in particular two with positive definite kernels) have the same limit, zero in the positive definite case. If in addition* J *has full column rank,* $J' = JQ$ *and* $J'_{\mathrm{qu}} = J_{\mathrm{qu}} Q$ *for an invertible* Q *and* $f_{\mathrm{qu}}(\theta_0) = f'_{\mathrm{qu}}(\theta'_0)$*, the limiting query predictions* $f_{\mathrm{qu}}(\theta_0) - J_{\mathrm{qu}} J^+ r_0$ *also coincide. Without full column rank the minimum-norm least-squares parameter solutions can yield different query predictions under a change of parameter basis.*

Proof. The residual $r_0 = f(\theta_0) - y$ is the same for both. (i) With $\ker \mathcal{K}' = 0$, equation 9 with $a \ge \lambda_{\min}(\mathcal{K}')$ gives the upper bound. For the lower bound keep in equation 7 only the modes with $\lambda_q \le \lambda_{\max}/(2m)$, for which $(1 - \lambda_q/\lambda_{\max})^{2m} \ge (1 - 1/(2m))^{2m} \ge 1/4$. (ii) Compare equation 7 term by term: for $0 \le u < v \le 1$, $0 \le (1-v)^{2m} < (1-u)^{2m}$ when $m \ge 1$. (iii) $\delta_m = \theta_m - \theta_0$ lies in $\mathrm{range}(J^\top)$ and obeys $\delta_{m+1} = (I - \eta J^\top J)\delta_m - \eta J^\top r_0$. On $\mathrm{range}(J^\top)$, $J^\top J$ is positive definite with the non-zero eigenvalues of $\mathcal{K}$, so for $0 < \eta < 2/\lambda_{\max}(\mathcal{K})$ the iteration contracts to the unique

fixed point there, which solves $J^\top J\delta = -J^\top r_0$ and is $-J^+ r_0$; then $r_\infty = (I - JJ^+)r_0 = \Pi_0 r_0$. Under full column rank, $(JQ)^+ = Q^{-1}J^+$, so $J_{\mathrm{qu}}QQ^{-1}J^+ = J_{\mathrm{qu}}J^+$. □

A.3 Several tasks: the block structure of the kernel

Section A.2 adapts one task. In parallel multi-task training all tasks move the same parameters, so the kernel of all training inputs splits into one block per pair of tasks. The diagonal blocks contribute to per-task fitting rates, and the off-diagonal blocks carry first-order transfer and interference. Qin et al. (2025) use per-task kernel spectra to balance the convergence speeds of tasks. As in Section A.2, every rate below concerns the *linearised* model, whose kernel does not change during training. Proposition 6(iii) gives a first-order step statement for the network itself, while Proposition 6(iv) factorises its instantaneous gradients at differentiable parameters. The proofs of Propositions 6 to 8 follow by applying the arguments of Propositions 3 and 4 to each block of the kernel.

Setting. Keep the setting of Section A.2. Tasks $i = 1, \dots, T$ have support inputs $x_{i,1}, \dots, x_{i,n_i}$, stacked outputs $f_i(\theta) \in \mathbb{R}^{d_i}$ with $d_i = n_i n_L$, targets $y_i \in \mathbb{R}^{d_i}$, Jacobians $J_i = \partial f_i(\theta_0)/\partial\theta \in \mathbb{R}^{d_i \times p}$ and losses $L_i(\theta) = \frac{1}{2}\|f_i(\theta) - y_i\|^2$. The kernel of all training inputs is $\mathcal{K} = JJ^\top$ for the stacked Jacobian J; its blocks are $\mathcal{K}_{ij} = J_i J_j^\top$. The training loss is $\sum_i w_i L_i$ with weights $w_i > 0$, and $\mathcal{W} = \mathrm{blockdiag}_i(w_i I_{d_i})$. Linearised parallel gradient descent is $\theta_{m+1} = \theta_m - \eta \sum_j w_j J_j^\top r_j^{(m)}$ from θ_0 with $\eta > 0$, where $r_i^{(m)} = f_i^{\mathrm{lin}}(\theta_m) - y_i$; we write $r^{(m)}$ for the stacked residual, $L_i^{(m)} = \frac{1}{2}\|r_i^{(m)}\|^2$ and $e_i^{(m)} = \|r_i^{(m)}\|$. For every task and pair of tasks let

$$a_i = \lambda_{\min}(\mathcal{K}_{ii}), \qquad b_i = \lambda_{\max}(\mathcal{K}_{ii}), \qquad \bar{\lambda}_i = \mathrm{tr}(\mathcal{K}_{ii})/d_i,$$
$$c_{ij} = \|\mathcal{K}_{ij}\|_2, \qquad \rho_{ij} = \frac{\|\mathcal{K}_{ij}\|_F}{(\|\mathcal{K}_{ii}\|_F \|\mathcal{K}_{jj}\|_F)^{1/2}}, \tag{12}$$

the last one when $\mathcal{K}_{ii}, \mathcal{K}_{jj} \neq 0$. The scale $\bar{\lambda}_i \in [a_i, b_i]$ is the mean eigenvalue of the block of task i, and ρ_{ij} is the normalised coupling of tasks i and j.

Proposition 6 (Block dynamics, transfer and interference)**.** *In this setting:*

(i) $r^{(m+1)} = (I - \eta\mathcal{K}\mathcal{W})r^{(m)}$*, that is,* $r_i^{(m+1)} = r_i^{(m)} - \eta \sum_j w_j \mathcal{K}_{ij} r_j^{(m)}$*. With* $\mathcal{K}_{\mathcal{W}} = \mathcal{W}^{1/2}\mathcal{K}\mathcal{W}^{1/2}$ *and* $\tilde{r}^{(m)} = \mathcal{W}^{1/2} r^{(m)}$*,* $\tilde{r}^{(m+1)} = (I - \eta\mathcal{K}_{\mathcal{W}})\tilde{r}^{(m)}$ *and* $\sum_i w_i L_i^{(m)} = \frac{1}{2}\|\tilde{r}^{(m)}\|^2$*. Hence Proposition 3(i), (iii) and (iv) hold for the weighted loss with* $\mathcal{K}$ *replaced by* $\mathcal{K}_{\mathcal{W}}$ *and* r_0 *by* $\tilde{r}^{(0)}$*.*

(ii) For every task i *and step* m*, exactly,*

$$L_i^{(m+1)} - L_i^{(m)} = -\eta \sum_j w_j\, r_i^{(m)\top} \mathcal{K}_{ij} r_j^{(m)} + \frac{\eta^2}{2} \Big\| \sum_j w_j \mathcal{K}_{ij} r_j^{(m)} \Big\|^2. \tag{13}$$

Thus L_i *increases at step* m *whenever* $\sum_j w_j r_i^{(m)\top} \mathcal{K}_{ij} r_j^{(m)} < 0$*, and decreases whenever this sum exceeds* $\frac{\eta}{2}\|\sum_j w_j \mathcal{K}_{ij} r_j^{(m)}\|^2$*.*

(iv) (Layers.) In the layered setting of Proposition 4, let H_{l-1}^i *have rows* $h_{l-1}(x_{i,a})^\top$*, let* $r_{i,a} \in \mathbb{R}^{n_L}$ *be the block of* r_i *for input* $x_{i,a}$*, and let* E_l^i *have rows* $\delta_{i,a}^{l\top}$ *with* $\delta_{i,a}^l = \Delta_l(x_{i,a})^\top r_{i,a}$*, the derivative of* $\frac{1}{2}\|r_{i,a}\|^2$ *with respect to* z_l *away from kinks. Then* $\nabla_{W_l} L_i = E_l^{i\top} H_{l-1}^i$ *(the selected surrogate at kinks), and*

$$r_i^\top \mathcal{K}_{ij} r_j = \sum_{l=1}^{L} \big\langle E_l^i E_l^{j\top},\, H_{l-1}^i H_{l-1}^{j\top} \big\rangle = \sum_{l=1}^{L} \langle \nabla_{W_l} L_i, \nabla_{W_l} L_j \rangle,$$
$$|r_i^\top \mathcal{K}_{ij} r_j| \le \sum_{l=1}^{L} \|E_l^i E_l^{j\top}\|_F \|H_{l-1}^i H_{l-1}^{j\top}\|_F. \tag{14}$$

Each term is equation 4 of Proposition 1 for layer l*, with error rows* $\delta_{t,a}^l$ *and inputs* $h_{l-1}(x_{t,a})$*.*

Proposition 7 (Per-task rates and the implicit curriculum)**.** *In the linearised setting, let $\eta w_i b_i \le 1$ for every i. Let $C \in \mathbb{R}^{T\times T}$ have entries $C_{ij} = w_j c_{ij}$ for $j \ne i$ and $C_{ii} = 0$, and let $M = I - \eta\,\mathrm{diag}(w_1 a_1, \dots, w_T a_T) + \eta C$. Inequalities between vectors are entrywise.*

(i) For every i and m,

$$(1-\eta w_i b_i)e_i^{(m)} - \eta \sum_{j\ne i} w_j c_{ij} e_j^{(m)} \le e_i^{(m+1)} \le (1-\eta w_i a_i)e_i^{(m)} + \eta \sum_{j\ne i} w_j c_{ij} e_j^{(m)}. \quad (15)$$

(ii) M is entrywise non-negative and $e^{(m)} \le M^m e^{(0)}$. If $v \in \mathbb{R}^T$ has positive entries and $Mv \le \mu v$ for some $\mu \ge 0$, then $e_i^{(m)} \le \mu^m v_i \max_k e_k^{(0)}/v_k$ for all i and m. Such a v exists for every $\mu > \rho(M)$, the spectral radius of M; hence if $\rho(M) < 1$, every task converges geometrically at any rate $\mu \in (\rho(M), 1)$. In particular, if the blocks are diagonally dominant in the sense that

$$\gamma = \min_i \Big(w_i a_i - \sum_{j\ne i} w_j c_{ij} \Big) > 0, \qquad \textit{then} \qquad \max_i e_i^{(m)} \le (1-\eta\gamma)^m \max_i e_i^{(0)}. \quad (16)$$

Proposition 8 (Layerwise control of the blocks)**.** *In the layered setting of Proposition 4, let H_{l-1}^i have rows $h_{l-1}(x_{i,a})^\top$, $X_i = H_0^i$ and $\mathcal{G}_{l-1}^{ij} = H_{l-1}^i H_{l-1}^{j\top}$, so that $\mathcal{G}_{l-1}^{ii}$ is the realised feature Gram of task i.*

(a) For all tasks i, j,

$$\begin{aligned} a_i &\ge \textstyle\sum_l \lambda_{\min}(\mathcal{G}_{l-1}^{ii}) \min_a \sigma_{n_L}(\Delta_l(x_{i,a}))^2, \\ b_i &\le \textstyle\sum_l \lambda_{\max}(\mathcal{G}_{l-1}^{ii}) \max_a \|\Delta_l(x_{i,a})\|_2^2, \\ c_{ij} &\le \textstyle\sum_l \|\mathcal{G}_{l-1}^{ij}\|_2 \max_a \|\Delta_l(x_{i,a})\|_2 \max_b \|\Delta_l(x_{j,b})\|_2. \end{aligned} \quad (17)$$

(c) (Linear.) Under the hypotheses of Proposition 4(c), let $u_\varepsilon = (1+\varepsilon)^{L-1} - 1$ and $s_i^0 = \|X_i\|_F^2/n_i$. Then

$$L(1-\varepsilon)^{L-1}(X_i X_i^\top \otimes I_{n_L}) \preceq \mathcal{K}_{ii} \preceq L(1+\varepsilon)^{L-1}(X_i X_i^\top \otimes I_{n_L}), \quad (18)$$

so a_i, b_i and $\bar\lambda_i$ lie between $L(1-\varepsilon)^{L-1}$ and $L(1+\varepsilon)^{L-1}$ times $\lambda_{\min}(X_i X_i^\top)$, $\lambda_{\max}(X_i X_i^\top)$ and s_i^0 respectively, and for $X_i, X_j \ne 0$,

$$\Big(\frac{1-\varepsilon}{1+\varepsilon}\Big)^{L-1} \le \frac{\bar\lambda_i/\bar\lambda_j}{s_i^0/s_j^0} \le \Big(\frac{1+\varepsilon}{1-\varepsilon}\Big)^{L-1}. \quad (19)$$

For all i, j,

$$\begin{aligned} \|\mathcal{K}_{ij} - L\,X_i X_j^\top \otimes I_{n_L}\|_2 &\le L u_\varepsilon \|X_i\|_2 \|X_j\|_2, \\ \|\mathcal{K}_{ij} - L\,X_i X_j^\top \otimes I_{n_L}\|_F &\le L u_\varepsilon \sqrt{n_L} \|X_i\|_F \|X_j\|_F. \end{aligned} \quad (20)$$

With the input coupling $\rho_{ij}^0 = \|X_i X_j^\top\|_F / (\|X_i X_i^\top\|_F \|X_j X_j^\top\|_F)^{1/2}$ and $\xi_i = \|X_i\|_F^2/\|X_i X_i^\top\|_F \in [1, \sqrt{\mathrm{rank}\, X_i}]$, for $X_i, X_j \ne 0$,

$$\frac{\rho_{ij}^0 - u_\varepsilon\sqrt{\xi_i\xi_j}}{(1+\varepsilon)^{L-1}} \le \rho_{ij} \le \frac{\rho_{ij}^0 + u_\varepsilon\sqrt{\xi_i\xi_j}}{(1-\varepsilon)^{L-1}}. \quad (21)$$

For $\varepsilon = 0$, $\mathcal{K}_{ij} = L\,X_i X_j^\top \otimes I_{n_L}$ for all i, j, so $\bar\lambda_i = L s_i^0$ and $\rho_{ij} = \rho_{ij}^0$ at every depth. For non-zero $h, h' \in \mathbb{R}^{n_0}$ and $1 \le l \le L$, the cosine between $W_{l-1:1}h$ and $W_{l-1:1}h'$ differs from that between h and h' by at most $((1+\varepsilon)/(1-\varepsilon))^{l-1} - 1$.

Proposition 9 (The held-out block)**.** *Let u be a held-out task with support inputs, Jacobian J_u at the pretrained parameters θ_0, kernel $\mathcal{K}_{uu} = J_u J_u^\top \ne 0$ and cross blocks $\mathcal{K}_{uj} = J_u J_j^\top$. In (i)–(iii), adapt on u alone by linearised gradient descent (Proposition 3 with $\mathcal{K} = \mathcal{K}_{uu}$) with $0 < \eta \le 1/\lambda_{\max}(\mathcal{K}_{uu})$ and $r_0 \ne 0$, and let*

$$q_u = \frac{\eta\, r_0^\top \mathcal{K}_{uu} r_0}{\|r_0\|^2} \in [\eta\lambda_{\min}(\mathcal{K}_{uu}), 1]; \qquad \textit{at } \eta = 1/\lambda_{\max}(\mathcal{K}_{uu}), \quad q_u = \frac{r_0^\top \mathcal{K}_{uu} r_0}{\lambda_{\max}(\mathcal{K}_{uu})\|r_0\|^2}. \quad (22)$$

(i) $\ell_1 = (1 - d_0)\ell_0$ *with* $q_u \le d_0 \le 2q_u - q_u^2$.

(iii) $\ell_m \ge (1 - q_u)^{2m}\ell_0$ *for every* $m \ge 0$. *Hence, if* $0 < q_u < 1$, *reducing the loss by a factor* $\delta \in (0, 1)$ *takes at least* $\log(1/\delta)/(2\log(1/(1 - q_u))) \ge (1 - q_u)\log(1/\delta)/(2q_u)$ *steps, if that threshold is reachable. Proposition 3(iv) separately bounds the reducible loss from above using the smallest positive eigenvalue carried by* r_0 *and the chosen step size.*

B Protocol of the NineRooms experiments

Environment and agent. We use the NineRooms environment and the sequential protocol of the released Parseval code (Chung et al., 2024).[1] The grid has 15×15 cells divided into nine rooms, and the agent has four actions. The reward is -0.01 times the shortest-path distance to the goal at every step, plus 3 on reaching the goal. In the parallel pretraining of Section 6.5 the observation includes the goal, a goal is drawn for each episode, and 16 environments run in parallel. Actor and critic are separate networks with two tanh hidden layers of width 64. The PPO settings are those of the released code: 128 steps per rollout, 4 epochs, 4 minibatches, clipping 0.2, discount 0.99, GAE parameter 0.95, entropy coefficient 0.01, value coefficient 0.5, gradient-norm clipping 0.5, Adam $\epsilon = 10^{-5}$ and a constant learning rate.

Arms. Muon uses five Newton–Schulz iterations, the reference form of the released comparison code: bias-corrected momentum 0.95, the update of a matrix scaled by $\sqrt{\max(1, n_{\text{out}}/n_{\text{in}})}$, every two-dimensional kernel including the output layers, and NAdam with the same learning rate for the other parameters. All regularised arms use the released Parseval target. Every non-output dense kernel K is pulled towards $K^\top K = 2I$, and all hidden kernels start from an orthogonal initialisation with gain $\sqrt{2}$, so every arm starts at the target. Parseval adds $10^{-3}\sum_K \|K^\top K - 2I\|_F^2$ to the loss. AdamO and MuonO apply the decoupled step $K \leftarrow K - \eta c \nabla_K \frac{1}{2}\|K^\top K - 2I\|_F^2 = K - 2\eta c\, K(K^\top K - 2I)$ with $c = 0.1$ after each optimiser step, the same step as Equation (2) with target $2I$.

Seeds and compute. The depth-2 held-out experiment uses the seeds 960100 to 960163. All 64 seeds of one arm run as one vectorised batch on one GH200 GPU.

C Protocol and full results of held-out adaptation

We freeze one split of the goals into 20 training, 5 validation and 6 test goals, shared by all seeds. Each optimiser family selects its pretraining learning rate on separate calibration goals by the largest mean training success, with ties going to the smaller rate. Adam and AdamO use 10^{-4}, and Muon, MuonO and Parseval use 2.5×10^{-4}. The Parseval-against-Adam contrast therefore compares two learning rates. The adaptation procedure is selected once on the validation goals and is the same for every arm. It is Adam with learning rate 10^{-3}, entropy coefficient 0.03, fresh optimiser moments and no regulariser, run for $262{,}144$ transitions. The endpoint is the mean success on the six test goals over the whole adaptation (full AUC). The early AUC averages the first $65{,}536$ transitions.

D Depth-16 held-out adaptation: protocol

The depth-16 experiment uses the split, adaptation procedure and learning rates of the depth-2 protocol (Section C) without new selection: Adam, AdamO, L2-init, shrink-and-perturb and entropy-tuned Adam pretrain at 10^{-4}, and Muon, MuonO and Parseval at 2.5×10^{-4}. The plain network has 16 tanh layers of width 64 with orthogonal initialisation of gain $\sqrt{2}$, pretrained for $4{,}194{,}304$ transitions on 16 environments with entropy coefficient 0.01, except for entropy-tuned Adam. The primary contrast (Parseval against L2-init), the secondary contrasts and the seeds (990000 to 990063) were fixed in a pre-registration before submission, from a 16-seed pilot on validation goals, and the adaptation uses the six test goals. Contrasts marked exploratory in Table 1 were computed after

[1] `https://github.com/wechu/parseval_reg`; the paper describes the task as a 15×15 grid-world.

the run from the same data. Further depth-16 contrasts: shrink-and-perturb against Adam $+0.025$ $[-0.034, 0.085]$, Adam with entropy coefficient 0.1 against fresh -0.003 $[-0.023, 0.017]$, L2-init against fresh $+0.036$ $[0.016, 0.057]$. At initialisation the singular values of the Jacobian of the last hidden state with respect to the first (width 64, 3 seeds) lie in $[1.22, 1.41]$ for the plain network at depth 2, and in $[2.9 \times 10^{-5}, 30.1]$ at depth 16.

E Controls and robustness at depth 16

All runs in Table 2 were pre-registered before they were run and use the depth-16 protocol of Section D, 64 paired seeds and the six test goals, unless stated otherwise. An entropy-matched arm changes its entropy coefficient after every update by bounded feedback, so that its policy entropy follows the pretraining trajectory of the paired AdamO or MuonO run; an arm counts as matched if the root mean squared entropy difference is at most 0.03 nats. The first, integral controller met this criterion in 0 of 64 Adam seeds and 2 of 64 Muon seeds, mostly because of a lag in the first 0.5M transitions, although the final entropies match closely. A second, proportional-integral controller in log space, with gains selected on two separate calibration seeds, matches Muon in 64 of 64 seeds; Adam still meets the criterion in only 2 of 64 seeds, because early in pretraining its entropy stays 0.1 to 0.3 nats above that of AdamO even with no entropy bonus. Entropy-tuned Adam uses the coefficient 0.1, the value that every arm selected when the coefficient was tuned on the sequential stream. The depth sweep uses the plain network at $L \in \{2, 4, 8, 32\}$ in addition to $L = 16$; its prediction that the AdamO gain grows monotonically with depth does not hold. At width 256 every pretrained arm reaches a training success above 0.95; the retuned run selects each arm's pretraining learning rate on validation goals with 16 tuning seeds, from three to four values around the width-64 rate.

Table 2: Controls and robustness, NineRooms depth 16, 64 paired seeds, test goals. Paired differences in held-out full AUC with 95% bootstrap intervals; the depth sweep lists means for $L = 2, 4, 8, 16, 32$.

Control	Contrast	Full AUC difference
Entropy matching, integral	AdamO − matched Adam	$+0.088$ $[0.068, 0.110]$
	MuonO − matched Muon	$+0.053$ $[0.034, 0.074]$
Entropy matching, PI	AdamO − matched Adam (0.814)	$+0.084$ $[0.060, 0.113]$
	MuonO − matched Muon (0.846)	$+0.046$ $[0.030, 0.063]$
Entropy-tuned Adam (0.1)	AdamO − tuned Adam (0.824)	$+0.075$ $[0.053, 0.100]$
	tuned Adam − fresh	-0.003 $[-0.023, 0.017]$
L2-init at 2.5×10^{-4} (0.890)	Parseval − L2-init	$+0.026$ $[0.015, 0.037]$
	AdamO − L2-init	$+0.008$ $[-0.005, 0.021]$
Depth sweep	AdamO − Adam	$+0.47$, $+0.41$, $+0.39$, $+0.36$, $+0.64$
	MuonO − Muon	$+0.16$, $+0.13$, $+0.14$, $+0.37$, $+0.46$
	best regularised − best other	$+0.09$, $+0.05$, $+0.07$, $+0.05$, $+0.08$
	Parseval − fresh	$+0.03$, $+0.04$, $+0.10$, $+0.09$, $+0.12$
16.8M pretraining	AdamO (0.891) − fresh	$+0.064$ $[0.050, 0.079]$
	AdamO − L2-init (0.874)	$+0.016$ $[0.006, 0.027]$
	MuonO − L2-init	-0.018 $[-0.030, -0.005]$
Width 256, width-64 rates	AdamO − Adam	$+0.183$ $[0.148, 0.220]$
	Parseval − fresh	-0.076 $[-0.123, -0.030]$
	Parseval − L2-init, both 2.5×10^{-4}	-0.038 $[-0.090, 0.014]$
Width 256, retuned rates	AdamO − Adam	$+0.178$ $[0.127, 0.230]$
	best regularised − best other	-0.060 $[-0.102, -0.014]$

F A second task family: DoorKey with held-out layouts

Task family. We use `MiniGrid-DoorKey-5x5` and `-6x6` of XLand-MiniGrid (Nikulin et al., 2024). A task is one layout, fixed by a reset key. From a pool of 1,024 keys we hold out 5 validation and 6 test tasks, and each seed draws its 20 training tasks from the remaining 1,013. The observation

Table 3: Other settings in which we tested the regulariser, one row each. Each either lacks the loss of adaptability that the main claim concerns, so it cannot test it, or shows no gain beyond what entropy control gives. Intervals are 95% bootstrap intervals over paired seeds.

Setting	What was run	Outcome and reading
NineRooms, depth 2	Held-out protocol, two tanh layers (Chung et al., 2024), pretraining 0.26M to 16.8M, 64 seeds	At 4.19M AdamO − Adam $+0.478$ $[0.429, 0.523]$, MuonO − Muon $+0.128$ $[0.096, 0.160]$; entropy matching recovers $+0.411$ of it for Adam and all of it for Muon. Only MuonO exceeds a fresh network, and after 16.8M no arm does.
NineRooms stream	20-goal stream of Chung et al. (2024), 64 seeds	AdamO − Adam $+0.389$ $[0.365, 0.413]$, but entropy control alone gives Adam $+0.413$ $[0.390, 0.437]$.
PointRobot stream	Eight groups of goals in sequence, width 16, 72 seeds, pre-registered	Mean peak forgetting 0.178 (AdamO) against 0.046 (Adam), 0.368 (MuonO) against 0.040 (Muon).
Small multi-task fit	Multi-task CartPole with PQN (32 seeds), MinAtar, supervised teacher-student toy (16 seeds), CartPole blocks (16 seeds)	No gain in training fit; the toy is worse for every regularised arm. On CartPole blocks Muon gains more over Adam ($+0.170$ $[0.114, 0.226]$) than norm-preserving AdamO ($+0.066$ $[0.016, 0.115]$).
CartPole family	Signed-permutation CartPole, parallel pretraining, held-out tasks; pre-registered pilot, 16 seeds	Gate failed: Adam-pretrained networks adapt better than fresh ones (0.462 against 0.335). No loss of adaptability to prevent; AdamO − Adam $+0.027$ $[0.011, 0.047]$.
Brax HalfCheetah, Ant	Target velocity and direction tasks, depth 16 gate, 16 seeds, after two pre-registered learning qualifications	Gate failed: Adam-pretrained networks adapt far better than fresh ones ($+5.3$ $[4.5, 6.2]$ and $+9.4$ $[8.6, 10.4]$ normalised area).
MinAtar, held-out game	Five games, each held out in turn, depth 16 and 2, 16-seed pilot, pre-registered gate	Gate failed: pretraining did not reach the required returns. Pilot AdamO − Adam $+0.000$ $[-0.066, 0.070]$.
XLand-MiniGrid	RL^2 on `small-1m`, 10^{10} transitions, 2 seeds per arm; gradient-adaptation gate, 16 seeds	In-context return Adam 13.3, Muon 11.1, MuonO 10.8, AdamO 7.6. The gradient gate could not test the effect: Adam did not learn the training rulesets (return 0.013).
MetaMaze, in-context	Recurrent agent, four-episode trials, 64 and 1024 tasks, 8 to 32 seeds	The full regulariser lowers held-out in-context adaptation (level 3: AdamO 0.030 against Adam 0.275); restricted to the recurrent kernels it gives no gain.
Reptile meta-training	Five PPO inner iterations per outer step, same budget, depth 16, 16-seed pilot	Gate failed: meta-trained Adam does not lose adaptability (0.834 against 0.794 fresh); AdamO − Adam $+0.021$ $[-0.026, 0.068]$.

is the egocentric 7×7 view (one-hot tile and colour) and the direction. The score is the MiniGrid return, $1 - 0.9\, t/t_{\max}$ on success and 0 otherwise, averaged over 8 sampled-policy episodes per task. The layout variant uses the layouts only. The permuted variant adds a per-task permutation of the six actions and a 20-slot one-hot task ID, so held-out tasks conflict with pretraining (zero-shot score about 0.01). The 5×5 family has only 48 distinct layouts, so in the layout variant about a third of the test layouts also occur among a seed's training layouts; for 6×6 it is about 2%.

Gate. A cell passed if, at depth 16 on 16 seeds and validation tasks, the fresh network adapted at least 0.05 better than Adam in full AUC, Adam reached a training score of at least 0.85, and the fresh network reached a final held-out score of at least 0.6. We took the first passing cell in the fixed order Layout 5×5, Layout 6×6, Permuted 5×5, Permuted 6×6. Only Permuted 5×5 passed.

Contrasts. At depth 16: AdamO against Adam $+0.275$ $[0.205, 0.345]$ (primary), Parseval against fresh $+0.218$ $[0.172, 0.262]$ (primary), MuonO against Muon $+0.443$ $[0.377, 0.504]$, Parseval against Adam $+0.346$ $[0.276, 0.415]$; AdamO, MuonO and Parseval against entropy-tuned Adam $+0.170$ $[0.097, 0.243]$, $+0.244$ $[0.178, 0.312]$ and $+0.241$ $[0.170, 0.311]$, and against L2-init $+0.143$ $[0.069, 0.216]$, $+0.216$ $[0.153, 0.282]$ and $+0.213$ $[0.143, 0.282]$. At depth 2: AdamO against Adam $+0.241$ $[0.192, 0.292]$, Parseval against fresh $+0.076$ $[0.025, 0.127]$, MuonO against Muon $+0.063$ $[0.034, 0.093]$; AdamO, MuonO and Parseval against entropy-tuned Adam -0.093 $[-0.135, -0.051]$, $+0.033$ $[-0.014, 0.078]$ and -0.133 $[-0.180, -0.087]$, and against L2-init -0.054 $[-0.095, -0.014]$, $+0.071$ $[0.033, 0.106]$ and -0.095 $[-0.136, -0.054]$.

G Loss of adaptability and what the regulariser preserves: details

We analyse the weights handed to adaptation in the NineRooms depth-16 experiment (64 paired seeds, Section 6.1) on a probe of the 225 positions of each held-out goal; the analysis and its primary quantity, tanh saturation in layers 9 to 16, were pre-registered. Saturation is higher under Adam and Muon than under the regularised arms in every seed, but it is not the cause: at depth 2 Adam

loses the most adaptability with almost no saturation, and Adam at the larger learning rate is more saturated yet adapts better. Every Adam and Muon variant has a deep feature rank of 10 to 23 and a Gram defect of 0.23 to 0.44, against 37 to 48 and 0.02 to 0.05 for the regularised arms, with L2-init between (Figure 4). Against entropy-matched Adam, AdamO keeps a higher deep feature rank (37.0 against 18.2) and a smaller Gram defect (0.050 against 0.233), but also a higher held-out entropy (0.66 against 0.46), so these data cannot separate the two, and the evidence is correlational.

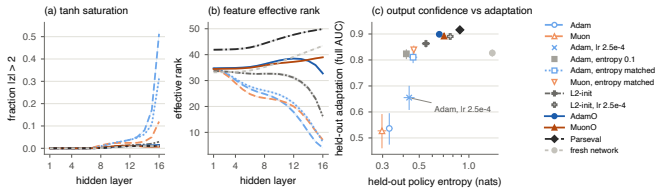

Figure 4: Feature rank and entropy give different accounts of held-out adaptation in NineRooms (64 seeds, weights at the start of adaptation, probe on the held-out goals). (a) Fraction of actor tanh pre-activations with $|z| > 2$ per hidden layer. (b) Effective rank of the features per layer. (c) Held-out policy entropy against full AUC, arm means with 95% bootstrap intervals. Adam at learning rate 2.5×10^{-4}, L2-init at 2.5×10^{-4} and the entropy-matched arms are exploratory controls.

What tracks the loss of adaptability. Over all 12 depth-16 arms, the held-out policy entropy and the logit scale have the strongest mean within-arm correlations with the full AUC ($+0.29$ $[0.22, 0.35]$ and -0.30 $[-0.37, -0.23]$), and these two are the only quantities that move with the full AUC in every control that restores the adaptability of Adam or Muon.

H MULTI-TASK KERNEL DURING DEPTH-16 PRETRAINING

Prediction 1: per-task scales (supported). At the end of pretraining the paired differences in the coefficient of variation of s_i over the 20 training goals are, for the actor, -0.239 $[-0.260, -0.218]$ (AdamO against Adam), -0.197 $[-0.213, -0.181]$ (MuonO against Muon) and -0.302 $[-0.324, -0.280]$ (Parseval against Adam), with intervals below zero also for the critic.

Low-reward goals. In a pre-registered experiment the reward of the first 10 of each seed's 20 training goals is multiplied by 0.1 during pretraining (64 new paired seeds); the held-out goals keep their reward.

At depth 16 the actor kernel scale of the starved goals ends at 0.734 $[0.677, 0.790]$ of that of the other goals under Adam and 0.800 under Muon, against 0.849, 0.873 and 0.909 under AdamO, MuonO and Parseval, so the regularised arms reduce the loss of kernel scale but do not remove it.

Number of training goals. Repeating the depth-16 runs with $T = 5$ and $T = 60$ training goals (64 seeds), the pre-registered prediction that the advantage of the regularised arms grows with T is falsified: it shrinks from about $+0.60$ at $T = 5$ to $+0.11$ at $T = 60$.