Selective Predictive Training and Structured Predictability in Neural Training Dynamics

A Dissertation Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

Fanqi Wang

August 2026

© by Fanqi Wang, 2026
All Rights Reserved.

Abstract

The training of modern deep neural networks remains governed by long sequences of iterative gradient-based updates, while most acceleration techniques reduce the cost of individual steps without changing this step-by-step structure. This dissertation investigates a different possibility: whether recent parameter trajectories contain short-horizon temporal structure that can be selectively exploited to replace expensive optimization steps and, independently, measured to reveal how training dynamics evolve within a model.

The first contribution is Predictive Differential Training (PDT), which combines Koopman/DMD-based weight forecasting with standard gradient-based optimization. Instead of applying forecasts uniformly, PDT uses a coordinate-wise mask based on scale control and dynamic consistency. Coordinates that pass this conservative reliability test advance through prediction, while the remainder continue under the baseline optimizer. This hybrid update preserves gradient-based correction as a safeguard while exploiting local forecast reliability. Across CIFAR-10 and ImageNet-1K classification, multiple architectures and optimizers, and a representative self-supervised setting, PDT reduces time to matched baseline training targets while maintaining comparable final performance.

The second contribution is a convention-aware, multiscale framework for measuring structured predictability in stored training trajectories. Complementary displacement-direction, subspace-residual, and predictor-based probes characterize short-horizon temporal organization across aggregate, tensor, and within-tensor

parameter scales. Applied to controlled multi-pass vision training and public one-pass Pythia pretraining trajectories, the framework reveals a recurring but non-uniform organization: auxiliary parameters generally exhibit simpler short-horizon dynamics, whereas bulk transformation tensors contain localized, time-varying predictable pockets. These patterns change with optimization position, architecture, and training recipe, and parameter-resolved measurements expose temporal events that are not reducible to a scalar loss curve. Because the framework operates retrospectively on sufficiently dense checkpoints, it also provides a general diagnostic for comparing how training conditions reshape local parameter dynamics.

The central conclusion is that predictability is not a global or stationary property of a training run, but a local and evolving property of its parameter dynamics. This reframes predictive acceleration from a whole-model forecasting problem into a problem of locating, validating, and exploiting transient short-horizon structure. Prediction is therefore useful only where such structure can be detected, protected by reliability safeguards, and converted into a net computational benefit.

Table of Contents

1 Introduction **1**

1.1 The Persistent Cost of Iterative Neural Network Training 1

1.2 The Open Question of Temporal Structure in Training Trajectories . 2

1.3 From Predictive Training to Selective Predictive Training 4

1.4 From Selective Prediction to Structured Predictability 5

1.5 Dissertation Contributions 6

1.6 Organization of the Dissertation 7

2 Foundations of Predictive Training and Neural Training Dynamics **8**

2.1 Iterative Training and Adaptive Update Rules 8

2.2 Using Training History 9

2.3 Koopman/DMD Foundations for Predictive Training 11

2.4 Structure in Neural Training Dynamics 13

2.5 Research Gap and Dissertation Positioning 15

3 Predictive Differential Training Guided by Training Dynamics **17**

3.1 Problem Formulation 17

3.2 Methodology 19

3.2.1 Koopman/DMD-Based Multi-Step Prediction 19

3.2.2 Selective Masking via Dynamic Consistency 21

3.3 Experimental Evaluation 24

3.3.1 Experimental Setup and Metrics 25

3.3.2 Cross-Architecture Evaluation: Convergence and Net Efficiency 26
3.3.3 Masking Necessity: Ablation of Selection Criteria 29
3.3.4 Plug-in Generality Across Optimizers 32
3.3.5 Generalization to Self-Supervised Learning 34
3.3.6 Mask Dynamics and Distribution 34
3.3.7 Sensitivity to Prediction Hyperparameters 36
3.4 Summary and Limitations 39

4 Measuring Structured Predictability in Neural Training Dynamics 41
4.1 From Selective Prediction to Predictability Measurement 41
4.2 Measurement Framework 43
4.2.1 Anchors, Histories, and Measurement Units 43
4.2.2 Displacement-Direction Probes 44
4.2.3 Subspace-Residual Probes 45
4.2.4 Predictor-Based Probes 47
4.2.5 Grouping Conventions and Reference Calibration 49
4.3 Empirical Analysis 50
4.3.1 Experimental Settings and Measurement Protocol 50
4.3.2 Probe Families Are Coherent but Non-equivalent 52
4.3.3 Predictability Is Spatially Localized Across Parameter Scales and Training Regimes 57
4.3.4 Predictability and Realized Optimization Dynamics 60
4.3.5 Training Conditions and State-Dependent Temporal Redundancy 64
4.3.6 Selective Prediction in a Higher-Confidence Parameter Region 69
4.4 Predictability Measurement as a Training-Dynamics Diagnostic . . . 71

5 Discussion, Scope, and Future Directions 73
5.1 Limitations 76
5.2 Future Directions 78

Bibliography **82**

Appendices **93**

A Supplementary Details for Predictive Differential Training **94**

A.1 Toy Example for Binary Differential Learning 94

A.2 Reduced DMD Implementation for High-Dimensional Weights 96

A.3 Convergence Analysis of the Masked Hybrid Update 98

A.4 Computational Complexity and Overhead Accounting 102

A.5 Runtime, Memory, and FLOPs Profiling 103

A.5.1 AlexNet on CIFAR-10 104

A.5.2 ResNet-50 on ImageNet-1K 104

A.5.3 ViT-Base on ImageNet-1K 107

A.5.4 Scalability Interpretation 108

A.5.5 FLOPs Analysis 108

A.6 Supplementary Robustness and Cross-Domain Results 110

A.6.1 Behavior Under Non-i.i.d. Training 110

A.6.2 Cross-Domain Evaluation on Natural Language Processing . . 112

B Supplementary Details for Structured Predictability Measurement **114**

B.1 Experimental Coverage, Protocol, and External Quantities 115

B.1.1 Vision Panel: Architectures, Optimizers, and Recipes 115

B.1.2 Additional Run Families 117

B.1.3 Pretraining Checkpoint Coverage 117

B.1.4 Measurement Clocks, Anchors, and Seeds 121

B.1.5 External State Quantities and Their Interpretation 122

B.1.6 Relation to Existing Coherence and Noise Measures 123

B.2 Measurement Conventions, Calibration, and Robustness 125

B.2.1 Measurement Units and Grouping Conventions 125

B.2.2 Predictor Constructions 130

B.2.3 Rank Selection . . . 131
B.2.4 Null Generation and the Calibrated Readout . . . 132
B.2.5 Numerical Conventions and the Sign Rule . . . 133
B.2.6 Grouping Sensitivity and Within-Family Checks . . . 134
B.2.7 Reporting Discipline . . . 142
B.2.8 Full Multi-Probe Agreement and Protocol Robustness . . . 143
B.2.9 Representative Multi-Seed Details . . . 147
B.3 Auxiliary–Bulk Attribution and Cross-Regime Mapping . . . 149
B.3.1 The Partition Rule . . . 149
B.3.2 Vision Panel: Attribution and Dual-Definition Audit . . . 150
B.3.3 Pythia: Segment Taxonomy and the Vector Segment . . . 151
B.3.4 Cross-Regime Comparability . . . 154
B.4 Spatial and Temporal Localization of Predictable Pockets . . . 155
B.4.1 Additional Vision-Panel Heterogeneity Diagnostics . . . 155
B.4.2 Pocket Trajectories, Persistence, and Overlap Registrations . . . 161
B.4.3 Pythia Scale Extensions . . . 182
B.5 Additional Architecture-Conditioned Diagnostics . . . 193
B.6 Additional Optimizer-Conditioned Diagnostics . . . 197

Vita **201**

List of Tables

3.1 Runtime comparison. FCN and AlexNet are trained on a single NVIDIA RTX A6000 GPU, while ResNet-50, ViT-Base, and ViT-Huge are trained on three NVIDIA H100 (80 GB) GPUs. For each model, PDT and its paired baseline use the same model-specific optimizer and training configuration. 27

3.2 Comprehensive masking ablation results on AlexNet trained on CIFAR-10. Success rates are shown as successful/total runs. Final accuracy is reported as mean ± std across five seeds, using successful runs only. Accuracy values at chance level (10% on CIFAR-10) indicate runs that failed to learn. 30

3.3 Impact of baseline optimizers on PDT performance. Trained on CIFAR-10 using AlexNet, batch size 256, momentum 0.9 where applicable, with a cosine-annealing learning-rate scheduler. 33

3.4 Performance comparison of SimSiam pre-training on CIFAR-10 with a ResNet-18 backbone, trained for 200 epochs, learning rate 0.03, batch size 256, momentum 0.9, with a cosine-annealing learning-rate scheduler. 34

3.5 Mask acceptance ratio and validation accuracy across learning rates. Mask acceptance ratio is reported by training stage and final test accuracy is reported as mean ± std across three seeds. Training stages: Early (epochs 5–20), Mid (epochs 21–40), Late (epochs 41–60). 37

4.1 Validation-selected PDT comparison on CIFAR-10 over three seeds under pure SGD. PDT-small restricts forecast eligibility to auxiliary parameters while retaining the Chapter 3 predictor and coordinate-wise PDT acceptance mask. 70

A.1 Memory and runtime overhead of PDT compared to the baseline. AlexNet is trained on CIFAR-10 using SGD with batch size 256, learning rate 0.05, and a cosine-annealing learning-rate scheduler. All experiments use a single Nvidia RTX A6000 GPU. 105

A.2 Memory and runtime overhead of PDT compared to the baseline across batch sizes. ResNet-50 is trained on ImageNet-1K using SGD with momentum, learning rate 0.1, momentum 0.9, and a cosine-annealing learning-rate scheduler. All experiments use a single Nvidia H100 GPU. 106

A.3 Memory and runtime overhead of PDT compared to the baseline. ViT-Base is trained on ImageNet-1K using AdamW with batch size 256, learning rate 0.0006, weight decay 0.05, and a cosine-annealing learning-rate scheduler. All experiments use a single Nvidia H100 GPU. 107

A.4 Performance and runtime comparison between SGD and PDT under i.i.d. and non-i.i.d. training data distributions. AlexNet is trained on CIFAR-10 with batch size 128, learning rate 0.05, and a cosine-annealing learning-rate scheduler. 110

A.5 Performance comparison on the AG News text-classification task using a deep LSTM model. Training is performed on a single Nvidia RTX A6000 GPU for 30 epochs with learning rate 0.1, batch size 128, and a cosine-annealing learning-rate scheduler. 112

B.1 Vision panel: baseline recipe, optimizer definitions, and per-cell learning rates. **(a)** the shared baseline recipe used unless a family-specific delta is stated; **(b)** the six training configurations of the main panel; **(c)** the five architectures with their parameter counts and the learning rate of each cell. 116

B.2 Run families of the vision corpus. The panel is the set analyzed throughout Section 4.3.2 to Section 4.3.4; the Muon arm and the dose ladders supply the controlled comparisons of Section 4.3.5; the remaining families support the robustness and calibration checks. The two collapsed momentum runs are kept on disk and excluded from every readout. 118

B.3 Pretraining checkpoint coverage. 119

B.4 Grouping conventions on the vision panel. D is the number of scalars per group. "Immune" means the membership-resample null passes θ=0.7 at a rate near zero, so the raw pass fractions carry a negligible chance contribution. The flat K=16 protocol is the one cut where a large share of raw passes is chance, which is what motivated the calibrated readout $\mathrm{LRGF}_{\mathrm{cal}}$. Levels are not comparable across cuts, since group sizes and chance floors differ, so every level claim in this paper names its cut. Structural and magnitude readouts are pooled over the 18 recipe cells of the group-level lanes and anchors 5 to 40. . 126

B.5 Grouping conventions on the Pythia side. All Pythia readings are registered as correspondences and are not used to certify one another. Column conventions follow Table B.4. The last two rows are the recut used for the granularity ladder in Appendix B.2.6. 128

B.6 Partial-anchor Spearman correlations across DMD history length n_{past}. The $n_{\mathrm{past}} = 3$ column is an estimator-boundary case, not successful robustness evidence. Valid robustness claims use $n_{\mathrm{past}} \in \{5, 7, 10\}$. . . 146

B.7 Range of partial-anchor Spearman correlations across 17 exclusion patterns. Strong DMD-free and DMD agreement pairs remain stable; subspace-related pairs remain sensitive or weak. 146
B.8 Robust magnitude comparison between DMD and linear trend. Relative errors are summarized in log10 space to avoid domination by raw blow-up values. 146
B.9 DMD win counts under robust metric definitions. The both-in-range comparison is conditional on both predictors having clipped log10 relative error in range. 147
B.10 Representative multi-seed stability. Values are mean Spearman correlations over seed pairs on the four-configuration by five-anchor subset. Code names follow the implementation used for the measurement panel: `inv_perp_fut` is $1 - \mathrm{FRR}$; `frac_pred_lt07` is LRGF with threshold $\theta = 0.7$; `cos_inertia`, `cos_linear`, and `cos_DMD` are cosine agreements between inertia, linear-trend, and DMD predictions and the realized τ-step displacement; and `lift_linear` and `lift_DMD` are the corresponding selectivity lifts. `pure_sgd` and `cos_1_tau` are the implementation names for the directional-persistence probes reported in Chapter 4. 147
B.11 Final test accuracy across seeds for the representative multi-seed subset. 148
B.12 Anchor-averaged bulk probe values across seeds for the representative subset. Values are seed 42 / seed 43 / seed 44. 148

B.13 Auxiliary–bulk attribution and dual-definition audit on the vision panel. **(a)** Per-model tensor attribution under the operational split, with the composition of the auxiliary class by coarse role (ln/bn/bias/tok; the single defector appears as *linear*). **(b)** Agreement between the morphological and operational definitions at three audit levels; the complete defector list is the single tensor discussed in Appendix B.3.2. Definitions are given in Appendix B.3.1. Data: the tensor-level table of the canonical group-level panel, with attribution fields constant across anchors and optimizer cells. Parameter counts cover the parameter vector only, with no batch-normalization running statistics. 150

B.14 Pythia segment scheme and its mapping to the auxiliary–bulk partition, with shapes and parameter counts at 70M. 152

B.15 Pocket stability across training regimes. Top: Jaccard overlap of pocket membership between consecutive anchors (k=1, median over anchors), at the group-level unit defined at the start of Appendix B.4.2. Bottom: residence run lengths on the pretraining side, counting consecutive anchors a group spends inside the pocket set, with n the number of runs. Stability is stratified by segment and the stratification pattern differs between the two regimes, so a blanket ordering between them does not hold. At k=1 one pretraining anchor spans about 0.8% of training against about 2% for one vision epoch, so the sparser vision sampling mechanically depresses its values; the transient MLP contrast runs against that bias and is the most robust entry. These overlaps are registered as descriptions of pocket dynamics under the two regimes and are not treated as cross-regime confirmation of a shared mechanism. 165

B.16 Row-level emergence readings across five pretraining scales, with the onset criteria and the per-segment conventions given in the table footer. 166

B.17 Registry of measured overlaps between pocket membership and indicators defined outside the subspace-residual probe family. Each row records the direction, the key readings, the null convention, and the caveats that travel with it. The rows are not treated as evidence for one another, and none of them supports a statement about what pocket membership represents. 175
B.18 Cross-scale readings. 183

List of Figures

3.1 Motivation for selective predictive training. Non-selective prediction may remain stable in simpler models but becomes unreliable as model complexity increases. Coordinate-level examples further show that the same forecasting mechanism can produce high-fidelity predictions for some parameters and low-fidelity predictions for others. 18

3.2 Illustration of the PDT framework and the detailed mechanism for a τ-step prediction, where qualified predicted weights (red) and standard optimizer-derived weights (blue) are integrated to accelerate training; the selection mechanism that determines which predicted weights are qualified is detailed in Section 3.2.2. 22

3.3 Analysis of PDT's optimization and generalization efficiency. Trained on CIFAR-10 using AlexNet, batch size 256, learning rate 0.05, with a cosine-annealing learning-rate scheduler. 27

3.4 Performance comparison between baseline optimization and PDT. . . 28

3.5 Mean training curves with standard-deviation shading for baseline optimization and PDT variants with different masking criteria. Trained on CIFAR-10 using AlexNet, batch size 256, with a cosine-annealing learning-rate scheduler. Only successful runs are included. 31

3.6 Temporal evolution of masking-criteria contributions during training. Results are from full PDT with both criteria applied. Trained on CIFAR-10 using AlexNet, learning rate 0.05, batch size 256, with a cosine-annealing learning-rate scheduler. Only successful runs are included. 32
3.7 Analysis of mask distribution in AlexNet. (a) Layer-wise mask evolution over training epochs. (b) Comparison of prediction ratios between convolutional and fully connected layers. 35
3.8 Influence of PDT prediction hyperparameters. (a) Prediction horizon, (b) prediction interval, (c) starting epoch, and (d) past snapshot count. Trained on CIFAR-10 using AlexNet, batch size 256. 37
3.9 Mask acceptance ratio evolution across different learning rates (mean ± std shading across three seeds). Trained on CIFAR-10 using AlexNet, batch size 256, with a cosine-annealing learning-rate scheduler. 38
3.10 Temporal evolution of masking-criteria contribution across learning rates. Trained on CIFAR-10 using AlexNet, batch size 256, with a cosine-annealing learning-rate scheduler. 38

4.1 Within-family agreement on the controlled CIFAR-10 vision panel. (a) ESA versus group-mean VCS; (b) out-of-window versus in-window FRR; and (c) SESA under DMD and linear-trend selectors. Circled points indicate cells flagged by the near-zero-displacement diagnostic. 54
4.2 Cross-family relations on the controlled CIFAR-10 vision panel. (a) ESA versus out-of-window FRR; (b) out-of-window FRR versus DMD direction cosine; and (c) ESA versus DMD selectivity lift. Circled points indicate cells flagged by the near-zero-displacement diagnostic. 55

4.3 Auxiliary–bulk separation across training regimes. Each line connects matched auxiliary and bulk summaries within one CIFAR-10 model–optimizer cell or one displayed Pythia trajectory. Circles denote CIFAR-10 cells and stars denote representative Pythia trajectories. Cross-regime comparisons concern within-trajectory ordering rather than absolute probe values. 58

4.4 Distribution of FRR excess over the approximate isotropic reference for structurally grouped bulk parameters. The retained history rank is $r = 5$; approximately 6% of pooled group–anchor observations exceed the reported margin of 0.3 under this grouping and reference convention. 59

4.5 Predictability measurements in relation to realized optimization motion in multi-pass CIFAR-10 training. (a) ESA versus movement directedness, defined as net displacement divided by path length over the future window. (b) ESA versus the loss-side gradient-coherence estimate. Horizontal lines mark the random sign-agreement level of 0.5. 60

4.6 Pythia-70M case study. (a) ESA by parameter role; (b) released training loss and its smoothed trajectory; (c) fraction of *qkv* row groups satisfying FRR < 0.7; and (d) head-aligned *qkv* $\mathrm{LRGF}_{\mathrm{cal}}$ excess. Vertical lines mark the early fading of head-aligned excess, the loss minimum, and the later *qkv* ESA excursion. 62

4.7 Size-controlled architecture comparison under SAM. Curves report canonical bulk ESA across ten anchors for ViT-tiny, ViT-mid, and the parameter-matched CIFAR ResNet. The dashed line marks the random sign-agreement level of 0.5. 65

4.8 Optimizer-family profiles for ViT-mid aligned by training loss. The horizontal axis is reversed, and the vertical axis reports canonical bulk ESA. Curves are shown only over the loss range shared by the displayed optimizers; the dashed line marks the random sign-agreement level of 0.5. 66

4.9 Controlled weight-decay and momentum dose ladders for ViT-mid. Panels report (a) bulk ESA, (b) bulk VCS, and (c) bulk FRR against reversed training loss under the canonical $K = 16$ vision grouping. Each arm changes only the indicated weight-decay or momentum value relative to plain SGD, and curves are compared over their overlapping loss ranges. 67

4.10 Multi-seed online PDT comparison under pure SGD for ViT-tiny and ViT-mid. Curves show seed means with ± 1 standard-deviation bands. 70

A.1 Binary differential-learning trajectory for the toy problem in Eq. A.1. Only the x and y dimensions are shown. The selected subset (x, y, z) uses a learning rate three times larger than the baseline rate, while the remaining variables use the baseline rate. 95

A.2 Performance comparison between standard gradient descent and PDT on the toy optimization problem. Standard gradient descent reaches the threshold in 53 steps, while PDT reaches it in 27 steps. (a) Optimization trajectories in the x-y plane. (b) Loss values during optimization. 96

A.3 Performance comparison between baseline optimization and PDT, with (a) epochs and (b) accumulated TFLOPs as the horizontal axis. AlexNet is trained on CIFAR-10 with batch size 256, learning rate 0.05, and a cosine-annealing learning-rate scheduler. 109

A.4 Performance comparison between SGD and PDT under i.i.d. and non-i.i.d. training data distributions. AlexNet is trained on CIFAR-10 with batch size 128, learning rate 0.05, and a cosine-annealing learning-rate scheduler. Shaded regions indicate standard deviation across five random seeds. 111

A.5 Performance comparison between baseline SGD and PDT on the AG News text-classification task using a deep LSTM model. Training uses a single Nvidia RTX A6000 GPU for 30 epochs, learning rate 0.1, batch size 128, and a cosine-annealing learning-rate scheduler. Shaded regions indicate standard deviation across three random seeds. 113

B.1 Checkpoint and anchor coverage across the pretraining scales. Each row is one model scale on a shared training-step axis, with tokens on the top axis at 2,097,152 per step. Ticks give the 154 public checkpoint revisions, released on the schedule $0, 1, 2, 4, \ldots, 512$ and every 1k steps thereafter up to step 143k. Bars give the anchor domain actually measured, 123 anchors on a 1k grid spanning 7k to 129k for every scale. The domain is trimmed at both ends for protocol reasons rather than for data availability: the head drops the warmup phase and leaves room for the backward history window, and the tail reserves a complete future window for the longest horizon ($143\text{k} - 14\text{k} = 129\text{k}$ at τ=14). Lines give the span of public training-loss telemetry, and the two smallest scales carry no line because no public run exists for them. Triangles mark splices where the released telemetry is assembled from more than one run. Hatching on the 1B row marks the step-116k checkpoint defect and the anchors masked because of it, with short ticks at the horizon-specific future endpoints; the telemetry line and the official evaluations are unaffected by the masking. 120

B.2 Probes that use no grouping convention, against the flat readout (bulk scope; one line per run, 8 matched ViT-mid and ResNet pairs; the long-cosine ViT-mid run extends to epoch 90). On all three cut-free probes, (a) element-level ESA, (b) tensor-level VCS, and (c) tensor-level FRR, where lower means more predictable, ViT-mid reads as more predictable than ResNet at every anchor, while (d) the flat K=16 LRGF reference orders the same runs and anchors the other way (7 to 8 of 8 pairs). Canonical protocol (n_{past}=5, τ=5, θ=0.7); dashed grey lines mark chance (ESA 0.5, VCS 0). 135

B.3 Permutation-null decomposition of the flat K=16 LRGF (bulk scope; one line per run): (a) structural blocking, (b) the membership-permutation null (mean of 5 draws, shading gives min to max), and (c) their difference Δ, which is the intra-block coherence contribution. Permutation destroys 66 to 87% of ResNet's flat readout and the destroyed part localizes on the 3×3 kernels (1×1 shortcut convolutions: $\Delta \approx .003$), whereas 82 to 94% of ViT-mid's readout survives and the architecture ordering flips to ViT-mid at or above ResNet on the null column (24/24 pairs). A flat-LRGF level is therefore a statement about a grouping convention as well as about an architecture. 136

B.4 Structural-axis cuts under the calibrated readout: excess = $\text{LRGF}_{\text{cal}} - \alpha$ (α=0.05, so zero is chance), median over runs. (a) ViT-mid: physically defined groups surface above-chance structure that the flat axis does not resolve, with whole patch filters reaching .61 and head-level qkv groups .21 at anchor 45, while slab and projection cuts sit near chance. (b) ResNet: whole 3×3 filters reach .55 to .62 and 1×1 rows .18. Each cut carries a size-matched membership-permutation null that preserves the group-size multiset, and groups use only pre-anchor information. Group size differs across cuts, so levels are comparable only through the calibrated excess. 137

B.5 Pocket density by grouping cut and architecture on the vision panel: share of groups with FRR < 0.7, pooled over the 18 recipe cells and anchors 5 to 40 of the group-level lanes (bulk tensors, out-of-window target). The structural cuts and the magnitude cut (top three of 16 causal bins) are chance-immune at θ=0.7, so the chance contribution to their bars is negligible. The flat K=16 bars are pooled over the same 18 cells weighted by group count and are stacked: the hatched portion is the measured chance-pass fraction under the membership-resample null (6.0, 15.5, and 26.6 pp of the 23.5/18.6/29.8% totals for ResNet, ViT, and ViT-mid). Most of the ViT-family flat readout is chance passing, and most of the ResNet flat readout is intra-kernel coherence. Group size differs across cuts, so levels are comparable within a bar group only in the chance-immune sense. 139

B.6 Magnitude-stratified cut at anchor 45: groups are 16 equal-width bins of the per-tensor pre-anchor $\|\Delta w\|_2$ rank (bin 0 weakest, bin 15 strongest), reported as excess over chance, median over runs; the floor at -0.05 means no group passes. (a) ViT-mid feature-dimension tensors show a monotone concentration signature (top bins .38 to .66), while the patch embedding stays at the floor because its structure lies on the spatial axis (Figure B.4). (b) ResNet reads near zero under the same cut. This is the strongest single-axis sorter on the ViT side, and its architecture contrast runs opposite to the flat readout. 140

B.7 Spectrum diagnostic at the late anchors (anchor 45 on the vision panel, step 70k on Pythia): top-five eigenvalue mass of the coordinate-trajectory correlation spectrum per tensor family, against a simulated independent null of matching shape (grey band gives min to max over 5 draws), median over runs. Every family of all three architectures exceeds the null by two to four times, so co-moving structure is present in every family; which grouping axis can access it is what differs by architecture. 141

B.8 Full 9×9 pairwise partial-anchor Spearman heatmaps for the probe panel. The heatmap records the full multi-probe structure behind the representative agreement pairs in Chapter 4. It shows both strong agreement among directional, extrapolation, and DMD probes and nontrivial decoupling of subspace and selector-lift probes. 144

B.9 Leave-one-out and exclusion sensitivity for representative probe pairs. Full-data correlations are shown as points and exclusion ranges as intervals. The main DMD-free and DMD directional correlations remain stable, while subspace-related pairs are more sensitive and should be interpreted as diagnostic rather than primary evidence. . . 144

B.10 Protocol sensitivity of representative probe-pair agreement over group size K, horizon τ, and DMD history length n_{past}. The $n_{\text{past}} = 3$ column is an estimator-boundary case and is not used as successful robustness evidence. DMD-free versus DMD directional and selectivity agreement is stable for valid short-horizon settings, while bulk subspace-related pairs are more protocol-sensitive. 145

B.11 DMD value-add over linear-trend extrapolation under the canonical protocol. Direction-only and direction-plus-clipped-magnitude win rates support DMD as a useful predictor-based probe beyond a simple baseline. The magnitude-only both-in-range column is conditional and should not be read as a full-panel win rate. 145

B.12 Where the auxiliary–bulk partition sits in the architecture. One box per parameter tensor (LayerNorm boxes carry weight and bias jointly; linear biases are drawn as chips on their matrix box), with fill colour giving the partition class: orange for auxiliary, which is the vector segment and covers all vector-like parameters including the final LayerNorm; blue for bulk, the weight matrices (qkv, `attn_out`, MLP); hatched grey for tensors excluded from the aggregation. **(a)** Pythia-70M: one parallel-residual block, repeated six times, plus the top level; the two untied embedding matrices are their own segments and are excluded. The marked bias pair (≡) coincides under the parallel residual, so both rows are kept in the panel while aggregates count the pair once. **(b)** ViT-mid: sequential residuals, repeated seven times; here the token and positional embedding vectors are vector-like and therefore auxiliary, which is the one cross-regime difference from (a), and the ten-parameter head bias (†) is below the K=16 grouping minimum and is the only tensor excluded on the vision side. Shapes are the 70M reference. 153

B.13 Cross-probe support for the auxiliary–bulk contrast. Panel A reports the fraction of matched evaluations where the auxiliary score exceeds the corresponding bulk score. Panel B reports the distribution of $\Delta = \text{probe}_{\text{aux}} - \text{probe}_{\text{bulk}}$. Scores are oriented so that larger values indicate stronger predictability. Embedded panel labels using "specialized/auxiliary" refer to the same auxiliary class. 156

B.14 Representative anchor-wise DMD-lift profiles. The curves illustrate recurring temporal shapes, including early peak and decay, persistent or mid-high profiles, late emergence, and disrupted-low behavior. They are illustrative examples rather than a formal taxonomy; the full small-multiple grids are shown in Fig. B.15. 156

B.15 Full temporal grids for DMD-selector lift. These small multiples extend Fig. B.14 by showing anchor-wise profiles across model, optimizer, and role combinations. 157
B.16 Full temporal grids for `pure_sgd`, a predictor-free directional diagnostic. These curves show that temporal heterogeneity is also visible without fitting or applying DMD, although exact role and scope orderings can differ across probes. 157
B.17 Predictor-free directional support for spatial and temporal heterogeneity. The top panels repeat the scope-gap analysis using `pure_sgd`; the lower panels show representative temporal profiles by role. This is corroborating evidence that spatial and temporal heterogeneity are not visible only through DMD-selector lift. 158
B.18 Probe-dependent caveat on AlexNet. Under DMD-based `lift_DMD`, AlexNet still shows an auxiliary-over-bulk majority pattern, while `pure_sgd` can invert the ordering. This boundary case supports a careful interpretation: the evidence is for structured heterogeneity, not for a universal scope ordering that is invariant to probe and architecture. 159
B.19 Representative multi-seed check for spatial and temporal heterogeneity. The tested three-seed, four-cell subset preserves the main auxiliary–bulk selector-lift gap and temporal profiles within the subset. This is a representative check against seed-specific artifacts, not an exhaustive multi-seed sweep. 159
B.20 Full within-bulk pocket grid across the four main models. This is the expanded version of the within-bulk pocket analysis in Chapter 4, showing per-layer and per-optimizer predictable-group fraction and per-layer FRR spread. It is retained as supplementary evidence because the full grid is too dense for the main text. 160

B.21 FRR-spread diagnostics for the two representative within-bulk cases shown in Chapter 4. Horizontal bars show the p10–p90 range of FRR across layers within each role, and filled markers show the p50 median. These diagnostics complement the LRGF view by showing that bulk predictability is localized and role-dependent rather than uniform. . . 161

B.22 Row-level pocket emergence across four pretraining scales (qkv). Every panel shows the fraction of rows with FRR < 0.7 resolved by depth and training step, where one row is one W_{qkv} row ($D = d_{\text{model}}$); at these group sizes the membership-resample pass rate at θ=0.7 is near zero, so the fraction is read as an absolute population rather than as excess over a null. Panels (a) to (d) share one colour scale and one colour bar, so the four sizes are directly comparable; the pooled maximum is 0.4323 (410M layer 21 at 116k). Dashed vertical lines mark the row-level pocket onset, the earliest anchor at which some layer stays above 5% for three consecutive anchors of the 4k scan grid: 20k at 70M and 160M, 48k at 410M, and none inside the window at 1.4B, whose layer-15 population is still rising at the edge. Rings mark the maximum over layer and anchor. Panel (e) shows the carrier layer of each size on the same step axis with the 5% criterion as a reference line. The 1B series is excluded from this figure on the grounds registered in Appendix B.4.3, and its row-level fraction stays at or below 0.33% throughout in any case. All entries are single-run readings on the public checkpoint series, registered as descriptions of those runs; no ordering between the row-level onset and the pool-coherence onset is asserted here. 167

B.23 The same row-level population on three segments. (a) peak fraction of rows with FRR < 0.7, maximized over layer and anchor, for qkv, the attention output projection, and the MLP output half at four sizes, with the bar label giving the carrier layer; row scans of the attention output projection exist for 70M and 160M only, and the missing cells are drawn as explicit markers rather than as gaps. (b) and (c) trajectories of the carrier layer of each segment at 70M and 160M. Two conventions travel with the figure: the attention-output granularity is less clean than qkv's, since groups are flattened on $(d_{\text{model}}, d_{\text{head}})$ with d_{head} as the unit and 70M reads 19.3% at D=64 against 13.1% at D=512; and group size differs between segments, so levels are compared within a segment across sizes rather than across segments. The 1B series is excluded on the grounds registered in Appendix B.4.3. 169

B.24 Where the ∼20k *qkv* event sits, in the measurement or in its null. (a) and (c) layer-level calibrated readout for *qkv* (τ=5, raw) at 70M and 160M on a common scale. (b) and (d) the same lane read as two curves: the measured median FRR of those layers (solid) and the pooled membership-resample q_{05} (dashed), with the FRR axis inverted so that upward means lower FRR and the null pool merged across layers per segment and group size. At 70M the pocket set collapses from 51 groups to 3 between anchors 19k and 20k and the layer-level median calibrated readout falls from .354 to .000, while the measured median FRR of the same groups is unchanged (.9959 to .9952). What moves is the null, whose q_{05} falls from .9937 to .7201 and stays at .59 to .72 through 25k. The calibrated readout is a difference between a measurement and its null, so the event is located on the null side of that difference and is not described here as a change in the predictability of the groups. The 160M null line descends in two stages, a moderate descent at 13k and a deep one over about 72 to 76k, the latter in the neighbourhood of that run's smoothed loss minimum (77.3k); the time co-location with the loss landmark is registered as a correspondence only. 171

B.25 Turnover of the pocket set itself, at group level, for the one-pass 70M run (top row, dashed vertical at the 19 to 20k anchor pair) and the two multi-pass vision runs (bottom row). Columns: set size on a log scale, Jaccard against the previous anchor, and Jaccard against the final anchor. On *qkv* the set collapses from 51 groups to 3 and then holds 2 to 11 groups, with Jaccard against the final set at or below .06 before 48k, rising to .5 to .8 over 52 to 70k, while the adjacent-anchor Jaccard after the collapse has median .88. The attention output projection never stabilizes. The multi-pass runs show no event of this kind, with counts growing monotonically and the Jaccard against the final set rising steadily, which is cumulative expansion rather than replacement. This panel sets the reading conditions for the two-anchor movement overlaps: because the pocket set changes both size and membership between anchors, an anchor-to-anchor difference in overlap contains both a change in overlap and turnover of the set, and the two are separated only by reporting them side by side. 173

B.26 Overlap between pocket coordinates and the top-K set of the signed norm-change proxy, labelled movement in the panels, registered against an equal-cardinality hypergeometric null. The proxy $S_i = \sum_t w_i(t)\Delta w_i(t)$ is built from checkpoint differences alone; preconditioning is not removed and S is not a curvature-based importance measure. (a) observed precision of the top-K movement set restricted to pocket coordinates, against the null K/N and its 95% interval, with the pretraining rows annotated by the ratio. (b) quantile position of the pocket coordinates inside their family's movement distribution, with the diagonal as the no-association reference; the qkv set at 129k sits at median quantile .93 against about .50 for the vision reference, so the whole set is displaced rather than a tail of it. Pocket sets are the group-level flag at τ=5, raw, head granularity. The two anchors carry different pocket sets in both size and membership, so a two-anchor comparison mixes a change in overlap with turnover of the set itself. Each reading is a registration of a measured overlap and is not used to support any statement about what pocket membership represents. . . 178

B.27 The same overlap resolved anchor by anchor, with the pocket set rematched at every anchor. The readout is the ratio of the observed overlap to its hypergeometric expectation, with the grey band giving the null 95% interval. Two time conventions for the movement proxy are reported side by side: one accumulates from the start of training, the other over the last five checkpoint or epoch pairs, matching the probe's history window. (a) *qkv*: the ratio hugs the random line over 7 to 19k, steps to 3.77 at 20k in one anchor, and stays high for the rest of training. (b) the attention output projection, on small sets with heavy turnover, where the two time conventions disagree and both are reported. (c) ViT-mid patch filters sit slightly above the band early and inside it from epoch 23 apart from single-epoch excursions; the late approach to one has a mechanical component, since the cut approaches saturation. (d) ResNet 3×3 sits below the band over epochs 9 to 16 and above it from epoch 17. Every curve is a registration of a measured overlap over time. 179

B.28 Normalization gain outliers and per-coordinate dynamics on the same gain trajectories. A coordinate is an outlier when the robust z of its absolute gain at the late anchor exceeds 3. (a) and (b) per-tensor medians for outlier against non-outlier coordinates across the 13 pretraining gain vectors: median ESA is higher for the outliers in 12 of 13 tensors and median drift coherence in 11 of 13, with connectors marking the tensors where it is not. The final normalization tensor is saturated, with agreement and coherence equal to 1.0 on all 512 coordinates, so it carries no resolution and is drawn only for completeness. (c) rank correlation between the outlier score and coherence per tensor, positive in 11 of 13 pretraining tensors and negative in two. On the ViT-mid control one chain reads with the same sign and the others with the opposite sign; its gain range is narrow and its outlier sets hold at most four coordinates. A top-1% control cut reproduces every direction in both regimes. 181

B.29 Auxiliary and bulk coordinates stay separated at every size and every anchor. Rows: (a) ESA, (b) $1 - \mathrm{FRR}$, (c) DMD direction cosine; columns: the six sizes. The solid line is the group median over the pooled group set and the shaded band the interquartile range; orange is auxiliary, the vector segment, and blue is bulk. Grey dashed references mark 0.5 in (a) and 0 in (c). All panels use one readout protocol at τ=5 over the 123 shared anchors, and because the suite trains every size with the same 2M-token batch, a matched step grid is also a matched token grid. Horizontal comparisons between sizes are therefore at matched steps and tokens, and not at matched loss, compute, or training progress. The median separation holds at every anchor at all six sizes under all three probes; the smallest median gaps are 0.036, 0.351, and 0.115 for the three rows, all at 1.4B late in training. Interquartile ranges are disjoint at every anchor for (b) and (c) at all sizes, and for (a) at every size except 1.4B, where they overlap on 30% of anchors. Bulk is the strict matrix set, with both embedding segments dropped; under the looser alternative the bulk means move by at most 1.2×10^{-4} for (a) and (b) but by up to 6.3×10^{-3} for (c), which is why the strict rule is used throughout. Pythia-1B is excluded here and in every other cross-scale panel, on the three independent grounds of Table B.18(c). 186

B.30 Segment division of labour across sizes: ESA barely separates the matrix segments while the calibrated readout does. Columns are the six sizes. (a) ESA group median per segment. (b) pooled calibrated excess per segment, drawn as a five-anchor centred rolling median, with the onset markers computed on the unsmoothed series. The three matrix segments share a blue system (qkv solid, attention output dashed, MLP dotted) and the vector segment is orange; grey dashed references mark 0.5 in (a) and the null level 0 in (b), where the floor at -0.05 is the zero-fraction case. The dark dotted line is the sustained collapse onset of the qkv series, and where the series dips and recovers the first crossing is given in grey parentheses. Onsets are 20k (70M), 25k (410M), 28k (31M), 43k (14M), and 71k (160M), with no sustained collapse at 1.4B, so the ordering is not monotone in parameter count under either crossing. In (a) the three matrix segments stay within a median of 0.048, 0.038, 0.027, 0.021, 0.014, and 0.011 of one another in ascending size order, so the spread between matrix segments shrinks monotonically with size, while the vector segment stays above all of them by at least 0.034 at every anchor and size. 187

B.31 Where the *qkv* calibrated signal survives: layer against step at six sizes on one colour scale. Each panel is one model, rows are transformer blocks with layer 0 at the bottom, and columns are the 123 shared anchors. The probe is reported as excess over its α=0.05 null level, so 0 is exactly the null and the raw pocket fraction is the excess plus 0.05; the colour ramp is anchored with white at 0, every cell at or below the null renders white, and the ramp is clipped at 0.60, which affects 0.21% of cells. Ticks on the right edge mark the layers whose tail-phase mean stays above the null. Early in training the signal covers 5/6, 6/6, 6/6, 10/12, 23/24, and 23/24 layers in ascending size order, and then contracts onto a persistent band whose absolute layer index moves deeper as the model deepens. 1.4B does not contract inside the window: 22 of 24 layers are still above the null in the tail phase, and it shows instead an extinction window near 96 to 118k followed by a recovery from about 120k. The calibrated readout is identical under both bulk rules, since the vector segment does not enter it. 189

B.32 Internal predictor-family diagnostics on the bulk set across six sizes. (a) median predicted-over-realized step amplitude of the fit; (b) median largest eigenvalue modulus of the fitted operator; (c) the onset step of the amplitude deficit against non-embedding parameter count on a log axis. Line shade encodes size on a single ramp and every curve is additionally labelled at its right end, so colour is never the only encoding. Onset is the first anchor opening five consecutive anchors whose median amplitude ratio is below 0.5, marked by the grey dashed reference in (a). The deficit sets in earlier the larger the model, at 58k, 31k, 21k, 15k, 14k, and 11k in ascending size order, which is the only cross-scale ordering in this appendix that is monotone in parameter count. By the last anchor the bulk median amplitude ratio has fallen to 0.13 to 0.38 at every size while the eigenvalue modulus settles into 0.59 to 0.68, with 14M the exception at 0.78. 191

B.33 Input-geometry negative diagnostic. The pre-declared hypothesis that whole-layer input concentration or global input-subspace stability explains predictable pockets is not supported. These quantities measure spatial representation geometry, whereas the architecture-conditioned pocket analysis concerns local temporal update pockets. . 193

B.34 Within-layer group heterogeneity details. ViT-mid retains a higher high-predictability tail, while spread metrics such as IQR and top-quartile minus bottom-quartile separation are comparable across architectures. This supports a level and persistence interpretation rather than an overclaim that ViT has uniformly larger within-layer separation. 194

B.35 Full role-enrichment diagnostics. We report role-permutation z-scores and odds ratios for high-predictability groups. Large z-scores reflect both effect size and group count; odds ratios provide a complementary effect-size view. Negative enrichment indicates that high-predictability groups are under-represented in that role relative to other bulk roles. 194

B.36 Pocket identity persistence and anchor-pair Jaccard details. The bar plot expands the role-level persistence summary, and the heatmap reports Jaccard overlap of high-predictability group sets across anchor pairs. The same groups reappear more consistently in ViT-like models than in the large ResNet convolutional bulk. 195

B.37 Phase-aware architecture comparison. The left panel summarizes mid and late-plateau phase statistics; the right panel plots bulk selector lift against relative training progress. This diagnostic addresses the concern that equal raw epochs may correspond to different optimization phases across architectures. 195

B.38 Representative multi-seed architecture check. ResNet/SAM is highly seed-stable in this subset, while ViT-tiny/SAM shows seed-dependent absolute pocket mass. The role-enrichment and persistence directions remain consistent in the tested seeds. This is a representative check, not an exhaustive multi-seed sweep. 196

B.39 Auxiliary SGD-wd replication for architecture-conditioned pockets. The architecture-associated role-pocket pattern also appears under SGD-wd, with especially strong attention-`qkv` enrichment in ViT-mid. We treat this as auxiliary robustness for architecture-conditioned predictability and reserve optimizer-mechanism interpretation for the optimizer analysis. 196

B.40 Full weight-decay residual diagnostic for SGD-wd cells. The solid curves report raw update metrics, while dashed curves remove the instantaneous projection of each update onto $-w_t$. AlexNet/SGD-wd is the strongest shrinkage-drift case: late-anchor `pure_sgd` remains high in the raw update, FRR becomes low, and the fraction of predictable groups rises sharply, but these effects collapse after the shrinkage projection is removed. These residual metrics are diagnostic decompositions, not counterfactual optimizer trajectories. 197

B.41 SAM versus SGD across architectures and anchors. The curves show SAM minus SGD for bulk DMD-selector lift, bulk `pure_sgd`, and bulk fraction of predictable groups. SAM amplifies predictable structure most clearly in ViT-class models, while the effect is weaker or more transient in other architectures. We use this as architecture-dependent optimizer modulation, not as evidence that SAM is universally more predictable. 198
B.42 SGD-momentum versus SGD. Momentum is a natural smoothing mechanism, but under the current bulk probes it does not consistently increase DMD-selector lift or predictable-group mass relative to SGD. We retain this as a cautious supplement because momentum effects are protocol- and optimizer-state-dependent. 198
B.43 Adaptive optimizer profiles for Adam and AdaGrad. These plots summarize bulk lift, `pure_sgd`, and predictable-group fraction over anchors. The profiles are architecture-dependent, so we do not use them as a universal adaptive-optimizer mechanism. 199
B.44 Neighborhood-curvature diagnostic for adaptive optimizers and other baselines. The metric is $1-\cos(g(w), g(w+\rho\nabla L/\|\nabla L\|))$ with $\rho = 0.05$ on fixed batches. AdaGrad is an outlier in several architectures. This diagnostic is correlational, not causal. For ResNet and AlexNet, batch-normalization statistics are evaluated in train mode and can shift between the two gradient evaluations, so the metric is partially confounded with the BN-statistic Jacobian. ViT, ViT-mid, and MLP have no batch normalization. 199

B.45 Muon same-step spectrum sanity check. The pre-Newton–Schulz matrix buffer B_t and post-Newton–Schulz update Q_t are compared at the same optimizer step. The post-Newton–Schulz update has higher effective rank and spectral entropy, as expected from the orthogonalized matrix update. This is a same-step diagnostic only; it is not a pre-trajectory versus post-trajectory comparison. 200
B.46 ViT and ViT-mid neighborhood-curvature sanity check. The original, RNG-fixed, and evaluation-mode versions match because the TinyViT implementation used here has no `nn.Dropout`. This diagnostic is retained only to document that the ViT-class neighborhood-curvature values are not inflated by dropout-mask noise. 200

Nomenclature

$\Delta \mathbf{w}_t$	Parameter update at iteration t
$\Delta \theta_{i,t}$	Update of parameter i at iteration t
η	Learning rate
$\hat{\mathbf{w}}_{t+k}$	Predicted parameter vector at future iteration $t + k$
$\lambda_{\max}(H)$	Maximum eigenvalue of the Hessian
$\mathbf{A}$	Finite-dimensional Koopman approximation matrix
$\mathbf{a}_t$	Optimization acceleration (second-order difference of weights)
$\mathbf{v}_t$	Optimization velocity (first-order difference of weights)
$\mathbf{w}$	Neural network parameter (weight) vector
$\mathbf{w}_t$	Model parameters at iteration t
$\mathcal{K}$	Koopman operator
$\mathcal{L}(\mathbf{w})$	Training loss function
$\mathcal{M}$	Prediction mask in PDT
$\mathcal{M}_i$	Mask value for parameter i
$\nabla \mathcal{L}(\mathbf{w})$	Gradient of the loss with respect to parameters

$\odot$	Element-wise (Hadamard) product
$\Phi(\cdot)$	Koopman observable mapping
τ	Temporal shift for windowed predictiveness
$\theta_{i,t}$	Value of parameter i at iteration t
d_{traj}	Total trajectory distance in weight space
$H(\mathbf{w})$	Hessian matrix of the loss
k	Prediction horizon
$P_i^{(\text{win})}$	Windowed predictiveness of parameter i
$s_i^{(\text{Hess})}$	Hessian-style directional smoothness of parameter i
$s_i^{(\text{SAM})}$	SAM-style directional smoothness of parameter i
t	Training iteration index
w	Window length for predictiveness evaluation
DMD	Dynamic Mode Decomposition
DNN	Deep Neural Network
FIM	Fisher Information Matrix
KOT	Koopman Operator Theory
PDT	Predictive Differential Training
SAM	Sharpness-Aware Minimization
SGD	Stochastic Gradient Descent
ViT	Vision Transformer

Chapter 1

Introduction

1.1 The Persistent Cost of Iterative Neural Network Training

Despite major advances in hardware design, distributed training, and numerical efficiency, the training of modern deep neural networks remains fundamentally governed by iterative gradient-based optimization. Parallel and distributed computation have reduced the wall-clock cost of large-scale learning [43], while mixed-precision arithmetic and related numerical techniques have improved hardware utilization [58]. Optimization algorithms such as momentum, adaptive learning-rate methods, and second-order or curvature-aware methods have also been proposed to improve the effectiveness and stability of individual optimization steps [76, 17, 78, 36, 50]. These advances make each step faster, more stable, or better conditioned, but they preserve the same basic structure: training still proceeds through a long sequence of explicitly executed forward passes, backward passes, and parameter updates.

This step-by-step structure creates a persistent computational burden. Each optimization step evaluates the model, computes gradient information, and applies parameter updates under the current objective, data stream, and optimizer state. For large models, large datasets, and long training horizons, the cost of repeatedly

executing this loop accumulates rapidly. Classical and modern optimizers use local gradients, historical gradient statistics, curvature information, or layer-wise scaling to determine the next update. Outside of the predictive-training and weight-nowcasting literature discussed in Section 2.2, however, mainstream optimization methods do not usually treat the recent parameter trajectory itself as a signal for replacing a short sequence of future forward–backward updates. They improve or modulate the next step, but the trajectory is still explicitly traversed step by step.

This observation motivates two related questions. Methodologically, can selected expensive optimization steps be replaced by cheaper trajectory-based predictions without destabilizing training? Scientifically, do recent update trajectories contain measurable short-horizon temporal redundancy—that is, is recent motion informative about nearby future parameter displacement? This dissertation addresses both questions through an online selective-prediction method and a complementary retrospective measurement framework.

1.2 The Open Question of Temporal Structure in Training Trajectories

The possibility of predictive replacement depends on whether recent training history contains usable information about near-future parameter motion. Neural network training is globally nonlinear, stochastic, and high-dimensional, and there is no reason to assume that an entire model or training trajectory is predictable. Locally, however, short windows of the update trajectory may evolve more regularly than the full optimization process suggests. Such regularity may depend on optimization position, optimizer state, learning-rate schedule, architecture, data sampling, and other aspects of the training recipe. The relevant question is therefore not whether training is globally simple, but where and when recent updates contain short-horizon temporal structure and how that structure changes across parameters and training conditions.

Prior work has shown that neural optimization exhibits substantial structure, much of it characterized through spatial or geometric perspectives. Loss-landscape and mode-connectivity studies reveal smoother relationships among optimization paths and trained solutions than a naive view of high-dimensional nonconvexity would suggest [24, 42, 16, 21]. Studies of intrinsic dimension and low-dimensional training subspaces further show that optimization can evolve in spaces much smaller than the full parameter dimension [28, 1]. These results indicate that neural training is not arbitrary high-dimensional noise, but they mainly characterize geometric structure, solution connectivity, or low-dimensional optimization subspaces rather than the temporal question addressed here: whether recent updates contain actionable information about nearby future motion.

A growing body of weight-prediction and nowcasting work makes this temporal question concrete [73, 34, 37]. These methods use recent training history to forecast near-future weights and show that forecast-based interventions can be interleaved with continued gradient-based optimization. Their success demonstrates that useful short-horizon information can exist, but it does not imply that this information is uniformly distributed across parameters or training stages, nor does the performance of one predictor fully characterize the temporal organization of the underlying trajectory.

Taken together, these observations broaden the role of predictability beyond direct acceleration. In addition to determining whether a forecast can be used, a useful diagnostic should distinguish complementary forms of temporal organization, resolve where they occur in parameter space, and track how they change over training. Such a diagnostic remains informative even when no prediction is written back into optimization.

1.3 From Predictive Training to Selective Predictive Training

Predictive training provides one way to act on this local temporal structure by using recent optimization history to estimate a near-future parameter state. Let w_t denote the model parameters at checkpoint t, and let a baseline optimizer generate the explicit trajectory

$$w_t, w_{t+1}, \ldots, w_{t+\tau}$$

through repeated forward propagation, backward propagation, and parameter updates. A predictive training method forms a recent history window $\mathcal{H}_t$ and estimates

$$\widehat{w}_{t+\tau} = P(\mathcal{H}_t),$$

where P is a forecasting rule and τ is a short prediction horizon. The predicted state may then be used to replace, guide, or modify part of the explicitly executed optimization trajectory.

This idea has appeared in several forms. Learned weight-evolution methods train predictors from historical parameter trajectories [73]; weight-nowcasting methods periodically forecast near-future weights during training [34]; neuron-interaction nowcasting incorporates additional structural information into the forecasting process [37]; and Koopman/DMD-based approaches model weight trajectories through data-driven linear dynamical approximations [15, 77]. In each case, recent training history is treated as informative enough to support some form of near-future prediction.

In practice, however, forecast reliability can be highly non-uniform within a single training run. A forecast may be useful for one subset of coordinates while being unreliable elsewhere under the same model, optimizer, and data stream, and this heterogeneity can change over training. Producing a future estimate therefore does

not by itself determine where that estimate should be applied. This distinction motivates the step from predictive training to selective predictive training.

1.4 From Selective Prediction to Structured Predictability

Because forecast reliability is uneven, selective application becomes central to the first contribution of this dissertation. This contribution is Predictive Differential Training (PDT), which combines Koopman/DMD-based forecasts with a coordinate-wise mask based on scale control and dynamic consistency. Coordinates that pass this conservative reliability test may advance through prediction, while the remaining coordinates continue under the baseline optimizer. PDT therefore turns non-uniform forecast reliability into an operational training strategy while retaining gradient-based correction as a safeguard.

Yet an operational selector does not by itself explain where short-horizon structure resides or how it changes across training conditions. The second contribution addresses this gap through a structured predictability framework that uses complementary probes at multiple parameter scales to analyze stored trajectories retrospectively. Across controlled multi-pass vision training and one-pass language-model pretraining, the measurements reveal a recurring but non-uniform organization: auxiliary parameters generally exhibit simpler short-horizon dynamics, while bulk transformation tensors contain localized, time-varying predictable pockets. The measured organization also changes with optimization position, architecture, and training recipe, so predictability is not a whole-model scalar or a ranking of optimizer quality.

The resulting two contributions are closely connected but non-equivalent. Both use recent training history, and their DMD- and subspace-based components rely on local low-rank representations that make high-dimensional trajectory analysis

tractable. However, PDT's online coordinate-level acceptance rule and the measurement framework's tensor- and group-level retrospective readings use different criteria, granularities, and settings and should not be numerically equated. Their shared conclusion is structural: useful short-horizon regularity can exist, but it is selective, localized, and time-dependent.

1.5 Dissertation Contributions

This dissertation studies short-horizon predictability as a form of temporal organization in neural training dynamics and examines how such structure can be used and measured. Its two primary contributions correspond to the two parts of the dissertation.

- **Selective Predictive Differential Training.** This dissertation develops Predictive Differential Training (PDT), a selective predictive training framework that combines Koopman/DMD-based short-horizon forecasts with standard gradient-based optimization. A coordinate-wise mask based on scale control and dynamic consistency rejects unsupported forecasts and retains the baseline optimizer as a fallback. The dissertation analyzes the masked hybrid update and evaluates its convergence behavior, efficiency, robustness, mask dynamics, and computational overhead across multiple architectures, optimizers, datasets, and learning settings.

- **Structured measurement of short-horizon predictability.** This dissertation develops a convention-aware, multiscale framework for retrospectively measuring temporal organization in stored training trajectories. Complementary displacement-direction, subspace-residual, and predictor-based probes provide parameter-resolved and time-local readings across aggregate, tensor,

and within-tensor scales. Applied to multi-pass vision training and one-pass language-model pretraining, the framework identifies recurring auxiliary–bulk separation, localized predictable pockets, and systematic variation with optimization position, architecture, and training recipe.

Together, these contributions support a conditional account of predictive acceleration: prediction-based intervention is justified only where suitable short-horizon structure can be detected, used with reliability safeguards, and made worthwhile after accounting for prediction overhead. Chapter 5 develops this synthesis rather than treating it as a third independent technical contribution.

1.6 Organization of the Dissertation

Chapter 2 reviews the background and related work on gradient-based optimization, predictive and nowcasting approaches, Koopman/DMD foundations, and structure in neural training dynamics. Chapter 3 presents Predictive Differential Training, including its prediction mechanism, coordinate-wise masking strategy, analysis, and empirical evaluation. Chapter 4 develops the structured predictability measurement framework and studies its spatial, temporal, architecture-conditioned, and optimizer-conditioned behavior across vision and language-model trajectories. Chapter 5 synthesizes the two contributions, discusses their scope and limitations, and outlines future directions for measurement-guided predictive training and parameter-resolved analysis of training dynamics.

Chapter 2

Foundations of Predictive Training and Neural Training Dynamics

2.1 Iterative Training and Adaptive Update Rules

Modern deep neural networks are usually trained through repeated stochastic-gradient updates to a high-dimensional parameter vector. Let $w_t \in \mathbb{R}^d$ denote the trainable parameters at iteration t, and let $g_t = \nabla \ell_{\mathcal{B}_t}(w_t)$ be the gradient computed on a mini-batch $\mathcal{B}_t$. A broad class of optimizers can be written as

$$w_{t+1} = w_t + \mathcal{U}(w_t, g_t, s_t),$$

where s_t denotes optimizer state, such as momentum buffers, accumulated gradient statistics, or preconditioning matrices. Despite differences in $\mathcal{U}$, these methods share the same basic procedure: each iteration executes a forward pass, a backward pass, and a parameter update.

Stochastic gradient descent (SGD) updates parameters in the negative stochastic-gradient direction [66]. Momentum methods maintain a running average of past gradients, amplifying persistent descent directions and damping oscillatory components [62, 76]. Nesterov acceleration modifies this idea by evaluating the gradient at

a look-ahead point [59]. These methods alter the optimization trajectory but still execute every update explicitly.

Adaptive optimizers treat parameters more selectively. AdaGrad, RMSProp, Adam, and AdamW use accumulated gradient statistics to produce coordinate-dependent effective step sizes [17, 78, 36, 45]. Adam, for example, maintains exponential moving estimates of the first and second moments of the stochastic gradients. Its base learning rate is global, but the update applied to each coordinate depends on that coordinate's gradient history and optimizer state.

Adaptation also occurs at coarser structural scales. Curvature-aware and preconditioned methods such as K-FAC and Shampoo use structured approximations to second-order information [51, 27]. Layer-wise methods such as LARS and LAMB scale updates using parameter and gradient norms and are particularly useful in large-batch settings [85, 86]. Together, these methods show that effective optimization need not treat all coordinates, blocks, or layers identically.

These approaches improve how the next update is computed, but they retain the step-by-step structure of training. Predictive training asks a different question: whether recent trajectory history can support replacing selected expensive optimization steps with lower-cost forecast operations. The next section reviews methods that use training history for this purpose.

2.2 Using Training History

Training history can be used more directly than as optimizer state. Relevant methods differ in where history enters the training process. Extrapolation methods combine previous iterates to propose a new state; learned optimizers use history to construct the update rule; and predictive-training methods explicitly forecast future parameters or updates.

Classical extrapolation provides an early example. Anderson acceleration forms a new iterate from a linear combination of previous iterates and residuals, accelerating

fixed-point iterations when the local dynamics are sufficiently regular [3]. Regularized nonlinear acceleration stabilizes this construction through regularization and provides guarantees in convex or locally well-behaved settings [70]. Unlike an adaptive learning rate, which rescales a gradient-based step, these methods form a candidate iterate directly from a history of states and residuals. Their assumptions, however, need not hold in stochastic, non-convex neural-network training.

Learned optimizers provide a second use of history. Instead of applying a fixed analytic rule such as SGD or Adam, they train a neural network to produce updates from gradient, parameter, or optimizer-state histories [4, 82, 53]. They demonstrate that optimization histories contain learnable structure, but introduce additional requirements, including auxiliary training, transfer across tasks and architectures, and inference or state overhead.

Predictive training forecasts future parameters or updates more explicitly. Introspection learns patterns of weight evolution from historical trajectories [73]. Weight nowcasting uses recent checkpoints to forecast near-future weights and periodically interleaves these forecasts with conventional optimization [34]. Neuron-interaction nowcasting incorporates structural relations among neural components rather than treating coordinates independently [37]. XGrad uses weight prediction within an optimizer-level rule, providing another way to incorporate a future-state estimate into gradient-based training [25].

These methods establish several viable uses of training history, but they rely on either structural assumptions, separately learned predictors, or specific rules for incorporating a forecast. This dissertation focuses on a complementary approach: fitting a predictive model directly to the recent trajectory of the current training run, without a separately trained forecasting network. Koopman operator theory and Dynamic Mode Decomposition provide the foundation for this approach.

2.3 Koopman/DMD Foundations for Predictive Training

Koopman operator theory and Dynamic Mode Decomposition (DMD) provide a mathematical framework for modeling observed dynamics from temporal snapshots. Koopman theory represents nonlinear state evolution through linear evolution of observables, while DMD constructs a finite-dimensional approximation from data [54, 56, 69, 40].

Consider a discrete-time nonlinear dynamical system

$$x_{t+1} = T(x_t),$$

where x_t is the state and T is the transition map. For an observable g, the Koopman operator $\mathcal{K}$ is defined by

$$(\mathcal{K}g)(x_t) = g(T(x_t)) = g(x_{t+1}).$$

Although T may be nonlinear, $\mathcal{K}$ is linear on the space of observables [57, 54, 55, 56]. This property permits spectral and multi-step analysis using linear-operator tools.

For neural-network training, the state may be taken as the parameter vector w_t, or as a related representation of its updates. With the identity observable $g(w_t) = w_t$, saved checkpoints become direct observations of the optimizer-induced training dynamics. This choice is simple and practical, but it does not exploit nonlinear observables that might better linearize the dynamics. The dissertation therefore uses Koopman/DMD as a local, data-driven model of recent weight evolution rather than as an exact representation of the complete training process.

DMD begins with paired snapshot matrices

$$X = [x_1, x_2, \ldots, x_{h-1}], \qquad Y = [x_2, x_3, \ldots, x_h],$$

and fits a linear map that advances the observations in X toward those in Y. In the standard reduced workflow, a thin or truncated singular value decomposition of X is used to construct and eigendecompose an operator on the snapshot subspace. Full-state DMD modes can then be reconstructed without forming an operator over the full ambient parameter dimension [80, 40]. This reduction makes high-dimensional weight sequences tractable when a short history is well represented by a small number of modes. Its computational advantage diminishes as the retained rank or history length grows, and longer windows may be less appropriate when training dynamics are nonstationary.

Several studies have applied Koopman and DMD ideas to neural-network training. Dietrich et al. develop an operator-theoretic view of iterative algorithms [13]. Dogra and Redman model neural-network weight evolution as a dynamical system and use Koopman/DMD representations to analyze and accelerate optimization [14, 15]. Tano et al. first extract dominant layer-wise weight modes using proper orthogonal decomposition (POD)/SVD, fit a reduced DMD model, and extrapolate the resulting modal trajectory before resuming backpropagation [77]. Their method applies the predicted state to the modeled weights without a parameter-wise acceptance rule. Related work uses Koopman mode analysis for neural-network analysis, pruning, and the identification of equivalent training dynamics [49, 65, 64].

These studies establish DMD as a practical forecasting mechanism for observed weight sequences. Its usefulness nevertheless depends on whether recent training history contains stable local structure that can support short-horizon extrapolation. The next section reviews the forms of structure previously identified in neural training dynamics.

2.4 Structure in Neural Training Dynamics

Prior work provides several reasons to expect structure in neural training, but most studies characterize loss-landscape geometry, low-dimensional subspaces, representations, or named training phases rather than the short-horizon temporal organization of parameter updates.

The closest trajectory-level precedent analyzes the optimization path itself. Singh et al. [72] show that the directional geometry of raw checkpoint paths saturates early at coarse temporal scales, continues to oscillate at finer scales, and is shaped in part by weight decay and momentum. The measurement problem in this dissertation is different: it concerns the sequence of updates rather than the raw checkpoint path, asks whether recent updates are informative about the next few steps, and resolves this structure at aggregate, tensor, and within-tensor scales.

Loss-landscape studies show interpolation structure and low-loss connectivity among trained solutions [24, 42, 21, 16, 20, 2]. Snapshot ensembles, stochastic weight averaging, and model soups further exploit useful relations among points sampled along or across optimization trajectories [31, 32, 83]. These results establish geometric regularity in the loss landscape and solution space.

Low-dimensional and spectral analyses provide a related view. Hessian studies find anisotropic curvature and evolving spectral structure [67, 23]. Gradient descent can align with low-dimensional Hessian subspaces [28], and intrinsic-dimension experiments show that successful optimization may occur in spaces much smaller than the full parameter dimension [41, 1]. Low-rank methods have also been used to reduce optimization cost by estimating gradients in dominant directions [11]. However, spatial confinement does not imply temporal predictability: a trajectory may remain in a low-dimensional region while changing direction irregularly within it. Recent work likewise cautions against interpreting gradient–Hessian alignment as evidence that SGD simply follows a fixed projected subspace [74].

Training structure also changes over time. Edge-of-stability analyses describe optimization near curvature stability boundaries [10]; neural collapse characterizes terminal-phase representation geometry [61]; and grokking reveals delayed transitions between memorization and generalization [63, 44, 38, 46]. Lazy-training and neural-tangent-kernel analyses identify regimes in which network evolution remains close to a linearized model around initialization [33, 9]. These phenomena indicate that any usable temporal structure may vary across training stages.

Released checkpoint suites enable this evolution to be studied over long pretraining runs. Pythia provides intermediate checkpoints across model scales [5], and has supported studies of capability and circuit formation, including induction heads and circuit consistency across training and scale [60, 79]. Scaling-law work separately characterizes relationships among model size, data, compute, and training duration [30]. This dissertation uses Pythia for a different purpose: to measure temporal redundancy in parameter updates derived from the released checkpoints. Temporal alignment with reported capability or circuit events is treated as correspondence, not as evidence of a shared mechanism.

Architecture and optimization choices further shape the trajectory. Normalization, attention, and adaptive methods affect Transformer optimization [84, 87, 90, 39]. Momentum, weight decay, sharpness-aware perturbations, slow–fast coupling, and matrix-structured updates alter parameter-update geometry [36, 45, 19, 88, 47, 35]. Weight decay is especially relevant because its interaction with normalization and effective learning rates can introduce slowly varying structure into parameter trajectories [81, 12].

Taken together, these studies show that neural training is structured but that its structure depends on scale, training stage, architecture, and optimization recipe. They motivate the more specific question addressed in Chapter 4: whether recent updates contain measurable information about near-future parameter motion, and where that information is located within the model.

2.5 Research Gap and Dissertation Positioning

The literature leaves three issues unresolved for predictive training.

Forecast reliability.

A method that produces a future parameter estimate does not establish that the estimate is safe to use. Forecast quality may vary across parameters and across training stages. DMD is therefore useful as a proposal mechanism, but it does not determine which parts of a prediction should replace optimizer-derived updates.

Temporal evidence.

Loss-landscape geometry, curvature, and low-dimensional subspaces show that training is organized, but they do not directly test whether recent updates are informative about nearby future motion. The relationship between recent and future motion is the form of structure needed to justify predictive replacement of optimization steps.

Localization and conditioning.

Short-horizon predictability need not be a whole-model scalar or a fixed property of a training run. It may differ across parameter roles, tensors, and within-tensor groups, and may change with optimization position, architecture, optimizer state, regularization, and training recipe. Predictability should therefore be interpreted as a property of an observed trajectory under a stated measurement scale, not as a ranking of optimizer or architecture quality.

Several parallel and subsequent studies support the broader importance of these questions. Predictable Gradient Manifolds uses path-length and rank measures of temporal predictability [6]. Leap+Verify uses speculative weight prediction with verification before acceptance [52]. TeZO studies temporal low-rank structure in zeroth-order optimization for LLM fine-tuning [75]. Gradient Flow Matching instead learns weight-update dynamics using flow-matching models [71]. These works address related problems through complexity analysis, verification, specialized optimization,

or learned forecasting; they do not duplicate the two-part treatment developed in this dissertation.

Chapter 3 addresses the operational problem through Predictive Differential Training. DMD-generated coordinate predictions are accepted only when they satisfy scale-control and dynamic-consistency criteria; all other coordinates remain under the baseline optimizer. Chapter 4 addresses the complementary measurement problem retrospectively and at multiple parameter scales. Its probe families measure displacement direction, subspace residuals, and predictor-based selectivity. The chapters are connected but not equivalent: the measurements diagnose where temporal redundancy appears, but they do not constitute a completed online pocket detector or an automatic acceptance rule.

The dissertation therefore adopts a conditional view of predictive training. Neural trajectories can contain exploitable short-horizon redundancy, but any predictive reuse must remain selective, safeguarded, and sensitive to where and when that redundancy occurs.

Chapter 3

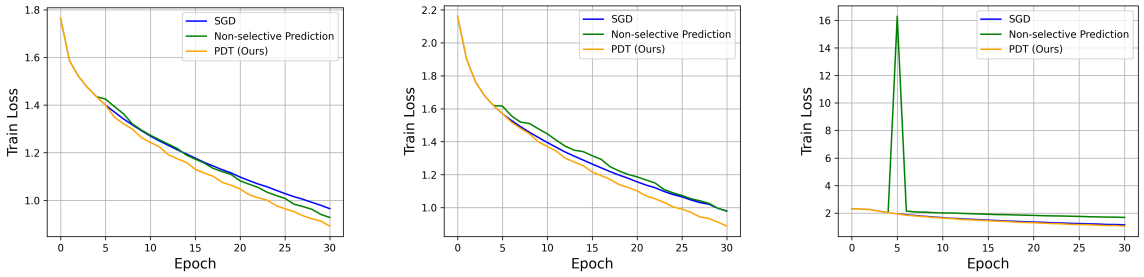
Predictive Differential Training Guided by Training Dynamics

3.1 Problem Formulation

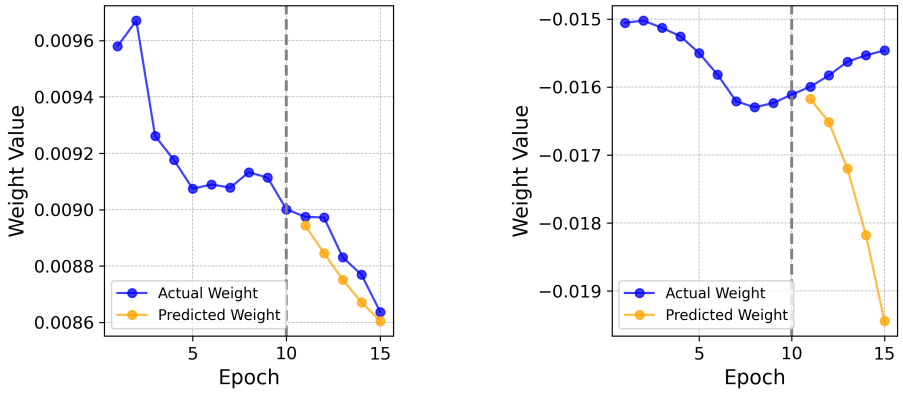
Predictive training was introduced in Chapter 1, Section 1.3, as a forecast-based alternative to purely step-by-step optimization: a recent history window $\mathcal{H}_t$ is used to estimate a near-future parameter state $\widehat{w}_{t+\tau} = P(\mathcal{H}_t)$. The central unresolved issue is not only how to construct the forecasting rule P, but also where the resulting forecast can be trusted.

Figure 3.1 exhibits this non-uniform reliability concretely. At the network level (top row), uniformly applied Koopman/DMD predictions remain stable on shallow FCNs but diverge on deeper ones. At the coordinate level (bottom row), the same forecasting mechanism yields very different fidelity for different parameters. Together these patterns indicate that the relevant design question is not only how to forecast a future weight vector, but how to decide where a forecast may be written back into the model.

A natural way to express this selectivity is through a differential learning viewpoint. Differential learning does not require all parameters to move with the



(a) 2-layer FCN **(b)** 4-layer FCN **(c)** 6-layer FCN



(d) High-fidelity coordinate prediction **(e)** Low-fidelity coordinate prediction

Figure 3.1: Motivation for selective predictive training. Non-selective prediction may remain stable in simpler models but becomes unreliable as model complexity increases. Coordinate-level examples further show that the same forecasting mechanism can produce high-fidelity predictions for some parameters and low-fidelity predictions for others.

same effective learning rate or update scale. If g_t denotes an optimizer-induced update direction, a coordinate-wise differential update can be written as

$$w_{t+1}^{\text{diff}} = w_t - \eta_t \odot g_t, \qquad \eta_t \in \mathbb{R}_+^d, \tag{3.1}$$

where η_t is a vector of effective step sizes. Standard gradient descent corresponds to the special case $\eta_t = \eta \mathbf{1}$, and adaptive optimizers can be viewed within this differential family, with η_t varying smoothly across coordinates rather than taking a binary form. A binary analogue is given in Appendix A.1, where one subset of variables uses a

larger effective learning rate while the remaining variables retain the baseline rate; in that coupled system, subset acceleration improves overall convergence.

Predictive Differential Training (PDT) applies this differential viewpoint to predictive training. It adopts a binary differential-training form: coordinates whose short-horizon forecasts are reliable are advanced by prediction, while the remaining coordinates continue to follow the baseline optimizer. The central problem is therefore not merely to forecast a future weight vector, but to identify the subset of coordinates for which prediction can be trusted without degrading training fidelity. In this form, PDT replaces selected expensive optimizer advances with cheaper trajectory-based prediction, rather than eliminating computation.

Thus, PDT solves the selective prediction problem:

Identify a subset of parameters for which a short-horizon prediction can replace selected explicit optimizer updates without degrading training fidelity.

The remainder of this chapter develops the DMD-based forecaster (Section 3.2.1), the masking rule that operationalizes this selection (Section 3.2.2), and the experiments validating the design (Section 3.3).

3.2 Methodology

3.2.1 Koopman/DMD-Based Multi-Step Prediction

Following the Koopman/DMD framework of Section 2.3, we view the epoch-level weight sequence $\mathbf{w}_1, \mathbf{w}_2, \ldots \in \mathbb{R}^d$ as observations of the optimizer-induced dynamics. PDT uses this sequence to fit a local DMD approximation and predict future weights several epochs ahead.

Specializing the abstract snapshots of Section 2.3 to the epoch-level weight setting, PDT forms paired snapshot matrices $\mathbf{W}_i$ and $\mathbf{W}_{i+1}$ from a short history of h weights

and fits a finite-dimensional approximation of the local Koopman evolution:

$$\mathbf{W}_{i+1} \approx A\,\mathbf{W}_i, \tag{3.2}$$

where A approximates the local action of the Koopman operator (Section 2.3) on the subspace spanned by the recent snapshots. In our implementation, the snapshot matrices are constructed as

$$\mathbf{W}_i = \begin{bmatrix} \mathbf{w}_{i-h+1} & \mathbf{w}_{i-h+2} & \cdots & \mathbf{w}_{i-1} \end{bmatrix}, \qquad \mathbf{W}_{i+1} = \begin{bmatrix} \mathbf{w}_{i-h+2} & \mathbf{w}_{i-h+3} & \cdots & \mathbf{w}_{i} \end{bmatrix}. \tag{3.3}$$

Here the matrix subscript indexes the snapshot window rather than its latest column: $\mathbf{W}_i$ collects the $h-1$ weights up to but not including $\mathbf{w}_i$, while $\mathbf{W}_{i+1}$ is the one-epoch-shifted version that contains $\mathbf{w}_i$ as its rightmost column. Thus $\mathbf{W}_i, \mathbf{W}_{i+1} \in \mathbb{R}^{d\times(h-1)}$, where d is the number of trainable parameters and $h \ll d$.

Rather than constructing or diagonalizing the full $d \times d$ operator in Eq. 3.2, PDT computes DMD in the low-rank snapshot subspace. Following the standard reduced-DMD construction [69, 80, 40], a rank-r truncated SVD of the snapshot matrix, with $r \leq h-1$,

$$\mathbf{W}_i \approx U_r \Sigma_r V_r^\top, \tag{3.4}$$

yields a reduced operator whose eigendecomposition gives the eigenvalues Λ and the DMD modes Φ:

$$\widetilde{A} = U_r^\top \mathbf{W}_{i+1} V_r \Sigma_r^{-1} \qquad \widetilde{A} Z = Z\Lambda \qquad \Phi = U_r Z \in \mathbb{R}^{d\times r}. \tag{3.5}$$

This keeps the spectral computation at the scale of the short history rank r rather than the full parameter dimension d.

PDT then performs a direct multi-step spectral prediction:

$$\mathbf{w}_{i+\tau}^{\text{pred}} = \mathrm{Re}\left\{\Phi\,\Lambda^\tau\,\Phi^\dagger\,\mathbf{w}_i\right\}, \qquad \tau \in \mathbb{Z}_+, \tag{3.6}$$

where τ is the prediction horizon and $\Phi^\dagger$ denotes the Moore–Penrose pseudoinverse. The current weight vector is projected onto the DMD modes, each mode is advanced by τ steps according to its eigenvalue, and the result is reconstructed as a candidate future weight vector. The real part is taken because the network parameters are real-valued. We use the spectral form $\Phi\Lambda^\tau\Phi^\dagger$ rather than the matrix power A^τ, which can be numerically unstable in high-dimensional systems [40].

Training workflow. PDT inserts prediction steps intermittently into training. After an initial burn-in stage of T_0 epochs, during which only the baseline optimizer runs and weight snapshots are accumulated, a prediction step is performed every T_i epochs, each advancing the weights by τ epochs into the future. The hyperparameters (T_0, T_i, τ, h) govern how often and how aggressively prediction is applied. PDT remains effective across a range of these values, and in practice they are kept relatively small (e.g., a short history h and a short prediction interval T_i); their effect is studied systematically in Section 3.3.

The predicted vector in Eq. 3.6 is not applied uniformly to all parameters. It serves as a candidate forecast whose coordinates are subsequently filtered by the masking mechanism. The overall workflow and the detailed mechanism of a τ-step prediction are illustrated in Fig. 3.2, where the prediction module (`Pred`) is embedded among standard optimization steps (`OPT`). Qualified predicted weights and optimizer-derived weights are then integrated through the selective update described in the next subsection.

3.2.2 Selective Masking via Dynamic Consistency

As discussed in Section 2.4, predictability is not uniformly distributed across parameters: some coordinates follow the dominant recent dynamics, while others do not. Given the candidate DMD forecast $\mathbf{w}_{i+\tau}^{\text{pred}}$ from Section 3.2.1, PDT constructs a coordinate-wise reliability mask $\mathbf{m}_i \in \{0,1\}^d$. Each entry $m_{i,j}$ determines whether coordinate j receives the predicted value or falls back to the one-step optimizer update

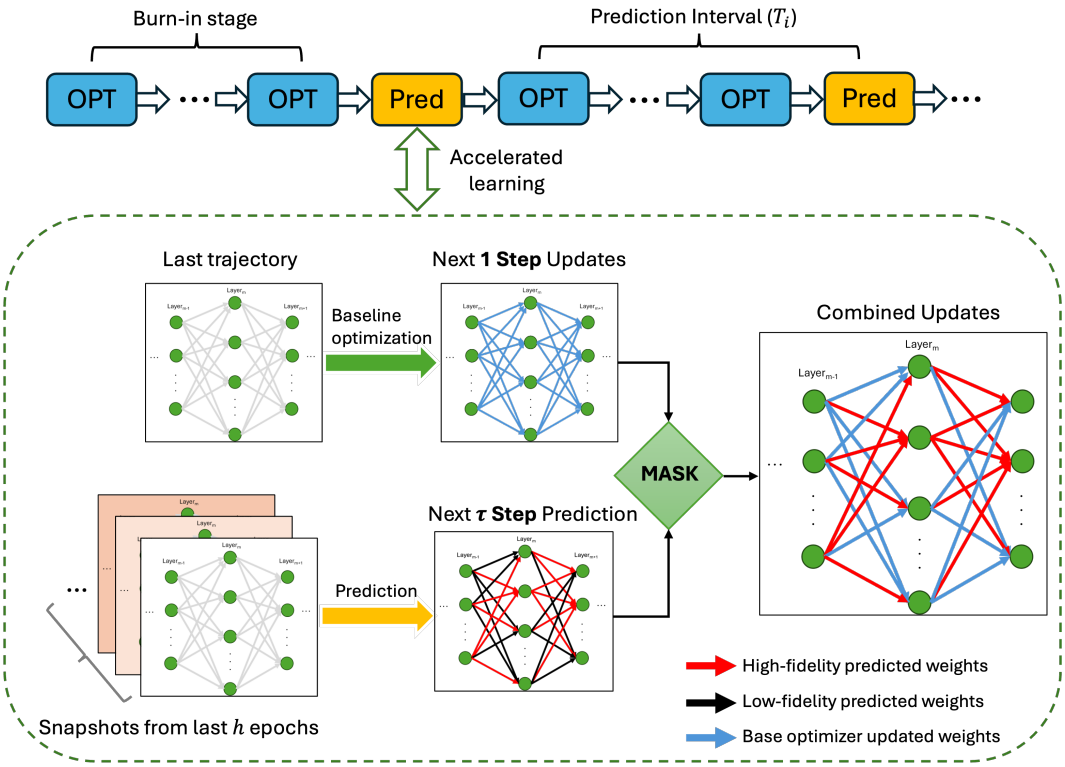


Figure 3.2: Illustration of the PDT framework and the detailed mechanism for a τ-step prediction, where qualified predicted weights (red) and standard optimizer-derived weights (blue) are integrated to accelerate training; the selection mechanism that determines which predicted weights are qualified is detailed in Section 3.2.2.

$\mathbf{w}_{i+1}^{\text{opt}}$. The mask is defined by two criteria: acceleration effectiveness and dynamic consistency.

Acceleration effectiveness. Let $\mathbf{w}_i$ denote the current weights, $\mathbf{w}_{i+1}^{\text{opt}}$ the one-step baseline optimizer result, and $\mathbf{w}_{i+\tau}^{\text{pred}}$ the τ-step DMD prediction from Eq. 3.6. For each coordinate j, the acceleration-effectiveness criterion requires the predicted displacement to be larger than the corresponding one-step optimizer displacement while remaining bounded by the prediction horizon:

$$\left|w_{i+1,j}^{\text{opt}} - w_{i,j}\right| < \left|w_{i+\tau,j}^{\text{pred}} - w_{i,j}\right| \leq \tau \left|w_{i+1,j}^{\text{opt}} - w_{i,j}\right|. \tag{3.7}$$

The lower bound ensures that an accepted prediction advances coordinate j farther than one explicit optimizer step. The upper bound prevents the predicted displacement from exceeding the coordinate-wise scale of τ such steps.

Dynamic consistency. The dynamic-consistency criterion checks whether the predicted trajectory follows the local optimizer-induced direction. The spectral predictor produces intermediate states at horizons $k = 1, \ldots, \tau$. For coordinate j, PDT requires each predicted intermediate increment to align with the one-step optimizer direction:

$$\operatorname{sign}\left(w_{i+k,j}^{\text{pred}} - w_{i+k-1,j}^{\text{pred}}\right) = \operatorname{sign}\left(w_{i+1,j}^{\text{opt}} - w_{i,j}\right), \qquad k = 1, \ldots, \tau. \tag{3.8}$$

For $k = 1$, the predicted trajectory is anchored at the current state, $\mathbf{w}_i^{\text{pred}} = \mathbf{w}_i$, so the first predicted increment is compared against the local optimizer step from the same starting point. This criterion enforces not only that the final predicted displacement has the correct sign, but also that the intermediate spectral trajectory remains directionally consistent with the local optimizer step.

Mask Construction and Hybrid Update. Based on these two criteria, PDT constructs the coordinate-wise binary mask $\mathbf{m}_i$. For each coordinate j,

$$m_{i,j} = \begin{cases} 1, & \text{if both criteria are satisfied,} \\ 0, & \text{otherwise.} \end{cases} \tag{3.9}$$

The resulting hybrid update is

$$\mathbf{w}_{i+1} = \mathbf{m}_i \odot \mathbf{w}_{i+\tau}^{\text{pred}} + (\mathbf{1} - \mathbf{m}_i) \odot \mathbf{w}_{i+1}^{\text{opt}}, \tag{3.10}$$

where $\odot$ denotes elementwise multiplication. Coordinates passing both criteria are treated as high-fidelity predictions and written back through the first term, while

the remaining low-fidelity coordinates fall back to the baseline optimizer through the second term.

This mask acts as a local reliability filter on top of the global DMD forecast. Coordinates whose predicted evolution is both sufficiently large and directionally consistent with the local optimizer step are accelerated with bounded deviation. Coordinates that exhibit irregular, oscillatory, or transitional behavior remain under the baseline optimizer. In this way, prediction is applied only where the global spectral forecast agrees with the local update direction and scale.

PDT also uses an acceleration scheduler to control how aggressively prediction is attempted during training. The scheduler adjusts the prediction interval and can rectify off-predictions by returning control to the baseline optimizer when predicted updates deviate from the training trajectory. It determines when prediction is attempted and how quickly the method falls back after unreliable prediction events.

The scale-control condition also gives a direct bounded-step property. Under the idealized gradient-descent setting $\mathbf{w}_{i+1}^{\text{opt}} = \mathbf{w}_i - \eta \nabla \mathcal{L}(\mathbf{w}_i)$, the hybrid step $\mathbf{d}_i := \mathbf{w}_{i+1} - \mathbf{w}_i$ satisfies

$$\|\mathbf{d}_i\| \leq \tau\eta \, \|\nabla \mathcal{L}(\mathbf{w}_i)\| \, . \tag{3.11}$$

Under a standard step-size condition and L-smoothness, the masked hybrid update admits a gradient-descent-style descent guarantee and the loss sequence converges. The full proof is given in Appendix A.3.

3.3 Experimental Evaluation

This section evaluates Predictive Differential Training (PDT) as a selective predictive acceleration method. The experiments are organized around the evidence chain established in the previous sections: first, whether PDT achieves net acceleration while preserving final performance; second, whether the selective masking mechanism is necessary for stability; third, whether PDT can be inserted on top of different

Algorithm 1 Predictive Differential Training

Require: Baseline optimizer; history length h; burn-in T_0; prediction interval T_i; horizon τ

Initialize model weights $\mathbf{w}_0$ and an empty weight-history buffer
for epoch $i = 0, 1, 2, \ldots$ **do**
    Run the baseline optimizer for one epoch to obtain $\mathbf{w}_{i+1}^{\text{opt}}$
    Store the current weight snapshot in the history buffer
    if $i < T_0$ or fewer than h snapshots are available or prediction is not scheduled **then**
        $\mathbf{w}_{i+1} \leftarrow \mathbf{w}_{i+1}^{\text{opt}}$
    else
        Construct $\mathbf{W}_i$ and $\mathbf{W}_{i+1}$ from the recent history
        Compute DMD modes Φ and eigenvalues Λ from $\mathbf{W}_i, \mathbf{W}_{i+1}$
        Predict $\mathbf{w}_{i+\tau}^{\text{pred}}$ and intermediate states $\mathbf{w}_{i+k}^{\text{pred}}$, $k = 1, \ldots, \tau$
        Construct $\mathbf{m}_i$ using Eqs. 3.7–3.9
        $\mathbf{w}_{i+1} \leftarrow \mathbf{m}_i \odot \mathbf{w}_{i+\tau}^{\text{pred}} + (\mathbf{1} - \mathbf{m}_i) \odot \mathbf{w}_{i+1}^{\text{opt}}$
    end if
end for

optimizers and learning settings; and finally, how the accepted prediction ratio and key prediction hyperparameters behave during training.

3.3.1 Experimental Setup and Metrics

We evaluate PDT on supervised image classification and on one self-supervised task for representation learning. The supervised experiments cover CIFAR-10 and ImageNet-1K. The architectures range from smaller fully convolutional and AlexNet-style networks to ResNet-50, ViT-Base, and ViT-Huge. Unless otherwise specified, tabular results are averaged over multiple random seeds and reported as mean ± standard deviation.

Following the evaluation protocol of the PDT paper, performance is measured from two perspectives. **Optimization efficiency** is measured by the **Time to Baseline Best Train Loss** (**TTB-Loss**), defined as the wall-clock time required for PDT to reach the best training loss achieved by the baseline optimizer. **Generalization efficiency** is measured by the **Time to Baseline Best Validation Accuracy**

(**TTB-Acc**), defined as the wall-clock time required for PDT to reach the best validation accuracy achieved by the baseline. These metrics evaluate whether PDT reaches the same optimization or generalization target earlier, rather than merely reporting final accuracy.

All runtime metrics reported in this dissertation represent total wall-clock time, fully inclusive of all computational overhead introduced by PDT, including snapshot collection, SVD/DMD computation, multi-step prediction, mask generation, and rejected prediction attempts. Thus, the reported runtime reduction is a net acceleration after accounting for the cost of the predictive module. Detailed runtime, memory, and FLOPs profiling are provided in Appendices A.4 and A.5; the main text reports net wall-clock acceleration and the primary experimental evidence.

For the main PDT configuration, the predictor uses a short history of recent epoch-level weight snapshots and performs short-horizon prediction at the epoch level, consistent with the workflow described in Section 3.2.1. The effects of the prediction horizon τ, prediction interval T_i, starting epoch T_0, and history length h are studied separately in Section 3.3.7.

3.3.2 Cross-Architecture Evaluation: Convergence and Net Efficiency

PDT is evaluated across neural network architectures of different scales, including Fully-Convolutional Network (FCN, 3.9M parameters), AlexNet (57M parameters), ResNet-50 (25.6M parameters), ViT-Base (86.4M parameters), and ViT-Huge (632M parameters). The experiments span CIFAR-10 and ImageNet-1K, and use baseline optimizers including SGD, SGD with momentum, and AdamW. This range is used to test whether selective prediction remains effective beyond small networks where earlier non-selective Koopman-based prediction methods were primarily demonstrated.

Figure 3.3 illustrates the two time-to-target metrics on AlexNet trained on CIFAR-10. PDT reaches the baseline optimizer's best training loss and best validation

accuracy earlier in wall-clock time, while maintaining comparable final performance. This representative case illustrates the distinction between reducing training loss faster and obtaining a useful model earlier.

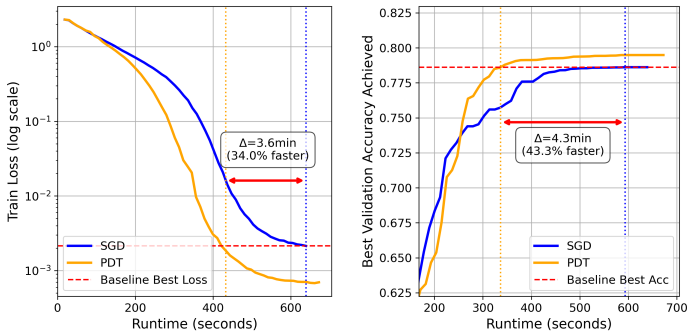
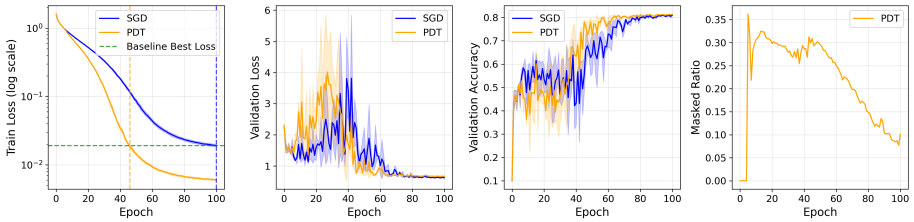


Figure 3.3: Analysis of PDT's optimization and generalization efficiency. Trained on CIFAR-10 using AlexNet, batch size 256, learning rate 0.05, with a cosine-annealing learning-rate scheduler.

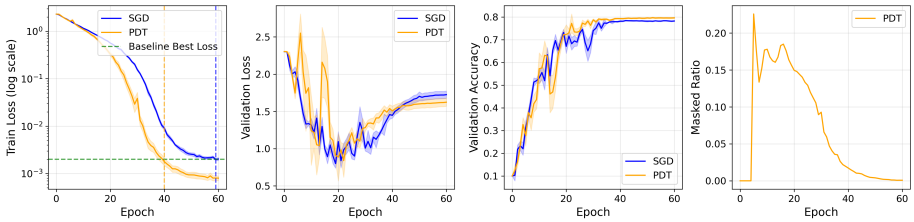
Figure 3.4 shows convergence behavior across architectures and datasets. On CIFAR-10, FCN and AlexNet exhibit a clear leftward shift in both training loss and validation accuracy, indicating faster progress toward the baseline targets. On ImageNet-1K, the effect is smaller but remains visible for ResNet-50 and ViT-Base, suggesting that the selective prediction mechanism scales beyond the small-network regime.

Table 3.1: Runtime comparison. FCN and AlexNet are trained on a single NVIDIA RTX A6000 GPU, while ResNet-50, ViT-Base, and ViT-Huge are trained on three NVIDIA H100 (80 GB) GPUs. For each model, PDT and its paired baseline use the same model-specific optimizer and training configuration.

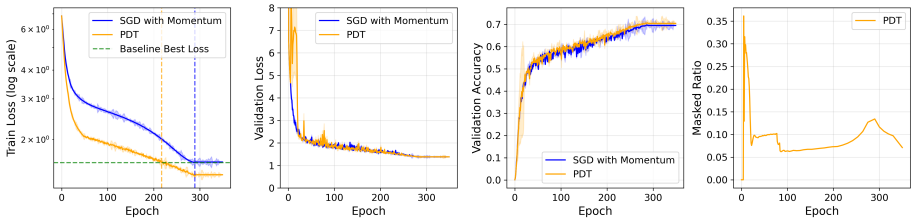
Model	Base Opt.	TTB-Loss (s)		TTB-Acc (s)		Runtime Reduction (%)	
		Baseline	PDT	Baseline	PDT	Train Loss	Val. Acc.
FCN	SGD	2174.32	1313.52	2088.58	1424.14	**39.59**	**31.81**
AlexNet	SGD	683.93	430.91	531.30	347.11	**37.00**	**34.67**
ResNet-50	SGD+M	110063.72	88752.33	121449.60	92133.34	**19.36**	**24.14**
ViT-Base	AdamW	259241.21	232810.62	296028.36	243097.58	**10.20**	**17.88**
ViT-Huge	AdamW	725564.86	653854.05	741220.54	660711.80	**9.88**	**10.86**



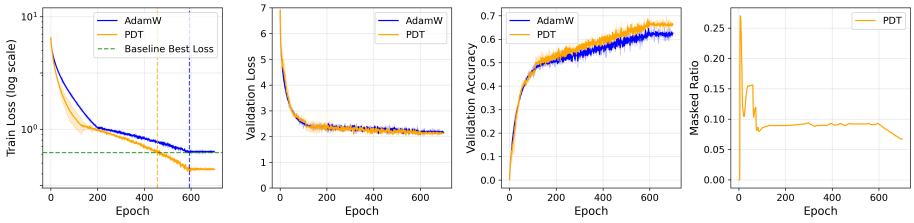
(a) Trained on CIFAR-10 using FCN, batch size 256, learning rate 0.01, with a cosine-annealing learning-rate scheduler.



(b) Trained on CIFAR-10 using AlexNet, batch size 256, learning rate 0.05, with a cosine-annealing learning-rate scheduler.



(c) Trained on ImageNet-1K using ResNet-50, batch size 1800, learning rate 0.1, momentum 0.9, with a cosine-annealing learning-rate scheduler.



(d) Trained on ImageNet-1K using ViT-Base with AdamW, batch size 256, learning rate 0.0006, weight decay 0.05, and a cosine-annealing learning-rate scheduler.

Figure 3.4: Performance comparison between baseline optimization and PDT.

Table 3.1 reports the corresponding net wall-clock reduction. Across these settings, PDT reduces TTB-Loss and TTB-Acc by approximately 10–40%, with larger reductions on CIFAR-10 and smaller but still positive reductions on ImageNet-scale models. The magnitude of acceleration varies with model scale, dataset, and optimizer, but the results consistently show that the selective prediction module can reduce time to the baseline's best loss and validation accuracy without degrading the target performance.

Additional training curves, accumulated-FLOPs comparisons, and runtime and memory profiling are reported in Appendix A.4 and Appendix A.5. In particular, the FLOPs analysis reports a representative AlexNet/CIFAR-10 reduction of approximately 23.74% in total computation required to reach the baseline target loss.

3.3.3 Masking Necessity: Ablation of Selection Criteria

The previous subsection establishes that PDT can reduce time to target metrics. The next question is whether the selective mask is necessary, or whether a simpler selection rule would suffice. This subsection evaluates the masking mechanism from two angles: first, by comparing PDT with ad hoc selection strategies; and second, by ablating the two criteria used in Section 3.2.2.

The PDT paper compares the proposed mask with random-selection alternatives. Randomly accelerating a subset of weights with the same selection ratio does not reproduce PDT's improvement and can introduce instability. Randomly selecting subsets of Koopman-predicted weights is even less stable, frequently leading to gradient explosions and non-recoverable NaN values. We also evaluate an alternative strategy that switches between prediction and SGD based on the validation-loss trend; it is likewise insufficient. Although it initially activates prediction when validation loss decreases, later prediction errors can cause a loss surge from which reverting to SGD does not recover. These results indicate that selection must depend on local prediction fidelity, not merely on selection ratio or delayed validation-loss feedback.

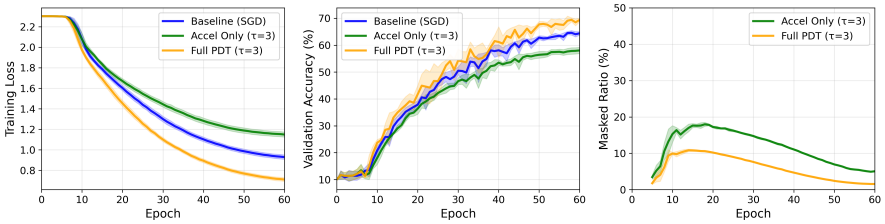
Table 3.2 reports a controlled ablation on AlexNet trained on CIFAR-10. The four variants are: the baseline optimizer, PDT using only the acceleration-effectiveness criterion, PDT using only the dynamic-consistency criterion, and full PDT using both criteria. The results show that neither criterion alone is sufficient. Consistency-only fails in nearly all trials, while acceleration-only becomes unstable for larger prediction horizons. Full PDT achieves stable convergence across all tested settings.

Table 3.2: Comprehensive masking ablation results on AlexNet trained on CIFAR-10. Success rates are shown as successful/total runs. Final accuracy is reported as mean ± std across five seeds, using successful runs only. Accuracy values at chance level (10% on CIFAR-10) indicate runs that failed to learn.

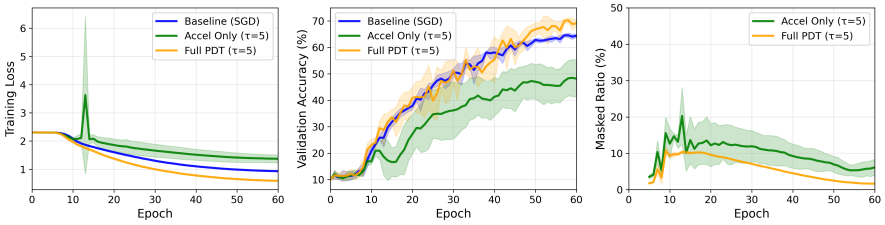
Configuration	τ	LR=0.01		learning rate 0.05		Overall
		Success	Acc. (%)	Success	Acc. (%)	Success
Baseline (SGD)	—	5/5	64.40 ± 0.71	5/5	78.68 ± 0.21	**10/10**
Accel. Only	3	5/5	58.04 ± 1.29	5/5	78.23 ± 0.64	16/20
	5	4/5	48.20 ± 8.19	2/5	10.00 ± 0.00 (chance)	
Consistency Only	3	0/5	*Crashed*	1/5	10.00 (chance)	1/20
	5	0/5	*Crashed*	0/5	*Crashed*	
Full PDT	3	5/5	$\mathbf{69.39 \pm 0.79}$	5/5	78.98 ± 0.18	**20/20**
	5	5/5	69.10 ± 1.33	5/5	$\mathbf{79.48 \pm 0.20}$	

The corresponding training curves are shown in Fig. 3.5. Full PDT maintains stable loss trajectories and low variance across seeds, whereas ablated variants either fail numerically or converge to inferior solutions. This supports the need to combine a scale-control criterion with a directional-consistency criterion.

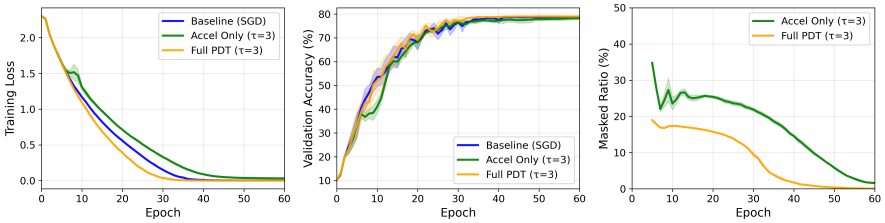
Figure 3.6 further decomposes the contribution of the masking criteria during successful full-PDT runs. The dynamic-consistency criterion remains active throughout training and serves as a directional filter. The acceleration-effectiveness criterion controls the update scale: the lower bound ensures that accepted predictions provide a meaningful advancement, while the upper bound rejects over-aggressive predictions, especially near convergence.



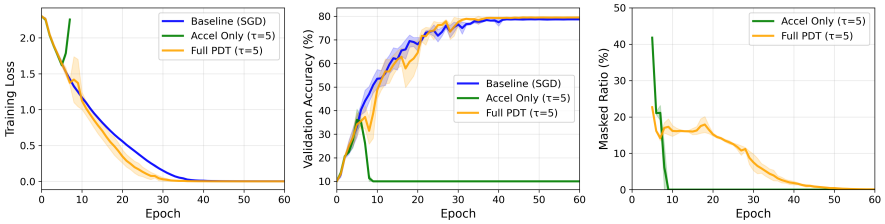
(a) LR=0.01, prediction step (τ) = 3.



(b) LR=0.01, prediction step (τ) = 5.

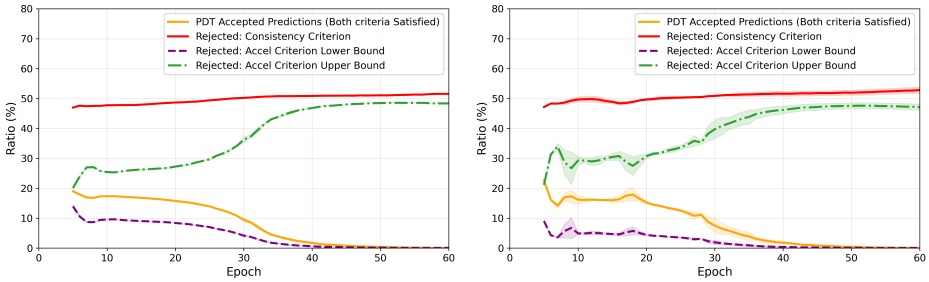


(c) learning rate 0.05, prediction step (τ) = 3.



(d) learning rate 0.05, prediction step (τ) = 5.

Figure 3.5: Mean training curves with standard-deviation shading for baseline optimization and PDT variants with different masking criteria. Trained on CIFAR-10 using AlexNet, batch size 256, with a cosine-annealing learning-rate scheduler. Only successful runs are included.



(a) learning rate 0.05, prediction step (τ) = 3.

(b) learning rate 0.05, prediction step (τ) = 5.

Figure 3.6: Temporal evolution of masking-criteria contributions during training. Results are from full PDT with both criteria applied. Trained on CIFAR-10 using AlexNet, learning rate 0.05, batch size 256, with a cosine-annealing learning-rate scheduler. Only successful runs are included.

3.3.4 Plug-in Generality Across Optimizers

PDT is also evaluated as a plug-in on top of several baseline optimizers, including SGD, SGD with momentum, Adam, AdamW, RMSProp, Shampoo, and LAMB. The goal of this experiment is not to rank optimizers, but to test whether the PDT mechanism can be inserted into different optimizer-induced trajectories. Table 3.3 reports final accuracy, best training loss, TTB-Loss, and runtime reduction on AlexNet trained on CIFAR-10.

Across all listed optimizers, PDT reduces the time required to reach the baseline optimizer's best training loss, with final accuracy preserved or slightly improved in these runs. Since each optimizer induces a different update trajectory, these results suggest that the DMD prediction and masking mechanism can be applied as an optimizer-agnostic wrapper, rather than being tied to a single baseline update rule. The variation in runtime reduction should be interpreted as optimizer- and configuration-dependent behavior, not as an optimizer ranking.

Table 3.3: Impact of baseline optimizers on PDT performance. Trained on CIFAR-10 using AlexNet, batch size 256, momentum 0.9 where applicable, with a cosine-annealing learning-rate scheduler.

Optimizer	lr	Final Accuracy	Best Train Loss	TTB-Loss (s)	Runtime Reduction (%)
SGD	0.1	0.7930 ± 0.0023	0.0002 ± 0.0000	665.27 ± 9.08	**19.67**
PDT		0.7978 ± 0.0032	0.0002 ± 0.0000	534.41 ± 12.64	
Momentum	0.001	0.6672 ± 0.0068	0.8609 ± 0.0166	752.74 ± 9.62	**41.06**
PDT		0.7298 ± 0.0051	0.5358 ± 0.0165	443.68 ± 8.75	
Adam	0.0005	0.7952 ± 0.0063	0.0001 ± 0.0000	779.13 ± 11.81	**14.87**
PDT		0.8050 ± 0.0050	0.0002 ± 0.0000	663.28 ± 15.30	
AdamW	5e-5	0.8031 ± 0.0021	0.0077 ± 0.0002	652.92 ± 5.26	**28.36**
PDT		0.8149 ± 0.0037	0.0013 ± 0.0002	467.76 ± 6.05	
RMSProp	0.0001	0.7996 ± 0.0032	$(4.1 \pm 1.6) \times 10^{-5}$	661.08 ± 0.42	**15.35**
PDT		0.8108 ± 0.0026	$(2.4 \pm 0.6) \times 10^{-5}$	559.61 ± 0.00	
Shampoo	0.001	0.8012 ± 0.0071	0.0040 ± 0.0005	736.56 ± 10.58	**16.03**
PDT		0.8101 ± 0.0043	0.0033 ± 0.0003	618.49 ± 12.72	
LAMB	0.001	0.8034 ± 0.0025	0.1215 ± 0.0085	663.25 ± 5.44	**44.24**
PDT		0.8140 ± 0.0027	0.0036 ± 0.0006	369.82 ± 10.59	

3.3.5 Generalization to Self-Supervised Learning

To test whether PDT is restricted to supervised classification losses, we also evaluate it in a self-supervised learning setting using SimSiam [8] with a ResNet-18 backbone on CIFAR-10. SimSiam differs from supervised training because its dynamics are shaped by a stop-gradient mechanism and a negative cosine similarity objective. The experiment therefore tests whether PDT can accelerate a qualitatively different training objective.

Table 3.4: Performance comparison of SimSiam pre-training on CIFAR-10 with a ResNet-18 backbone, trained for 200 epochs, learning rate 0.03, batch size 256, momentum 0.9, with a cosine-annealing learning-rate scheduler.

Optimizer	Final Accuracy	TTB-Loss (s)	TTB-Acc (s)	Runtime Reduction (%)	
				Train Loss	Val. Acc.
SGD with momentum	0.7285 ± 0.0166	9611.35 ± 837.98	7353.12 ± 1063.96	**48.78**	**16.10**
PDT	0.7685 ± 0.0144	4922.92 ± 712.55	6169.04 ± 1142.97		

Table 3.4 shows that PDT also reduces time to target metrics in the self-supervised setting, while preserving representation quality as measured by downstream validation accuracy. This result indicates that PDT is not limited to standard supervised cross-entropy training, although the scope of this experiment is one representative SSL configuration rather than a comprehensive survey of self-supervised methods.

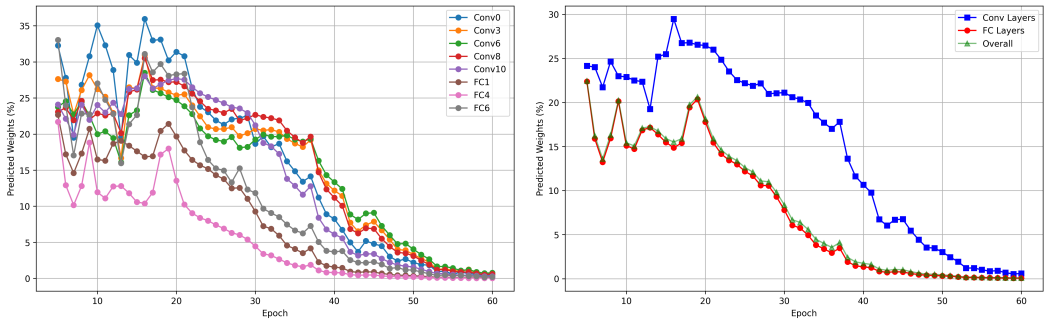
3.3.6 Mask Dynamics and Distribution

The accepted prediction ratio over training is shown alongside the convergence curves in Fig. 3.4. This ratio is determined by the masking criteria in Section 3.2.2; it should be interpreted as the fraction of coordinates accepted by PDT's reliability filter, not as a ground-truth measure of predictability.

Across the models in Fig. 3.4, the accepted ratio is higher near the beginning of training and generally decreases as training proceeds. This trend is consistent with the changing character of optimization: early updates often contain larger and

more coherent directional signals, while later-stage updates near a local minimum become smaller and more oscillatory. Smaller models on simpler tasks, such as FCN and AlexNet on CIFAR-10, show a more gradual decrease. Larger models and more complex settings, such as ResNet-50 and ViT on ImageNet, show a sharper reduction, indicating that a smaller fraction of coordinates satisfies the PDT acceptance criteria as training proceeds. This acceptance ratio is specific to PDT's predictor and safeguards and should not be interpreted as a direct measurement of trajectory-wide predictability.

Layer-wise mask distributions for AlexNet are shown in Fig. 3.7. The convolutional layers maintain a higher accepted prediction ratio than fully connected layers for much of training, although individual early layers can show noticeable fluctuations. Since most AlexNet parameters are in the fully connected layers, the overall accepted ratio closely follows the behavior of those layers. These layer-wise differences provide one source of evidence that predictive reliability is spatially nonuniform within a network.



(a) **(b)**

Figure 3.7: Analysis of mask distribution in AlexNet. (a) Layer-wise mask evolution over training epochs. (b) Comparison of prediction ratios between convolutional and fully connected layers.

These observations do not by themselves provide a complete measurement framework for predictability. Rather, they show how the PDT reliability filter behaves inside the acceleration method. Chapter 4 builds on this observation by

measuring structured predictability more systematically across parameters, layers, training stages, architectures, and optimizers.

3.3.7 Sensitivity to Prediction Hyperparameters

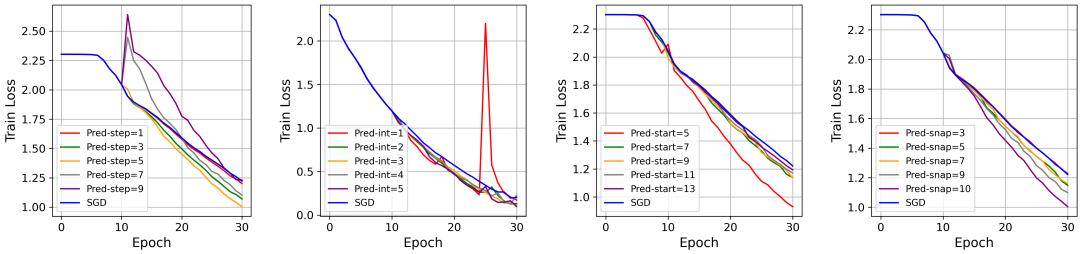
PDT introduces several prediction-related hyperparameters: the prediction horizon τ, the prediction interval T_i, the starting epoch T_0, and the history length h. These parameters control how far the method predicts, how often prediction is attempted, when prediction begins, and how much recent history is used to fit the DMD model.

Prediction horizon (τ). The prediction horizon determines the number of future epochs being forecast by the DMD module. As shown in Fig. 3.8(a), increasing τ can increase acceleration up to a practical range, but overly long horizons can introduce larger prediction errors and destabilize training.

Prediction interval (T_i). The prediction interval controls how frequently PDT attempts prediction. A shorter interval can increase acceleration when predictions are reliable, but it also increases the number of prediction attempts and can expose training to more unreliable forecasts later in training. Fig. 3.8(b) shows that the prediction interval therefore mediates the trade-off between aggressiveness and stability.

Starting epoch (T_0). Prediction cannot begin until sufficient snapshots have been collected. In practice, T_0 also controls whether early transient dynamics are included in the first DMD fit. Fig. 3.8(c) shows the effect of changing this starting point.

History length (h). The history length controls how many past snapshots are used to fit the local DMD model. If h is too small, the snapshot window provides insufficient information about recent dynamics. If it is too large, the window may average over changes in local dynamics and produce a less responsive predictor. Fig. 3.8(d) illustrates this sensitivity.



(a) learning rate 0.01, int=1, start=10, snap=10. **(b)** learning rate 0.05, step=9, start=10, snap=10. **(c)** learning rate 0.01, int=1, step=5, snap=5. **(d)** learning rate 0.01, step=5, int=1, start=10.

Figure 3.8: Influence of PDT prediction hyperparameters. (a) Prediction horizon, (b) prediction interval, (c) starting epoch, and (d) past snapshot count. Trained on CIFAR-10 using AlexNet, batch size 256.

The interaction between learning rate and accepted prediction ratio is reported in Table 3.5 and Fig. 3.9. Larger learning rates tend to produce higher early-stage accepted ratios, while very small learning rates suppress prediction usage throughout training. In the tested AlexNet/CIFAR-10 setting, PDT improves final accuracy over the baseline across the learning rates in the table.

Table 3.5: Mask acceptance ratio and validation accuracy across learning rates. Mask acceptance ratio is reported by training stage and final test accuracy is reported as mean ± std across three seeds. Training stages: Early (epochs 5–20), Mid (epochs 21–40), Late (epochs 41–60).

LR	Early (%)	Mid (%)	Late (%)	Overall (%)	Base Acc (%)	PDT Acc (%)
0.001	0.06 ± 0.00	0.07 ± 0.00	0.10 ± 0.02	0.08 ± 0.01	11.83 ± 1.04	14.92 ± 2.44
0.01	8.12 ± 0.20	7.10 ± 0.06	$\mathbf{2.77 \pm 0.02}$	5.77 ± 0.07	63.96 ± 0.14	68.43 ± 0.64
0.05	16.94 ± 0.20	$\mathbf{8.52 \pm 0.5}$	0.68 ± 0.15	$\mathbf{7.94 \pm 0.31}$	78.67 ± 0.13	$\mathbf{79.24 \pm 0.22}$
0.1	$\mathbf{18.77 \pm 0.24}$	7.10 ± 0.52	0.37 ± 0.05	7.91 ± 0.20	79.76 ± 0.27	$\mathbf{80.06 \pm 0.20}$

The contribution of the acceleration and consistency criteria across learning rates is further decomposed in Fig. 3.10. The lower-bound component of the acceleration-effectiveness criterion is most relevant early in training, while the upper-bound component becomes increasingly important near convergence, when gradients are smaller and over-aggressive predictions are more likely to destabilize training.

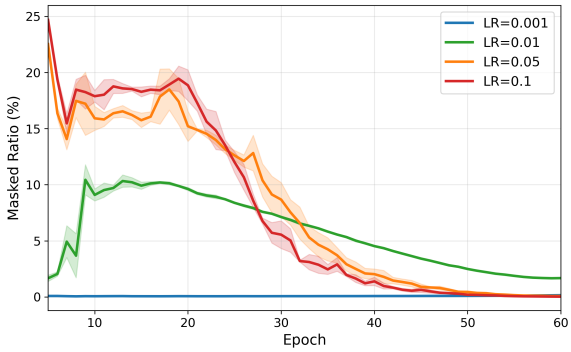


Figure 3.9: Mask acceptance ratio evolution across different learning rates (mean ± std shading across three seeds). Trained on CIFAR-10 using AlexNet, batch size 256, with a cosine-annealing learning-rate scheduler.

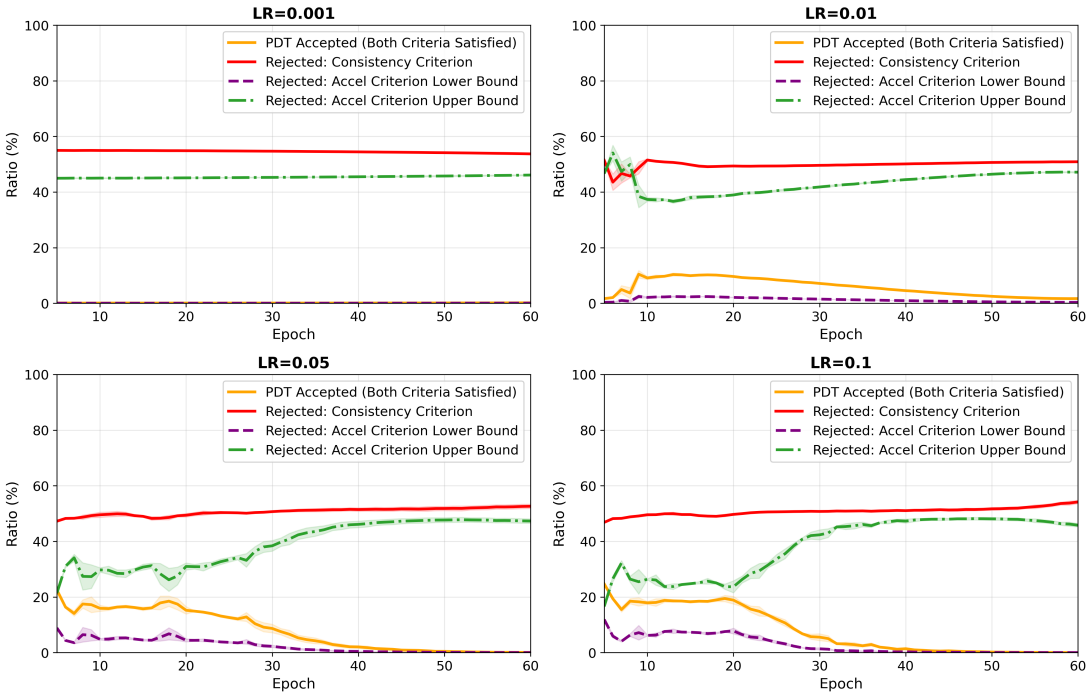


Figure 3.10: Temporal evolution of masking-criteria contribution across learning rates. Trained on CIFAR-10 using AlexNet, batch size 256, with a cosine-annealing learning-rate scheduler.

Supplementary robustness. Additional robustness experiments under non-i.i.d. training data are reported in Appendix A.6.1. These experiments are not included in the main evidence chain because they are supplementary to the primary PDT design claims, but they provide additional evidence that PDT remains effective when the stochastic-gradient sequence deviates from standard i.i.d. mini-batch sampling.

3.4 Summary and Limitations

This chapter presented Predictive Differential Training (PDT), a selective predictive acceleration method for neural network training. PDT uses a Koopman/DMD-based predictor to generate short-horizon candidate weight forecasts from recent training trajectories, and applies these forecasts only through a coordinate-wise masking mechanism. The mask combines acceleration effectiveness and dynamic consistency criteria, so that accepted coordinates receive predicted updates while unaccepted coordinates fall back to the baseline optimizer. In this way, PDT retains the baseline training trajectory as a stabilizing reference while using prediction only where the local dynamics satisfy the reliability criteria.

The experimental results show that PDT can reduce time to target loss and validation accuracy across the tested supervised settings, including CIFAR-10 and ImageNet-1K, and across architectures ranging from FCN and AlexNet to ResNet-50, ViT-Base, and ViT-Huge. In these experiments, PDT achieves approximately 10–40% net wall-clock reduction in time-to-baseline targets, with all reported runtime measurements including the overhead of DMD computation, prediction, mask construction, and rejected prediction attempts. The ablation results further show that both masking criteria are needed: scale control alone can admit over-aggressive predictions that destabilize training at larger horizons, while dynamic consistency alone fails in nearly all trials; neither criterion is sufficient on its own. Additional

optimizer and self-supervised experiments suggest that PDT can be used as a plug-in mechanism across several optimizer-induced trajectories and at least one non-supervised training objective.

Several limitations remain. PDT introduces additional hyperparameters, including the history length, prediction horizon, burn-in period, and prediction interval. Although the experiments indicate that PDT is effective over a practical range of these values, they still influence the aggressiveness and stability of prediction. PDT also requires storing recent weight snapshots and performing DMD computation during training. These costs are included in the reported wall-clock measurements and are amortized in the tested settings, but they remain implementation considerations for regimes with very short epochs, limited memory, or extremely large parameter counts. More memory-efficient variants, such as streaming or incremental DMD implementations [29], may reduce this cost in future systems.

A more fundamental limitation is that PDT provides an online reliability filter, not a complete measurement of predictability in neural training dynamics. The coordinate-wise mask identifies which predicted coordinates satisfy the two acceptance criteria at a particular prediction event, but it does not fully explain where predictability comes from, how it varies across layers or training stages, or how architecture and optimizer choices shape predictable regions of the trajectory. The mask is therefore useful as a practical control mechanism, but it should not be interpreted as a complete map of training-dynamics predictability.

These limitations shift the remaining question from acceleration design to measurement. PDT shows that selective prediction can accelerate training when its coordinate-wise reliability criteria identify usable forecasts, while the variation of the accepted ratio across layers and training stages suggests that forecast reliability is neither uniform nor stationary. Chapter 4 therefore studies short-horizon temporal structure directly, using retrospective probes to determine where it appears across parameter scales, training stages, architectures, and optimizers.

Chapter 4

Measuring Structured Predictability in Neural Training Dynamics

4.1 From Selective Prediction to Predictability Measurement

Chapter 3 developed Predictive Differential Training (PDT) as a practical framework for applying prediction selectively during neural network training. PDT uses a coordinate-wise mask to accept a candidate forecast only when it satisfies the acceleration-effectiveness and dynamic-consistency criteria; otherwise, the corresponding coordinates remain under the baseline optimizer. This mask provides an operational rule for runtime acceptance, but it is not designed to characterize how short-horizon predictability is organized in the training dynamics. This chapter takes the PDT mask as a starting point and asks a broader measurement question: where, when, and under what training conditions do recent updates retain information about near-future parameter motion?

The background in Chapter 2 establishes that neural training contains geometric, low-dimensional, and phase-dependent structure, but it does not show whether a short update history is informative about nearby future motion. Accordingly, this

chapter does not seek a stronger forecaster or an optimizer ranking. It measures how short-horizon temporal structure is distributed within training trajectories and how that structure changes with optimization.

Short-horizon temporal structure can be represented at different levels of complexity. At the simplest level, recent motion can be summarized by whether its latest displacement direction persists over the near future. A richer description retains the low-dimensional span of several recent updates, allowing future motion to remain structured even when no single direction persists. Explicit local models can then test whether coupled temporal evolution supported by this history is recoverable. The scope is therefore limited to low-order directional relations and locally low-dimensional structure recoverable from short histories, rather than arbitrary nonlinear predictability. Within this scope, the chapter uses three related but non-equivalent probe families of increasing representational complexity: displacement-direction probes, subspace-residual probes, and predictor-based probes. Agreement within and across the families tests whether measured patterns recur beyond a single construction, while their differences distinguish structure visible only through richer representations. The probes are evaluated at aggregate, tensor, and within-tensor group scales under explicit grouping and calibration conventions.

The empirical analysis combines controlled multi-pass vision training on CIFAR with public one-pass Pythia pretraining trajectories. The controlled vision panel supports matched comparisons across architectures and training recipes, while the Pythia checkpoints provide long, parameter-resolved trajectories from language-model pretraining. Together, the two settings test whether the same measurement framework remains informative across substantially different training regimes, while allowing regime-specific analyses of how short-horizon organization evolves with the realized optimization trajectory.

Section 4.2 defines the measurement units, probe families, grouping conventions, and calibration rules. Section 4.3 analyzes probe relations, spatial localization, optimization evolution, training-condition dependence, and an illustrative bridge

to selective prediction. Section 4.4 synthesizes these results as a retrospective, parameter-resolved diagnostic and delineates their conditional relevance to predictive reuse.

4.2 Measurement Framework

This section formalizes the measurement units, probe constructions, grouping conventions, and calibration rules used throughout the chapter. All probe constructions use only updates preceding an anchor, and their scores are evaluated retrospectively against the displacement realized after that anchor. This protocol supports analysis across training stages and spatial granularities without modifying the underlying optimization process.

4.2.1 Anchors, Histories, and Measurement Units

Let $w_t \in \mathbb{R}^d$ denote the trainable parameters at saved checkpoint t. In this chapter, t indexes recorded checkpoints rather than individual mini-batch iterations. The displacement between consecutive saved checkpoints is

$$\Delta w_t = w_{t+1} - w_t. \tag{4.1}$$

For an anchor checkpoint a and measurement horizon τ, the realized future displacement is

$$\Delta w_a^{(\tau)} = w_{a+\tau} - w_a. \tag{4.2}$$

All probes compare information available before the anchor with this realized future displacement.

The history length n_{past} specifies how many past checkpoint-to-checkpoint displacements are used to form the local history matrix:

$$H_a = \left[\Delta w_{a-n_{\text{past}}}, \Delta w_{a-n_{\text{past}}+1}, \ldots, \Delta w_{a-1}\right]. \tag{4.3}$$

A measurement unit S is either a whole trainable tensor or one of K disjoint within-tensor parameter groups. After vectorizing the unit, $\Delta w_{t,S}$, $\Delta w_{a,S}^{(\tau)}$, and $H_{a,S}$ denote the corresponding restrictions of the one-step update, realized future displacement, and local history matrix. For group-level analysis, a tensor is partitioned into $\{G_g\}_{g=1}^{K}$ and every probe is evaluated separately on each group. Every probe uses only history available before a and is scored retrospectively against the displacement after a. Architectural information may be used to define measurement units or grouping conventions, but it does not enter the probe definitions or calibration thresholds. Architectural labels are used to organize and interpret the resulting measurements.

Concrete choices of τ, n_{past}, K, and the grouping convention are specified in the empirical analysis and robustness studies.

4.2.2 Displacement-Direction Probes

Displacement-direction probes use the simplest representation considered in this chapter. They construct no explicit prediction and ask whether the latest observed update persists into the displacement realized over the next τ checkpoints. For element j of a measurement unit S, define the direction-agreement indicator

$$e_{a,S,j}^{(\tau)} = \mathbf{1}\Big\{\operatorname{sign}(\Delta w_{a-1,S,j}) = \operatorname{sign}(\Delta w_{a,S,j}^{(\tau)})\Big\}. \tag{4.4}$$

At the element level, we measure this behavior using **element-wise sign agreement** (ESA):

$$\mathrm{ESA}_S(a,\tau) = \frac{1}{|S|}\sum_{j\in S} e_{a,S,j}^{(\tau)}. \tag{4.5}$$

ESA treats all elements in S equally and measures whether their most recent update direction persists toward the future displacement.

A corresponding vector-level version is **vector cosine similarity** (VCS):

$$\mathrm{VCS}_S(a,\tau) = \frac{\left\langle \Delta w_{a-1,S}, \Delta w_{a,S}^{(\tau)} \right\rangle}{\|\Delta w_{a-1,S}\|_2 \left\|\Delta w_{a,S}^{(\tau)}\right\|_2}. \tag{4.6}$$

Unlike ESA, VCS is magnitude-weighted through the vector inner product. Thus, ESA measures element-wise directional persistence, whereas VCS measures vector-level alignment dominated more strongly by coordinates with larger update magnitudes.

4.2.3 Subspace-Residual Probes

A richer representation retains the span of several recent updates rather than only the latest direction. Subspace-residual probes ask whether the realized future displacement remains within this locally low-dimensional span, even when no single recent direction persists.

For a measurement unit S, let

$$H_{a,S} \in \mathbb{R}^{|S| \times n_{\mathrm{past}}} \tag{4.7}$$

be the restricted history matrix. We compute its thin singular value decomposition:

$$H_{a,S} = U_{a,S} \Sigma_{a,S} V_{a,S}^{\top}. \tag{4.8}$$

Let $U_{r,a,S}$ denote the retained left singular vectors after applying the Gavish–Donoho hard-threshold rule [22], and define the projection onto the retained history subspace as

$$P_{a,S} = U_{r,a,S} U_{r,a,S}^{\top}. \tag{4.9}$$

We measure the fraction of future displacement lying outside this recent-update subspace using the **future residual ratio** (FRR):

$$\mathrm{FRR}_S(a,\tau) = \frac{\left\|(I - P_{a,S})\Delta w_{a,S}^{(\tau)}\right\|_2}{\left\|\Delta w_{a,S}^{(\tau)}\right\|_2}. \tag{4.10}$$

A lower FRR indicates that the actual future displacement is better contained in the span of recent updates, while a higher FRR indicates stronger deviation from the local history subspace. The cell-wise rank-selection procedure and the stated full-history-rank exceptions for group-level analyses are documented in Appendix B.2.3.

An *in-window* control uses the same construction but evaluates the latest observed update $\Delta w_{a-1,S}$ against the span of the preceding $n_{\mathrm{past}} - 1$ updates. It therefore tests local subspace consistency within the observed history, whereas the out-of-window FRR in Eq. 4.10 evaluates continuation beyond the anchor.

For grouped analysis, we summarize this structure using the **low-residual group fraction** (LRGF):

$$\mathrm{LRGF}(a,\tau;\theta) = \frac{1}{K}\sum_{g=1}^{K} \mathbf{1}\left\{\mathrm{FRR}_{G_g}(a,\tau) < \theta\right\}. \tag{4.11}$$

Here, θ is a diagnostic threshold specified in the experimental protocol rather than a universal boundary. Thus, FRR measures subspace containment for a given unit or group, while LRGF summarizes how broadly low-residual behavior appears across groups.

Because a fixed threshold can also be passed by chance, we additionally use a null-calibrated readout, $\mathrm{LRGF}_{\mathrm{cal}}$. Its threshold is the $\alpha = 0.05$ quantile of a size-matched permutation-null FRR distribution. Under the null, the expected selected fraction is therefore approximately α; where indicated, we report the excess over α rather than interpreting the raw fraction without calibration.

4.2.4 Predictor-Based Probes

Predictor-based probes use explicit local models to test whether richer structure in the recent history can be exploited. A predictor maps the observed history to a predicted future displacement $\widehat{f}^{(\tau)}_{\mathrm{P},a,S}$, where $\mathrm{P} \in \{\mathrm{in}, \mathrm{lin}, \mathrm{DMD}\}$. We use three increasingly expressive models: an inertia predictor, a per-element linear-trend predictor, and a DMD-based local dynamics predictor.

At the same time, the magnitude of a predicted displacement is difficult to interpret in isolation. It depends on the learning-rate schedule, optimizer state, regularization, and the chosen prediction horizon. A prediction with imperfect scale may still point in a useful future direction, while a directionally reversed prediction directly conflicts with the realized trajectory. We therefore evaluate predictor outputs through both selector-free directional agreement and predictor-guided selection.

The **inertia predictor** assumes that the most recent update persists over the horizon:

$$\widehat{f}^{(\tau)}_{\mathrm{in},a,S} = \tau \Delta w_{a-1,S}. \tag{4.12}$$

The **linear-trend predictor** fits an element-wise affine trend to the recent one-step displacements. With local history index $s = 0, \ldots, n_{\mathrm{past}} - 1$, it fits

$$\Delta w_{a-n_{\mathrm{past}}+s,S} \approx \alpha_{a,S} + s\beta_{a,S}, \tag{4.13}$$

and integrates the fitted update trend over the next τ steps:

$$\widehat{f}^{(\tau)}_{\mathrm{lin},a,S} = \sum_{q=0}^{\tau-1} \left(\widehat{\alpha}_{a,S} + \widehat{\beta}_{a,S}(n_{\mathrm{past}} + q) \right). \tag{4.14}$$

This predictor models a linear trend in the checkpoint-to-checkpoint displacements, rather than directly extrapolating the checkpoints.

The **DMD-based predictor** fits a local linear dynamics model to the recent update sequence. This use of DMD follows the dynamical-systems viewpoint reviewed

in Section 2.3, but differs from the PDT forecaster in Chapter 3: here DMD models update dynamics for measurement, rather than producing a weight-snapshot forecast to be written back into training. Define consecutive history matrices

$$\begin{aligned} X_{a,S} &= \left[\Delta w_{a-n_{\text{past}},S}, \ldots, \Delta w_{a-2,S}\right], \\ Y_{a,S} &= \left[\Delta w_{a-n_{\text{past}}+1,S}, \ldots, \Delta w_{a-1,S}\right]. \end{aligned} \tag{4.15}$$

DMD fits the local map

$$A_{a,S} = Y_{a,S} X_{a,S}^{\dagger}. \tag{4.16}$$

Starting from the most recent observed displacement, the learned map generates a sequence of predicted future displacements, which are accumulated over the horizon:

$$\widehat{f}^{(\tau)}_{\text{DMD},a,S} = \sum_{k=1}^{\tau} A_{a,S}^{k} \Delta w_{a-1,S}. \tag{4.17}$$

Equations (4.15)–(4.17) give the compact conceptual construction. The implementation row-standardizes the history, fits and rolls out the map in reduced SVD coordinates, applies explicit singular-value and degeneracy guards, and then restores the original coordinate scales; the exact construction is given in Appendix B.2.2.

The first predictor-based readout is the selector-free **predicted-direction cosine**:

$$\cos_{\text{P},S}(a,\tau) = \frac{\left\langle \widehat{f}^{(\tau)}_{\text{P},a,S}, \Delta w^{(\tau)}_{a,S} \right\rangle}{\left\| \widehat{f}^{(\tau)}_{\text{P},a,S} \right\|_2 \left\| \Delta w^{(\tau)}_{a,S} \right\|_2}. \tag{4.18}$$

Here, $\text{P} \in \{\text{in}, \text{lin}, \text{DMD}\}$. Because cosine ignores scale and the inertia prediction is a positive multiple of the latest update, $\cos_{\text{in},S}(a,\tau) \equiv \text{VCS}_S(a,\tau)$.

Given a predicted future displacement $\widehat{f}^{(\tau)}_{a,S}$, we also define a selector by agreement between the predicted future direction and the last observed update direction:

$$m_{a,S,j} = \mathbf{1}\left\{ \text{sign}(\widehat{f}^{(\tau)}_{a,S,j}) = \text{sign}(\Delta w_{a-1,S,j}) \right\}. \tag{4.19}$$

This selector is not evaluated in isolation, because some units may already have strong directional persistence without predictor-guided selection. We therefore compare the selected subset against the unselective ESA baseline. The **selected element-wise sign agreement** (SESA) is

$$\mathrm{SESA}_S(m; a, \tau) = \frac{\sum_{j \in S} m_{a,S,j} e^{(\tau)}_{a,S,j}}{\sum_{j \in S} m_{a,S,j}}, \tag{4.20}$$

when the selected set is nonempty. We then report the **selectivity lift** over the unselective ESA baseline:

$$\mathrm{Lift}_S(m; a, \tau) = \mathrm{SESA}_S(m; a, \tau) - \mathrm{ESA}_S(a, \tau). \tag{4.21}$$

A positive lift indicates that the predictor selects a subset whose recent update direction is more reliable for the future displacement than the measurement unit as a whole. In this sense, lift measures whether the predictor provides additional directional selectivity beyond simple direction persistence.

Under this construction, the DMD prediction remains in the span of the recent update history. Its attainable directional agreement is therefore linked to FRR: a large component of the realized future displacement outside the history span limits how well any history-span-constrained DMD prediction can align with that future displacement.

4.2.5 Grouping Conventions and Reference Calibration

The grouping convention is part of the measurement definition because it determines the spatial scale at which local subspace structure can be resolved. Consequently, FRR- and LRGF-based results are interpreted only together with their stated grouping convention. Both flat and architecture-aligned cuts are considered, but they are not interchangeable: changing group membership and group size changes the chance floor, absolute level, and in some comparisons even the ordering. On

the CIFAR side, flat $K = 16$ groups support convention-matched panel analyses, while architecture-aligned and magnitude-based cuts test structure along native model axes. On the Pythia side, architecture-aligned cuts are emphasized because they preserve the organization of large tensors and support role-, layer-, and group-resolved interpretation. Every group-level claim therefore travels with its cut, and raw levels are not compared across cuts. The complete grouping definitions and sensitivity analysis are provided in Appendices B.2.1 and B.2.6.

The two LRGF readouts serve different comparison goals. The canonical fixed-threshold readout uses $\theta = 0.7$ only under an explicitly shared grouping convention and is interpreted together with the matched null pass rate; a raw fraction is treated as an absolute group population only when that null pass rate is negligible. The calibrated readout replaces the fixed threshold by the $\alpha = 0.05$ quantile of a size-matched membership-resampling null and reports excess over α. This calibration supports comparisons within a stated cut and geometry, but it does not make raw levels from different cuts interchangeable. The null construction and reporting rules are given in Appendices B.2.4 and B.2.7. More generally, probe construction uses only information available before the anchor, but the resulting scores are retrospective because evaluation requires the subsequently realized displacement.

4.3 Empirical Analysis

4.3.1 Experimental Settings and Measurement Protocol

We analyze stored parameter trajectories in two complementary training regimes: controlled multi-pass vision training and public one-pass language-model pretraining. In both regimes, probe construction and all probe inputs use only history preceding an anchor. The reported scores are nevertheless retrospective because they are evaluated against the displacement realized after that anchor.

Multi-pass vision training. The main CIFAR-10 panel contains 30 model–optimizer cells, crossing five architectures—a CIFAR-style ResNet, ViT-tiny, ViT-mid, AlexNet, and an MLP—with six training configurations: SGD, SGD with momentum, SGD with weight decay, SAM, Adam, and AdaGrad. Each model is trained for 50 epochs with batch size 128 and a cosine learning-rate schedule, and parameters are recorded at every epoch boundary. Unless otherwise stated, probes are evaluated at anchor epochs $\{5, 10, 20, 30, 40\}$. Additional analyses in Section 4.3.5 use Muon and controlled momentum- and weight-decay-dose arms; these additional arms are not included in the 30-cell panel.

One-pass language-model pretraining. The main pretraining panel analyzes the public checkpoint trajectories of Pythia 70M, 160M, and 410M [5]. After the dense early release grid, checkpoints are spaced by 1,000 optimization steps. For each main-panel model, the standard protocol evaluates 123 admissible post-warmup anchors spanning steps 7,000–129,000, unless a specific analysis uses a subset of scales or anchors. Appendix B.1.3 extends the same accounting to 14M, 31M, 1B, and 1.4B and records the exclusion rules for the defective 1B checkpoint.

Measurement clocks and shared protocol. Unless otherwise stated, both regimes use horizon $\tau = 5$ and history length $n_{\text{past}} = 5$. The measurement clock is regime-specific: one probe step corresponds to one epoch for CIFAR and one released checkpoint for Pythia. Thus, the nominal horizon spans five epochs in the CIFAR runs and five checkpoint intervals, or 5,000 optimization steps, in the Pythia trajectories. The shared index-space protocol standardizes the local construction of the probes but does not make the two measurement clocks directly comparable.

Grouping and calibration. Both flat partitions and groupings aligned with model architecture are considered. Convention-matched vision-panel analyses use flat $K = 16$ groups, while structural and magnitude-based cuts are used to test

whether a native model axis exposes localized coherence that the flat cut may obscure. These cuts answer different scale-specific questions and are not treated as equivalent. In the Pythia analyses, architecture-aligned groupings are emphasized within the qkv, attention-output, MLP, and vector segments, including row- and head-aligned cuts where appropriate. Architectural information may define measurement units or grouping conventions, but it does not enter the probe formulas or calibration thresholds.

FRR- and LRGF-based results therefore state the grouping convention explicitly. The canonical fixed-threshold readout uses $\theta = 0.7$ and is interpreted with its matched null pass rate. For localized within-layer and cross-size analyses, $\mathrm{LRGF}_{\mathrm{cal}}$ uses the size-matched membership-resampling null specified in Appendix B.2.4.

Cross-regime interpretation. Because the checkpoint clocks and grouping conventions differ, cross-regime comparisons concern recurring within-trajectory orderings and qualitative organization rather than equality of absolute probe magnitudes. The complete vision recipes, additional run families, pretraining checkpoint accounting, and anchor construction are provided in Appendices B.1.1–B.1.4.

4.3.2 Probe Families Are Coherent but Non-equivalent

With these measurement conventions fixed, we first ask whether the probe panel recovers a coherent but non-redundant organization of short-horizon structure. We test alternative readouts within each family under their stated conventions and then compare representative readouts across families to determine which relations are shared and which remain distinct. These analyses use the controlled CIFAR-10 panel of 30 model–optimizer cells and five anchors described in Section 4.3.1.

Within-Family Coherence under Explicit Conventions

Figure 4.1 summarizes one within-family comparison for each probe family. The purpose is to test whether the measured patterns recur under alternative readouts rather than arising from the idiosyncratic behavior of a single score. Coherence within a family does not make its readouts interchangeable, but it provides convergent evidence that the observed structure is not merely an artifact of one formula or scoring convention.

Displacement-direction probes. ESA and group-mean VCS evaluate the same pair of displacements but weight their coordinates differently: ESA assigns one sign vote to each coordinate, whereas VCS weights coordinates by displacement magnitude. Under a centered bivariate Gaussian reference, the corresponding sign agreement and correlation satisfy [68]

$$\mathrm{ESA} = 1 - \frac{\arccos(\mathrm{VCS})}{\pi}. \tag{4.22}$$

As shown in Figure 4.1(a), 28 of the 30 cells closely follow this parameter-free curve across anchors. The principal departures occur in cells containing many coordinates whose displacement is zero or numerically negligible over one or both intervals. Such entries contribute little to VCS but remain sensitive to the coordinate-wise sign convention used by ESA. Recomputing both readouts after restricting the two displacement vectors to their common active coordinate support largely restores the agreement.

Subspace-residual probes. The out-of-window FRR compares the post-anchor displacement with the preceding history span, whereas the in-window control compares the latest observed update with the span of the earlier updates in the same history. The two readouts are strongly correlated, with a median per-cell Spearman coefficient of $\rho_S = 0.83$ (Figure 4.1(b)), consistent with similar local

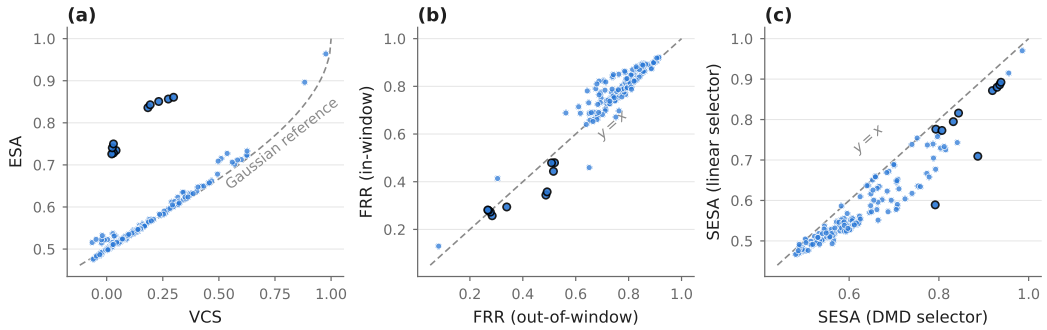


Figure 4.1: Within-family agreement on the controlled CIFAR-10 vision panel. (a) ESA versus group-mean VCS; (b) out-of-window versus in-window FRR; and (c) SESA under DMD and linear-trend selectors. Circled points indicate cells flagged by the near-zero-displacement diagnostic.

subspace structure across these neighboring target windows. For this comparison, FRR is computed within contiguous, structure-aligned parameter groups at the model-specific granularity used in the measurements study. This convention fixes the spatial scale at which local subspace structure is resolved.

Predictor-based probes. DMD and linear-trend selectors induce strongly consistent SESA orderings, with a median per-cell Spearman coefficient of $\rho_S = 0.89$. DMD also attains systematically higher SESA values (Figure 4.1(c)), consistent with its use of coupled low-rank dynamics in the recent trajectory rather than a separate linear trend for each coordinate. Alternative grouping granularities are analyzed in Appendix B.2.6, and the full multi-probe and protocol-robustness checks are reported in Appendix B.2.8. Across all three families, the recurring relationships support a shared underlying trajectory signal while preserving the measurement convention specific to each readout.

Cross-Family Relations across Levels of Complexity

We next compare representative readouts across probe families at the tensor level. For each model–optimizer cell, correlations are computed across tensors and then summarized by the median over the 30 cells. This cell-wise procedure avoids inducing

correlations by pooling measurements from different training recipes. Figure 4.2 summarizes the three representative cross-family relations.

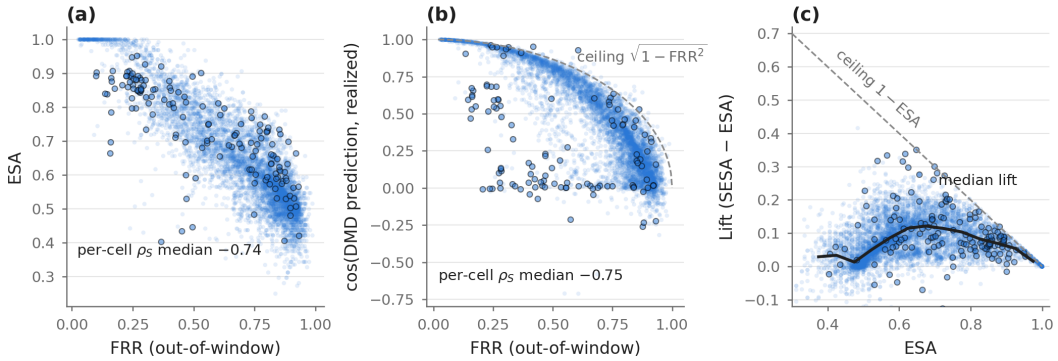


Figure 4.2: Cross-family relations on the controlled CIFAR-10 vision panel. (a) ESA versus out-of-window FRR; (b) out-of-window FRR versus DMD direction cosine; and (c) ESA versus DMD selectivity lift. Circled points indicate cells flagged by the near-zero-displacement diagnostic.

Direction–subspace relation. ESA and out-of-window FRR are strongly anticorrelated, with a median per-cell Spearman coefficient of $\rho_S = -0.74$ (Figure 4.2(a)). Tensors with more persistent coordinate-wise directions therefore also tend to keep their future displacement closer to the span of recent updates. The two probe families detect related temporal structure from different geometric perspectives, but the measurements are not identical: one evaluates persistence of the latest direction, whereas the other allows arbitrary combinations of directions in the recent-history span.

Subspace–predictor relation. Under the construction in Section 4.2.4, the DMD prediction lies in the recent-history subspace. The magnitude of its directional agreement with the realized future displacement is therefore bounded by

$$|\cos_{\mathrm{DMD}}| \leq \sqrt{1 - \mathrm{FRR}^2}. \tag{4.23}$$

DMD direction cosine is correspondingly strongly anticorrelated with FRR, with a median per-cell Spearman coefficient of $\rho_S = -0.75$ (Figure 4.2(b)). Across the tensor-level observations, 37% lie within 0.15 of the geometric bound. FRR therefore describes the directional headroom available to a predictor constrained to the recent-history subspace; it does not by itself specify which direction within that subspace the predictor will select.

Direction–predictor relation. Because selectivity lift is the improvement of SESA over ESA, it satisfies

$$\text{Lift} = \text{SESA} - \text{ESA} \leq 1 - \text{ESA}. \tag{4.24}$$

The binned median DMD lift peaks at 0.118 for ESA $\in [0.6, 0.7]$, rather than at the largest ESA values (Figure 4.2(c)). When ESA is already high, simple directional persistence leaves little residual error for predictor-based selection to remove, so a small lift in this region does not indicate weak predictability. At very low ESA, the trajectory also provides less stable structure for the predictor to exploit. The largest lift consequently occurs at intermediate ESA, where additional low-rank temporal structure can be recovered while sufficient selection headroom remains.

The resulting pattern is graded rather than interchangeable. Coordinate-wise persistence, history-subspace containment, and structure recoverable by an explicit local predictor overlap strongly enough to corroborate a shared temporal organization, yet their systematic differences show that no single readout subsumes the others. We therefore retain the three families as complementary diagnostics rather than collapsing them into a single predictability score.

4.3.3 Predictability Is Spatially Localized Across Parameter Scales and Training Regimes

With this coherent but non-redundant probe panel established, we next ask where the measured structure appears in parameter space. For a coarse tensor-scale summary, we interpret the resulting scores through two descriptive morphological classes. *Auxiliary parameters* comprise vector-like tensors such as normalization parameters and biases, whereas *bulk parameters* comprise matrix-like feature-transforming weights such as attention projections, MLP weights, and convolutional kernels. This classification is applied after measurement to summarize the spatial distribution of the scores. The vision analysis uses an operational role-class assignment that closely approximates the morphological definition, while the Pythia analysis uses an explicit segment taxonomy; the full attribution and audit are given in Appendix B.3. Viewed through this lens, the measurements reveal a recurring auxiliary–bulk separation across both multi-pass vision training and one-pass language-model pretraining.

Auxiliary–Bulk Separation across Training Regimes

Figure 4.3 compares matched auxiliary and bulk summaries within the controlled CIFAR panel and representative Pythia trajectories.

The paired comparisons show a stable tensor-scale separation within matched trajectories. Auxiliary parameters exceed bulk parameters in 29/32 displayed trajectory-level comparisons under ESA and in all 32/32 comparisons under the selector-free DMD direction cosine. Before temporal aggregation, the same ordering holds at all 246 model–checkpoint observations represented by the two Pythia trajectories shown in the panel under both probes. The paired ordering therefore recurs across multi-pass vision training and one-pass language-model pretraining, and it appears under both a displacement-direction probe and an explicit predictor-based probe. This agreement indicates that the coarse auxiliary–bulk separation is not specific to a single readout or training regime. It does not, however, isolate tensor

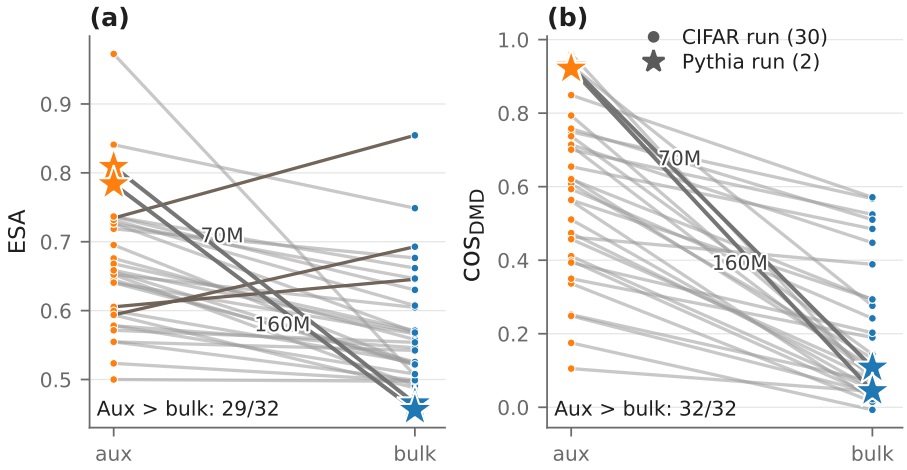


Figure 4.3: Auxiliary–bulk separation across training regimes. Each line connects matched auxiliary and bulk summaries within one CIFAR-10 model–optimizer cell or one displayed Pythia trajectory. Circles denote CIFAR-10 cells and stars denote representative Pythia trajectories. Cross-regime comparisons concern within-trajectory ordering rather than absolute probe values.

morphology as the causal source of the separation. In the released Pythia recipe, the auxiliary vector segment is exempt from weight decay while the matrix segments are not, so the paired ordering is interpreted as a descriptive role-and-recipe contrast rather than a shape-only mechanism.

The broader analysis also includes additional Pythia scales; only two trajectories are drawn in the main panel so that the paired markers remain legible. We do not compare the numerical sizes of the CIFAR and Pythia gaps because the two regimes use different measurement clocks and grouping conventions. The full partition audit and cross-scale extension are provided in Appendices B.3 and B.4.3.

Predictable Pockets within Bulk Parameters

The lower aggregate predictability of bulk parameters does not imply that every bulk region is equally irregular. At within-tensor resolution, a small minority of structurally defined groups exhibit substantially elevated short-horizon predictability. We refer to these localized, time-varying subsets as *predictable pockets*. Figure 4.4 summarizes their scarcity under one explicitly stated structural convention.

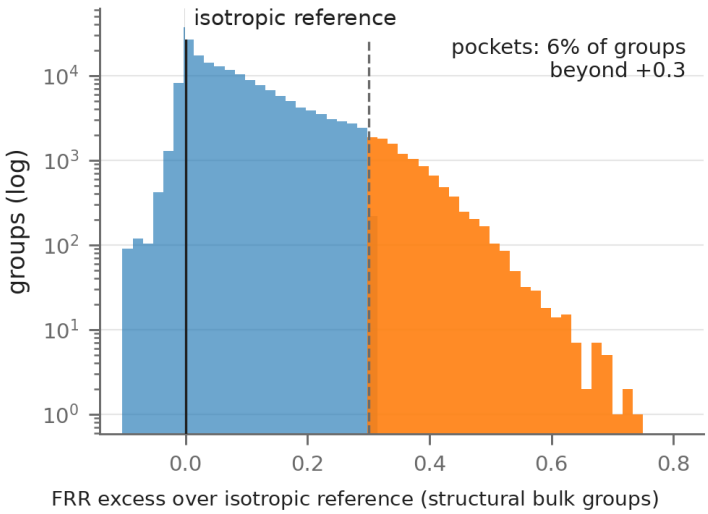


Figure 4.4: Distribution of FRR excess over the approximate isotropic reference for structurally grouped bulk parameters. The retained history rank is $r = 5$; approximately 6% of pooled group–anchor observations exceed the reported margin of 0.3 under this grouping and reference convention.

For a group of dimension D and retained history rank $r = 5$, the plotted statistic measures how far its realized FRR falls below the approximate isotropic reference $\sqrt{(D-r)/D}$. Approximately 6% of the pooled group–anchor observations exceed the reported margin of 0.3. This percentage is specific to the structural grouping, reference definition, and threshold used in the figure; it is not a universal boundary or a model-wide fraction of predictable parameters.

These pockets are transient rather than permanent features of a tensor. Their identities and prominence vary across anchors, while the extent of persistence over sustained portions of training depends on parameter structure and training regime. Bulk predictability is therefore localized both spatially and temporally: whole-tensor aggregation can conceal small high-predictability subsets, and the same subset need not remain dominant throughout the trajectory. The resulting picture is one of spatially localized and temporally shifting structure. We next ask how these changes correspond to realized optimization dynamics.

4.3.4 Predictability Measurements Correspond to Realized Optimization Dynamics

Short-horizon predictability varies with the realized course of optimization rather than with checkpoint index alone. To examine whether these changes correspond to meaningful trajectory evolution, we compare the probe readings with independently defined summaries of parameter motion, scalar training telemetry, and released loss curves. These comparisons provide an additional check that the measurements respond systematically to optimization position, while the parameter- and group-resolved probes reveal where the corresponding changes are expressed within the model. All temporal correspondences reported below are retrospective and observational. Figure 4.5 compares ESA with two independently defined summaries of realized optimization motion.

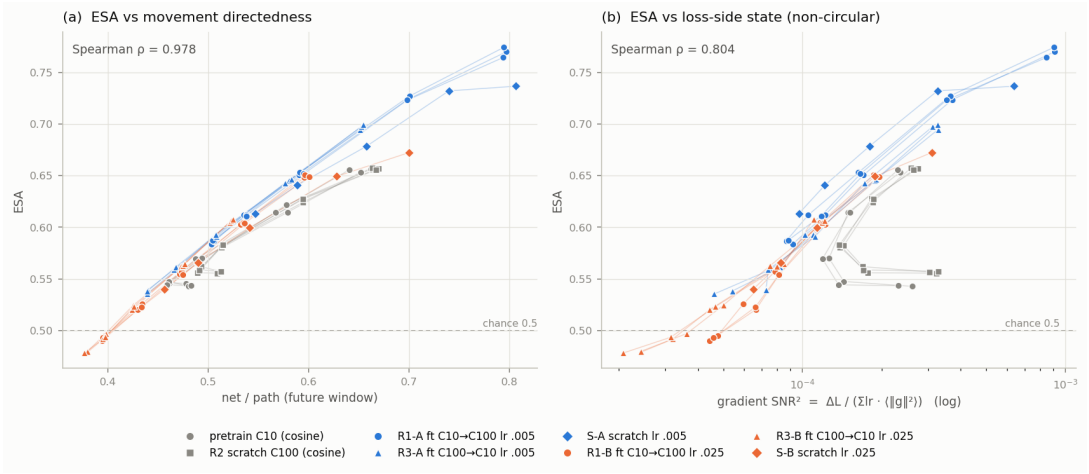


Figure 4.5: Predictability measurements in relation to realized optimization motion in multi-pass CIFAR-10 training. (a) ESA versus movement directedness, defined as net displacement divided by path length over the future window. (b) ESA versus the loss-side gradient-coherence estimate. Horizontal lines mark the random sign-agreement level of 0.5.

Across heterogeneous CIFAR conditions, ESA forms a tight monotone relation with movement directedness (Fig. 4.5a). Trajectories whose future movement covers more net distance relative to their path length retain greater coordinate-wise

directional agreement. A strong association also appears with the loss-side gradient-coherence estimate (Fig. 4.5b), although this quantity is constructed only from scalar telemetry and contains no parameter-direction information. The two comparisons show that ESA responds systematically to independently defined properties of realized optimization motion across architectures and training conditions, rather than simply following checkpoint index. Formal definitions, averaging conventions, and interpretive limits are provided in Appendix B.1.5.

We next use Pythia-70M as a case study to examine how these correspondences unfold within a one-pass pretraining trajectory. Figure 4.6 jointly resolves the public loss trajectory and probe readings across parameter roles, layers, and grouping scales.

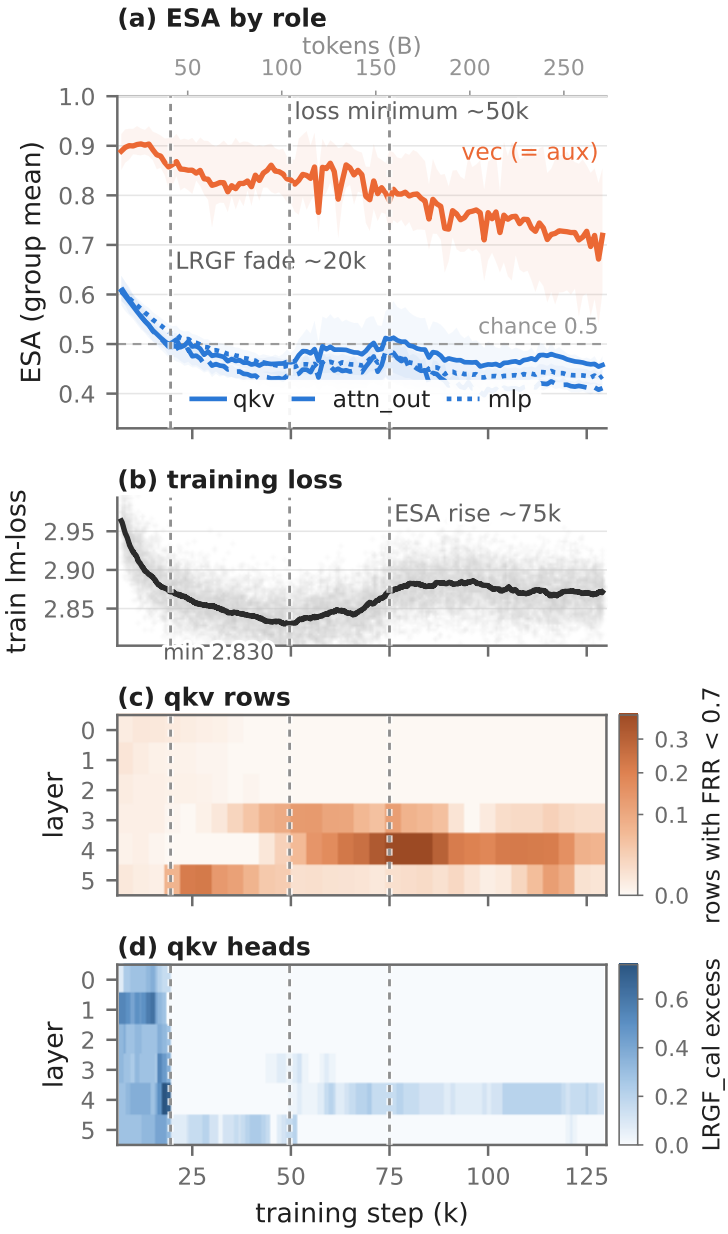


Figure 4.6: Pythia-70M case study. (a) ESA by parameter role; (b) released training loss and its smoothed trajectory; (c) fraction of qkv row groups satisfying FRR < 0.7; and (d) head-aligned qkv $\mathrm{LRGF}_{\mathrm{cal}}$ excess. Vertical lines mark the early fading of head-aligned excess, the loss minimum, and the later qkv ESA excursion.

The auxiliary vector segment remains clearly separated from the bulk roles throughout training. Although the bulk tensors contain most of the model parameters, all three bulk roles undergo a rapid initial decline in ESA and cross below the 0.5 sign-agreement reference. They continue to evolve within this sub-0.5 regime rather than settling to a common floor. The clearest mid-to-late turn occurs in *qkv*: its ESA bends upward near the smoothed loss minimum around 50k steps, reaches a local maximum near 75k after the loss has begun to rise, and subsequently declines again. Attention-output and MLP exhibit smaller, non-identical variations over the same interval.

The finer-resolution views expose an earlier reorganization within *qkv* at a different spatial scale. Head-aligned $\mathrm{LRGF}_{\mathrm{cal}}$ excess is concentrated near the beginning of training and largely fades around 20k steps. Row-level low-FRR pockets, however, are initially sparse and emerge only after some training, first in deeper layers and later across deep and intermediate layers. The fading of head-aligned excess therefore should not be read as a global disappearance of predictable structure. Rather, the detectable *qkv* structure changes its localization and its relation to the chosen grouping scale.

The temporal proximity between the loss reversal and the later *qkv* ESA excursion is a retrospective correspondence, not evidence that either event causes or uniquely explains the other. The role-, depth-, and grouping-resolved readings nevertheless reveal trajectory reorganizations that are not localized by the scalar loss curve alone. Pocket census, persistence, turnover, and overlap diagnostics are provided in Appendix B.4.2; the cross-scale extension is provided in Appendix B.4.3. These cross-regime correspondences motivate a controlled analysis of how architecture and training recipe modulate the same state-sensitive temporal organization.

4.3.5 Training Conditions Modulate State-Dependent Temporal Redundancy

The preceding analyses showed that the measured short-horizon structure evolves with the realized optimization trajectory. We next ask how architecture and training recipe modulate this state-sensitive organization. We consider three representative comparisons: a size-controlled architecture contrast, optimizer-family profiles along a shared training-loss coordinate, and paired weight-decay and momentum dose ladders. Because the compared runs are not matched in every aspect of their realized state, each contrast is interpreted under its stated controls. In particular, aligning runs by loss reduces the most direct checkpoint-position confound but does not imply matched optimizer state.

Architecture-Associated Differences at Controlled Scale

Figure 4.7 compares three architectures trained on the same dataset with SAM and a cosine learning-rate schedule.

ViT-mid and the ResNet have nearly matched parameter counts, yet ViT-mid retains higher bulk ESA at every measured anchor. The separation is already present near the beginning of training: the two ViTs have ESA values of approximately 0.72 and 0.71, compared with 0.65 for the ResNet at epoch 5. Around epoch 40, the corresponding values are approximately 0.60, 0.60, and 0.52. The smaller ViT follows the same higher-ESA profile despite containing approximately one fifth as many parameters as ViT-mid.

Parameter count alone therefore does not explain the observed ordering. Because the models use their standard model-specific learning rates, this experiment does not strictly isolate architecture from every hyperparameter difference. It nevertheless shows a stable architecture-associated contrast under the reported conditions: ViT-class models retain greater bulk directional persistence than the similarly sized ResNet throughout the measured trajectory.

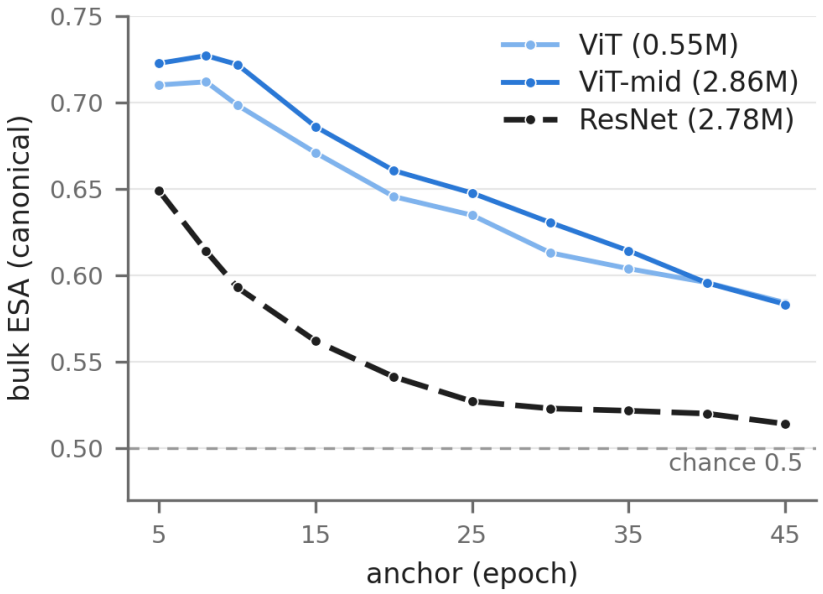


Figure 4.7: Size-controlled architecture comparison under SAM. Curves report canonical bulk ESA across ten anchors for ViT-tiny, ViT-mid, and the parameter-matched CIFAR ResNet. The dashed line marks the random sign-agreement level of 0.5.

Optimizer-Family Profiles along Training Loss

Because optimizer families traverse training at different rates, Figure 4.8 compares their bulk ESA profiles along training loss rather than at matched epochs. The comparison is restricted to the loss range shared by all displayed optimizers; as noted above, this alignment reduces an elapsed-training confound without equating their optimizer states.

Distinct optimizer-conditioned profiles remain visible. SAM occupies the highest-ESA regime, with values of approximately 0.58–0.73 over the displayed interval, whereas Muon remains near the 0.5 sign-agreement reference. Adam lies approximately 0.08 below SGD over most of their overlapping loss range. SGD and AdaGrad approach a similar low-loss ESA level of approximately 0.545, despite following different profiles earlier in training. Optimization position therefore organizes the evolution of predictability, while the update rule also conditions its level and local directional structure.

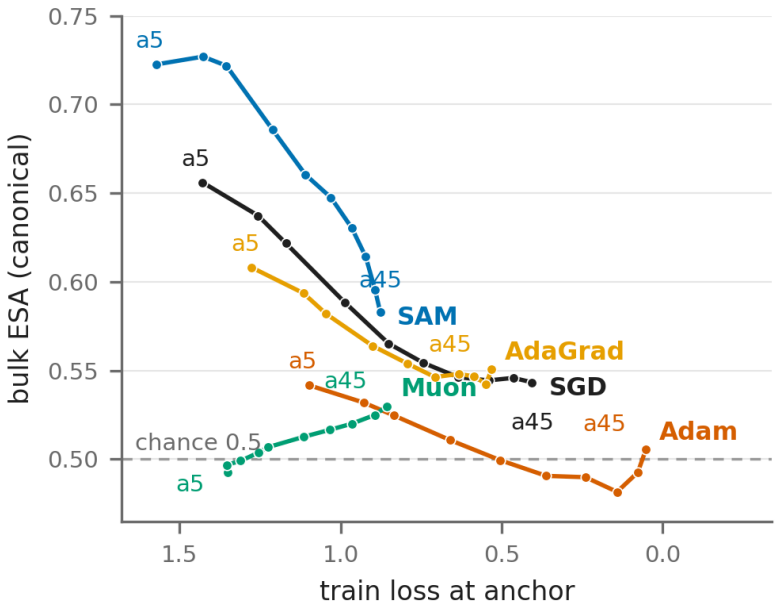


Figure 4.8: Optimizer-family profiles for ViT-mid aligned by training loss. The horizontal axis is reversed, and the vertical axis reports canonical bulk ESA. Curves are shown only over the loss range shared by the displayed optimizers; the dashed line marks the random sign-agreement level of 0.5.

This ordering is not an optimizer-quality ranking. Optimizers with strong final performance occupy both high- and low-predictability regimes. ESA measures the amount of short-horizon directional redundancy in the realized parameter trajectory; it does not directly measure final optimization quality or generalization.

Controlled Weight-Decay and Momentum Dose Effects

Figure 4.9 presents controlled weight-decay and momentum dose ladders for ViT-mid under a shared measurement protocol.

The dose ladders provide a more controlled comparison than the optimizer-family panel because each arm changes only one named hyperparameter relative to plain SGD. They also show why the three probe families should be retained jointly: the same intervention can register differently through element-wise signs, vector direction, and recent-history subspace containment.

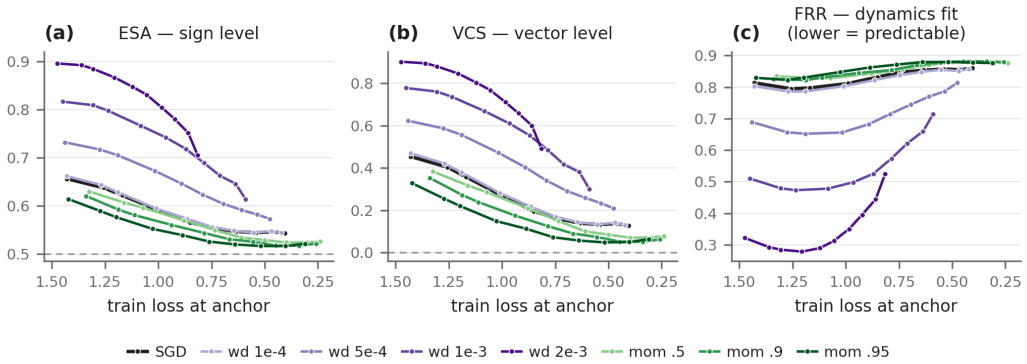


Figure 4.9: Controlled weight-decay and momentum dose ladders for ViT-mid. Panels report (a) bulk ESA, (b) bulk VCS, and (c) bulk FRR against reversed training loss under the canonical $K = 16$ vision grouping. Each arm changes only the indicated weight-decay or momentum value relative to plain SGD, and curves are compared over their overlapping loss ranges.

Weight decay. The weight-decay ladder exhibits a consistent dose ordering across all three readouts. ESA increases with the decay coefficient at shared loss positions and approaches 0.90 at the largest dose. VCS rises toward a similar level, showing that the full displacement direction becomes more persistent, while FRR falls to approximately 0.28, indicating stronger containment of the future displacement within the span of recent updates.

The readouts differ in their sensitivity at smaller doses. At $\lambda = 10^{-4}$, ESA and VCS remain near the plain-SGD profile, whereas FRR has already separated from it, at approximately 0.66 rather than 0.79. In this setting, the subspace-residual probe detects a low-dose change that is less visible through coordinate signs or vector alignment.

For SGD with weight decay, the update contains a component proportional to $-\lambda w$, introducing a slow, coherent shrinkage drift into the parameter trajectory. The monotone changes in ESA, VCS, and FRR show that this temporal component is captured consistently at the sign, vector, and recent-subspace levels. The raw probe profiles summarize the full realized trajectory rather than providing a complete

decomposition of the weight-decay mechanism, but they offer a trajectory-level view of how the intervention changes temporal organization.

The increase in predictability does not imply improved training performance. The 2×10^{-3} run terminates at a higher loss and reaches final accuracy of approximately 0.707, compared with 0.755 for the 10^{-3} run. Temporal redundancy therefore increases monotonically across this dose ladder even though training quality does not.

Momentum. The momentum ladder shows the opposite dose ordering. Increasing momentum lowers ESA and VCS over the shared loss range and produces a smaller increase in FRR. The clearest response appears at the vector-direction level: VCS decreases more substantially than ESA, while the ESA change is approximately -0.02 to -0.03.

Under this protocol, higher momentum therefore reduces the forms of bulk short-horizon redundancy captured by these probes. This statement concerns the realized parameter trajectory rather than the smoothness of the underlying gradient estimate: gradient smoothing and short-horizon predictability of the resulting parameter displacement are distinct properties. The contrast with weight decay further shows that two common training interventions can reorganize the realized update trajectory in different—and here opposite—directions across the probe panel.

Scope of the comparisons. These comparisons are representative rather than exhaustive: the architecture and optimizer-family panels use their stated standard configurations, and the controlled dose ladders cover one ViT-mid setting. The main conclusion is therefore not a universal ordering of architectures or optimizers, but that the probe families resolve systematic changes in how model structure and training recipe organize the realized trajectory. Additional architecture- and optimizer-conditioned diagnostics, including the weight-decay residual and momentum comparisons, are collected in Appendices B.5 and B.6. This condition

dependence motivates a narrow online test of whether restricting forecast eligibility to a coarse higher-confidence region can be useful.

4.3.6 Selective Prediction in a Higher-Confidence Parameter Region

Chapter 3 introduced PDT as an online selective-prediction method: candidate forecasts are written back coordinate-wise only when they satisfy the scale-control and dynamic-consistency criteria. The measurements developed in this chapter address a complementary question at coarser spatial scales: where does the realized training trajectory exhibit stronger short-horizon temporal structure? To connect these two views, we consider a simple illustrative test: whether a role-level restriction motivated by the auxiliary–bulk separation can provide a useful prior over where PDT attempts predictive write-back. This experiment is not a complete online realization of the measurement framework, nor a separate wall-clock acceleration study. It serves only as an initial indication that the measured spatial organization may be useful for narrowing the scope of online prediction.

The comparison uses the ViT-tiny and ViT-mid models under pure SGD, with momentum and weight decay both set to zero. Standard training is compared with two PDT variants. PDT-full makes all trainable parameters eligible for the prediction-and-masking procedure from Chapter 3. PDT-small retains the same selective PDT mechanism but restricts forecast eligibility to the operational auxiliary class documented in Appendix B.3.1; all remaining parameters continue to follow the baseline optimizer. The restriction is deliberately coarse. It uses the role-level tendency identified by the retrospective measurements rather than attempting to detect the time-varying predictable pockets within bulk tensors online.

Table 4.1 and Figure 4.10 summarize the online comparison. On both architectures, PDT-small gives the largest mean validation-selected test-accuracy gain while

Table 4.1: Validation-selected PDT comparison on CIFAR-10 over three seeds under pure SGD. PDT-small restricts forecast eligibility to auxiliary parameters while retaining the Chapter 3 predictor and coordinate-wise PDT acceptance mask.

Arch.	Method	Test acc.	Gain (pp)	Accepted (%)
ViT-tiny	Baseline	0.7422±0.0057	—	—
	PDT-full	0.7484±0.0018	+0.62±0.43	23.9%
	PDT-small	**0.7516±0.0058**	**+0.94±0.09**	**0.80%**
ViT-mid	Baseline	0.7425±0.0032	—	—
	PDT-full	0.7498±0.0032	+0.73±0.18	23.2%
	PDT-small	**0.7511±0.0029**	**+0.87±0.09**	**0.32%**

The accepted-coordinate column reports the aggregate fraction of trainable coordinates receiving predictive write-back under the coordinate-wise PDT mask; it is not the fraction made eligible by the role restriction.

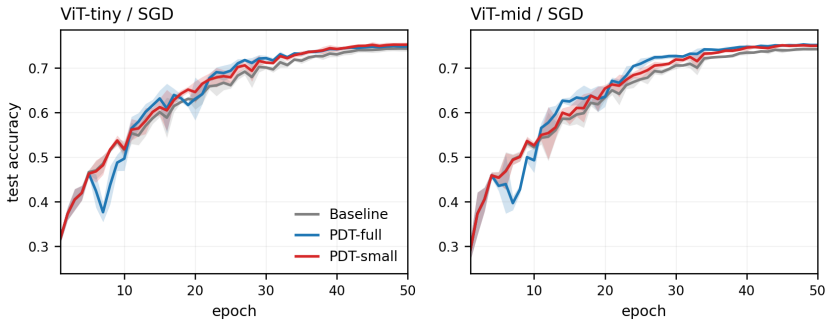


Figure 4.10: Multi-seed online PDT comparison under pure SGD for ViT-tiny and ViT-mid. Curves show seed means with ±1 standard-deviation bands.

applying predictive write-back to less than 1% of trainable coordinates: 0.80% on ViT-tiny and 0.32% on ViT-mid. PDT-full applies predictive write-back to approximately 24% of trainable coordinates and yields smaller mean gains of 0.62 percentage points on ViT-tiny and 0.73 percentage points on ViT-mid. The gain from PDT-small is also more stable across the three seeds: its standard deviation is 0.09 percentage points for both model scales, compared with 0.18–0.43 percentage points for PDT-full.

Within these two model–training cells, restricting forecast eligibility to the auxiliary region therefore yields a larger and more stable validation-selected gain than full-model eligibility, despite a much smaller accepted fraction. This result is consistent with the auxiliary–bulk separation identified by the retrospective

measurements, but it does not isolate the role restriction as the causal source of the improvement or imply that auxiliary parameters are uniformly predictable. The original coordinate-wise PDT mask remains the immediate online safeguard; the static role-level restriction neither implements the retrospective probe panel nor detects the localized, time-varying pockets within bulk tensors.

4.4 Predictability Measurement as a Training-Dynamics Diagnostic

This chapter develops a measurement view of short-horizon temporal redundancy in neural training. Instead of treating predictability as the outcome of a single forecasting method, the proposed probes examine whether recent update history contains information about near-future motion. The joint probe panel separates three related but non-equivalent forms of this structure: persistence of the latest displacement direction, containment of future motion in the recent-history subspace, and temporal structure recoverable by an explicit local predictor. This parameter-resolved view complements scalar loss and any single forecasting score by showing where, when, and at what spatial scale changes in temporal organization are expressed within the model.

Applying the same measurement framework to controlled multi-pass vision training and one-pass language-model pretraining demonstrates its cross-regime applicability. In both regimes, the probes provide interpretable analyses of optimization evolution across parameter roles, spatial scales, and training positions. Auxiliary and bulk parameters exhibit a recurring separation, while elevated bulk predictability is concentrated in localized, time-varying pockets rather than distributed uniformly. The finer analyses also reveal regime-specific temporal events and condition-dependent changes. Because CIFAR and Pythia use different checkpoint

clocks, grouping conventions, and training processes, this evidence concerns recurring organization rather than equality of absolute probe values.

The predictor-based results add one qualification to the simpler directional view. As shown in Section 4.3.2, explicit local predictors can recover temporal information beyond persistence alone, but the gain depends on the amount of structure and residual selection headroom already present. This supports retaining the three probe families without treating DMD as universally superior to simpler predictors.

This provides a measurement-side explanation for the value of prediction in PDT: the predictor can contribute information beyond the naive directional baseline, while selective masking determines where that additional information may be used reliably. The retrospective selector studied in this chapter is not identical to PDT's online coordinate-wise acceptance mask, so the connection explains the respective roles of prediction and selectivity without identifying the two mechanisms. The restricted-scope experiment in Section 4.3.6 illustrates this distinction: role-level scope restriction can improve reliability in the tested settings, but the retrospective probes are not themselves an online decision rule.

Short-horizon predictability should therefore be interpreted as a retrospective, parameter-resolved, and state-sensitive diagnostic of temporal redundancy. It is not a whole-model scalar, an optimizer- or generalization-quality ranking, or a causal account of training events. Nor do offline probe values automatically determine which predictions should be accepted during training. Chapter 5 develops the dissertation-level synthesis of these findings, their limitations, and the remaining steps toward broader validation and online measurement-guided predictive training.

Chapter 5

Discussion, Scope, and Future Directions

This dissertation addresses one central question from two complementary directions: when can short-horizon structure in neural training trajectories be used rather than ignored? Chapter 3 develops Predictive Differential Training (PDT) as an online method for exploiting such structure. It combines Koopman/DMD-based forecasting with a coordinate-wise reliability mask, accepts only forecasts that satisfy conservative scale and dynamic-consistency criteria, and leaves the remaining coordinates under the baseline optimizer. Chapter 4 approaches the same question as a measurement problem. Instead of evaluating whether one online mask succeeds, it examines where short-horizon temporal redundancy appears in recorded trajectories across parameter roles, tensors, within-tensor groups, training stages, architectures, optimizers, and training regimes.

The methodological connection between the two chapters is as important as their conceptual connection. At their shared core, both use short local histories and reduced representations of recent trajectory evolution. PDT computes its DMD forecast through the low-dimensional span identified from recent weight snapshots. The subspace-residual and DMD-based measurements in Chapter 4 likewise ask

whether near-future updates remain within, or can be modeled from, the span of recent updates. This locally low-rank restriction makes otherwise high-dimensional trajectories computationally tractable and exposes coherent short-horizon structure. It is also a common boundary of the current work: neither approach is designed to recover arbitrary nonlinear, long-memory, or high-rank dependence. The predictor-free directional probes in Chapter 4 complement this view by testing lower-order persistence without assuming that all useful structure is captured by a fitted low-rank predictor.

The main empirical synthesis is therefore about selectivity and localization, not a shared numerical fraction. PDT's acceptance ratio is a coordinate-level outcome of its online scale-control and dynamic-consistency criteria. By contrast, Chapter 4 reports several non-equivalent quantities at tensor and group scales, including directional persistence, recent-history subspace residuals, predictor-based selectivity, and convention-dependent pocket readouts. These quantities should not be collapsed into one universal percentage or compared directly with the PDT mask ratio. What the two chapters jointly establish is that short-horizon structure is unevenly distributed: auxiliary parameters often exhibit simpler local dynamics, while bulk feature-transforming tensors contain localized predictable pockets rather than uniformly predictable motion. PDT operationalizes this heterogeneity through conservative coordinate-wise selection; the measurement framework resolves its spatial organization at additional parameter scales.

Spatial localization is only part of the synthesis. PDT acceptance varies throughout training, and Chapter 4 shows why a fixed eligible subset would be insufficient: predictable pockets can emerge, weaken, migrate, and disappear across anchors. In the Pythia trajectories, the measured organization also changes with parameter role, depth, and grouping scale, so fading at one spatial resolution does not imply that all short-horizon structure has disappeared. These results support a dynamic view of selection in which the candidate scope, confidence threshold, or

prediction schedule may need to change with optimization position rather than remain fixed by layer or by a predetermined acceptance ratio.

The contribution of Chapter 4 is broader than an explanation of PDT. Because its core probes operate on stored parameter checkpoints, the framework can be applied retrospectively to any training run that retains a sufficiently dense trajectory. It provides a parameter-resolved and time-local view that complements scalar loss by showing where changes in temporal organization are expressed across tensors, roles, layers, and within-tensor groups. The controlled vision experiments demonstrate that the readings correspond with independently defined movement directedness and a loss-side gradient-coherence estimate, while the Pythia analysis resolves role-, depth-, and scale-dependent events that are not localized by the loss curve alone. The present experiments establish sensitivity to architecture, optimizer, and training recipe; the same retrospective framework can in future be used to diagnose how changes in data ordering, sampling, or distribution reshape training dynamics at the parameter level. These uses remain observational and do not by themselves identify a causal mechanism.

This diagnostic perspective also clarifies the role of Koopman/DMD prediction. Simple directional persistence already captures a substantial part of short-horizon organization, so DMD should not be presented as universally superior to simpler predictors. Within the predictor family, however, DMD and linear-trend selectors produce strongly consistent reliability orderings, while DMD achieves higher selected sign agreement in the reported panel. Its additional value is most visible at intermediate directional persistence, where recent motion retains exploitable structure but leaves sufficient residual variation for a richer local model to capture. The resulting division of labor is consistent across the dissertation: a predictor proposes candidate short-horizon motion, while selective safeguards determine where that proposal is reliable enough to use.

Taken together, the two chapters support a conditional rather than universal view of predictive acceleration. Predictable pockets may be interpreted cautiously

as locally coherent or slowly varying components of the update sequence, but the dissertation does not claim to identify a formal slow–fast decomposition, center manifold, or spectral submanifold of neural training. Prediction-based acceleration is most promising when explicit forward–backward updates are expensive, the recent trajectory contains measurable local redundancy, and prediction overhead can be amortized. Very short runs, small models, highly memory-constrained settings, or trajectories with weak local structure may not benefit. The relevant question is not whether prediction should be applied universally, but whether a given training process contains sufficient local temporal structure for predictive reuse to outweigh its cost.

5.1 Limitations

The empirical coverage is broader than a vision-only study but remains limited. The measurement framework is evaluated on a controlled CIFAR-10 panel spanning multiple architectures and training recipes, together with the public one-pass checkpoint trajectories of Pythia 70M, 160M, and 410M. This combination supports cross-regime conclusions about recurring organization, including auxiliary–bulk separation and localized bulk pockets. It does not establish universality across larger language models, alternative pretraining corpora, multimodal systems, reinforcement-learning pipelines, or other foundation-model families. The Pythia analysis also relies on released trajectories from one model suite rather than matched multi-seed or intervention-controlled language-model runs.

A shared methodological limitation of PDT and the deeper Chapter 4 probes is their reliance on short histories and locally low-rank representations. PDT forecasts within a reduced subspace identified from recent weight snapshots, while the subspace and DMD-based probes evaluate structure relative to the span or fitted dynamics of recent updates. This restriction is what makes local analysis of very high-dimensional trajectories tractable, but it can miss nonlinear, long-memory, or high-rank temporal dependence outside the retained history space. The predictor-free directional probes

do not fit a low-rank model, but they measure only lower-order directional relations. All readings also depend on checkpoint spacing, prediction horizon, history length, grouping convention, rank selection, and null calibration. These choices define the temporal and spatial scale of observation rather than universal constants of neural training. Very short histories, such as $n_{\text{past}} = 3$ in the current DMD setting, should therefore be treated as estimator-boundary cases rather than evidence that temporal structure is absent.

The measurements are retrospective or delayed. ESA, FRR, and the predictor-evaluation quantities use the realized post-anchor displacement, which is available only after the future window has occurred. Recorded trajectories provide the ground truth needed to determine whether recent history was informative, but an online mask must approximate this judgment causally from pre-anchor information. The restricted-scope experiment in Section 4.3.6 is therefore an illustrative bridge, not a complete implementation of the measurement framework as an online mask. It uses a coarse auxiliary-parameter eligibility rule while retaining PDT's original coordinate-wise safeguard, and it is not a separate wall-clock acceleration claim.

The most direct methodological limitation of the current online PDT mechanism is false-positive control. Accepting an unreliable prediction can be more damaging than rejecting a useful one, because an incorrect write-back may move the model into a region from which later gradient updates cannot easily recover. PDT addresses this risk through scale control and dynamic consistency, and Chapter 4 makes reliability measurable rather than implicit. Still, the current mask does not provide an independently calibrated false-positive rate, nor does it explicitly test whether each accepted coordinate belongs to a retrospectively diagnosed pocket.

Magnitude calibration is another unresolved issue. Many of the most informative measurements emphasize direction, subspace continuation, or selector lift. These properties are important because a reversed direction directly conflicts with the future training motion. Yet a directionally correct forecast can still have the wrong magnitude, especially under learning-rate schedules, adaptive optimizer states, weight

decay, or changing curvature. A complete online reliability rule should therefore combine directional and subspace confidence with magnitude-aware calibration.

Optimizer-state interaction remains only partially addressed. PDT primarily predicts weights, but modern optimizers maintain state variables such as momentum buffers, adaptive first- and second-moment estimates, matrix-structured preconditioners, or explicit slow and fast weights. Advancing the parameter vector without consistently advancing or reconciling these states can create a mismatch between the weights and optimizer memory. Chapter 4 further shows that measured temporal regularity can reflect optimizer mechanics, such as coherent shrinkage under weight decay. Such decompositions are diagnostic and should not be interpreted as counterfactual optimizer trajectories.

Finally, predictive training must be evaluated under complete overhead accounting. Prediction is itself a computational operation. DMD fitting, spectral forecasting, mask construction, snapshot storage, eligibility checks, and rejected predictions all consume time or memory. PDT avoids the meta-training and auxiliary-network cost of learned predictors, but it still stores a short history of large parameter vectors. The practical value of predictive training therefore depends on the joint tradeoff among prediction fidelity, accepted scope, stability, storage, communication, and measured wall-clock savings.

5.2 Future Directions

A first direction is to extend structured-predictability measurement across broader scales, learning regimes, and data distributions. Controlled multi-seed studies are needed for larger language models and additional pretraining suites, as well as for multimodal, contrastive, reinforcement-learning, and multi-stage fine-tuning pipelines. Matched experiments that vary data ordering, curriculum, sampling balance, domain mixture, or distribution shift would test whether the framework can localize how these changes alter training dynamics rather than only whether final loss

or accuracy changes. Such studies should preserve the distinction between directly comparable within-regime readings and qualitative cross-regime organization, and should test whether auxiliary–bulk separation and localized pocket structure persist under different checkpoint clocks, architectures, distributions, and optimizer states.

A second direction is to study the emergence, migration, and disappearance of predictable pockets as training events in their own right. The role-, depth-, and grouping-resolved Pythia results show that temporal organization can reorganize in ways not summarized by scalar loss. Future work can test whether these events align with independently measured representation formation, circuit development, or interpretable parameter communities. The observed sub-0.5 ESA regimes also motivate careful investigation of possible relationships with edge-of-stability behavior [10] or with Ornstein–Uhlenbeck approximations of locally stationary, constant-step SGD [48]. These are hypotheses to test under their respective assumptions, not mechanistic interpretations of the current correlations.

The retrospective diagnostic should first be developed into a reusable analysis pipeline for densely checkpointed runs. Such a pipeline could report when and where movement directedness, recent-subspace retention, or predictor-based selectivity changes across tensors, roles, layers, and groups, providing a local parameter-level account of differences caused by architecture, optimizer, schedule, or data regime. A further extension is delayed-online or history-only operation. A causal or low-latency version could identify when a trajectory enters or leaves a temporally redundant phase while training is still in progress, while preserving the distinction between predictability, optimizer quality, and generalization.

A central methodological direction is to improve the differential selection component of PDT. Future masks may be stage-aware, role-aware, group-aware, or subspace-aware rather than purely coordinate-wise. Adaptive schedulers could adjust the prediction horizon, prediction interval, candidate scope, or acceptance threshold using online confidence signals. If recent updates exhibit stable, reusable

structure, prediction can be attempted more aggressively; when the trajectory becomes irregular, the method should fall back quickly to the baseline optimizer.

Converting offline measurements into such causal confidence signals is the main bridge between Chapters 3 and 4. Residual- and reconstruction-based quantities are natural candidates, including DMD fitting residuals, pre-anchor subspace consistency, history-only analogues of FRR, and agreement among predictors of different complexity. These signals should be calibrated against false positives and evaluated for whether they predict future reliability before write-back. The goal is not to copy a retrospective threshold directly into training, but to learn which pre-anchor statistics anticipate the pockets diagnosed offline.

One related direction is to improve the prediction component itself. The current method uses finite-window linear DMD because it is data-driven, lightweight, and naturally suited to short-horizon trajectory modeling. More expressive predictors, including nonlinear Koopman variants, learned observables, kernel or EDMD-style methods, and hybrid learned-dynamical models, may capture structure that linear DMD misses. This direction must remain overhead-aware: a richer predictor is useful only when its improvement in reliability, accepted scope, or usable horizon exceeds the additional storage and computation it requires.

The measurement results also motivate prediction in transformed coordinates and across multiple time scales. PDT currently applies its mask in raw weight coordinates, whereas predictable structure can appear at group, role, and recent-update-subspace scales. Future work can test whether SVD coordinates, DMD modes, PCA-like bases, or other learned local representations separate coherent components from transient ones more effectively than coordinate-wise selection. This program could connect empirical predictability measurement with slow–fast decomposition, center-manifold ideas, and spectral-submanifold analysis without assuming that such formal structures have already been identified.

Overhead-aware implementation must develop alongside these methodological improvements. Possible directions include compressed or low-precision snapshot

storage, streaming or incremental DMD [29], sparse history retention, and localized analysis restricted to candidate regions. If predictable structure is concentrated, a scalable implementation need not store and process the full model uniformly. It can allocate memory and computation where the expected benefit of prediction is most likely to exceed its cost.

Structured predictability may ultimately guide prediction, update skipping, compression, and local acceleration. Its immediate value, however, is diagnostic: it resolves where recent updates retain usable short-horizon structure and where that structure breaks down. Higher predictability is not evidence of a better optimizer or stronger generalization; diagnostic interpretation, causal explanation, and acceleration benefit should remain separate questions.

The measurements explain why selective predictive training is necessary and identify promising directions for future online masking, but converting these diagnostics into a robust online acceleration mechanism remains open.

Bibliography

[1] Aghajanyan, A., Zettlemoyer, L., and Gupta, S. (2020). Intrinsic dimensionality explains the effectiveness of language model fine-tuning. *arXiv preprint arXiv:2012.13255*. 3, 13

[2] Ainsworth, S. K., Hayase, J., and Srinivasa, S. (2022). Git re-basin: Merging models modulo permutation symmetries. *arXiv preprint arXiv:2209.04836*. 13

[3] Anderson, D. G. (1965). Iterative procedures for nonlinear integral equations. *Journal of the ACM*, 12(4):547–560. 10

[4] Andrychowicz, M., Denil, M., Gomez, S., Hoffman, M. W., Pfau, D., Schaul, T., Shillingford, B., and De Freitas, N. (2016). Learning to learn by gradient descent by gradient descent. *Advances in neural information processing systems*, 29. 10

[5] Biderman, S., Schoelkopf, H., Anthony, Q. G., Bradley, H., O'Brien, K., Hallahan, E., Khan, M. A., Purohit, S., Prashanth, U. S., Raff, E., Skowron, A., Sutawika, L., and van der Wal, O. (2023). Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR. 14, 51

[6] Calvo, A. (2026). Predictable gradient manifolds in deep learning: Temporal path-length and intrinsic rank as a complexity regime. *arXiv preprint arXiv:2601.04270*. 15

[7] Chatterjee, S. (2020). Coherent gradients: An approach to understanding generalization in gradient descent-based optimization. *arXiv preprint arXiv:2002.10657*. 123

[8] Chen, X. and He, K. (2021). Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15750–15758. 34

[9] Chizat, L., Oyallon, E., and Bach, F. (2019). On lazy training in differentiable programming. *Advances in neural information processing systems*, 32. 14

[10] Cohen, J. M., Kaur, S., Li, Y., Kolter, J. Z., and Talwalkar, A. (2021). Gradient descent on neural networks typically occurs at the edge of stability. *arXiv preprint arXiv:2103.00065*. 14, 79

[11] Cosson, R., Jadbabaie, A., Makur, A., Reisizadeh, A., and Shah, D. (2023). Low-rank gradient descent. *IEEE Open Journal of Control Systems*, 2:380–395. 13

[12] d'Angelo, F., Andriushchenko, M., Varré, A., and Flammarion, N. (2024). Why do we need weight decay in modern deep learning? In *Advances in Neural Information Processing Systems*, volume 37, pages 23191–23223. 14

[13] Dietrich, F., Thiem, T. N., and Kevrekidis, I. G. (2020). On the koopman operator of algorithms. *SIAM Journal on Applied Dynamical Systems*, 19(2):860–885. 12

[14] Dogra, A. S. (2020). Dynamical systems and neural networks. *arXiv preprint arXiv:2004.11826*. 12

[15] Dogra, A. S. and Redman, W. (2020). Optimizing neural networks via koopman operator theory. *Advances in Neural Information Processing Systems*, 33:2087–2097. 4, 12

[16] Draxler, F., Veschgini, K., Salmhofer, M., and Hamprecht, F. (2018). Essentially no barriers in neural network energy landscape. In *International conference on machine learning*, pages 1309–1318. PMLR. 3, 13

[17] Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7). 1, 9

[18] Feng, Y. and Tu, Y. (2021). Phases of learning dynamics in artificial neural networks: with or without mislabeled data. *arXiv preprint arXiv:2101.06509*. 124

[19] Foret, P., Kleiner, A., Mobahi, H., and Neyshabur, B. (2020). Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*. 14

[20] Frankle, J., Dziugaite, G. K., Roy, D., and Carbin, M. (2020). Linear mode connectivity and the lottery ticket hypothesis. In *International conference on machine learning*, pages 3259–3269. PMLR. 13

[21] Garipov, T., Izmailov, P., Podoprikhin, D., Vetrov, D. P., and Wilson, A. G. (2018). Loss surfaces, mode connectivity, and fast ensembling of dnns. *Advances in neural information processing systems*, 31. 3, 13

[22] Gavish, M. and Donoho, D. L. (2014). The optimal hard threshold for singular values is $4/\sqrt{3}$. *IEEE Transactions on Information Theory*, 60(8):5040–5053. 45

[23] Ghorbani, B., Krishnan, S., and Xiao, Y. (2019). An investigation into neural net optimization via hessian eigenvalue density. In *International Conference on Machine Learning*, pages 2232–2241. PMLR. 13

[24] Goodfellow, I. J., Vinyals, O., and Saxe, A. M. (2014). Qualitatively characterizing neural network optimization problems. *arXiv preprint arXiv:1412.6544*. 3, 13

[25] Guan, L., Li, D., Shi, Y., and Meng, J. (2024). Xgrad: Boosting gradient-based optimizers with weight prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(10):6731–6747. 10

[Gulli] Gulli, A. AG News Dataset. http://groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html. Accessed: 2025-12-01. 112

[27] Gupta, V., Koren, T., and Singer, Y. (2018). Shampoo: Preconditioned stochastic tensor optimization. In *International Conference on Machine Learning*, pages 1842–1850. PMLR. 9

[28] Gur-Ari, G., Roberts, D. A., and Dyer, E. (2018). Gradient descent happens in a tiny subspace. *arXiv preprint arXiv:1812.04754*. 3, 13, 124

[29] Hemati, M. S., Williams, M. O., and Rowley, C. W. (2014). Dynamic mode decomposition for large and streaming datasets. *Physics of Fluids*, 26(11). 40, 81

[30] Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., de Las Casas, D., Hendricks, L. A., Welbl, J., Clark, A., et al. (2022). Training compute-optimal large language models. In *Advances in Neural Information Processing Systems*. 14

[31] Huang, G., Li, Y., Pleiss, G., Liu, Z., Hopcroft, J. E., and Weinberger, K. Q. (2017). Snapshot ensembles: Train 1, get m for free. *arXiv preprint arXiv:1704.00109*. 13

[32] Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D., and Wilson, A. G. (2018). Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*. 13

[33] Jacot, A., Gabriel, F., and Hongler, C. (2018). Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31. 14

[34] Jang, J., Yun, W.-h., Kim, W. H., Yoon, Y., Kim, J., Lee, J., and Han, B. (2023). Learning to boost training by periodic nowcasting near future weights. In *International Conference on Machine Learning*, pages 14730–14757. PMLR. 3, 4, 10

[35] Jordan, K., Jin, Y., Boza, V., Jiacheng, Y., Cesista, F., Newhouse, L., and Bernstein, J. (2024). Muon: An optimizer for hidden layers in neural networks, 2024. *URL https://kellerjordan.github.io/posts/muon*, 6(3):4. 14

[36] Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. *International Conference on Machine Representations (ICLR)*. 1, 9, 14

[37] Knyazev, B., Moudgil, A., Lajoie, G., Belilovsky, E., and Lacoste-Julien, S. (2025). Accelerating training with neuron interaction and nowcasting networks. In *International Conference on Learning Representations*, volume 2025, pages 29569–29588. 3, 4, 10

[38] Kumar, T., Bordelon, B., Gershman, S. J., and Pehlevan, C. (2023). Grokking as the transition from lazy to rich training dynamics. In *The twelfth international conference on learning representations*. 14

[39] Kunstner, F., Milligan, A., Yadav, R., Schmidt, M., and Bietti, A. (2024). Heavy-tailed class imbalance and why adam outperforms gradient descent on language models. *Advances in Neural Information Processing Systems*, 37:30106–30148. 14

[40] Kutz, J. N., Brunton, S. L., Brunton, B. W., and Proctor, J. L. (2016). *Dynamic mode decomposition: data-driven modeling of complex systems*. SIAM. 11, 12, 20, 21

[41] Li, C., Farkhoor, H., Liu, R., and Yosinski, J. (2018a). Measuring the intrinsic dimension of objective landscapes. *arXiv preprint arXiv:1804.08838*. 13

[42] Li, H., Xu, Z., Taylor, G., Studer, C., and Goldstein, T. (2018b). Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, 31. 3, 13

[43] Li, S., Zhao, Y., Varma, R., Salpekar, O., Noordhuis, P., Li, T., Paszke, A., Smith, J., Vaughan, B., Damania, P., et al. (2020). Pytorch distributed: Experiences on accelerating data parallel training. *arXiv preprint arXiv:2006.15704*. 1

[44] Liu, Z., Michaud, E. J., and Tegmark, M. (2022). Omnigrok: Grokking beyond algorithmic data. *arXiv preprint arXiv:2210.01117*. 14

[45] Loshchilov, I. and Hutter, F. (2019). Decoupled weight decay regularization. In *International Conference on Learning Representations*. 9, 14

[46] Lyu, K., Jin, J., Li, Z., Du, S. S., Lee, J. D., and Hu, W. (2023). Dichotomy of early and late phase implicit biases can provably induce grokking. *arXiv preprint arXiv:2311.18817*. 14

[47] Ma, J. and Yarats, D. (2018). Quasi-hyperbolic momentum and adam for deep learning. *arXiv preprint arXiv:1810.06801*. 14

[48] Mandt, S., Hoffman, M. D., and Blei, D. M. (2017). Stochastic gradient descent as approximate bayesian inference. *Journal of Machine Learning Research*, 18(134):1–35. 79, 124

[49] Manojlovic, I., Fonoberova, M., Mohr, R., Andrejcuk, A., Drmac, Z., Kevrekidis, Y., and Mezić, I. (2020). Applications of koopman mode analysis to neural networks. *arXiv preprint arXiv:2006.11765*. 12

[50] Martens, J. (2010). Deep learning via hessian-free optimization. In *ICML*, pages 735–742. 1

[51] Martens, J. and Grosse, R. (2015). Optimizing neural networks with kronecker-factored approximate curvature. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, pages 2408–2417. 9

[52] McEntire, J. (2026). Leap+ verify: Regime-adaptive speculative weight prediction for accelerating neural network training. *arXiv preprint arXiv:2602.19580*. 15

[53] Metz, L., Maheswaranathan, N., Nixon, J., Freeman, D., and Sohl-Dickstein, J. (2019). Understanding and correcting pathologies in the training of learned optimizers. In *International Conference on Machine Learning*, pages 4556–4565. PMLR. 10

[54] Mezić, I. (2005). Spectral properties of dynamical systems, model reduction and decompositions. *Nonlinear Dynamics*, 41:309–325. 11

[55] Mezić, I. (2020). Spectrum of the Koopman operator, spectral expansions in functional spaces, and state-space geometry. *Journal of Nonlinear Science*, 30(5):2091–2145. 11

[56] Mezić, I. (2021). Koopman operator, geometry, and learning of dynamical systems. *Notices of the American Mathematical Society*, 68(7):1087–1105. 11

[57] Mezić, I. and Banaszuk, A. (2004). Comparison of systems with complex behavior. *Physica D: Nonlinear Phenomena*, 197(1-2):101–133. 11

[58] Micikevicius, P., Narang, S., Alben, J., Diamos, G., Elsen, E., Garcia, D., Ginsburg, B., Houston, M., Kuchaiev, O., Venkatesh, G., et al. (2017). Mixed precision training. *arXiv preprint arXiv:1710.03740*. 1

[59] Nesterov, Y. (1983). A method of solving a convex programming problem with convergence rate o (1/k** 2). *Doklady Akademii Nauk SSSR*, 269(3):543. 9

[60] Olsson, C., Elhage, N., Nanda, N., Joseph, N., DasSarma, N., Henighan, T., Mann, B., Askell, A., Bai, Y., Chen, A., et al. (2022). In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*. 14

[61] Papyan, V., Han, X., and Donoho, D. L. (2020). Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40):24652–24663. 14

[62] Polyak, B. T. (1964). Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17. 8

[63] Power, A., Burda, Y., Edwards, H., Babuschkin, I., and Misra, V. (2022). Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*. 14

[64] Redman, W., Bello-Rivas, J., Fonoberova, M., Mohr, R., Kevrekidis, Y., and Mezic, I. (2024). Identifying equivalent training dynamics. *Advances in Neural Information Processing Systems*, 37:23603–23629. 12

[65] Redman, W., Fonoberova, M., Mohr, R., Kevrekidis, I. G., and Mezić, I. (2022). An operator theoretic view on pruning deep neural networks. *International Conference on Learning Representations (ICLR)*. 12

[66] Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407. 8

[67] Sagun, L., Evci, U., Guney, V. U., Dauphin, Y., and Bottou, L. (2017). Empirical analysis of the hessian of over-parametrized neural networks. *arXiv preprint arXiv:1706.04454*. 13

[68] Schmid, F. and Schmidt, R. (2007). Nonparametric inference on multivariate versions of blomqvist's beta and related measures of tail dependence. *Metrika*, 66(3):323–354. 53

[69] Schmid, P. J. (2010). Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28. 11, 20

[70] Scieur, D., d'Aspremont, A., and Bach, F. (2016). Regularized nonlinear acceleration. In *Advances in Neural Information Processing Systems*. 10

[71] Shou, X., Ding, Y., and Gao, J. (2025). Gradient flow matching for learning update dynamics in neural network training. *arXiv preprint arXiv:2505.20221*. 15

[72] Singh, S. P., He, B., Hofmann, T., and Schölkopf, B. (2025). The directionality of optimization trajectories in neural networks. In *The Thirteenth International Conference on Learning Representations*. 13, 124

[73] Sinha, A., Sarkar, M., Mukherjee, A., and Krishnamurthy, B. (2017). Introspection: Accelerating neural network training by learning weight evolution. *arXiv preprint arXiv:1704.04959*. 3, 4, 10

[74] Song, M., Ahn, K., and Yun, C. (2024). Does sgd really happen in tiny subspaces? *arXiv preprint arXiv:2405.16002*. 13

[75] Sun, Y., Huang, T., Ding, L., Shen, L., and Tao, D. (2025). TeZO: Empowering the low-rankness on the temporal dimension in zeroth-order optimization for fine-tuning LLMs. *arXiv preprint arXiv:2501.19057*. 15

[76] Sutskever, I., Martens, J., Dahl, G., and Hinton, G. (2013). On the importance of initialization and momentum in deep learning. In *International Conference on Machine Learning (ICML)*, pages 1139–1147. PMLR. 1, 8

[77] Tano, M. E., Portwood, G. D., and Ragusa, J. C. (2020). Accelerating training in artificial neural networks with dynamic mode decomposition. *arXiv preprint arXiv:2006.14371*. 4, 12

[78] Tieleman, T. and Hinton, G. (2012). Rmsprop: Divide the gradient by a running average of its recent magnitude. coursera: Neural networks for machine learning. *COURSERA Neural Networks Mach. Learn*, 17. 1, 9

[79] Tigges, C., Hanna, M., Yu, Q., and Biderman, S. (2024). LLM circuit analyses are consistent across training and scale. In *Advances in Neural Information Processing Systems*, volume 37, pages 40699–40731. 14

[80] Tu, J. H., Rowley, C. W., Luchtenburg, D. M., Brunton, S. L., and Kutz, J. N. (2014). On dynamic mode decomposition: Theory and applications. *Journal of Computational Dynamics*, 1(2):391–421. 12, 20

[81] van Laarhoven, T. (2017). L_2 regularization versus batch and weight normalization. *arXiv preprint arXiv:1706.05350*. 14

[82] Wichrowska, O., Maheswaranathan, N., Hoffman, M. W., Colmenarejo, S. G., Denil, M., Freitas, N., and Sohl-Dickstein, J. (2017). Learned optimizers that scale and generalize. In *International conference on machine learning*, pages 3751–3760. PMLR. 10

[83] Wortsman, M., Ilharco, G., Gadre, S. Y., Roelofs, R., Gontijo-Lopes, R., Morcos, A. S., Namkoong, H., Farhadi, A., Carmon, Y., Kornblith, S., et al. (2022). Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pages 23965–23998. PMLR. 13

[84] Xiong, R., Yang, Y., He, D., Zheng, K., Zheng, S., Xing, C., Zhang, H., Lan, Y., Wang, L., and Liu, T. (2020). On layer normalization in the transformer architecture. In *International conference on machine learning*, pages 10524–10533. PMLR. 14

[85] You, Y., Gitman, I., and Ginsburg, B. (2017). Large batch training of convolutional networks. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*. 9

[86] You, Y., Li, J., Reddi, S., Hseu, J., Kumar, S., Bhojanapalli, S., Song, X., Demmel, J., and Keutzer, K. (2019). Large batch optimization for deep learning: Training BERT in 76 minutes. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*. 9

[87] Zhang, J., Karimireddy, S. P., Veit, A., Kim, S., Reddi, S., Kumar, S., and Sra, S. (2020). Why are adaptive methods good for attention models? *Advances in Neural Information Processing Systems*, 33:15383–15393. 14

[88] Zhang, M., Lucas, J., Ba, J., and Hinton, G. E. (2019). Lookahead optimizer: k steps forward, 1 step back. *Advances in neural information processing systems*, 32. 14

[89] Zhang, X., Zhao, J., and LeCun, Y. (2015). Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28. 112

[90] Zhang, Y., Chen, C., Ding, T., Li, Z., Sun, R., and Luo, Z.-Q. (2024). Why transformers need adam: A hessian perspective. *Advances in neural information processing systems*, 37:131786–131823. 14

Appendices

Appendix A

Supplementary Details for Predictive Differential Training

This appendix provides additional methodological, theoretical, and empirical details for Chapter 3. Appendix A.1 gives the toy differential-learning example used to motivate selective acceleration. Appendix A.2 details the reduced DMD implementation used for high-dimensional neural-network weights. Appendix A.3 provides the convergence analysis for the masked hybrid update under idealized gradient-descent assumptions. Appendices A.4–A.5 analyze computational complexity, memory use, and measured runtime overhead. Finally, Appendix A.6 reports supplementary robustness and cross-domain experiments.

A.1 Toy Example for Binary Differential Learning

To illustrate the intuition behind differential learning, we consider the toy optimization problem introduced in Chapter 3. The objective is

$$\begin{aligned} f(x, y, z, u, v, w) = x^2 + y^2 + \sin(z) + u^2 - \cos(v) + w^2 \\ + xy + y\sin(z) + uvw. \end{aligned} \tag{A.1}$$

Starting from the initial point

$$[2.0,\ 2.0,\ 1.0,\ 0.5,\ -0.5,\ 1.5],$$

standard gradient descent with learning rate 0.01 requires 53 steps to reach the threshold $f < 0.1$.

We then consider a binary differential-learning variant in which the variables x, y, z use a learning rate three times larger than the baseline rate, while u, v, w continue to use the baseline rate. This example does not require a prediction model; rather, it illustrates the broader principle that a selected subset of variables can be advanced more aggressively while the remaining variables remain under the baseline update rule. As shown in Fig. A.1, this binary differential-learning strategy follows a similar trajectory in the x-y plane but reaches the threshold in 25 steps.

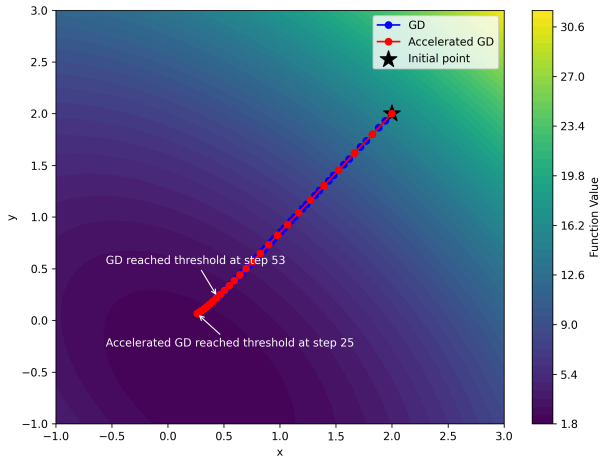
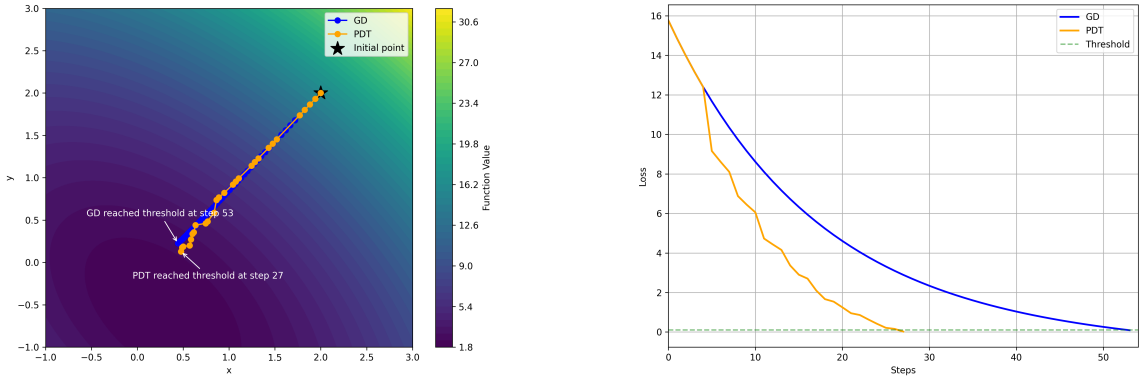


Figure A.1: Binary differential-learning trajectory for the toy problem in Eq. A.1. Only the x and y dimensions are shown. The selected subset (x, y, z) uses a learning rate three times larger than the baseline rate, while the remaining variables use the baseline rate.

We also apply PDT to the same toy objective. PDT reaches the same threshold in 27 steps, compared with 53 steps for standard gradient descent. The comparison in Fig. A.2 shows that PDT preserves the broad descent trajectory while reducing the number of optimization steps required to reach the target loss.



(a) **(b)**

Figure A.2: Performance comparison between standard gradient descent and PDT on the toy optimization problem. Standard gradient descent reaches the threshold in 53 steps, while PDT reaches it in 27 steps. (a) Optimization trajectories in the x-y plane. (b) Loss values during optimization.

A.2 Reduced DMD Implementation for High-Dimensional Weights

Chapter 3 introduces the DMD-based approximation

$$A = W_{i+1} W_i^{\dagger}, \tag{A.2}$$

where W_i and W_{i+1} are time-shifted snapshot matrices constructed from recent weight history. For a neural network with N trainable parameters, the full operator A is an $N \times N$ matrix and is therefore not formed explicitly. Instead, PDT uses a reduced DMD implementation that computes the spectral evolution in the low-dimensional subspace spanned by the recent snapshots.

Let $\mathbf{w}_i \in \mathbb{R}^N$ denote the flattened weight vector at epoch i, and let h be the number of saved snapshots in the history buffer. From the recent history, we form

the input and shifted snapshot matrices

$$\begin{aligned} W_i &= \left[\mathbf{w}_{i-h+1}, \mathbf{w}_{i-h+2}, \ldots, \mathbf{w}_{i-1}\right], \\ W_{i+1} &= \left[\mathbf{w}_{i-h+2}, \mathbf{w}_{i-h+3}, \ldots, \mathbf{w}_{i}\right], \end{aligned} \tag{A.3}$$

where $W_i, W_{i+1} \in \mathbb{R}^{N\times(h-1)}$. Because $h \ll N$, the effective rank of the snapshot sequence is at most $h-1$.

We first compute the reduced singular value decomposition

$$W_i \approx U_r \Sigma_r V_r^\top, \tag{A.4}$$

where $U_r \in \mathbb{R}^{N\times r}$, $\Sigma_r \in \mathbb{R}^{r\times r}$, $V_r \in \mathbb{R}^{(h-1)\times r}$, and $r \leq h-1$ is the retained rank. This step compresses the recent trajectory into the dominant snapshot subspace.

Instead of eigendecomposing the full operator A, we project it onto the subspace spanned by U_r:

$$\widetilde{A} = U_r^\top A U_r = U_r^\top W_{i+1} V_r \Sigma_r^{-1}. \tag{A.5}$$

The reduced operator $\widetilde{A} \in \mathbb{R}^{r\times r}$ is small because r is controlled by the short history length. Its eigendecomposition is

$$\widetilde{A} Z = Z\Lambda, \tag{A.6}$$

where $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_r)$ contains the DMD eigenvalues and $Z \in \mathbb{C}^{r\times r}$ contains the corresponding eigenvectors.

The high-dimensional DMD modes are reconstructed as

$$\Phi = U_r Z. \tag{A.7}$$

This is the standard projected-DMD form. An exact-DMD variant can instead use

$$\Phi_{\text{exact}} = W_{i+1} V_r \Sigma_r^{-1} Z, \tag{A.8}$$

but the projected form is more efficient and is sufficient for the short-window spectral prediction used in PDT.

Given Φ and Λ, the current weights are projected onto the DMD modes:

$$\mathbf{c}_i = \Phi^{\dagger}\mathbf{w}_i, \tag{A.9}$$

where $\Phi^{\dagger}$ denotes the Moore–Penrose pseudoinverse. The τ-step prediction is then computed by spectral evolution:

$$\mathbf{w}_{i+\tau}^{\text{pred}} = \text{Re}\left\{\Phi\Lambda^{\tau}\Phi^{\dagger}\mathbf{w}_i\right\}. \tag{A.10}$$

The real part is taken because the DMD eigendecomposition can be complex-valued, whereas neural-network parameters are real-valued.

This reduced formulation is important for scalability. The eigendecomposition is performed only on the $r \times r$ reduced operator, not on an $N \times N$ matrix. Thus, the computational bottleneck scales with the number of parameters and the short history length, rather than quadratically or cubically with the full parameter dimension.

A.3 Convergence Analysis of the Masked Hybrid Update

This section analyzes the masked hybrid update used in PDT under idealized deterministic gradient-descent assumptions. The analysis is intended to show that the PDT masking criteria provide a descent-style safeguard when the baseline update is a gradient step and the loss is smooth. It should not be interpreted as a full convergence proof for arbitrary stochastic optimizers with additional optimizer state.

Let $\mathcal{L} : \mathbb{R}^N \to \mathbb{R}$ denote the training loss and assume that $\mathcal{L}$ has L-Lipschitz continuous gradients:

$$\|\nabla\mathcal{L}(x) - \nabla\mathcal{L}(y)\| \leq L\|x - y\|, \qquad \forall x, y \in \mathbb{R}^N. \tag{A.11}$$

Let $\mathbf{w}_i^{\text{opt}}$ denote the current parameters before the hybrid update, and let the baseline optimizer step be idealized as one step of gradient descent:

$$\mathbf{w}_{i+1}^{\text{opt}} = \mathbf{w}_i^{\text{opt}} - \eta\nabla\mathcal{L}(\mathbf{w}_i^{\text{opt}}), \tag{A.12}$$

where $\eta > 0$ is the step size. Let $\mathbf{w}_{i+\tau}^{\text{pred}}$ denote the τ-step DMD prediction starting from $\mathbf{w}_i^{\text{opt}}$.

For a coordinate j accepted by the PDT mask, the scale-control criterion enforces the coordinate-wise bound

$$\left|w_{i+\tau,j}^{\text{pred}} - w_{i,j}^{\text{opt}}\right| \leq \tau\left|w_{i+1,j}^{\text{opt}} - w_{i,j}^{\text{opt}}\right| = \tau\eta\left|\nabla_j\mathcal{L}(\mathbf{w}_i^{\text{opt}})\right|. \tag{A.13}$$

For accepted coordinates, the acceleration-effectiveness lower bound also requires

$$\left|w_{i+\tau,j}^{\text{pred}} - w_{i,j}^{\text{opt}}\right| > \left|w_{i+1,j}^{\text{opt}} - w_{i,j}^{\text{opt}}\right| = \eta\left|\nabla_j\mathcal{L}(\mathbf{w}_i^{\text{opt}})\right|. \tag{A.14}$$

The dynamic-consistency criterion requires the predicted direction to agree with the baseline optimizer direction:

$$\operatorname{sign}\left(w_{i+k,j}^{\text{pred}} - w_{i+k-1,j}^{\text{pred}}\right) = \operatorname{sign}\left(w_{i+1,j}^{\text{opt}} - w_{i,j}^{\text{opt}}\right), \qquad k = 1, \ldots, \tau. \tag{A.15}$$

Consequently, for accepted coordinates,

$$\left(w_{i+\tau,j}^{\text{pred}} - w_{i,j}^{\text{opt}}\right)\nabla_j\mathcal{L}(\mathbf{w}_i^{\text{opt}}) \leq 0. \tag{A.16}$$

Coordinates with zero gradient contribute zero to the descent inner product and do not affect the argument.

Let $\mathbf{m}_i \in \{0,1\}^N$ denote the binary PDT mask. The hybrid update is

$$\mathbf{w}_{i+1} = \mathbf{m}_i \odot \mathbf{w}_{i+\tau}^{\text{pred}} + (\mathbf{1} - \mathbf{m}_i) \odot \mathbf{w}_{i+1}^{\text{opt}}, \tag{A.17}$$

where $\odot$ denotes element-wise multiplication. Define the hybrid step

$$\mathbf{d}_i := \mathbf{w}_{i+1} - \mathbf{w}_i^{\text{opt}}. \tag{A.18}$$

Coordinate-wise,

$$d_{i,j} = \begin{cases} w_{i+\tau,j}^{\text{pred}} - w_{i,j}^{\text{opt}}, & m_{i,j} = 1, \\ -\eta \nabla_j \mathcal{L}(\mathbf{w}_i^{\text{opt}}), & m_{i,j} = 0. \end{cases} \tag{A.19}$$

For accepted coordinates, Eq. A.13 gives

$$|d_{i,j}| \leq \tau\eta |\nabla_j \mathcal{L}(\mathbf{w}_i^{\text{opt}})|.$$

For unaccepted coordinates, Eq. A.19 gives

$$|d_{i,j}| = \eta |\nabla_j \mathcal{L}(\mathbf{w}_i^{\text{opt}})| \leq \tau\eta |\nabla_j \mathcal{L}(\mathbf{w}_i^{\text{opt}})|,$$

since $\tau \geq 1$. Therefore,

$$\|\mathbf{d}_i\| \leq \tau\eta \|\nabla \mathcal{L}(\mathbf{w}_i^{\text{opt}})\|. \tag{A.20}$$

For accepted coordinates, the lower bound in Eq. A.14 and the directional-consistency condition imply

$$\nabla_j \mathcal{L}(\mathbf{w}_i^{\text{opt}}) d_{i,j} \leq -\eta \left(\nabla_j \mathcal{L}(\mathbf{w}_i^{\text{opt}})\right)^2. \tag{A.21}$$

For unaccepted coordinates, the baseline gradient step gives

$$\nabla_j \mathcal{L}(\mathbf{w}_i^{\text{opt}}) d_{i,j} = -\eta \left(\nabla_j \mathcal{L}(\mathbf{w}_i^{\text{opt}})\right)^2 . \tag{A.22}$$

Summing over coordinates yields

$$\left\langle \nabla \mathcal{L}(\mathbf{w}_i^{\text{opt}}), \mathbf{d}_i \right\rangle \leq -\eta \left\| \nabla \mathcal{L}(\mathbf{w}_i^{\text{opt}}) \right\|^2 . \tag{A.23}$$

By the descent lemma for L-smooth functions,

$$\mathcal{L}(\mathbf{x} + \mathbf{d}) \leq \mathcal{L}(\mathbf{x}) + \langle \nabla \mathcal{L}(\mathbf{x}), \mathbf{d} \rangle + \frac{L}{2} \|\mathbf{d}\|^2. \tag{A.24}$$

Applying Eq. A.24 with $\mathbf{x} = \mathbf{w}_i^{\text{opt}}$ and $\mathbf{d} = \mathbf{d}_i$, and using Eqs. A.20 and A.23, we obtain

$$\begin{aligned} \mathcal{L}(\mathbf{w}_{i+1}) &\leq \mathcal{L}(\mathbf{w}_i^{\text{opt}}) - \eta \left\| \nabla \mathcal{L}(\mathbf{w}_i^{\text{opt}}) \right\|^2 + \frac{L}{2} \tau^2 \eta^2 \left\| \nabla \mathcal{L}(\mathbf{w}_i^{\text{opt}}) \right\|^2 \\ &= \mathcal{L}(\mathbf{w}_i^{\text{opt}}) - \eta \left(1 - \frac{L \eta \tau^2}{2} \right) \left\| \nabla \mathcal{L}(\mathbf{w}_i^{\text{opt}}) \right\|^2 . \end{aligned} \tag{A.25}$$

A sufficient condition for the coefficient to be nonnegative is

$$\eta \leq \frac{2}{L \tau^2}. \tag{A.26}$$

Under this condition,

$$\mathcal{L}(\mathbf{w}_{i+1}) \leq \mathcal{L}(\mathbf{w}_i^{\text{opt}}). \tag{A.27}$$

Thus, under the idealized deterministic assumptions above, the masked hybrid update is non-increasing in loss. If the loss is bounded below, the loss sequence converges. Standard arguments for bounded-step descent methods on smooth objectives then imply that limit points of the resulting sequence are stationary.

A.4 Computational Complexity and Overhead Accounting

This section analyzes the additional computation and memory introduced by PDT. The important distinction is that prediction is itself a computational operation: PDT does not remove work for free, but replaces selected expensive forward–backward optimization steps with cheaper DMD prediction, masking, and weight-assembly operations. Accordingly, all wall-clock metrics reported in Chapter 3 include PDT overhead.

Consider a neural network with N trainable parameters trained on a dataset with S samples and mini-batch size B. A standard first-order optimizer requires forward and backward propagation for each mini-batch. At a coarse level, the parameter-dependent cost of one batch scales as $\mathcal{O}(BN)$, and the per-epoch cost scales as $\mathcal{O}(SN)$, suppressing architecture-dependent constants.

PDT stores h recent weight snapshots and periodically constructs snapshot matrices of size $N \times h$. As shown in Appendix A.2, the main operations in one DMD prediction attempt are:

- reduced SVD of the snapshot matrix: $\mathcal{O}(Nh^2)$;
- construction of the reduced operator $\widetilde{A}$: $\mathcal{O}(Nh^2)$;
- eigendecomposition of $\widetilde{A}$: $\mathcal{O}(h^3)$;
- reconstruction of DMD modes: $\mathcal{O}(Nh^2)$;
- spectral prediction and projection: $\mathcal{O}(Nh^2)$;
- mask construction and hybrid assembly: $\mathcal{O}(N)$.

Thus, the prediction-stage complexity is dominated by $\mathcal{O}(Nh^2)$. In the experiments, h is small, typically $h = 5$, so the DMD overhead is linear in the number of model parameters up to a small history-dependent factor. Because prediction is attempted

at epoch-level intervals rather than at every mini-batch, this cost is amortized over many forward–backward passes.

The additional memory cost is dominated by the snapshot buffer:

$$\mathcal{O}(Nh). \tag{A.28}$$

For $h = 5$, this corresponds to storing five additional copies of the model weights, plus temporary buffers used by SVD, prediction, masking, and weight assembly. The cost scales with parameter count rather than activation size. Consequently, the relative memory overhead can appear large for small-input settings where baseline activation memory is small, but it becomes a smaller fraction of total memory in large-scale training where activations dominate peak memory.

Rejected predictions also contribute overhead. A prediction attempt requires constructing the DMD forecast and mask before the method determines which coordinates are accepted. Therefore, the relevant efficiency question is not whether every predicted coordinate is used, but whether the net reduction in training epochs and wall-clock time compensates for the total cost of prediction attempts, including rejected coordinates. The profiling results in Appendix A.5 report this net overhead.

A.5 Runtime, Memory, and FLOPs Profiling

To validate the complexity analysis above, we profile PDT on three representative settings: AlexNet on CIFAR-10, ResNet-50 on ImageNet-1K, and ViT-Base on ImageNet-1K. We measure wall-clock runtime, GPU memory usage, and floating-point operations for both the baseline optimizer and PDT.

Measurement methodology. Peak GPU memory is measured using PyTorch's peak-memory API. Currently allocated memory records memory actively allocated at the end of an epoch. Reserved memory records memory reserved by the PyTorch

caching allocator. PDT memory overhead includes snapshot storage, SVD workspace, and auxiliary buffers such as weight cloning buffers, optimizer-state interactions, temporary tensors, and allocator fragmentation.

Runtime overhead is decomposed into core PDT operations and auxiliary operations. Core PDT operations include SVD decomposition, DMD-based weight prediction, and masking. Auxiliary operations include snapshot cloning, CUDA synchronization, and data movement between CPU and GPU. All profiling experiments are conducted on a single GPU to avoid distributed-communication effects. Unless otherwise stated, PDT uses history length $h = 5$ and prediction interval $T_i = 1$.

A.5.1 AlexNet on CIFAR-10

Table A.1 reports memory and runtime profiling for AlexNet on CIFAR-10. PDT increases peak memory by approximately 8.4 GB. The relative increase appears large because the baseline memory footprint for AlexNet on CIFAR-10 is small, but the absolute peak memory remains within the capacity of a modern GPU. The total PDT overhead is approximately 0.333 seconds per epoch, corresponding to 4.30% of epoch time. The SVD operation itself accounts for roughly 98.3 ms.

A.5.2 ResNet-50 on ImageNet-1K

Table A.2 reports memory and runtime profiling for ResNet-50 on ImageNet-1K under batch sizes 64, 128, and 256. The absolute PDT memory overhead remains approximately constant at 1.46 GB across batch sizes, confirming that the additional storage is controlled primarily by parameter count and history length rather than data scale. As the batch size increases, baseline activation memory grows, so the relative PDT memory overhead decreases from 47.9% to 13.0%. The measured runtime overhead remains negligible relative to the epoch time.

Table A.1: Memory and runtime overhead of PDT compared to the baseline. AlexNet is trained on CIFAR-10 using SGD with batch size 256, learning rate 0.05, and a cosine-annealing learning-rate scheduler. All experiments use a single Nvidia RTX A6000 GPU.

Metric	Baseline	PDT
GPU Memory Usage		
Peak GPU Memory (MB)	735.9	9,158.2
Currently Allocated (MB)	468.7	5,078.0
Reserved by Allocator (MB)	994.0	11,454.0
PDT Memory Overhead		
Total PDT Overhead (MB)	—	8,422.4
Overhead Ratio (%)	—	**1,145%**
PDT Memory Overhead Breakdown		
Snapshot Storage (MB)	—	1,088.0
SVD Workspace (MB)	—	1,740.9
Other (MB)	—	5,593.4
PDT Runtime Overhead per Epoch		
Average Time per Epoch (s)	7.465	7.746
Total PDT Overhead (s)	—	0.333
PDT Overhead Ratio (%)	—	**4.30%**
PDT Runtime Overhead Breakdown		
Core PDT Operations (ms)	—	194.0
SVD Decomposition (ms)	—	98.3
Prediction (ms)	—	88.1
Masking (ms)	—	7.6
Auxiliary Operations (ms)	—	138.8

Table A.2: Memory and runtime overhead of PDT compared to the baseline across batch sizes. ResNet-50 is trained on ImageNet-1K using SGD with momentum, learning rate 0.1, momentum 0.9, and a cosine-annealing learning-rate scheduler. All experiments use a single Nvidia H100 GPU.

	Baseline			PDT		
Metric	bs=64	bs=128	bs=256	bs=64	bs=128	bs=256
GPU Memory Usage						
Peak GPU Memory (MB)	3,062	5,792	11,258	4,528	7,255	12,721
Currently Allocated (MB)	400	436	512	2,439	2,436	2,439
Reserved by Allocator (MB)	3,404	6,766	13,424	7,392	10,864	15,298
PDT Memory Overhead						
Total PDT Overhead (MB)	—	—	—	1,466	1,463	1,463
Overhead Ratio (%)	—	—	—	**47.9%**	**25.3%**	**13.0%**
PDT Memory Overhead Breakdown						
Snapshot Storage (MB)	—	—	—	585	585	585
SVD Workspace (MB)	—	—	—	780	780	780
Other (MB)	—	—	—	101	98	98
PDT Runtime Overhead per Epoch						
Average Time per Epoch (s)	813.9	653.1	626.5	804.0	651.5	635.3
Total PDT Runtime Overhead (ms)	—	—	—	90	90	110
Overhead Ratio (%)	—	—	—	**0.011%**	**0.013%**	**0.017%**
PDT Runtime Overhead Breakdown						
Core PDT Operations (ms)	—	—	—	51.5	50.7	51.1
SVD Decomposition (ms)	—	—	—	31.0	30.4	30.7
Prediction (ms)	—	—	—	19.7	19.5	19.6
Masking (ms)	—	—	—	0.8	0.8	0.8
Auxiliary Operations (ms)	—	—	—	38.5	39.3	58.9

A.5.3 ViT-Base on ImageNet-1K

Table A.3 reports profiling for ViT-Base on ImageNet-1K. ViT-Base has substantially more parameters than ResNet-50, so its absolute snapshot and SVD workspace costs are larger. Nevertheless, the runtime overhead remains small relative to total epoch time. The memory overhead breakdown is also consistent with the theoretical analysis: snapshot storage and SVD workspace dominate the additional memory footprint.

Table A.3: Memory and runtime overhead of PDT compared to the baseline. ViT-Base is trained on ImageNet-1K using AdamW with batch size 256, learning rate 0.0006, weight decay 0.05, and a cosine-annealing learning-rate scheduler. All experiments use a single Nvidia H100 GPU.

Metric	Baseline	PDT
GPU Memory Usage		
Peak GPU Memory (MB)	19,022	23,976
Currently Allocated (MB)	1,536	8,406
Reserved by Allocator (MB)	19,462	35,328
PDT Memory Overhead		
Total PDT Overhead (MB)	—	4,954
Overhead Ratio (%)	—	**26.0%**
PDT Memory Overhead Breakdown		
Snapshot Storage (MB)	—	1,981
SVD Workspace (MB)	—	2,642
Other (MB)	—	330
PDT Runtime Overhead per Epoch		
Average Time per Epoch (s)	831.3	832.1
Total PDT Runtime Overhead (ms)	—	877
Overhead Ratio (%)	—	**0.11%**
PDT Runtime Overhead Breakdown		
Core PDT Operations (ms)	—	193.2
SVD Decomposition (ms)	—	127.4
Prediction (ms)	—	62.8
Masking (ms)	—	3.0
Auxiliary Operations (ms)	—	683.6

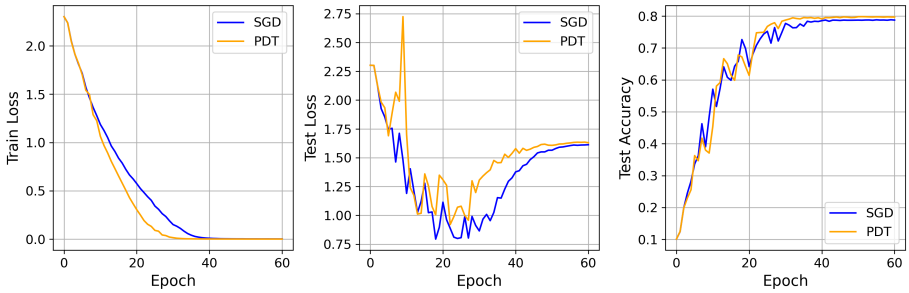
A.5.4 Scalability Interpretation

The profiling results are consistent with the $\mathcal{O}(Nh)$ memory and $\mathcal{O}(Nh^2)$ prediction-stage complexity derived in Appendix A.4. The apparent variation in relative memory overhead across architectures is explained by the composition of baseline memory. For AlexNet on CIFAR-10, baseline memory is small because the input resolution is 32×32 and activation maps are relatively inexpensive; therefore, parameter-history storage appears large relative to the baseline footprint. For ImageNet-1K training with 224×224 inputs, baseline memory is dominated by activation storage, so the same parameter-history mechanism becomes a smaller fraction of total memory.

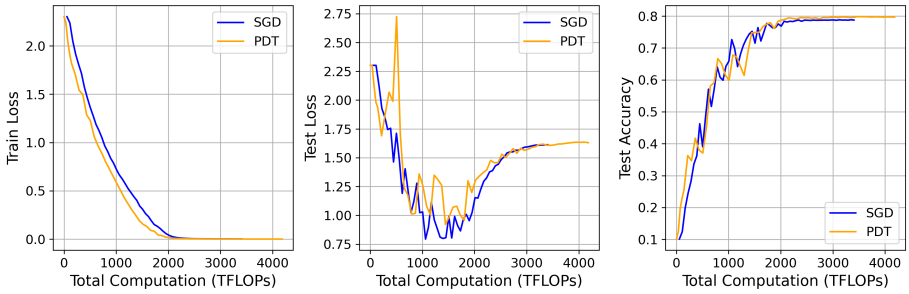
Thus, PDT's memory overhead scales primarily with parameter count and history length, whereas baseline training memory often scales strongly with activation size, batch size, and model architecture. The measured runtime overhead is small because DMD prediction is performed at epoch-level frequency and is amortized across many mini-batch forward–backward computations.

A.5.5 FLOPs Analysis

Beyond epoch-level acceleration, PDT also reduces total computation measured in floating-point operations. Figure A.3 compares AlexNet on CIFAR-10 with both epoch and accumulated TFLOPs as the horizontal axis. Although each PDT prediction attempt introduces additional snapshot-processing, DMD, and masking operations, the total TFLOPs needed to reach the baseline target loss is reduced by approximately 23.74%. This confirms that prediction overhead is compensated by the reduction in required training progress along the explicit optimization trajectory.



(a)



(b)

Figure A.3: Performance comparison between baseline optimization and PDT, with (a) epochs and (b) accumulated TFLOPs as the horizontal axis. AlexNet is trained on CIFAR-10 with batch size 256, learning rate 0.05, and a cosine-annealing learning-rate scheduler.

A.6 Supplementary Robustness and Cross-Domain Results

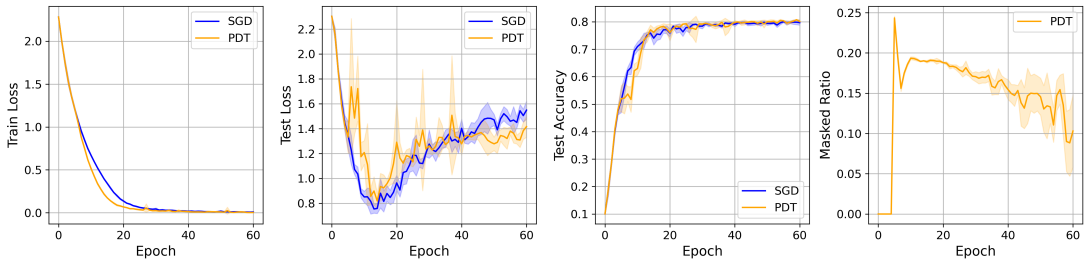
A.6.1 Behavior Under Non-i.i.d. Training

We evaluate PDT under a deliberately non-i.i.d. batching scheme on CIFAR-10 with AlexNet. In this setting, mini-batches are constructed from samples belonging to the same class, while the sequence of class-specific batches is randomized. This creates abrupt transitions between batch distributions and prevents DMD from exploiting a fixed deterministic class order. The validation set remains sampled in the standard i.i.d. manner.

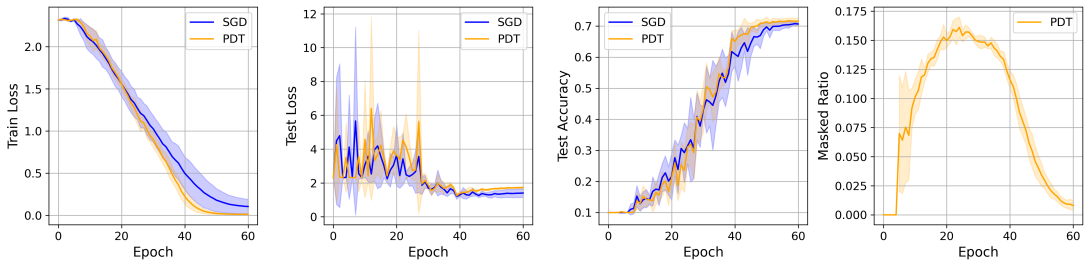
Figure A.4 and Table A.4 compare SGD and PDT under normal i.i.d. mini-batching and the non-i.i.d. scheme. Both methods converge more slowly under non-i.i.d. training, and the variance across seeds is larger. Nevertheless, PDT maintains a clear advantage in time to baseline best loss and achieves comparable or slightly improved final accuracy. This indicates that PDT is not restricted to the standard i.i.d. mini-batch condition, although irregular batch distributions can make the underlying weight trajectory more difficult to predict.

Table A.4: Performance and runtime comparison between SGD and PDT under i.i.d. and non-i.i.d. training data distributions. AlexNet is trained on CIFAR-10 with batch size 128, learning rate 0.05, and a cosine-annealing learning-rate scheduler.

Training Data Distribution	Method	Final Accuracy (mean ± std)	Best Train Loss (mean ± std)	TTB-Loss (s) (mean ± std)	Runtime Reduction (%)
i.i.d.	SGD	0.7969 ± 0.0093	0.0039 ± 0.0017	662.48 ± 7.73	9.15
	PDT	0.8011 ± 0.0067	0.0016 ± 0.0017	601.86 ± 17.78	
non-i.i.d.	SGD	0.7067 ± 0.0062	0.1053 ± 0.0874	806.83 ± 13.15	27.90
	PDT	0.7159 ± 0.0103	0.0119 ± 0.0057	581.73 ± 19.34	



(a) Normal i.i.d. training data distribution.



(b) Non-i.i.d. class-ordered mini-batch distribution.

Figure A.4: Performance comparison between SGD and PDT under i.i.d. and non-i.i.d. training data distributions. AlexNet is trained on CIFAR-10 with batch size 128, learning rate 0.05, and a cosine-annealing learning-rate scheduler. Shaded regions indicate standard deviation across five random seeds.

A.6.2 Cross-Domain Evaluation on Natural Language Processing

To evaluate whether PDT can transfer beyond image-classification settings, we also test it on a text-classification task. We train a deep LSTM model for four-class topic classification on the AG News dataset [89, Gulli]. The model consists of an embedding layer, a four-layer stacked LSTM with 512 hidden units per layer, and a linear classifier, for a total of approximately 8.25 million parameters. This setting differs from the vision experiments in both data modality and optimization dynamics, since the model is recurrent and trained through backpropagation through time.

The baseline uses SGD with learning rate 0.1 and batch size 128 for 30 epochs, together with a cosine-annealing learning-rate scheduler. PDT uses the same default hyperparameters as the vision experiments:

$$\tau = 5, \qquad h = 5, \qquad T_0 = 5, \qquad T_i = 1.$$

Results are averaged over three random seeds.

Table A.5: Performance comparison on the AG News text-classification task using a deep LSTM model. Training is performed on a single Nvidia RTX A6000 GPU for 30 epochs with learning rate 0.1, batch size 128, and a cosine-annealing learning-rate scheduler.

Method	Final Accuracy	TTB-Loss (s)	TTB-Acc (s)	Runtime Reduction (%)	
				Train Loss	Val. Acc.
SGD	86.10 ± 1.30	700.8 ± 72.1	694.4 ± 76.9	**26.1**	**20.5**
PDT	88.44 ± 0.59	517.9 ± 102.2	551.7 ± 120.5		

As summarized in Table A.5, PDT reaches the baseline's best training loss 26.1% faster and the baseline's best validation accuracy 20.5% faster. It also achieves a higher final validation accuracy, 88.44% compared with 86.10% for the baseline. Figure A.5 shows the corresponding training dynamics.

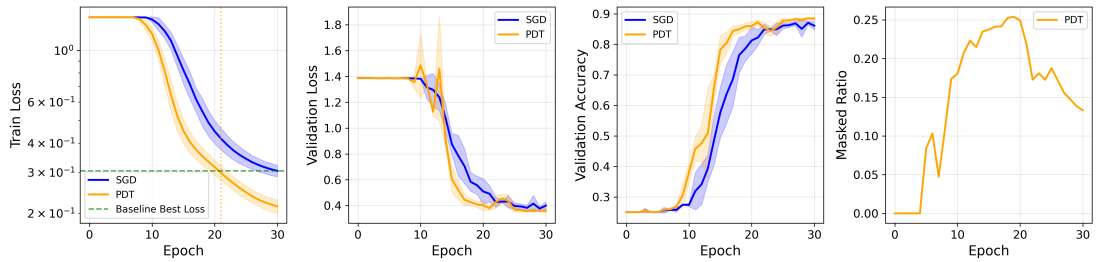


Figure A.5: Performance comparison between baseline SGD and PDT on the AG News text-classification task using a deep LSTM model. Training uses a single Nvidia RTX A6000 GPU for 30 epochs, learning rate 0.1, batch size 128, and a cosine-annealing learning-rate scheduler. Shaded regions indicate standard deviation across three random seeds.

These results provide supplementary evidence that PDT is not limited to convolutional or vision-transformer image models. However, because this experiment is a single cross-domain case study rather than a systematic NLP benchmark suite, it should be interpreted as evidence of possible generality rather than a complete characterization of PDT across language-modeling settings.

Appendix B

Supplementary Details for Structured Predictability Measurement

This appendix provides the protocol details, calibration analyses, robustness checks, attribution audits, and extended diagnostics supporting Chapter 4. Its organization follows the role of each result in the dissertation: first the experimental corpus and measurement conventions, then the auxiliary–bulk mapping, pocket dynamics, and condition-specific extensions.

Appendix B.1 records the complete experimental coverage, measurement clocks, and externally defined state quantities. Appendix B.2 specifies grouping, calibration, numerical conventions, and robustness checks. Appendix B.3 audits the auxiliary–bulk partition across the two training regimes. Appendix B.4 collects spatial, temporal, persistence, overlap, and cross-scale diagnostics. Appendices B.5 and B.6 retain additional architecture- and optimizer-conditioned analyses.

B.1 Experimental Coverage, Protocol, and External Quantities

The basic probe definitions are given in Section 4.2; they are not repeated here. This section records the training corpus behind Section 4.3, checkpoint accounting for both regimes, measurement clocks and anchors, and the two external quantities used in the state-correspondence analysis. Appendix B.1.1 covers the controlled vision panel and Appendix B.1.2 the further run families that Section 4.3.5 draws on. Appendix B.1.3 covers the pretraining checkpoints, and Appendix B.1.4 the measurement clocks, anchor construction, and seeds.

B.1.1 Vision Panel: Architectures, Optimizers, and Recipes

The controlled vision panel crosses five architectures with six training configurations. The architectures span three structural families: an MLP; AlexNet and a CIFAR-style ResNet-18 variant with its last stage removed; and two ViT scales. The smaller model is called ViT-tiny in Chapter 4 and appears as ViT in legacy code labels and embedded figure legends; these names denote the same architecture. The larger model is ViT-mid. The six configurations are plain SGD, SGD with momentum, SGD with weight decay, Adam, AdaGrad, and SAM. Table B.1 gives the shared recipe, the optimizer definitions, and the learning rate of every cell together with the parameter count of each architecture.

All cells train from scratch for 50 epochs at batch size 128 under a cosine schedule stepped once per epoch, with seed 42, and weights are recorded at every epoch boundary. That epoch-level snapshot cadence is what sets the probe time step on this side. Most cells take the model-default learning rate; the hand-tuned exceptions are listed in the table. AlexNet is the one architecture with a different input pipeline, since it resizes to 128 pixels and trains without augmentation.

(a) Shared recipe

Item	Setting
Dataset	CIFAR-10 (full 50k train / 10k test); CIFAR-100 only in the transfer family
Loss / batch size	cross-entropy; batch size 128 everywhere
Epochs / schedule	50 epochs; schedules stepped once per epoch
Cosine schedule	`CosineAnnealingLR` to 0 with $T_{\max}$ = epochs
Augmentation	`RandomCrop(32, padding=4)` + `RandomHorizontalFlip`
AlexNet exception	`Resize(128)`, `Normalize(0.5, 0.5)`, no augmentation
Normalization	CIFAR-10: mean $(0.4914, 0.4822, 0.4465)$, std $(0.2023, 0.1994, 0.2010)$ CIFAR-100: mean $(0.5071, 0.4865, 0.4409)$, std $(0.2673, 0.2564, 0.2762)$
Statistics pinning	the transfer and calibration families pin CIFAR-10 statistics for all their runs, including the CIFAR-100 ones, so transfer pairs see identical input statistics
Seed	42 for the panel; 43 and 44 for replicates
Snapshots	weights recorded at every epoch boundary

(b) Optimizer definitions

Name	Rule	Momentum	Weight decay	Other
SGD	SGD	0	0	—
SGD-mom	SGD	0.9	0	—
SGD-wd	SGD	0	5×10^{-4}	—
Adam	Adam	$\beta = (0.9, 0.999)$	0	—
AdaGrad	AdaGrad	—	0	—
SAM	SAM over plain SGD	0	0	$\rho = 0.05$

(c) Architectures and per-cell learning rates

Architecture	Params	SGD	SGD-mom	SGD-wd	Adam	AdaGrad	SAM
MLP	3,805,450	0.1	0.01	0.1	10^{-3}	10^{-2}	0.05
AlexNet	57,044,810	0.05	0.01	0.05	10^{-3}	5×10^{-3}	0.01
ResNet	2,777,674	0.1	0.1	0.1	10^{-3}	10^{-2}	0.1^{s}
ViT	546,186	0.1	0.01	0.1	$10^{-3\,s}$	10^{-2}	$0.05^{s\dagger}$
ViT-mid	2,855,562	0.05	0.05	0.05	5×10^{-4}	5×10^{-3}	$0.025^{s\dagger}$

All 30 cells are trained from scratch under the shared recipe of panel (a) with seed 42. Parameter counts are for the 10-class heads; ViT-mid has 2,875,812 parameters with the 100-class head used in the transfer family. Most cells take the model-default learning rate; a few were hand-tuned, including ViT with SAM (0.05) and ViT-mid with SAM (0.025). ^{s} Cell additionally re-trained with seeds 43 and 44. † Those two SAM re-runs used the model-default learning rate (0.1 and 0.05), twice the hand-tuned seed-42 value, so these two cells are not learning-rate-matched across seeds.

Table B.1: Vision panel: baseline recipe, optimizer definitions, and per-cell learning rates. **(a)** the shared baseline recipe used unless a family-specific delta is stated; **(b)** the six training configurations of the main panel; **(c)** the five architectures with their parameter counts and the learning rate of each cell.

B.1.2 Additional Run Families

Around the panel, six further families vary one ingredient at a time while holding the shared recipe fixed (Table B.2). Two of them supply the controlled comparisons of Section 4.3.5: a Muon arm over the same five architectures, and a weight-decay and momentum dose ladder on plain SGD. The remaining families support the robustness and calibration checks: seed replicates on five cells, a transfer-dynamics family between CIFAR-10 and CIFAR-100, and a probe-calibration batch covering a schedule zoo, basin slices, a learning-rate dose ladder, and a blind audit whose configurations were withheld from the analysis side. The corpus is 139 training runs and one frozen source checkpoint.

Two runs in the momentum ladder collapsed and are excluded from every readout while being kept on disk: MLP at momentum 0.95, which ends at 0.170 test accuracy, and AlexNet at momentum 0.95, which ends at 0.100, the chance level for ten classes. Neither contributes to the momentum comparisons.

B.1.3 Pretraining Checkpoint Coverage

Table B.3 lists the released checkpoint series we analyze, and Figure B.1 places the measured anchor domain on the training-step axis of each size. The main panel is Pythia-70M, 160M, and 410M. The extended sizes 14M, 31M, 1B, and 1.4B are processed under the identical protocol. Six of the seven sizes enter the cross-scale readings of Appendix B.4.3; 1B is covered by the coverage accounting here but is excluded from every cross-scale reading, on the three grounds given there.

Checkpoints are released every 1,000 steps after a dense early grid, giving 154 revisions up to step 143,000. We evaluate 123 anchors on a 1,000-step grid spanning steps 7,000 to 129,000. Both edges are protocol choices rather than data limits. The lower edge keeps the five-checkpoint history window, which reaches back to step 2,000, clear of the 1,430-step learning-rate warmup. The upper edge leaves a complete

Family	Runs	Recipe deltas with respect to the panel
Panel	30	The five architectures of Table B.1(c) crossed with the six configurations of Table B.1(b); from scratch, seed 42.
Muon arm	5	The same five architectures under Muon (momentum 0.95, `ns_steps`=5), learning rates 0.02 (MLP), 0.01 (AlexNet), 0.02 (ResNet), 0.005 (ViT), 0.01 (ViT-mid). One additional stalled diagnostic run at the ViT model-default learning rate 0.02 is kept on disk and excluded from readouts.
Weight-decay and momentum dose	26	Plain SGD at the model-default learning rate, cosine, 50 epochs, seed 42. Weight-decay ladder $\{10^{-4}, 10^{-3}, 2 \times 10^{-3}\}$ over the five architectures, with the override applied to plain SGD only; momentum ladder $\{0.5, 0.95\}$ over the five architectures; one further MLP run at momentum 0.7. Two runs collapsed and are kept on disk but excluded from every readout: MLP at momentum 0.95 (final test accuracy 0.170) and AlexNet at momentum 0.95 (final test accuracy 0.100, chance level).
Seed replicates	10	Five panel cells re-trained at seeds 43 and 44, recipe otherwise identical: ResNet–SAM, ResNet–Muon, ViT–Adam, ViT–SAM, ViT-mid–SAM. The two SAM re-runs are not learning-rate-matched (Table B.1 note †).
Transfer dynamics	22	All ViT-mid under plain SGD (momentum 0, weight decay 0), CIFAR-10 statistics pinned, plus one frozen source checkpoint. CIFAR-10 to CIFAR-100 and CIFAR-100 to CIFAR-10 at constant learning rate, two arms (0.005 and 0.025), seeds 42/43/44 with weight seed k and data-order seed $1000{+}k$; scratch controls on both datasets; a stationary continuation (+50 epochs at 0.025) and step learning-rate switches at epoch 25.
Probe calibration	45	Plain SGD, seed 42, CIFAR-10 statistics pinned. A 16-run schedule zoo (ResNet and ViT-mid over linear, two step schedules, exponential, warmup-cosine and cosine-with-floor, two ResNet constant-rate runs, a 100-epoch cosine run and a basin anneal); 10 basin slices (early and late basins crossed with five constant rates, 12 epochs); a 5-run learning-rate dose ladder ($\times\{\frac{1}{25}, \frac{1}{10}, \frac{1}{2}, 2, 4\}$ of base 0.025); two stationary continuations; and a 12-run blind audit whose configurations were withheld from the analysis side.
Total	139	training runs, plus one frozen source checkpoint

Table B.2: Run families of the vision corpus. The panel is the set analyzed throughout Section 4.3.2 to Section 4.3.4; the Muon arm and the dose ladders supply the controlled comparisons of Section 4.3.5; the remaining families support the robustness and calibration checks. The two collapsed momentum runs are kept on disk and excluded from every readout.

(a) Sizes, architecture, and coverage on disk

Model	Parameters		$L \times d_{\text{model}}$	Coverage
	total	non-embedding		
Main panel				
Pythia-70M	70,426,624	18,915,328	6×512	154 revisions; 123 anchors
Pythia-160M	162,322,944	85,056,000	12×768	154 revisions; 123 anchors
Pythia-410M	405,334,016	302,311,424	24×1024	123 anchors (246 shards); spectrum 154 rev.
Extended set, reported in Appendix B.4.3				
Pythia-14M	14,067,712	1,189,888	6×128	123 anchors (246 shards); spectrum 154 rev.
Pythia-31M	30,494,720	4,739,072	6×256	123 anchors (246 shards); spectrum 154 rev.
Pythia-1B†	1,011,781,632	805,736,448	16×2048	123 anchors (246 shards); spectrum 154 rev.
Pythia-1.4B	1,414,647,808	1,208,602,624	24×2048	123 anchors (246 shards); spectrum 154 rev.

"Total" counts pure parameters of the released fp16 weights, with no buffers; "non-embedding" excludes the untied input and output embeddings (vocabulary 50,304). The 70M and 160M values are read from the verified reference card; the five remaining sizes are recomputed from the on-disk group sizes, by a method that reproduces the reference card exactly on 70M. The extended sizes were processed at a second compute site under the identical engine and protocol, with per-size equivalence gates passing at zero maximum relative error (1260 units at 14M and 31M, 4824 at 410M and 1.4B, 3240 at 1B); the gate tables carry non-empty exemption counts for degenerate units, and every non-exempt row passes. Anchors are steps 7000 to 129000 at spacing 1000, and each anchor produces one panel and one segment shard.

(b) Public telemetry and evaluation provenance

Model	Public train-loss telemetry	Splice check	Official zero-shot evals
Pythia-14M	none public	—	none public
Pythia-31M	none public	—	none public
Pythia-70M	`32t0zbcs` (to 143k)	single run	27 JSON records
Pythia-160M	`3mvtbwii` (to 143k)	single run	27 JSON records
Pythia-410M	`3jg1upg7` $[1, 35\text{k})$ + `12j05401` $[35\text{k}, 143\text{k}]$	overlap $\lvert\Delta\text{loss}\rvert \leq 0.019$	28 JSON records
Pythia-1B†	`339s0ka6` + `1ena81fo` + `318z89xb` (restart chain, bf16)	49-step overlaps, $\Delta\text{loss} \leq 4 \times 10^{-16}$	28 JSON records
Pythia-1.4B	`qiu9n7a6` (fp16)	single run	28 JSON records

All telemetry comes from the public training project of the checkpoint release, in the run group matching the released weights, with configurations cross-checked item by item against the published training configurations. The 1B chain starts at step 2001; steps 0 to 2000 are missing and the analysis domain starts at 7000, so the gap is outside it. Official zero-shot evaluations come from the release repository and were produced by an older harness, so only within-harness trends are read from them.

† **Pythia-1B checkpoint defect.** Revision `step116000` of the 1B series has wrong content upstream: it is the only 1B revision exported by a different release pipeline, and tensor forensics place it as an inserted state from a different run, with $w_{115\text{k}} \approx w_{117\text{k}}$. The defect dates from the first weight upload and is not repairable downstream. Affected cells are masked at read time, by probe family: base-anchor rows at a=116k for all τ; history-window anchors a=116k to 121k for all τ; the future-endpoint cells $(a, \tau) \in \{(115\text{k}, 1), (114\text{k}, 2), (113\text{k}, 3), (111\text{k}, 5), (106\text{k}, 10), (102\text{k}, 14)\}$; and the `step116000` row of the embedding-spectrum series. Public telemetry and the official evaluations are unaffected, since neither is derived from local checkpoints.

Table B.3: Pretraining checkpoint coverage across the main and extended Pythia sets, including public telemetry provenance and the `step116000` checkpoint defect.

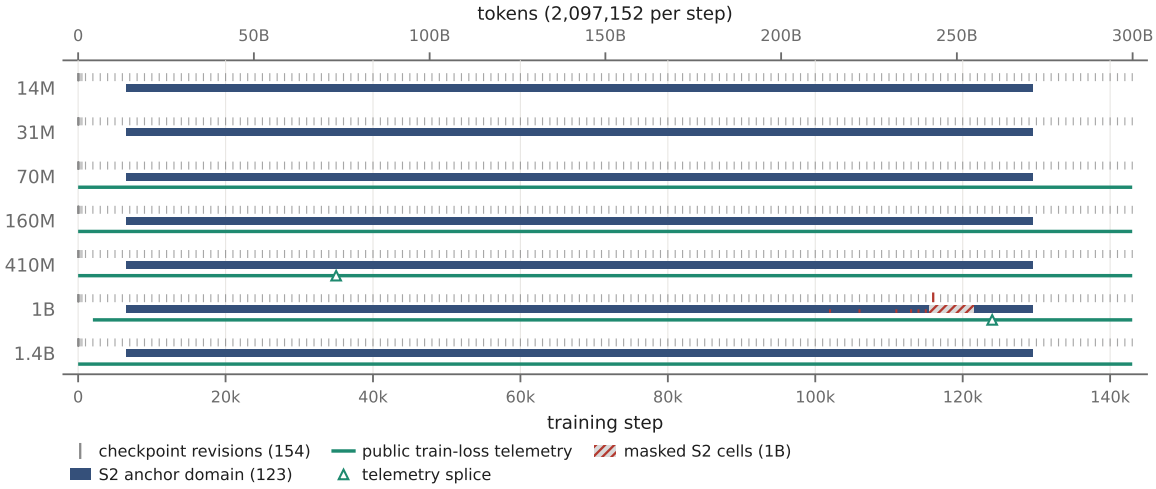


Figure B.1: Checkpoint and anchor coverage across the pretraining scales. Each row is one model scale on a shared training-step axis, with tokens on the top axis at 2,097,152 per step. Ticks give the 154 public checkpoint revisions, released on the schedule $0, 1, 2, 4, \ldots, 512$ and every 1k steps thereafter up to step 143k. Bars give the anchor domain actually measured, 123 anchors on a 1k grid spanning 7k to 129k for every scale. The domain is trimmed at both ends for protocol reasons rather than for data availability: the head drops the warmup phase and leaves room for the backward history window, and the tail reserves a complete future window for the longest horizon ($143\text{k} - 14\text{k} = 129\text{k}$ at τ=14). Lines give the span of public training-loss telemetry, and the two smallest scales carry no line because no public run exists for them. Triangles mark splices where the released telemetry is assembled from more than one run. Hatching on the 1B row marks the step-116k checkpoint defect and the anchors masked because of it, with short ticks at the horizon-specific future endpoints; the telemetry line and the official evaluations are unaffected by the masking.

future window for the longest horizon, since $143{,}000 - 14{,}000 = 129{,}000$ at $\tau = 14$ checkpoints.

Public training-loss telemetry and the official zero-shot evaluations are used only as external context, and their provenance is recorded in panel (b) of the table, including the two sizes whose released telemetry is assembled from more than one run. One size carries a checkpoint defect: revision `step116000` of the 1B series is wrong upstream, and the cells whose history or future window touches it are masked at read time. The masking windows are listed in the table footnote and are applied wherever 1B appears.

B.1.4 Measurement Clocks, Anchors, and Seeds

Clocks. One probe step is one epoch on the vision side and one released checkpoint on the pretraining side. A checkpoint step spans 1,000 optimization steps, or about 2.10 billion tokens at 2,097,152 tokens per step. The two clocks index different amounts of optimization, which is why cross-regime readings compare the ordering within a matched trajectory rather than levels across regimes.

Anchors and windows. Both regimes use horizon $\tau = 5$ and history length $n_{\text{past}} = 5$ unless stated otherwise, with the reference direction Δw_{a-1} and the future target $\Delta w_a^{(\tau)}$ as defined in Section 4.2. Vision probes are evaluated at anchor epochs $\{5, 10, 20, 30, 40\}$. An anchor is admissible only when its full history window and its full future window lie inside the recorded trajectory, which is what fixes the edges of the pretraining anchor domain above. The pretraining panel additionally records $\tau \in \{1, 2, 3, 5, 10, 14\}$ so that horizon sensitivity can be read from the same shards.

Seeds and determinism. The panel trains at seed 42, with replicates at 43 and 44 on the five cells listed in Table B.2. On the measurement side, three seeds are fixed and shared across every lane so that a recomputation reproduces the same numbers: the rank-selection sample (Appendix B.2.3), the per-layer group subsample

(Appendix B.2.1), and the permutation-null draws (Appendix B.2.4), whose seed is composed from the draw index, anchor, layer, and segment.

Compute. The vision corpus was trained on a single workstation with two NVIDIA RTX A6000 GPUs (48 GB each), one run per GPU. Median wall-clock for a 50-epoch run, taken over each architecture's optimizer cells, is about 12 minutes for the MLP, 15 for ViT, 22 for AlexNet, 23 for ViT-mid, and 38 for the ResNet; the slowest single run is AlexNet under Muon at about 4.8 hours, where the orthogonalization step dominates on the largest weight matrices. Probe and panel computation is CPU-only and in double precision, on a 16-core machine with 128 GB of memory: one pretraining anchor of the full panel protocol, including its permutation nulls, takes about 10 minutes at 160M and 25 at 410M. The panels for the two largest sizes ran on large-memory CPU nodes (64 cores, 2 TB) with anchors processed in 8 to 10 parallel lanes, at roughly 17 to 19 hours per size; the embedding-spectrum sweep is checkpoint-bound, at seconds to about a minute per revision. The stored checkpoint packs occupy 24, 54, 126, 311, and 436 GB for 70M, 160M, 410M, 1B, and 1.4B. Software: Python 3 with PyTorch, used only to deserialize checkpoints, together with NumPy, SciPy, and pandas. Figure 4.5 presents the state-correspondence results in the main text. The following subsection records the definitions, averaging objects, and interpretive scope needed to reproduce and delimit that comparison.

B.1.5 External State Quantities and Their Interpretation

Movement directedness. For an anchor a and horizon τ, movement directedness is the net displacement of the future window divided by the path length actually travelled inside it,

$$\mathrm{MD}_S(a,\tau) = \frac{\left\| w_{a+\tau,S} - w_{a,S} \right\|_2}{\sum_{t=a}^{a+\tau-1} \left\| \Delta w_{t,S} \right\|_2}. \tag{B.1}$$

It lies in $[0,1]$, equals one when the window is traversed in a straight line, and falls towards zero when the steps cancel. It is a property of the realized trajectory alone:

no probe, no history window, and no model of the dynamics enters it. It is also not causally prior to the probes, since both read the same stored weights; the comparison below is a correspondence between two summaries of one record.

Loss-side gradient coherence. The second quantity is built only from scalar training telemetry. Over the same window it compares the realized loss decrease with the loss decrease that the same gradient magnitudes would have produced had every step been perfectly aligned,

$$\mathrm{GC}(a,\tau) = \frac{L_{a-1} - L_{a+\tau-1}}{\left(\sum_{s\in W} \eta_s\right) \left\langle \|g_s\|_2^2 \right\rangle_{s\in W}}, \tag{B.2}$$

where W is the set of optimization steps inside the window, η_s is the learning rate at step s, and g_s is the gradient restricted to the bulk backbone. The numerator is read from the loss curve and the denominator from recorded per-step learning rates and gradient norms, so the estimate contains no parameter-direction information at all. This is what makes it a non-circular comparison for a directional probe.

B.1.6 Relation to Existing Coherence and Noise Measures

Both quantities have close relatives in the literature, and the differences matter for how the comparison should be read.

Gradient coherence in the sense of Chatterjee [7] is defined on *gradients*: it measures how much per-example gradients agree with one another within a batch, and it requires access to those per-example gradients. Equation (B.2) instead reads coherence off the loss curve, as the fraction of the available first-order decrease that the optimizer actually realized, and needs only telemetry that a training run already logs. It therefore measures agreement *across steps* within a window rather than across examples within a step, and the two can come apart whenever the loss surface curves inside the window.

Movement directedness is a trajectory-level quantity of the kind studied by Singh et al. [72], and it is related to, but not the same as, the observation that optimization concentrates in a low-dimensional subspace [28]. A trajectory can stay inside a small subspace while doubling back inside it, which lowers Eq. (B.1) without changing the subspace, and this is exactly the distinction that separates the displacement-direction family from the subspace-residual family in Section 4.2. Values of Eq. (B.1) well below one over short windows are also the signature of the diffusion-like regimes described by Mandt et al. [48] and Feng and Tu [18].

None of these quantities is used here to certify another. They are registered as independent summaries of the same stored trajectories.

B.2 Measurement Conventions, Calibration, and Robustness

This section makes explicit the implementation and reporting conventions behind the group-level measurements and then provides the full multi-probe robustness and representative multi-seed checks. Section 4.2 defers five implementation details to this appendix: the exact predictor constructions, the rank-selection rule, the null generation procedure, the grouping rules, and the numerical conventions. Appendices B.2.1 to B.2.5 give them, in that order. Appendix B.2.6 then reports what the group-level readouts do when the grouping convention is changed, together with the full versions of the within-family comparisons summarized in Section 4.3.2, and Appendix B.2.7 states the reporting rules that follow from those measurements.

B.2.1 Measurement Units and Grouping Conventions

Every group-level readout in Chapter 4 and this appendix is conditional on a rule that partitions a tensor's scalars into measurement units. We call such a rule a *cut*. Tables B.4 and B.5 list all cuts used in Chapter 4 together with their group sizes and their status with respect to chance passing.

Flat K=16, the main protocol. Each tensor's parameter vector is flattened in row-major order and split into consecutive disjoint blocks of K=16 scalars. A trailing block shorter than K is dropped, and a tensor holding fewer than K scalars carries no group at all; on the vision panel this removes only the ten-parameter classifier bias of each model. The cut is cheap, uniform in group size, and identical across architectures, which is what keeps the group size, and therefore the chance geometry, matched in cross-architecture comparisons. It is not aligned to the architecture. In row-major layout a flat block of a convolutional tensor is approximately one 3×3

Cut	One group is	D	What it reads	Chance status at θ=0.7
Flat protocol, the main measurement convention				
flat K=16	16 contiguous scalars of the row-major flattened tensor	16	Between-group ESA dispersion sits at the binomial sampling floor (1.1×, median over 30 cells), so blocks are statistically homogeneous. Bulk group-level FRR hugs the chance floor (median excess .014, against .148 for the auxiliary class). Cross-architecture ordering claims do not survive recutting.	Not immune: of the 31.2% bulk pass rate, about 25 pp is chance (null_θ median .23 to .25), so the calibrated readout is required.
Structural cuts, physical and functional axes				
filter 3×3	one ResNet output channel's full 3×3 kernel stack	27–2304	Names the unit behind ResNet's flat readout: 66 to 87% of ResNet flat LRGF is intra-kernel coherence, destroyed by within-tensor permutation (1×1 shortcut Δ=.0025). FRR<0.7: 1.5%. ESA dispersion 1.9× floor, not coupled to FRR.	Immune.
row 1×1	one 1×1-convolution row	64–128	Essentially no pockets: FRR<0.7 = 0.1%.	Immune.

Table B.4: Grouping conventions on the vision panel. D is the number of scalars per group. "Immune" means the membership-resample null passes θ=0.7 at a rate near zero, so the raw pass fractions carry a negligible chance contribution. The flat K=16 protocol is the one cut where a large share of raw passes is chance, which is what motivated the calibrated readout LRGF_{cal}. Levels are not comparable across cuts, since group sizes and chance floors differ, so every level claim in this paper names its cut. Structural and magnitude readouts are pooled over the 18 recipe cells of the group-level lanes and anchors 5 to 40.

Table B.4 (continued)

Cut	One group is	D	What it reads	Chance status at θ=0.7
Structural cuts, physical and functional axes (continued)				
head *qkv*	per-head q/k/v row block of the fused *qkv* tensor	4096 / 12544	Strongest structural unit: ESA dispersion 4.4× floor; FRR<0.7 of 19.8% (ViT-mid) and 5.2% (ViT); restores group-level ESA–FRR coupling (cell-wide median −.42 against −.16 for flat).	Immune (pass rate ≈0 for $D\geq$448).
head proj	per-head projection rows	4096 / 12544	FRR<0.7: 10.0% (ViT-mid), 1.2% (ViT).	Immune.
neuron pair	hidden neuron fan-in and fan-out concatenated	256 / 448	FRR<0.7: 12.7% (ViT-mid), 6.9% (ViT).	Immune.

Table B.4 (continued)

Cut	One group is	D	What it reads	Chance status at θ=0.7
Magnitude cut, causal activity axis				
magnitude, 16 bins	per-tensor rank by pre-anchor window $\|\Delta w\|_2$, 16 equal bins (causal)	108–36864	Strongest single-axis density sorter and not a raiser of totals: monotone 16-bin gradient (FRR<0.7 pooled 0% to 20.1%; ESA>0.7 0.1 to 27.2; VCS>0.4 0.2 to 36.4); the top three bins concentrate 2 to 3× (ViT-mid 27.9% against head *qkv* 19.8%); totals slightly lower than structural (0.4/3.2/8.1 against 1.3/6.7/12.9 for ResNet, ViT, ViT-mid).	Immune (all $D\geq$108).

Cut	One group is	D	What it reads	Chance status at θ=0.7
positional (reference)	*qkv*: per-head block; MLP: neuron pair	10^3–10^5	Reference convention of the Pythia protocol. Within-tensor permutation *raises* the flat *qkv* reading (Δ= − .140, structural .099 against permuted .239), the signature of comoving mass concentrated in a few strong coordinates.	Bare θ blind at large D; calibrated readout used.
`attn_out` head	per-head output block	32768	Excess +.263 at step 70k while the bare-θ column reads .000, the sharpest case of large-D blindness.	Calibrated only.
W_{qkv} row	one *qkv* row, the natural coherence scale	512 (70M)	Absolute pockets revive at row granularity: FRR<0.7 stable at 4 to 5%, against $\approx$ 0 at head level. Row-length invariance across D=128 to 3072 (410M peaks .432/.434/.438 at D=1024/512/256; 160M scan; 1B below 0.5% after masking).	θ legal (analytic floor $\approx$0).

Table B.5: Grouping conventions on the Pythia side. All Pythia readings are registered as correspondences and are not used to certify one another. Column conventions follow Table B.4. The last two rows are the recut used for the granularity ladder in Appendix B.2.6.

kernel, whereas for a feature-dimension linear tensor it is 16 adjacent input dimensions with no functional identity, and the two readings are not the same measurement.

Structural cuts (vision). Structural cuts take the unit from the architecture instead of from the memory layout. For the ViT family a group is one head's q, k, or v row block of the fused qkv tensor, one head's projection rows, or one hidden neuron's fan-in and fan-out concatenated. For ResNet a group is one output channel's full 3×3 kernel stack, or one 1×1 convolution row. Group sizes run from 27 to 12,544.

Magnitude cut (vision). Within each tensor, coordinates are ranked by the L_2 norm of their pre-anchor update window and split into 16 equal bins. The ranking reads only pre-anchor history, so the cut is causal and carries no information from the target window.

Pythia segments. Each transformer layer contributes four segments. The fused $[3d, d]$ qkv tensor is stored head-major, and one group is the block belonging to one head and one of q, k, v, of size $d_{\text{head}}\, d$ (32,768 at 70M); groups are enumerated part-major, that is, all q blocks, then all k, then all v. In `attention.dense` the input axis carries the concatenated head outputs, so one group is one head's column block. In the MLP one group is a neuron pair: row j of `dense_h_to_4h` joined with column j of `dense_4h_to_h`, of size $2d$. In the vector segment each vector parameter is its own group. The two embedding matrices form separate segments and stay outside every aggregation (Appendix B.3.3). Two finer conventions are used for the granularity studies in Appendix B.2.6: one W_{qkv} row ($D{=}d$), and the two MLP halves taken separately.

Group subsampling. Layers with more than 1,500 groups are subsampled to that cap before group-level statistics are formed, with a seed derived from the layer name so that the same groups are drawn on any recomputation. Tensor-level and pooled readouts use all groups.

B.2.2 Predictor Constructions

All three predictors read only the n_{past} one-step updates before the anchor. For a unit S write the history columns as $h_1, \ldots, h_m$ with $m = n_{\text{past}}$, so that $h_m = \Delta w_{a-1,S}$.

Inertia. $\widehat{f}_{\text{in}} = \tau\, h_m$.

Linear trend. Per coordinate, an affine model $h_s \approx \alpha + \beta s$ is fitted by least squares over $s = 0, \ldots, m-1$, and the fitted updates are summed over the next τ steps,

$$\widehat{f}_{\text{lin}} = \sum_{q=0}^{\tau-1} \big(\alpha + \beta(m+q)\big) = \tau\alpha + \beta \sum_{q=0}^{\tau-1} (m+q). \tag{B.3}$$

The fit is on updates rather than on weights, so the output is a displacement and not an extrapolated checkpoint.

DMD. The construction has three steps. *(i) Standardization.* Each coordinate row of the history is divided by its standard deviation over the m columns. Rows whose standard deviation falls below 10^{-30} are left unscaled, and any unit containing such a row is predicted as zero. Write $\tilde{H}$ for the standardized history. *(ii) Fit.* With $X = [\tilde{h}_1, \ldots, \tilde{h}_{m-1}]$ and $Y = [\tilde{h}_2, \ldots, \tilde{h}_m]$, take the thin SVD $X = U\Sigma V^\top$, truncate to $r = \min(|S|, m-1)$, which is four columns under the default protocol, and form the reduced map

$$\tilde{A} = U_r^\top\, Y\, V_r\, \Sigma_r^{-1}, \tag{B.4}$$

with singular values below 10^{-10} inverted to zero. *(iii) Rollout.* Eigendecompose $\tilde{A} = W\Lambda W^{-1}$, lift the eigenvectors to $\Phi = U_r W$, and expand the last standardized update in them, $b = \Phi^\dagger \tilde{h}_m$. The τ-step prediction is

$$\widehat{f}_{\text{DMD}} = \Re\Big\{ \Phi \Big(\sum_{k=1}^{\tau} \Lambda^k \Big) b \Big\}, \tag{B.5}$$

with each coordinate rescaled by the standard deviation removed in step (i). Summing Λ^k over $k = 1, \ldots, \tau$ instead of applying Λ^τ is what makes the output the cumulative displacement over the horizon, which is the quantity the target $\Delta w_{a,S}^{(\tau)}$ measures.

Containment in the recent-history span. Because $\Phi = U_r W$, the prediction lies in the column space of U_r, and the row standardization cancels exactly between the fit in step (ii) and the rescaling in step (iii). The DMD prediction therefore lies in the span of the first $m - 1$ raw history columns, which is contained in the span the FRR projector retains whenever that projector covers the full history rank. This is the containment behind reading FRR as the directional headroom available to a predictor built from the recent history, as in Section 4.3.2: the residual fraction that FRR measures bounds how well any vector drawn from that span can align with the realized displacement.

B.2.3 Rank Selection

The FRR projector retains a rank chosen by the Gavish–Donoho optimal hard threshold. For a unit whose history matrix has singular values $s_1 \geq \cdots \geq s_p$ and aspect ratio $\beta = \min(|S|, m)/\max(|S|, m)$, the threshold is

$$\begin{aligned} \theta_{\mathrm{GD}} &= \omega(\beta)\,\mathrm{median}(s), \\ \omega(\beta) &= 0.56\beta^3 - 0.95\beta^2 + 1.82\beta + 1.43, \end{aligned} \tag{B.6}$$

and the retained rank is $\max\big(2, |\{i : s_i > \theta_{\mathrm{GD}}\}|\big)$. The floor of two is part of the protocol rather than a numerical guard, and the rank is capped at the number of history columns.

Applying the rule per unit would force one projection per group. The default rule instead draws 256 units per model–anchor cell with a fixed seed, computes their Gavish–Donoho ranks, and applies the median rank uniformly to all units of that cell, which keeps the projection batched at one rank per cell. The flat-protocol group

readouts reported in Chapter 4 and this appendix are computed at the full history rank $r = n_{\text{past}} = 5$. The pooled bulk group-level FRR median is 0.815, just below the isotropic reference $\sqrt{(D-r)/D} = 0.829$ that this rank and D=16 imply. This is the same isotropic-reference construction that Figure 4.4 measures excess against, evaluated there at each structural group's own size rather than at D=16.

B.2.4 Null Generation and the Calibrated Readout

FRR compares a D-dimensional target with an r-dimensional subspace, so a target with no temporal structure at all still passes a fixed threshold at a rate that depends on D and r. A readout compared across group sizes therefore needs a chance reference computed at its own geometry.

Membership resampling. Within one pool of comparable units, a tensor on the vision side and a (layer, segment) block on the Pythia side, all scalar coordinate histories are collected, permuted, and re-chunked into groups whose size multiset is exactly that of the observed cut. FRR is then recomputed on the permuted groups under the same window, horizon, rank rule, and normalization. The permutation leaves every coordinate's own trajectory untouched and destroys only which coordinates share a group, so the resulting distribution isolates the contribution of group membership and nothing else.

Pooling and seeds. Null values are pooled per (segment, group size, horizon, protocol). The number of draws per segment is $\max\big(5, \lceil 200/n_{\text{groups}} \rceil\big)$, so that at least about 200 null values stand behind every threshold. Each draw takes a seed composed from the draw index, anchor, layer, and segment, so any lane that recomputes the null reproduces it.

The two readouts. The fixed-threshold readout LRGF_θ counts groups with FRR $< \theta$, with $\theta = 0.7$ throughout. The calibrated readout LRGF_{cal} replaces θ

by the $\alpha = 0.05$ quantile of the matched null pool and is reported as excess over α. $\mathrm{LRGF}_{\mathrm{cal}}$ is a comparability convention: it places segments whose group size and null geometry differ on a single axis. It does not correct individual FRR values, and it does not make a level comparable to a level read under a different cut. Where the fixed threshold is used, we also report the null pool's own pass rate at θ, written null_θ, which is the share of the observed fraction that membership alone would produce.

B.2.5 Numerical Conventions and the Sign Rule

The sign rule and zero displacement. Signs are compared exactly on the stored floating-point values, with $\mathrm{sign}(0) = 0$ and no tolerance band. A coordinate that does not move in either interval therefore counts as an agreement, and a coordinate that moves in exactly one of the two counts as a disagreement. We keep the exact rule because it leaves ESA a parameter-free function of the realized trajectory, but it makes ESA sensitive to coordinates whose displacement is zero or numerically negligible over one or both intervals, in a way that VCS is not: VCS weights each coordinate by its displacement magnitude, so those coordinates barely enter it.

This is the mechanism behind the two departures from the parameter-free reference curve $\mathrm{ESA} = 1 - \arccos(\mathrm{VCS})/\pi$ in Figure 4.1(a). At the five reported anchors the two departing cells are AlexNet with AdaGrad and AlexNet with Adam, in which up to 79% of coordinates have zero reference displacement and up to 69% are zero over both intervals. Their median vertical departure from the reference curve is +0.276 and +0.221. Restricting both probes to the coordinates that move in both intervals reduces those departures to +0.008 and +0.007, while the median absolute departure over the remaining 28 cells is 0.003 and is unchanged by the restriction.

Degenerate units. VCS is undefined when either vector has norm below 10^{-30}; such units are recorded as missing and dropped from denominators rather than counted as zero. SESA is undefined when the selector is empty, and those units are dropped in the same way. Non-finite FRR values leave the LRGF denominator.

Other numerical choices. The probe layer works in double precision. FRR denominators carry an additive 10^{-12} guard. FRR is computed on the raw history, whereas the DMD fit standardizes coordinate rows. On the Pythia side a second FRR protocol normalizes each history column to unit norm within the group, with a 10^{-30} guard, and degenerate columns are counted rather than absorbed; both FRR protocols are reported side by side wherever the column-normalized variant is used.

Group-mean and full-vector VCS. Figure 4.1(a) plots one point for each cell and anchor, pairing its ESA with its *group-mean* VCS, the mean of the per-group cosines. The full-vector cosine of the same cell is recorded alongside it and differs because it weights coordinates by magnitude across the whole unit rather than within each group. The two are close on most cells (median difference $+0.006$, interquartile range -0.017 to $+0.023$) but diverge sharply on a few, up to ± 0.45. The group mean is the readout that belongs on the parameter-free reference curve: its median absolute departure from $1 - \arccos(\mathrm{VCS})/\pi$ is 0.005, against 0.011 for the full-vector cosine on the same cells.

B.2.6 Grouping Sensitivity and Within-Family Checks

What the Flat Protocol Reads

A dedicated audit compared the flat readout with three probes that involve no grouping at all: element-level ESA, tensor-level VCS, and tensor-level FRR. On all three, ViT-mid reads as more predictable than ResNet at every anchor of every matched pair, whereas the flat LRGF reference orders the same runs and anchors the other way (Figure B.2). Under a within-tensor membership permutation the flat ordering reverses to ViT-mid at or above ResNet in 24 of 24 pair–anchor comparisons.

The decomposition names the mechanism (Figure B.3). Permuting block membership destroys 66 to 87% of ResNet's flat LRGF, and the destroyed part localizes on the 3×3 kernels, with 1×1 shortcut convolutions contributing $\Delta \approx .003$.

Table B.5 (continued)

Cut	One group is	D	What it reads	Chance status at θ=0.7
magnitude	within (layer, segment), coordinates sorted by pre-anchor L_2 and chunked to the same (n, D)	as reference	Loses to head grouping on qkv at step 10k (LRGF_{cal} .333/.389 against .104/.069 at 70M and 160M); MLP sensitivity is local (70M layer 5 mid-training, about 3×) with 7 of 10 grid cells negative; at most 0.12% on absolute θ across five sizes.	Null shared with the positional cut (grouping-independent).
MLP input half	neuron input row (w_{in})	d_{model}	Reads near zero (mean 0.04%).	θ legal.
MLP output half	neuron output column (w_{out})	d_{model}	Real emergence carried by the deepest layer: the 70M layer-5 wave rises from 64k to a peak of 5.4% at 96k. Across five model sizes the output half is the only MLP cut with signal (peaks 1.8 to 15.7%), with the input half and the magnitude cut near zero.	θ legal.

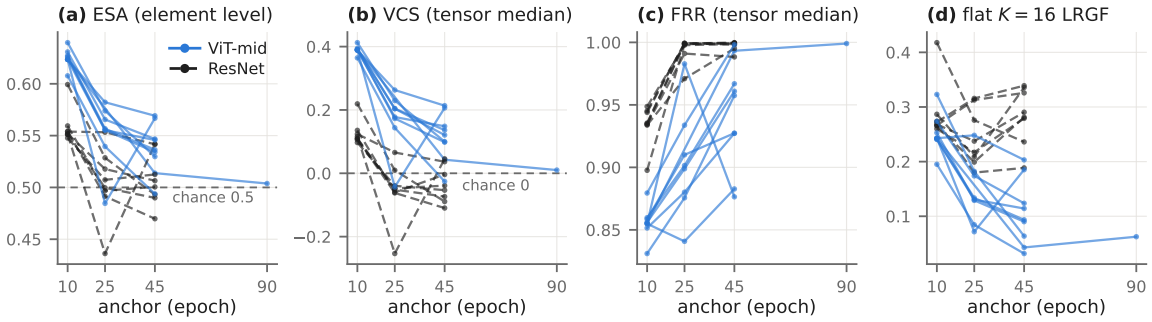


Figure B.2: Probes that use no grouping convention, against the flat readout (bulk scope; one line per run, 8 matched ViT-mid and ResNet pairs; the long-cosine ViT-mid run extends to epoch 90). On all three cut-free probes, (a) element-level ESA, (b) tensor-level VCS, and (c) tensor-level FRR, where lower means more predictable, ViT-mid reads as more predictable than ResNet at every anchor, while (d) the flat K=16 LRGF reference orders the same runs and anchors the other way (7 to 8 of 8 pairs). Canonical protocol (n_{past}=5, τ=5, θ=0.7); dashed grey lines mark chance (ESA 0.5, VCS 0).

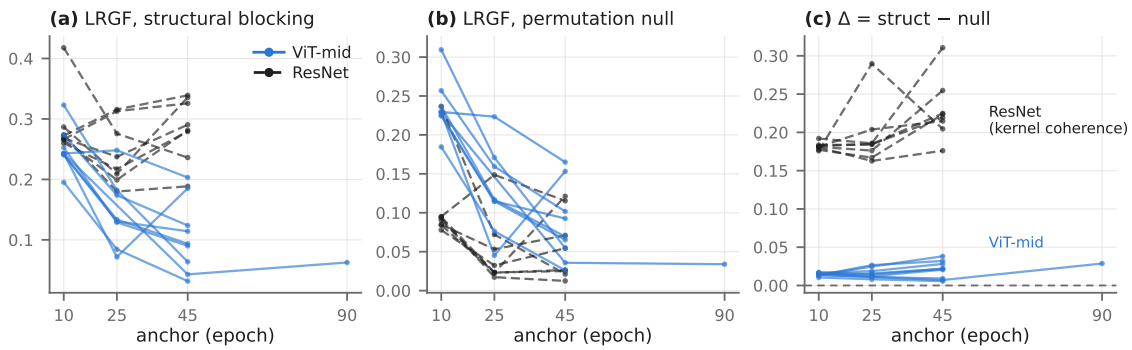


Figure B.3: Permutation-null decomposition of the flat K=16 LRGF (bulk scope; one line per run): (a) structural blocking, (b) the membership-permutation null (mean of 5 draws, shading gives min to max), and (c) their difference Δ, which is the intra-block coherence contribution. Permutation destroys 66 to 87% of ResNet's flat readout and the destroyed part localizes on the 3×3 kernels (1×1 shortcut convolutions: $\Delta \approx .003$), whereas 82 to 94% of ViT-mid's readout survives and the architecture ordering flips to ViT-mid at or above ResNet on the null column (24/24 pairs). A flat-LRGF level is therefore a statement about a grouping convention as well as about an architecture.

On ViT-mid, 82 to 94% of the reading survives the same permutation. A flat-LRGF level is therefore a statement about a grouping convention as much as about an architecture.

Two further properties of the flat protocol matter for reading the main paper. Between-group ESA dispersion sits at the binomial sampling floor (1.1×, median over 30 cells), so flat blocks are statistically homogeneous, and group-level coupling between ESA and FRR is largely erased (cell-wide median −.16). And bulk group-level FRR sits close to its chance floor: of the 31.2% of bulk groups passing θ=0.7, about 25 percentage points are chance passing (null$_\theta$ median .23 to .25). This is the concrete reason the time-axis readouts use LRGF$_{\text{cal}}$.

Structural and Magnitude Cuts on the Vision Panel

Recutting along physical or functional axes changes what the instrument resolves. Structural units raise between-group ESA dispersion off the floor (head qkv 4.4×, filter

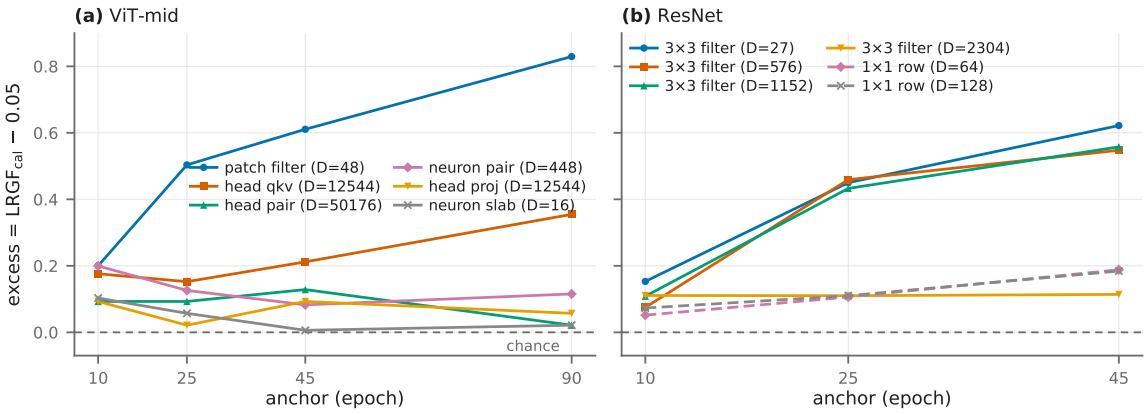


Figure B.4: Structural-axis cuts under the calibrated readout: excess = $\mathrm{LRGF}_{\mathrm{cal}} - \alpha$ (α=0.05, so zero is chance), median over runs. (a) ViT-mid: physically defined groups surface above-chance structure that the flat axis does not resolve, with whole patch filters reaching .61 and head-level *qkv* groups .21 at anchor 45, while slab and projection cuts sit near chance. (b) ResNet: whole 3×3 filters reach .55 to .62 and 1×1 rows .18. Each cut carries a size-matched membership-permutation null that preserves the group-size multiset, and groups use only pre-anchor information. Group size differs across cuts, so levels are comparable only through the calibrated excess.

3×3 1.9×, neuron pair 1.7×, row 1×1 1.2×, against 1.1× for flat) and restore group-level ESA–FRR coupling (cell-wide median −.42 against −.16). The restoration is architecture-resolved: ViT and ViT-mid couple in every recipe cell (−.28 to −.77), while ResNet's plain recipes stay flat (SGD −.14; momentum, Adam, and Muon near 0) and only the recipe cells couple (weight decay −.45, SAM −.35, AdaGrad −.30). Under the calibrated readout the same reordering is visible over training anchors (Figure B.4): whole patch filters and head-level *qkv* groups carry clear above-chance excess, while slab and projection cuts sit near chance.

Pocket density under structural cuts is architecture-stratified (Figure B.5): FRR < 0.7 reaches 19.8% of ViT-mid head-*qkv* groups and 12.7% of neuron pairs, against 1.5% for ResNet kernels and 0.1% for 1×1 rows. These cuts have large group sizes, where the membership-resample null passes θ=0.7 at a rate near zero, so essentially none of the passes is attributable to chance, unlike the flat bars in the

same figure. Structural excess decays over training, from a median of .070 at anchor 5 to .036 at anchor 40, which places pockets in an early-to-mid-training population.

The magnitude cut is the strongest single-axis sorter of pocket density and not a raiser of totals. Totals sit slightly below the structural cut (0.4/3.2/8.1% against 1.3/6.7/12.9% for ResNet, ViT, and ViT-mid), but the density gradient is monotone across all 16 bins and steep at the top: pooled FRR < 0.7 rises from 0% at bin 0 to 20.1% at bin 15, ESA > 0.7 from 0.1% to 27.2%, and VCS > 0.4 from 0.2% to 36.4%, three instrument families ordered the same way along one causal axis (Figure B.6). The top three bins concentrate density beyond the best structural cut (ViT-mid 27.9% against head *qkv* 19.8%). ResNet stays low under every probe of the magnitude cut (2.1/4.4/12.4% for FRR, ESA, and VCS on the top three bins), so the architecture stratification is itself cut-robust.

An axis-independent check bounds what any cut could reach (Figure B.7): the top-five eigenvalue mass of the coordinate-trajectory correlation spectrum exceeds a matched independent null by two to four times in every tensor family of all three architectures. Co-moving structure is present everywhere; which grouping axis can access it is what differs by architecture.

Pythia Cuts and the Granularity Ladder

The Pythia readings are registered as correspondences and are not used to certify one another. The flat audit finds a third permutation signature here, distinct from both vision architectures: on *qkv*, permuting block membership *raises* the flat reading ($\Delta = -.140$, structural .099 against permuted .239), which is what comoving mass concentrated in a few strong coordinates produces when random blocks inherit it and contiguous blocks isolate it.

Head grouping and magnitude grouping are adjudicated at anchor 10k. Head grouping retains far stronger calibrated signal on *qkv* (LRGF_{cal} .333/.389 against .104/.069 at 70M and 160M), so the head unit is an organizational axis of its own rather than a shadow of coordinate magnitude. On the MLP, magnitude sensitivity

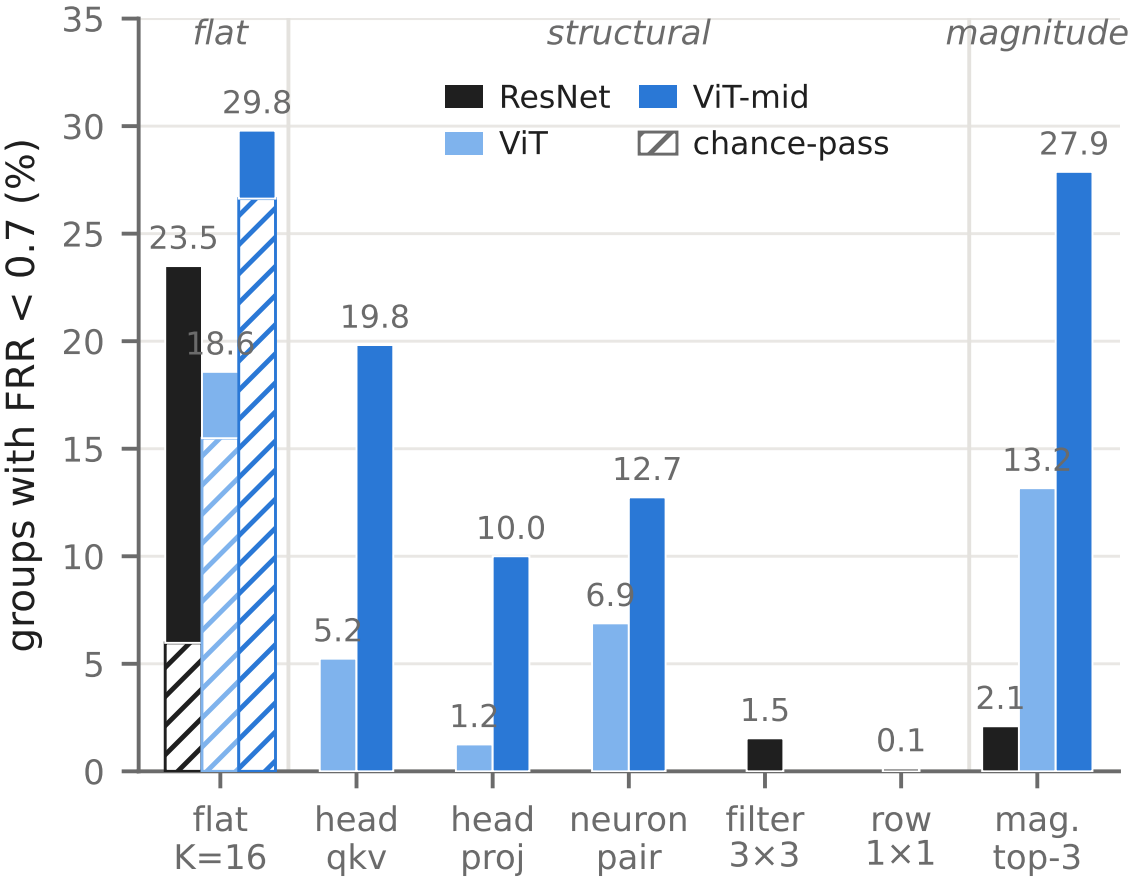


Figure B.5: Pocket density by grouping cut and architecture on the vision panel: share of groups with FRR < 0.7, pooled over the 18 recipe cells and anchors 5 to 40 of the group-level lanes (bulk tensors, out-of-window target). The structural cuts and the magnitude cut (top three of 16 causal bins) are chance-immune at θ=0.7, so the chance contribution to their bars is negligible. The flat K=16 bars are pooled over the same 18 cells weighted by group count and are stacked: the hatched portion is the measured chance-pass fraction under the membership-resample null (6.0, 15.5, and 26.6 pp of the 23.5/18.6/29.8% totals for ResNet, ViT, and ViT-mid). Most of the ViT-family flat readout is chance passing, and most of the ResNet flat readout is intra-kernel coherence. Group size differs across cuts, so levels are comparable within a bar group only in the chance-immune sense.

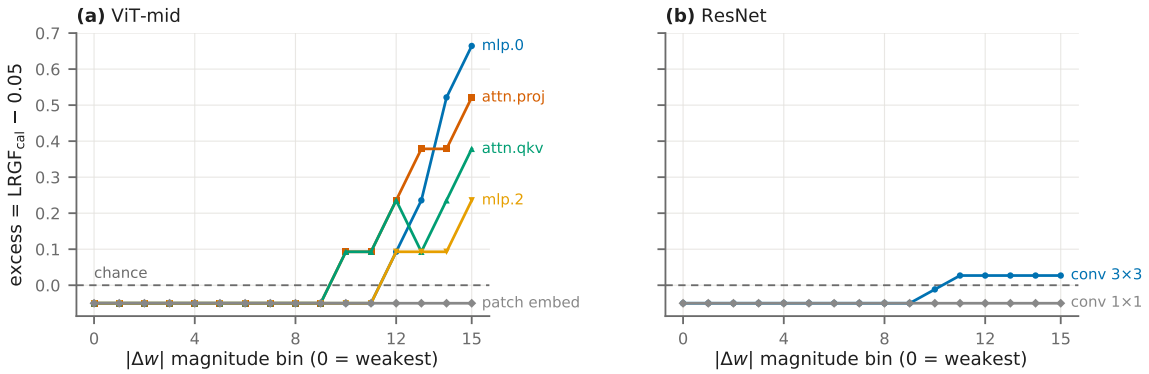


Figure B.6: Magnitude-stratified cut at anchor 45: groups are 16 equal-width bins of the per-tensor pre-anchor $\|\Delta w\|_2$ rank (bin 0 weakest, bin 15 strongest), reported as excess over chance, median over runs; the floor at -0.05 means no group passes. (a) ViT-mid feature-dimension tensors show a monotone concentration signature (top bins .38 to .66), while the patch embedding stays at the floor because its structure lies on the spatial axis (Figure B.4). (b) ResNet reads near zero under the same cut. This is the strongest single-axis sorter on the ViT side, and its architecture contrast runs opposite to the flat readout.

is local (70M layer 5 in mid-training, about 3×) and the grid is negative in 7 of 10 cells, so the MLP's modest calibrated readout is modest under both conventions.

Granularity is a separate axis. At head granularity (D=32,768) the absolute θ columns read near zero while calibrated excess is large, for instance +.263 for `attn_out` at anchor 70k, which is the sharpest case of large-D blindness in the fixed threshold. Stepping down to one W_{qkv} row (D=512) revives an absolute pocket population, with FRR < 0.7 stable at 4 to 5% of rows and the D=512 and D=128 distributions nearly coincident. The row convention is invariant across D=128 to 3072 (410M peak fractions .432/.434/.438 at D=1024/512/256; a 160M scan over the same range; 1B below 0.5% throughout after masking), so no per-size adjustment is needed. Recutting the MLP resolves an asymmetry between its halves: the input half and the magnitude cut read near zero on absolute θ (means 0.04% and 0.01%), while the output half carries a late wave in the deepest layer (70M layer 5: rising from 64k to a peak of 5.4% at 96k, fading by 128k). Across five model sizes the output half is the only MLP cut with signal (peaks 1.8 to 15.7%).

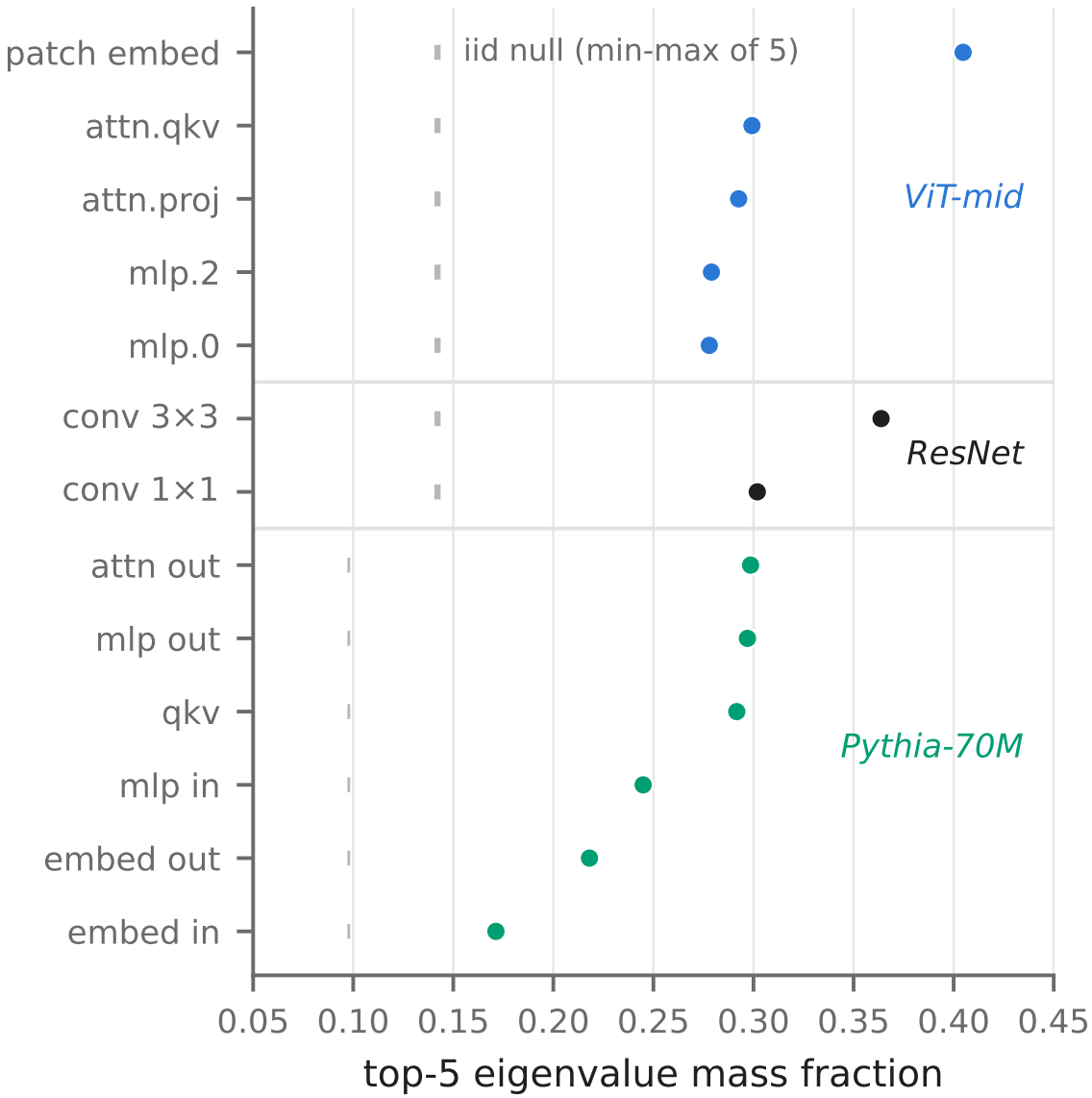


Figure B.7: Spectrum diagnostic at the late anchors (anchor 45 on the vision panel, step 70k on Pythia): top-five eigenvalue mass of the coordinate-trajectory correlation spectrum per tensor family, against a simulated independent null of matching shape (grey band gives min to max over 5 draws), median over runs. Every family of all three architectures exceeds the null by two to four times, so co-moving structure is present in every family; which grouping axis can access it is what differs by architecture.

Within-Family Agreement, per Cell

The two within-family summaries quoted in Section 4.3.2 are medians over per-cell Spearman correlations, each computed across the tensors of one model–configuration cell. Their dispersion is as follows. Out-of-window and in-window FRR correlate at a median $\rho_S = 0.83$, with per-cell values from 0.53 to 0.97; the low end is occupied by AdaGrad cells (ViT 0.62, ViT-mid 0.63) together with one further cell at 0.53. SESA under the DMD and linear-trend selectors correlates at a median $\rho_S = 0.89$, with per-cell values from 0.44 to 1.00; the one low value is AlexNet with AdaGrad (0.44), which is also one of the two cells identified in Appendix B.2.5. The two displacement-direction readouts correlate at a median $\rho_S = 0.93$. Correlations between the DMD and linear-trend *lift* readouts are much weaker (median 0.15); lift is a within-predictor quantity, and the two selectors define different denominators, so the comparison is reported only as dispersion and not as an agreement check.

B.2.7 Reporting Discipline

Five rules follow from the measurements above, and every group-level statement in Chapter 4 and this appendix follows them.

1. **Claims travel with the grouping convention.** Any group-level ordering or level statement names its cut, and an ordering that reverses under recutting is reported as conditional on the convention rather than as a property of the architectures.

2. **Levels are not comparable across cuts.** Paired flat and structural FRR readouts rank consistently ($\rho_S = .79$) but carry a systematic level offset (median $\Delta = +.082$, structural above flat), so cross-cut comparisons use ranks or calibrated excess and never raw levels.

3. **Chance immunity is cut-dependent.** Large-group cuts make θ=0.7 essentially chance-free, while the flat protocol passes about 25 percentage points

of bulk groups by chance. Raw θ fractions are quoted only for cuts whose null pass rate is near zero, and calibrated excess is used otherwise.

4. **Time-axis readouts use** $\mathrm{LRGF}_{\mathrm{cal}}$**.** Where a bare θ fraction is reported across layers, it is accompanied by its null_θ reference.

5. **Selectivity lift is not compared across predictors.** Lift is a within-predictor quantity, since SESA is evaluated on the selector that predictor produces.

B.2.8 Full Multi-Probe Agreement and Protocol Robustness

The following vision-panel diagnostics are reported under their stated flat and tensor-level conventions. They provide robustness evidence within those conventions and are not used to override the cut-dependence results in Appendix B.2.6. In the tables below, the legacy implementation scope *Small* is renamed *Auxiliary*; it denotes the same operational auxiliary class audited in Appendix B.3.2.

Anchor-conditioned Spearman agreement. For two probes p and q, let $R_\omega^{(p)}$ be the rank of probe p on matched evaluation ω, and let $a(\omega)$ denote the anchor associated with that evaluation. We subtract anchor-specific mean ranks,

$$\widetilde{R}_\omega^{(p)} = R_\omega^{(p)} - \overline{R}_{a(\omega)}^{(p)}, \tag{B.7}$$

and report the correlation of the residualized ranks:

$$\rho_{p,q}^{\mathrm{anchor}} = \frac{\sum_\omega \widetilde{R}_\omega^{(p)} \widetilde{R}_\omega^{(q)}}{\sqrt{\sum_\omega \left(\widetilde{R}_\omega^{(p)}\right)^2} \sqrt{\sum_\omega \left(\widetilde{R}_\omega^{(q)}\right)^2}}. \tag{B.8}$$

This removes anchor-level shifts before comparing the relative ordering induced by different probes.

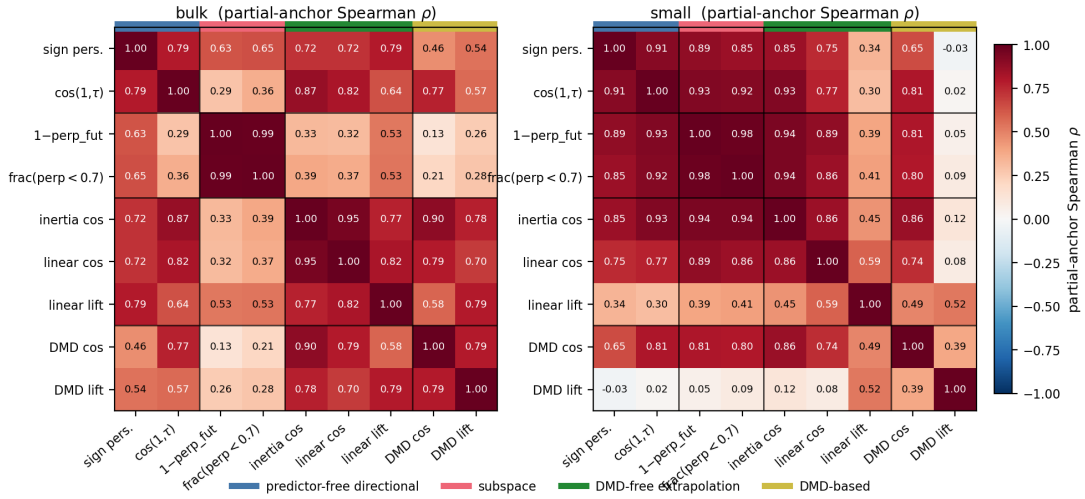


Figure B.8: Full 9×9 pairwise partial-anchor Spearman heatmaps for the probe panel. The heatmap records the full multi-probe structure behind the representative agreement pairs in Chapter 4. It shows both strong agreement among directional, extrapolation, and DMD probes and nontrivial decoupling of subspace and selector-lift probes.

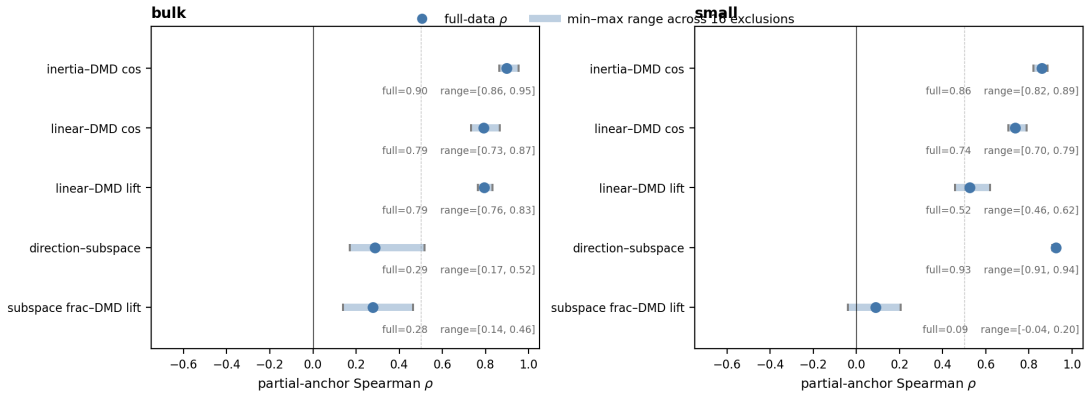


Figure B.9: Leave-one-out and exclusion sensitivity for representative probe pairs. Full-data correlations are shown as points and exclusion ranges as intervals. The main DMD-free and DMD directional correlations remain stable, while subspace-related pairs are more sensitive and should be interpreted as diagnostic rather than primary evidence.

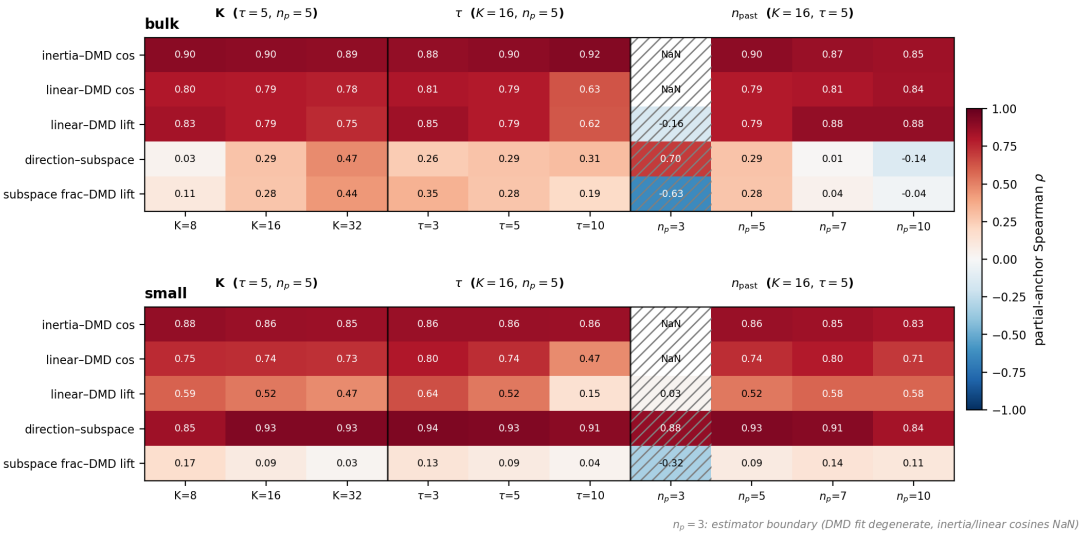


Figure B.10: Protocol sensitivity of representative probe-pair agreement over group size K, horizon τ, and DMD history length n_{past}. The $n_{\text{past}} = 3$ column is an estimator-boundary case and is not used as successful robustness evidence. DMD-free versus DMD directional and selectivity agreement is stable for valid short-horizon settings, while bulk subspace-related pairs are more protocol-sensitive.

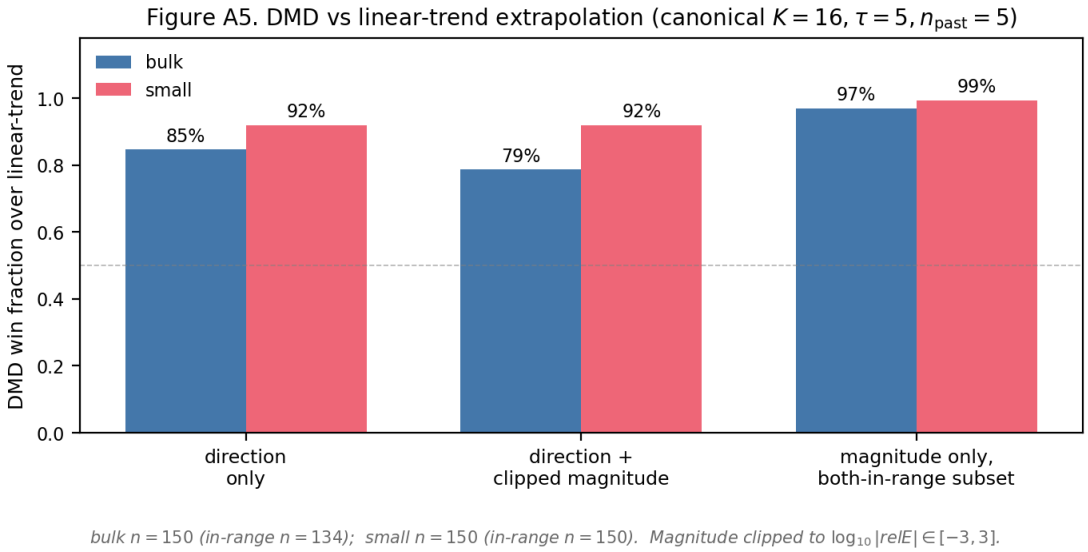


Figure B.11: DMD value-add over linear-trend extrapolation under the canonical protocol. Direction-only and direction-plus-clipped-magnitude win rates support DMD as a useful predictor-based probe beyond a simple baseline. The magnitude-only both-in-range column is conditional and should not be read as a full-panel win rate.

Table B.6: Partial-anchor Spearman correlations across DMD history length n_{past}. The $n_{\text{past}} = 3$ column is an estimator-boundary case, not successful robustness evidence. Valid robustness claims use $n_{\text{past}} \in \{5, 7, 10\}$.

Scope	Probe pair	$n_{\text{past}} = 3$	$n_{\text{past}} = 5$	$n_{\text{past}} = 7$	$n_{\text{past}} = 10$
Bulk	inertia–DMD cos	–	0.897	0.867	0.845
Bulk	linear–DMD cos	–	0.790	0.808	0.839
Bulk	linear–DMD lift	-0.158	0.793	0.882	0.879
Bulk	direction–subspace	0.697	0.287	0.009	-0.135
Bulk	subspace frac–DMD lift	-0.634	0.278	0.040	-0.037
Auxiliary	direction–subspace	0.877	0.926	0.910	0.836
Auxiliary	inertia–DMD cos	–	0.860	0.851	0.827
Auxiliary	linear–DMD cos	–	0.736	0.797	0.706
Auxiliary	linear–DMD lift	0.029	0.524	0.579	0.580
Auxiliary	subspace frac–DMD lift	-0.324	0.090	0.140	0.114

Table B.7: Range of partial-anchor Spearman correlations across 17 exclusion patterns. Strong DMD-free and DMD agreement pairs remain stable; subspace-related pairs remain sensitive or weak.

Scope	Probe pair	Min ρ	Max ρ	Range	Full ρ
Bulk	inertia–DMD cos	0.864	0.953	0.089	0.897
Bulk	linear–DMD cos	0.733	0.866	0.134	0.790
Bulk	linear–DMD lift	0.764	0.833	0.069	0.793
Bulk	direction–subspace	0.172	0.517	0.345	0.287
Bulk	subspace frac–DMD lift	0.138	0.465	0.327	0.278
Auxiliary	inertia–DMD cos	0.820	0.887	0.067	0.860
Auxiliary	linear–DMD cos	0.705	0.790	0.085	0.736
Auxiliary	linear–DMD lift	0.457	0.620	0.163	0.524
Auxiliary	direction–subspace	0.908	0.939	0.031	0.926
Auxiliary	subspace frac–DMD lift	-0.039	0.204	0.243	0.090

Table B.8: Robust magnitude comparison between DMD and linear trend. Relative errors are summarized in log10 space to avoid domination by raw blow-up values.

Scope	DMD median	DMD IQR	Linear median	Linear IQR
Bulk	0.16	[0.09, 0.41]	0.77	[0.64, 0.92]
Auxiliary	-0.01	[-0.13, 0.06]	0.57	[0.32, 0.76]

Table B.9: DMD win counts under robust metric definitions. The both-in-range comparison is conditional on both predictors having clipped log10 relative error in range.

Scope	Comparison	n	DMD wins	Fraction
Bulk	direction only	150	127	0.847
Bulk	magnitude clipped log10	150	130	0.867
Bulk	magnitude clipped both-in-range	134	130	0.970
Bulk	direction + magnitude clipped	150	118	0.787
Auxiliary	direction only	150	138	0.920
Auxiliary	magnitude clipped log10	150	148	0.987
Auxiliary	magnitude clipped both-in-range	150	148	0.987
Auxiliary	direction + magnitude clipped	150	138	0.920

B.2.9 Representative Multi-Seed Details

The following tables summarize a representative three-seed subset. These checks support the reproducibility of the tested patterns, but they are not a complete multi-seed sweep over every model–optimizer pair.

Table B.10: Representative multi-seed stability. Values are mean Spearman correlations over seed pairs on the four-configuration by five-anchor subset. Code names follow the implementation used for the measurement panel: `inv_perp_fut` is $1 - \mathrm{FRR}$; `frac_pred_lt07` is LRGF with threshold $\theta = 0.7$; `cos_inertia`, `cos_linear`, and `cos_DMD` are cosine agreements between inertia, linear-trend, and DMD predictions and the realized τ-step displacement; and `lift_linear` and `lift_DMD` are the corresponding selectivity lifts. `pure_sgd` and `cos_1_tau` are the implementation names for the directional-persistence probes reported in Chapter 4.

Probe	Bulk mean ρ	Auxiliary mean ρ
`pure_sgd`	0.980	0.984
`cos_1_tau`	0.971	0.986
`inv_perp_fut`	0.982	0.980
`frac_pred_lt07`	0.982	0.976
`cos_inertia`	0.991	0.979
`cos_linear`	0.988	0.969
`lift_linear`	0.994	0.896
`cos_DMD`	0.990	0.985
`lift_DMD`	0.989	0.943

Table B.11: Final test accuracy across seeds for the representative multi-seed subset.

Cell	Seed 42	Seed 43	Seed 44	Spread
ViT/SAM	0.6979	0.7257	0.7181	0.0278
ViT/Adam	0.7928	0.7957	0.8028	0.0100
ResNet/SAM	0.9215	0.9239	0.9213	0.0026
ResNet/Muon	0.9367	0.9380	0.9364	0.0016

Table B.12: Anchor-averaged bulk probe values across seeds for the representative subset. Values are seed 42 / seed 43 / seed 44.

Probe	ViT/SAM	ViT/Adam	ResNet/SAM	ResNet/Muon
`pure_sgd`	.753 / .735 / .735	.684 / .682 / .683	.708 / .707 / .706	.673 / .673 / .672
`cos_1_tau`	.696 / .654 / .653	.531 / .526 / .528	.580 / .577 / .577	.500 / .499 / .496
`inv_perp_fut`	.350 / .301 / .297	.171 / .170 / .170	.275 / .275 / .276	.162 / .161 / .161
`frac_pred_lt07`	.574 / .462 / .449	.155 / .151 / .152	.395 / .393 / .396	.134 / .132 / .134
`cos_inertia`	.439 / .355 / .347	.049 / .043 / .047	.191 / .191 / .198	.006 / .006 / .006
`cos_linear`	.263 / .207 / .200	.007 / .006 / .003	.082 / .083 / .086	-.004 / -.005 / -.005
`lift_linear`	.049 / .045 / .042	.007 / .007 / .006	.023 / .023 / .023	-.001 / -.002 / -.001
`cos_DMD`	.483 / .397 / .391	.115 / .111 / .111	.199 / .200 / .201	.019 / .019 / .018
`lift_DMD`	.083 / .074 / .072	.030 / .028 / .029	.047 / .046 / .047	.005 / .005 / .005

B.3 Auxiliary–Bulk Attribution and Cross-Regime Mapping

The auxiliary–bulk contrast is reported in the main text, so this section does not repeat the main result. Instead, it documents how the partition is assigned, audits the operational rule against the morphological definition, and records the Pythia segment and embedding conventions needed to interpret cross-regime results. Section 4.3.3 splits trainable tensors into two morphological classes and reads the auxiliary–bulk separation within matched trajectories. This appendix gives the membership rule, the tensor-by-tensor attribution it produces in each training regime, and the audits that bound how much the choice of rule could matter.

B.3.1 The Partition Rule

The primary definition is morphological. A parameter tensor is *auxiliary* if it is vector-like, which covers normalization scales and shifts, biases, and token and positional embedding vectors, and *bulk* if it is a matrix-like weight, which covers linear and attention projections, MLP weights, and convolutional kernels. The definition refers to the shape and function of the tensor, not to its size.

The vision pipeline realizes this rule through a size threshold on coarse role classes: a tensor is auxiliary if the total parameter count of its role class within that model falls below 10^5. The threshold applies to role-class totals rather than to individual tensors, so a small individual matrix inside a large class, such as the 1,280-parameter ViT classifier head, stays bulk. Every quantitative auxiliary–bulk readout in Chapter 4 uses this operational assignment, and Appendix B.3.2 audits it against the morphological definition. On Pythia the same rule is expressed through a segment taxonomy rather than a threshold, because the segment structure already separates matrices from vectors exactly (Appendix B.3.3). The Pythia partition additionally coincides with optimizer treatment: normalization parameters and all

(a) Per-model attribution

Model	Tensors		Parameters		aux share	aux composition (tensors)
	aux	bulk	aux	bulk		
MLP	3	4	1,792	3,803,648	0.05%	bias 3
ViT	37	18	14,464	531,712	2.65%	ln 18, bias 17, tok 2
ViT-mid	61	30	32,704	2,822,848	1.15%	ln 30, bias 29, tok 2
ResNet	31	15	7,040	2,770,624	0.25%	bn 26, bias 4, linear 1
AlexNet	7	8	9,344	57,035,456	0.02%	bias 7
All	139	75	65,344	66,964,288	0.10%	—

(b) Dual-definition audit

Audit level	total	agree	defect	agreement
model × tensor	214	213	1	99.53%
cell × tensor	1,203	1,196	7	99.42%
panel rows (5 anchors)	6,015	5,980	35	99.42%

Table B.13: Auxiliary–bulk attribution and dual-definition audit on the vision panel. **(a)** Per-model tensor attribution under the operational split, with the composition of the auxiliary class by coarse role (ln/bn/bias/tok; the single defector appears as *linear*). **(b)** Agreement between the morphological and operational definitions at three audit levels; the complete defector list is the single tensor discussed in Appendix B.3.2. Definitions are given in Appendix B.3.1. Data: the tensor-level table of the canonical group-level panel, with attribution fields constant across anchors and optimizer cells. Parameter counts cover the parameter vector only, with no batch-normalization running statistics.

biases are exempt from weight decay under the released recipe, so auxiliary–bulk contrasts at that scale register a parameter-role difference under the recipe rather than an isolated effect of tensor shape.

B.3.2 Vision Panel: Attribution and Dual-Definition Audit

Table B.13 lists the attribution per model, the composition of the auxiliary class by coarse role, and the audit of the two definitions against each other.

The two definitions agree on 213 of 214 unique model–tensor pairs (99.5%), and on 1,196 of 1,203 cell–tensor instances (99.4%). The defector list is a single tensor: the ResNet final classifier head `linear.weight` (2,560 parameters), which is matrix-like

under the morphological definition but auxiliary under the operational one, because its `linear` role class totals only those same 2,560 parameters. It recurs in all seven ResNet optimizer cells, which is 35 of 6,015 panel rows. Because the reported readouts use the operational assignment, this tensor is counted as auxiliary throughout.

Two properties of the audit are worth stating explicitly. First, the agreement is a property of the role-class form of the threshold: applying the same 10^5 threshold per tensor instead of per role class would agree with the morphological definition on only 175 of 214 tensors (81.8%). Second, the four ResNet tensors filed under the `bias` bucket are the shortcut batch-normalization affine parameters, which the coarse name rule places there because their names carry no `bn` marker. They are vector-like under either reading, so the audit is unaffected.

The panel covers the parameter vector only, with no batch-normalization running statistics, and excludes tensors holding fewer than 16 scalars, because the K=16 grouping cannot be formed there. On the vision side this excludes exactly one tensor per model, the ten-parameter classifier bias, for 50 parameters in total.

B.3.3 Pythia: Segment Taxonomy and the Vector Segment

Table B.14 lists the segment scheme with shapes, tensor counts, and parameter shares at 70M, and Figure B.12 places every parameter tensor in the architecture.

Each transformer layer contributes three matrix segments: qkv, the attention-output projection, and the two MLP weight matrices. Together they constitute bulk. One vector segment collects the two per-layer LayerNorms with their weights and biases, all four linear biases, and the final LayerNorm. The auxiliary series of the aggregation is an alias of the vector segment, with a per-anchor difference identically 0.0, so the vector segment is the auxiliary class: all vector-like parameters including the final LayerNorm. At 70M this is 50 tensors and 40,960 parameters, against 24 tensors and 18,874,368 parameters for bulk.

Segment	Class	Tensor patterns	Shape (70M)	n	Params (70M)	Share
qkv	bulk	`attention.query_key_value.weight`	1536×512	6	4,718,592	6.70%
`attn_out`	bulk	`attention.dense.weight`	512×512	6	1,572,864	2.23%
mlp	bulk	`mlp.dense_h_to_4h.weight`	2048×512	12	12,582,912	17.87%
		`mlp.dense_4h_to_h.weight`	512×2048			
vec	aux	`input_layernorm.{weight,bias}`	512	50	40,960	0.058%
		`post_attention_layernorm.{weight,bias}`	512			
		`attention.query_key_value.bias`	1536			
		`attention.dense.bias` ≡	512			
		`mlp.dense_h_to_4h.bias`	2048			
		`mlp.dense_4h_to_h.bias` ≡	512			
		`final_layer_norm.{weight,bias}` *	512			
`embed_in`	excluded	`gpt_neox.embed_in.weight` *	50304×512	1	25,755,648	36.57%
`embed_out`	excluded	`embed_out.weight` *	50304×512	1	25,755,648	36.57%
bulk	= qkv + `attn_out` + mlp			24	18,874,368	26.80%
aux	= vec (all vector-like parameters incl. final LN)			50	40,960	0.06%
excluded	= `embed_in` + `embed_out` (untied)			2	51,511,296	73.14%
total				76	70,426,624	100%

Per-layer patterns carry the prefix `gpt_neox.layers.`i`.` and repeat over the $L = 6$ layers; * marks whole-model tensors, with `final_layer_norm` counted inside vec and the two embedding matrices forming their own segments. Shapes are the 70M reference and scale with model size (160M: d=768, L=12, d_{mlp}=3072, so the qkv weights become 2304×768); shares are of the 70M parameter total 70,426,624. ≡ `attention.dense.bias` and `mlp.dense_4h_to_h.bias` coincide under the parallel residual, so both rows are kept in the panel with a duplicate flag and all aggregates count the pair once. The auxiliary series of the aggregation is an alias of the vec segment, with a per-anchor difference identically 0.0. The vec segment is exempt from weight decay in the released pretraining recipe, while the matrix segments and both embeddings take wd = 0.1.

Table B.14: Pythia segment scheme and its mapping to the auxiliary–bulk partition, with shapes and parameter counts at 70M.

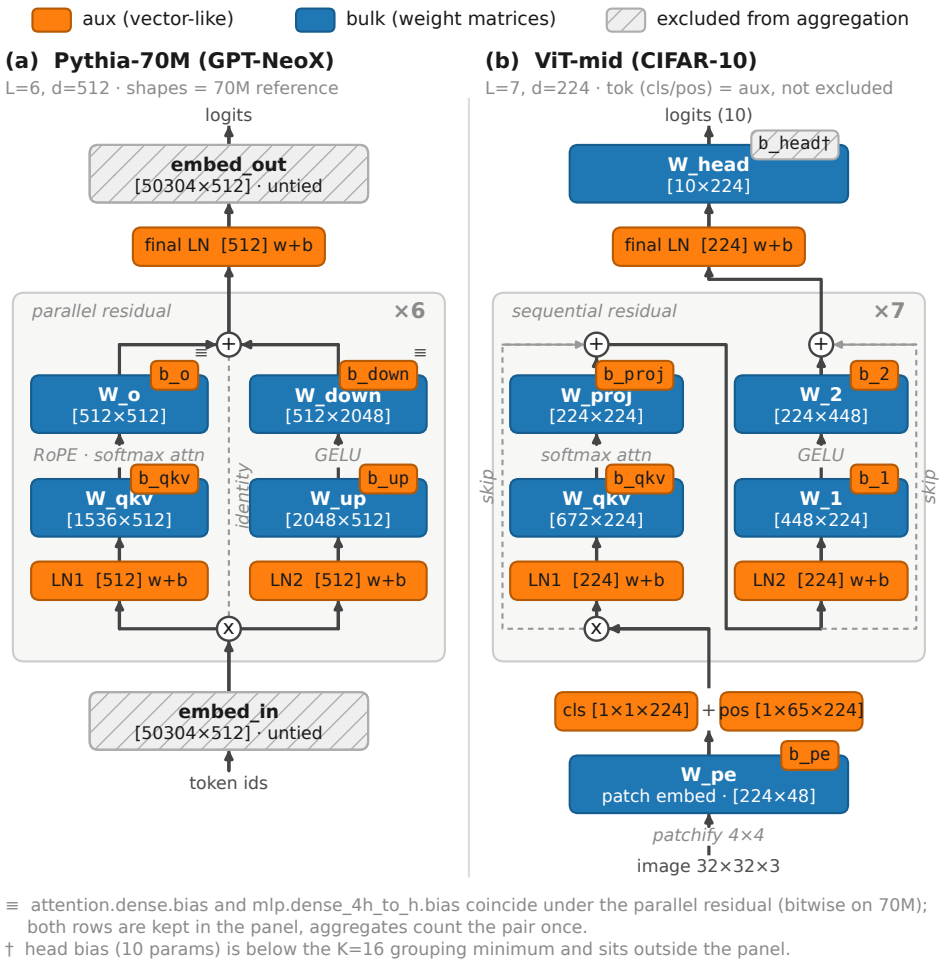


Figure B.12: Where the auxiliary–bulk partition sits in the architecture. One box per parameter tensor (LayerNorm boxes carry weight and bias jointly; linear biases are drawn as chips on their matrix box), with fill colour giving the partition class: orange for auxiliary, which is the vector segment and covers all vector-like parameters including the final LayerNorm; blue for bulk, the weight matrices (*qkv*, `attn_out`, MLP); hatched grey for tensors excluded from the aggregation. **(a)** Pythia-70M: one parallel-residual block, repeated six times, plus the top level; the two untied embedding matrices are their own segments and are excluded. The marked bias pair (≡) coincides under the parallel residual, so both rows are kept in the panel while aggregates count the pair once. **(b)** ViT-mid: sequential residuals, repeated seven times; here the token and positional embedding vectors are vector-like and therefore auxiliary, which is the one cross-regime difference from (a), and the ten-parameter head bias (†) is below the K=16 grouping minimum and is the only tensor excluded on the vision side. Shapes are the 70M reference.

Embeddings. The two untied embedding matrices, `embed_in` and `embed_out`, form their own segments and stay outside the auxiliary–bulk aggregation. They hold 73.1% of the parameter count, so the aggregation covers the remaining 26.9%. One labeling imprecision is registered: in the existing level–time aggregation the embedding rows enter the bulk side with 0.016% row weight, which has no numerical effect on any reported readout.

The duplicate bias pair. Under the parallel residual of GPT-NeoX, the attention-output bias and the MLP down-projection bias receive identical per-step gradients, and their trajectories coincide: bitwise on 70M over all released checkpoints, and within one half-precision unit in the last place on 160M. Both rows are kept in the panel with a duplicate flag, and every aggregate counts the pair once.

Weight decay. The vector segment is entirely exempt from weight decay in the released pretraining configuration, while the matrix segments and both embeddings take wd $= 0.1$. This is a property of the public training recipe rather than a choice of ours, and it is one of the conditions under which the auxiliary–bulk contrast in Figure 4.3 is read.

B.3.4 Cross-Regime Comparability

The membership rule, vector-like to auxiliary, is the same in both regimes. The one difference is which embedding-adjacent tensors exist and where they land. On the vision side the token and positional embedding *vectors* are vector-like and therefore auxiliary. On Pythia the embedding *matrices* are not vector-like, and they are excluded as separate segments rather than assigned to bulk.

Two further protocol differences keep absolute levels from being compared across regimes, and Figure 4.3 accordingly compares the within-line ordering rather than levels. The vision panel uses flat K=16 groups on an epoch clock, whereas the Pythia

panel uses architecture-aligned segment groups on a released-checkpoint clock, and the two clocks index different amounts of optimization per probe step.

B.4 Spatial and Temporal Localization of Predictable Pockets

This section combines cross-probe vision diagnostics with the detailed pocket census, persistence, overlap, and Pythia scale analyses. The results remain retrospective descriptions under their stated measurement units and grouping conventions; they are not treated as mutually independent evidence or as causal mechanism tests.

B.4.1 Additional Cross-Probe and Vision-Panel Heterogeneity Diagnostics

Some embedded figures retain legacy implementation labels. In those figures, "specialized/auxiliary" denotes the operational auxiliary class used in this dissertation, and `pure_sgd` denotes a predictor-free directional readout rather than a separate optimizer experiment. The surrounding text and captions use the current auxiliary–bulk terminology. Any group-level value or ordering remains conditional on the grouping convention documented in Appendix B.2.7.

The figures support the spatial and temporal heterogeneity analysis in Chapter 4: predictability forms a localized, stage-dependent map over tensor strata, roles, groups, and anchors. They are not intended to establish a universal auxiliary-over-bulk ordering or to explain architecture and optimizer mechanisms causally.

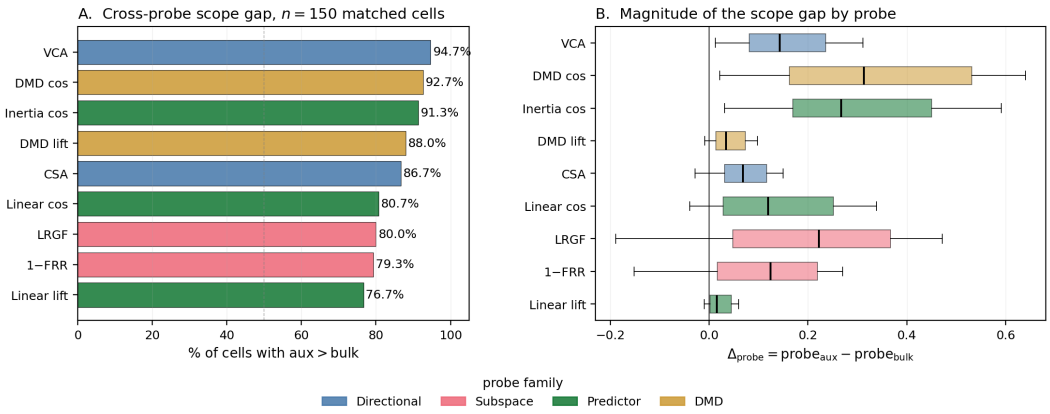


Figure B.13: Cross-probe support for the auxiliary–bulk contrast. Panel A reports the fraction of matched evaluations where the auxiliary score exceeds the corresponding bulk score. Panel B reports the distribution of $\Delta = \text{probe}_{\text{aux}} - \text{probe}_{\text{bulk}}$. Scores are oriented so that larger values indicate stronger predictability. Embedded panel labels using "specialized/auxiliary" refer to the same auxiliary class.

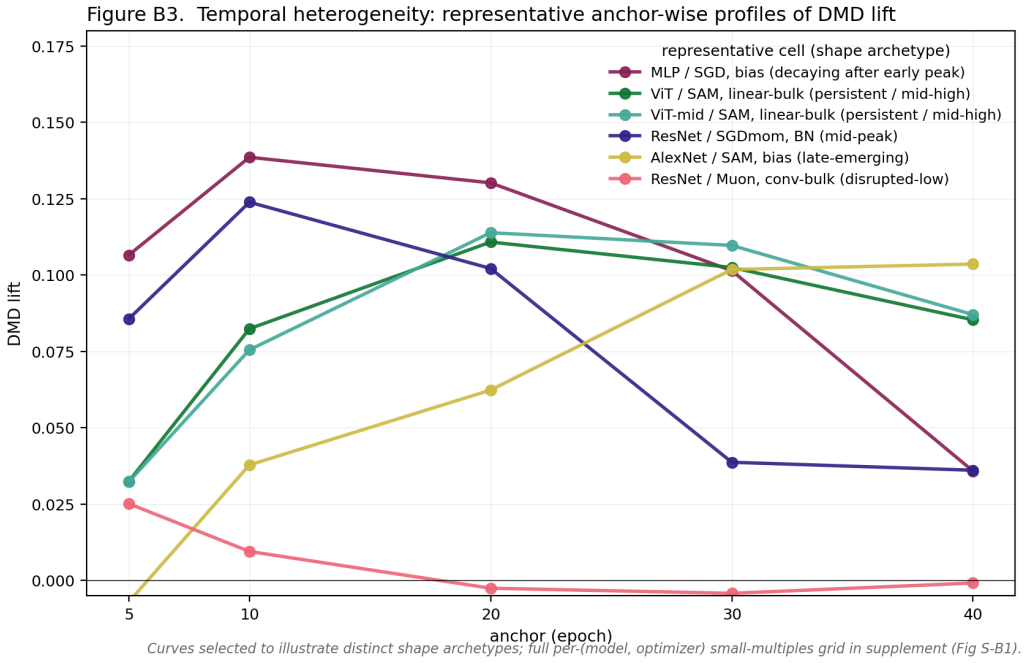


Figure B.14: Representative anchor-wise DMD-lift profiles. The curves illustrate recurring temporal shapes, including early peak and decay, persistent or mid-high profiles, late emergence, and disrupted-low behavior. They are illustrative examples rather than a formal taxonomy; the full small-multiple grids are shown in Fig. B.15.

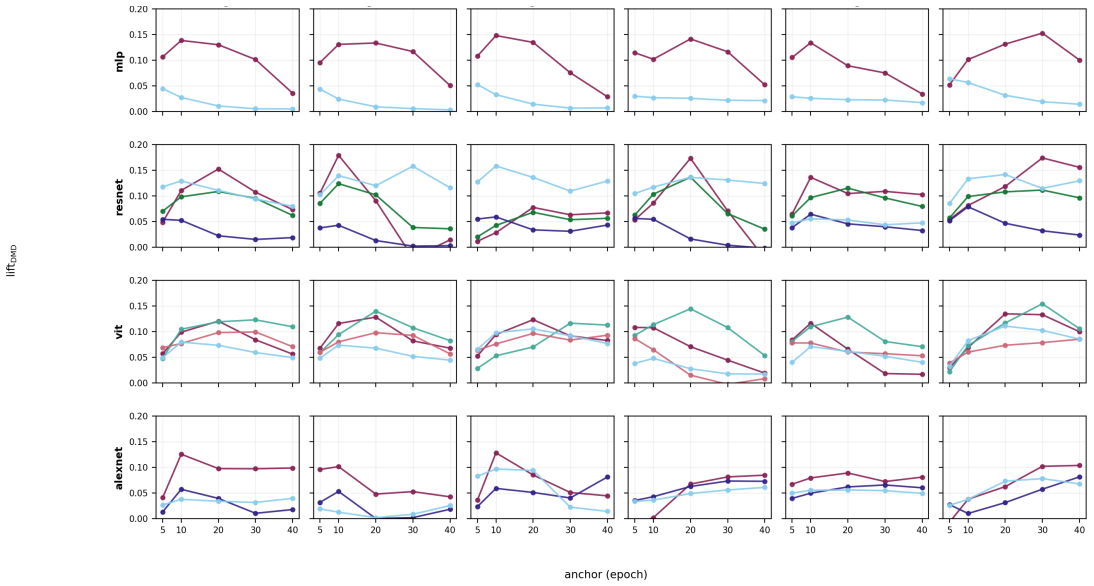


Figure B.15: Full temporal grids for DMD-selector lift. These small multiples extend Fig. B.14 by showing anchor-wise profiles across model, optimizer, and role combinations.

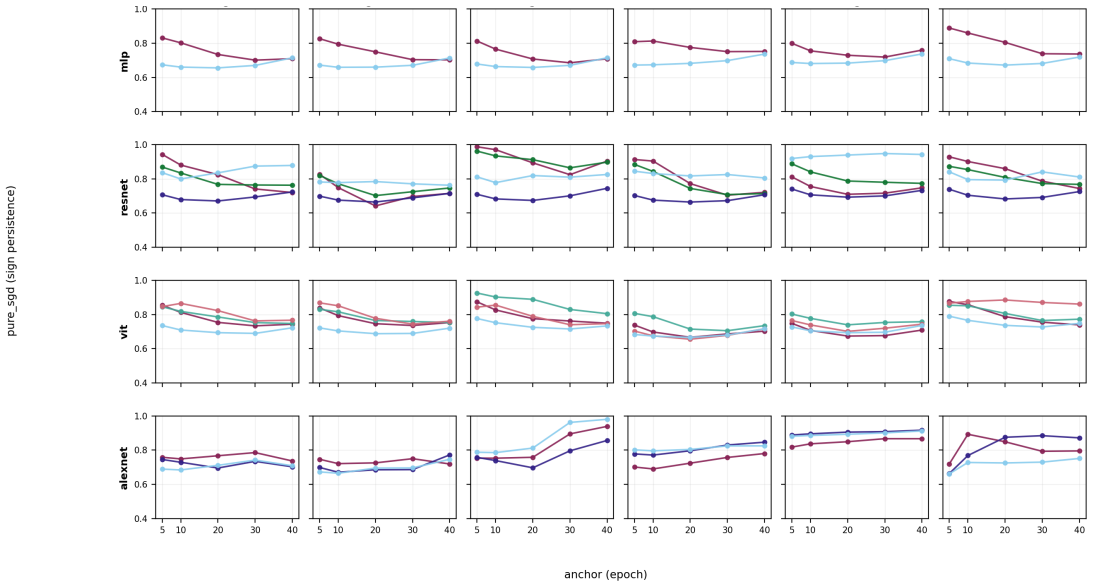


Figure B.16: Full temporal grids for `pure_sgd`, a predictor-free directional diagnostic. These curves show that temporal heterogeneity is also visible without fitting or applying DMD, although exact role and scope orderings can differ across probes.

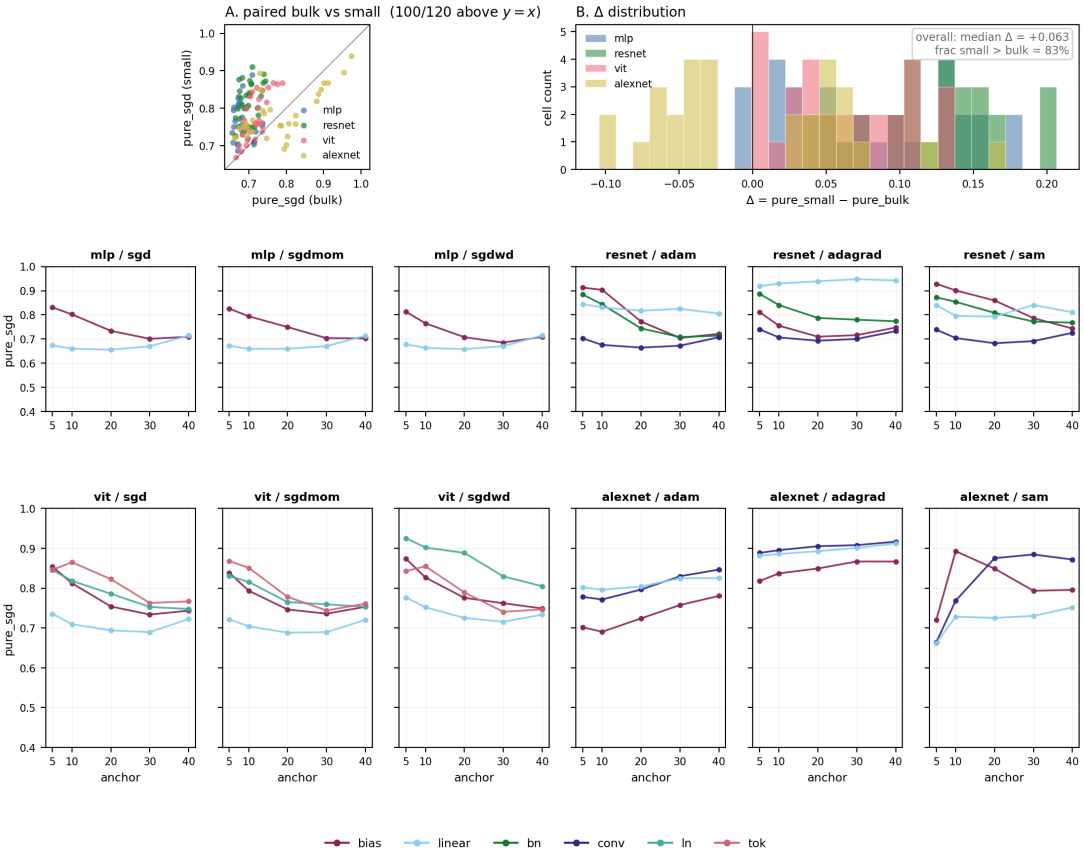


Figure B.17: Predictor-free directional support for spatial and temporal heterogeneity. The top panels repeat the scope-gap analysis using `pure_sgd`; the lower panels show representative temporal profiles by role. This is corroborating evidence that spatial and temporal heterogeneity are not visible only through DMD-selector lift.

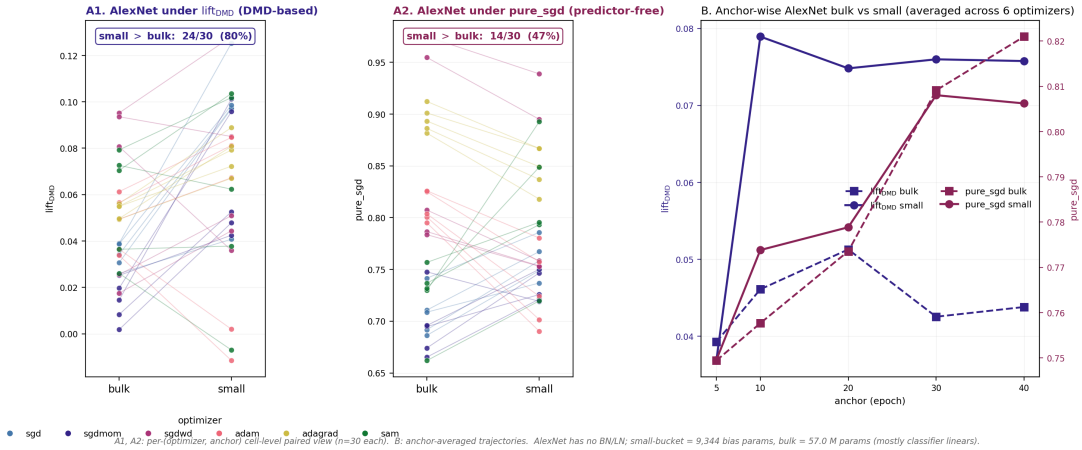


Figure B.18: Probe-dependent caveat on AlexNet. Under DMD-based `lift_DMD`, AlexNet still shows an auxiliary-over-bulk majority pattern, while `pure_sgd` can invert the ordering. This boundary case supports a careful interpretation: the evidence is for structured heterogeneity, not for a universal scope ordering that is invariant to probe and architecture.

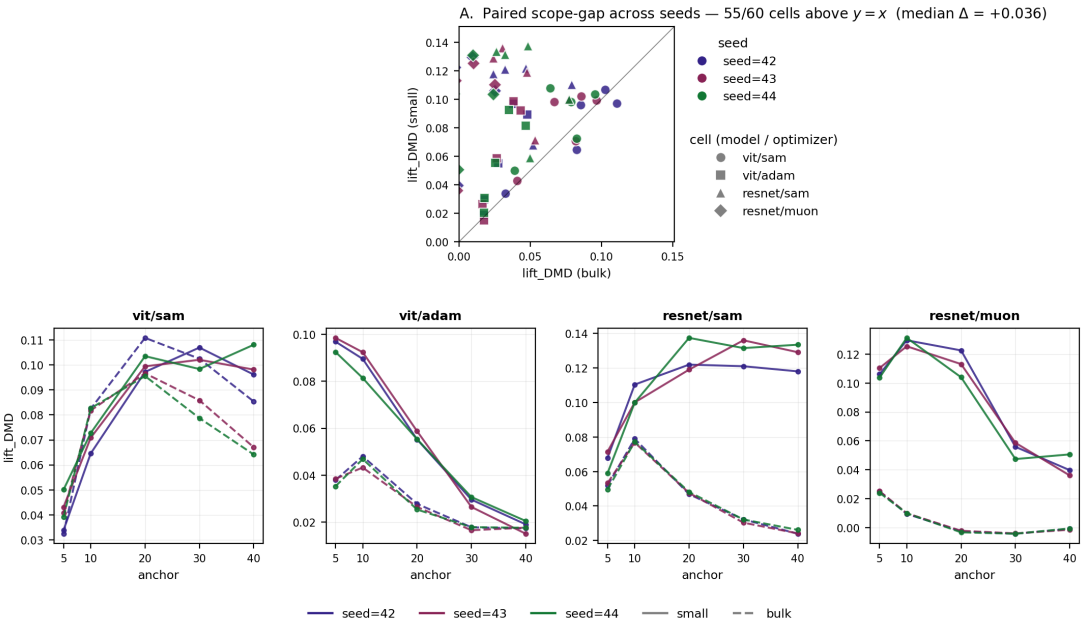


Figure B.19: Representative multi-seed check for spatial and temporal heterogeneity. The tested three-seed, four-cell subset preserves the main auxiliary–bulk selector-lift gap and temporal profiles within the subset. This is a representative check against seed-specific artifacts, not an exhaustive multi-seed sweep.

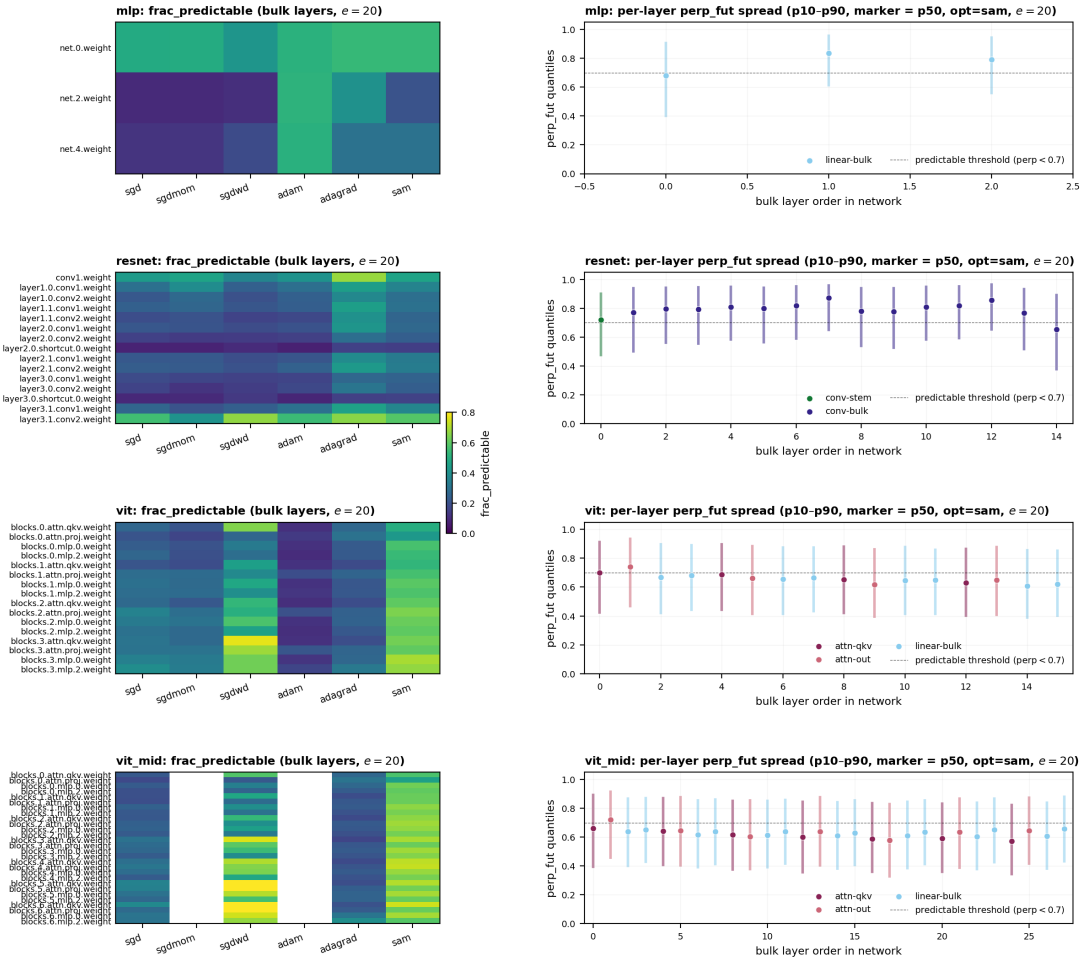


Figure B.20: Full within-bulk pocket grid across the four main models. This is the expanded version of the within-bulk pocket analysis in Chapter 4, showing per-layer and per-optimizer predictable-group fraction and per-layer FRR spread. It is retained as supplementary evidence because the full grid is too dense for the main text.

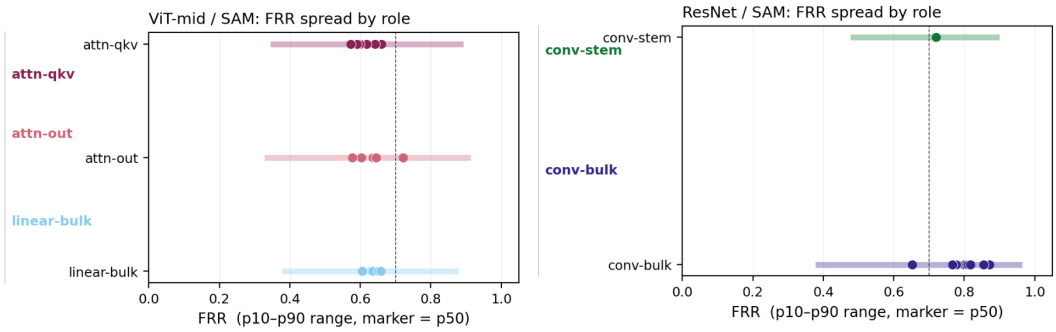


Figure B.21: FRR-spread diagnostics for the two representative within-bulk cases shown in Chapter 4. Horizontal bars show the p10–p90 range of FRR across layers within each role, and filled markers show the p50 median. These diagnostics complement the LRGF view by showing that bulk predictability is localized and role-dependent rather than uniform.

B.4.2 Pocket Trajectories, Persistence, and Overlap Registrations

Section 4.3.3 reports that predictable pockets are a small minority of bulk group–anchor observations, that most of them are transient, and that the degree of persistence varies by parameter role. This appendix gives the trajectories behind those statements: what the pocket population looks like at a given time, when it appears and how long individual pockets survive, the one event on the pretraining side at which the population changes discontinuously, and the measured overlaps between pocket membership and indicators defined outside the subspace-residual family.

Units. Throughout Chapter 4 and this appendix, each readout below names the unit it is read on and the threshold convention it uses (Appendix B.2.7). Two units appear. At *group level* a pocket is a functional group whose FRR falls below the size-matched 5% null quantile at that anchor (τ=5, raw protocol); this is the census unit, and it is read as excess over the null. At *row level*, meaning one W_{qkv} row, one d_{head} slice of the attention output projection, or one output column of an MLP neuron, the reported quantity is the fraction of rows with FRR < 0.7; at those group sizes the

membership-resample pass rate is near zero (Appendix B.2.4), so that fraction is read as an absolute population. The two are not interchangeable and are never pooled.

Reading discipline. The overlap subsection is written as a registry. Each entry is a measured overlap between pocket membership and one indicator; the entries are not treated as evidence for one another, and none of them supports a statement about what pocket membership represents.

Census and Residence

Group-level census. The census covers 123 anchors at 1k-step spacing for both 70M and 160M. The population is stratified by segment and, within a segment, concentrated in a few layers: averaging the bulk-segment share over anchors, 70M runs from 0.056 at layer 2 to 0.173 at layer 4, and 160M from 0.046 at layer 5 to 0.155 at layer 11, a factor of about three between the quietest and the busiest layer at both sizes.

Stability follows the same stratification. Table B.15 reports membership overlap between consecutive anchors together with residence run lengths. Vector and qkv pockets are close to persistent (vector Jaccard 1.000 at k=1, with residence median 7.5 anchors at 70M and maximum 62 at 160M; qkv Jaccard .875 and .842, with residence median 5 at 70M, p_{95} of 19 to 20, and maximum 56), attention-output is intermediate (.333 and .476), and MLP pockets are strongly transient (Jaccard .141 at both sizes, residence median 1 anchor).

Counted over residence runs rather than over segments, the population is dominated by the MLP: 103,465 of 104,030 runs at 70M and 239,191 of 240,808 at 160M, more than 99% at both sizes, and those runs have median length one anchor. The transient majority reported in Section 4.3.3 is therefore a statement about the pocket population by count, and the segment-resolved view here is what the same sentence means by persistence varying with parameter role.

Multi-pass recomputation. The same census recomputed on two vision runs with the same functional units, over anchors 5 to 45 at τ=5 epochs, moves in opposite directions for the two architectures. On ViT-mid the MLP, vector, and *qkv* shares decay (.233 → .104, .250 → .095, .219 → .182 from the early to the tail period) while the patch share rises steeply (.194 → .487 → .791). On ResNet both convolutional families rise (3×3 .143 → .381 → .472, 1×1 .106 → .162 → .261) while the vector share decays (.283 → .100). Adjacent-anchor Jaccard is .658 for ViT-mid *qkv*, .733 to .750 for vector groups, .480 for ViT-mid MLP, .446 for ResNet 3×3, and .244 for 1×1. The frozen-coordinate fraction is zero throughout both runs, so no reading in this census is carried by inactive units.

Two riders travel with the cross-regime comparison. The time unit differs: one pretraining anchor spans about 0.8% of training against about 2% for one vision epoch, so the sparser vision sampling mechanically depresses its Jaccard values. The contrast that survives that bias is the transient MLP population (.141 against ViT-mid MLP .480), which runs against the direction of the bias; the *qkv* contrast is partly attributable to it. And a blanket reading that one-pass pockets are more stable does not hold, because the stratification pattern itself differs between the two regimes.

Dependence on the unit. Persistence is a property of the unit as much as of the run. Under a top-*q* *group*-level agreement definition, the transfer family gives cross-run Jaccard 0.115 to 0.135 against a chance level of 0.111, with a within-run adjacent-anchor ceiling of only 0.127 to 0.156: at that unit the objects have almost no memory. Moving to *tensor*-level units on the same runs, at the same chance level, gives within-run persistence 0.57 to 0.73 in every condition. On the tensor-level definition the late-training pocket location is also conserved across transfer regimes: the three continuing-descent arms converge to 0.57 to 0.71 against a same-anchor cross-seed reference band of 0.58 to 0.80, while at early anchors every cross-regime overlap sits at 0.23 to 0.38, far below the band, and the two anti-persistent arms

stay below it throughout. We register that as a time-resolved reading: early pocket locations are regime-specific and late ones are not.

Emergence and Survival

Size-ordered timing. Table B.16 collects the row-level emergence readings for five sizes and Figure B.22 shows the underlying depth-by-time maps. Both the pool-coherence onset, which is where the null line descends, and the row-level pocket onset move later with model size, and the two do not move together. 70M has both at 20k. 160M has a two-stage null-line descent, moderate at 13k and deep over about 72 to 76k, with pocket onset at 20k. 410M has pocket onset at 48k with the null-line onset only at 92k. Neither 1B nor 1.4B meets either criterion inside the 129k window. The ordering is not a function of parameter count alone: at equal width 2048 the deeper model starts (1.4B, 24 layers, rising from about 100k) while the shallower one does not (1B, 16 layers), and at equal depth 24 the narrower model starts much earlier (410M at 48k against 1.4B at the window edge).

At 410M the row-level population starts at 48k while the pool-coherence onset is at 92k, so the row-level population is not gated by the pool-level onset. The two are separable events and no ordering between them is asserted beyond what each run's readings show.

Layer relay within a run. At 70M the row-level population is a sequence of layer-localized waves rather than a diffuse few percent: layer 5 rises first (7.6% at 20k, peaking at 22.1% at 28k, then decaying), layer 3 takes over from 32k with a double-humped trace (13.2% at 56k and 12.3% at 76k), the main layer-4 wave follows (peaking at 42.1% at 84k and staying elevated over 50 to 120k), and a later wave arrives near 115k. At 160M the relay runs layer 8 (peak 32.8% at 92k) into layer 7, which reaches 30.3% at 116k. At 410M the deep block, layers 18 to 21, rises together into a broad 110 to 120k band with the maximum at layer 21 (43.2% at 116k).

	Pythia (one-pass)		CIFAR (multi-pass)
Segment	70M	160M	counterpart
qkv	0.875	0.842	0.658 (ViT-mid qkv)
vec	1.000	1.000	0.733–0.750 (ViT-mid / ResNet vec)
mlp	**0.141**	**0.141**	**0.480** (ViT-mid mlp)
attn_out	0.333	0.476	—

	Pythia 70M				Pythia 160M			
Segment	p50	p95	max	n	p50	p95	max	n
qkv	5	19	56	175	2	20	56	612
vec	7.5	25.5	32	14	2	20	62	50
mlp	1	3	123	103 465	1	3	123	239 191
attn_out	1	6	40	376	1	10	30	955

Run lengths count consecutive anchors (1 anchor = 1000 steps) that a group spends inside the pocket; n is the number of residence runs. Key readings: vec p50 7.5 anchors (70M) with max 62 (160M); qkv p50 5 (70M), p95 19–20, max 56; mlp p50 = 1 at both sizes.

Table B.15: Pocket stability across training regimes. Top: Jaccard overlap of pocket membership between consecutive anchors (k=1, median over anchors), at the group-level unit defined at the start of Appendix B.4.2. Bottom: residence run lengths on the pretraining side, counting consecutive anchors a group spends inside the pocket set, with n the number of runs. Stability is stratified by segment and the stratification pattern differs between the two regimes, so a blanket ordering between them does not hold. At k=1 one pretraining anchor spans about 0.8% of training against about 2% for one vision epoch, so the sparser vision sampling mechanically depresses its values; the transient MLP contrast runs against that bias and is the most robust entry. These overlaps are registered as descriptions of pocket dynamics under the two regimes and are not treated as cross-regime confirmation of a shared mechanism.

	qkv, row level (D = one W_{qkv} row)				attn_out	mlp (out-half)
Model (L / d_{model})	Null-line onset	Pocket onset	Main peak (layer @ step)	Later wave (layer @ step)	Peak (layer @ step)	Peak (layer @ step)
70M (6 / 512)	20k	20k	**42.1**% (L4 @ 84k)	12.3% (L5 @ 116k)	19.3% (L4 @ 80k)	6.9% (L5 @ 84k)
160M (12 / 768)	13k†	20k	**32.8**% (L8 @ 92k)	30.3% (L7 @ 116k)	15.7% (L8 @ 104k)	15.7% (L11 @ 108k)
410M (24 / 1024)	92k	48k	**43.2**% (L21 @ 116k)	32–38% (L18–L20 @ 116k)	not scanned	5.2% (L23 @ 116k)
1B (16 / 2048)	none in window	none	0.3% (L1 @ 8k)	—	not scanned	1.8% (L15 @ 8k)
1.4B (24 / 2048)	none in window	none‡	3.0% (L15 @ 128k)	—	not scanned	1.8% (L23 @ 12k)

Criteria. Null-line onset: the pooled membership-resample q_{05} of qkv FRR (layer level, τ=5, raw; median over layers) stays below 0.90 for five consecutive 1k anchors. Pocket onset: some layer's row-level FRR<0.7 fraction stays above 5% for three consecutive anchors of the 4k row-scan grid. Peak = maximum over (layer, anchor) of the row-level FRR<0.7 fraction; "later wave" is the largest reading after the main peak's anchor. At these group sizes the membership-resample pass rate at θ=0.7 is ≈0, so the fractions are read as absolute rather than as excess over a null. **Notes.** †160M descends in two stages (moderate at 13k, deep at ∼72–76k); the deep stage sits near this run's smoothed loss minimum (77.3k). ‡1.4B does not meet the three-anchor criterion inside the window: its row-level fraction starts rising at ∼100k in L11–L15/L21/L23 and reaches 3.0% at the window edge (128k) while still increasing, so this is a window-edge case rather than a negative reading. 1B is read with the checkpoint-116000 masking window applied (anchors whose τ=5 window uses that checkpoint as base or history, 116–121k, plus the future endpoint 111k; 7 of 123 anchors dropped); its row-level fraction stays ≤0.33% throughout the remaining window. 1B is registered here for completeness but is left out of the cross-scale trend figures and of the cross-scale appendix: it is the one suite member trained in bf16 rather than fp16, and at 16 × 2048 its depth ordering crosses the parameter-count ordering, so it does not sit on the depth/width axis those readings use. Released checkpoint dtype is fp16 at every size, so the container precision our pipeline reads is not special for 1B. attn_out row scans exist for 70M/160M only (410M/1B/1.4B requested); groups are flattened on $(d_{\text{model}}, d_{\text{head}})$ with d_{head} as the unit, a convention less clean than for qkv (70M D=64 gives 19.3% vs. 13.1% at D=512; 160M 15.7% vs. 16.7%). The mlp column reports the out-half (w_{out} columns) cut, the only mlp cut carrying signal: in-half and magnitude cuts peak at ≤1.6% and ≤0.12% across all five sizes, and the mlp carrier is the deepest layer in every size. The row-length convention is invariant over D=128..3072, so no per-size adjustment is applied. Large-size mlp readings are quoted from full-protocol scans only; the 10%-sampled preview lane is not quoted as an S2 reading. All entries are single-run readings on the public Pythia checkpoint series, registered as descriptions of those runs.

Table B.16: Row-level emergence readings across five pretraining scales, with the onset criteria and the per-segment conventions given in the table footer.

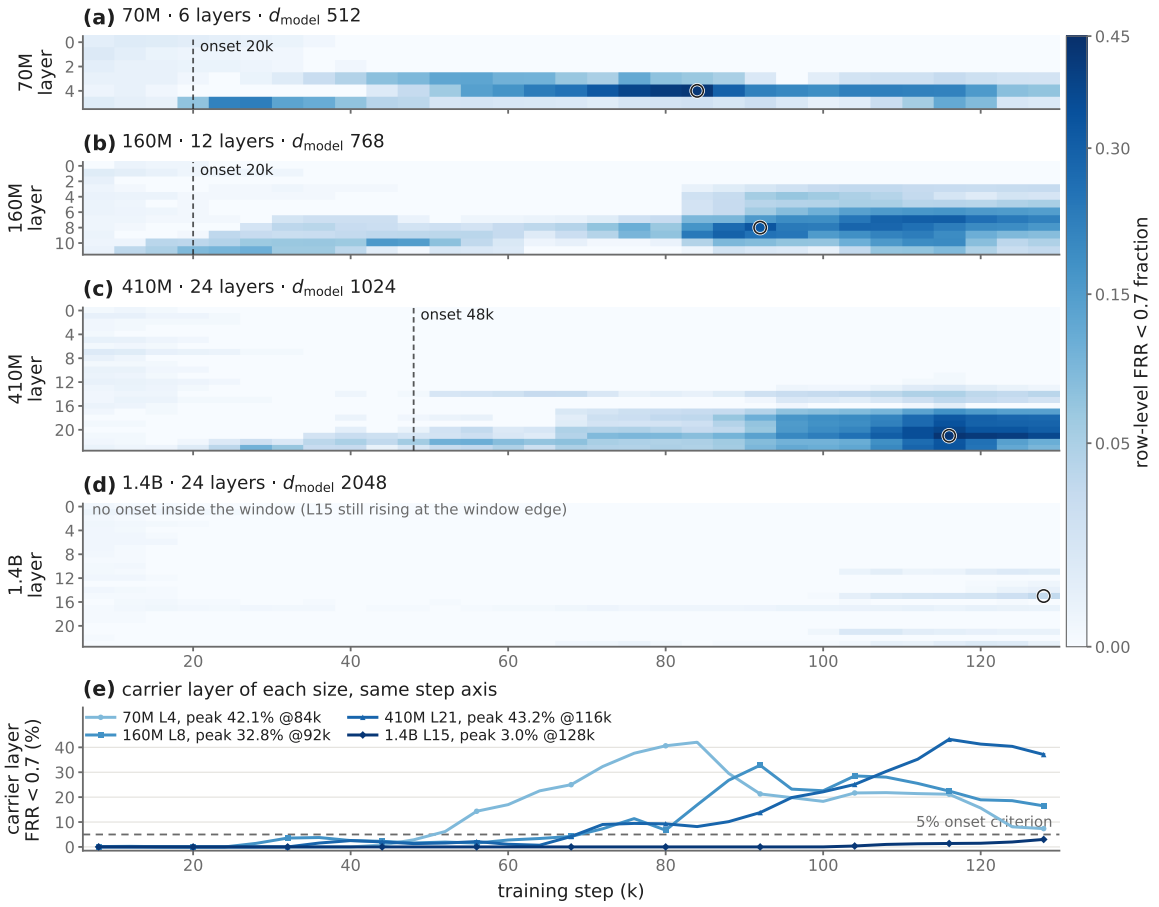


Figure B.22: Row-level pocket emergence across four pretraining scales (qkv). Every panel shows the fraction of rows with FRR < 0.7 resolved by depth and training step, where one row is one W_{qkv} row ($D = d_{\text{model}}$); at these group sizes the membership-resample pass rate at θ=0.7 is near zero, so the fraction is read as an absolute population rather than as excess over a null. Panels (a) to (d) share one colour scale and one colour bar, so the four sizes are directly comparable; the pooled maximum is 0.4323 (410M layer 21 at 116k). Dashed vertical lines mark the row-level pocket onset, the earliest anchor at which some layer stays above 5% for three consecutive anchors of the 4k scan grid: 20k at 70M and 160M, 48k at 410M, and none inside the window at 1.4B, whose layer-15 population is still rising at the edge. Rings mark the maximum over layer and anchor. Panel (e) shows the carrier layer of each size on the same step axis with the 5% criterion as a reference line. The 1B series is excluded from this figure on the grounds registered in Appendix B.4.3, and its row-level fraction stays at or below 0.33% throughout in any case. All entries are single-run readings on the public checkpoint series, registered as descriptions of those runs; no ordering between the row-level onset and the pool-coherence onset is asserted here.

The attention block moves together. Figure B.23 places the three bulk segments on a common scale. At head granularity the attention output projection shows no pocket population at all (median FRR 0.993 to 0.9996 across sizes). At row granularity it does: 19.3% at 70M layer 4 at 80k against *qkv*'s 42.1% at layer 4 at 84k, and 15.7% at 160M layer 8 at 104k against *qkv*'s 32.8% at layer 8 at 92k, about half the *qkv* fraction in the same layer over the same period. The carrier layers of the three segments agree closely within a size (*qkv*, attention output, MLP: 70M 4/4/5; 160M 8/8/11), and their peak times separate only at 160M, where *qkv* leads at 92k and the other two follow at 104k and 108k. We register this at the level of the attention block: the population appears on the read-in and the write-out projection together, not on *qkv* alone.

The MLP asymmetry. Re-cutting the MLP separates its two halves. Across all five sizes the output half is the only cut carrying signal, with peaks from 1.8 to 15.7%, while the input half and the magnitude cut stay at or below 1.6% and 0.12%. This is a third independent negative reading for the magnitude hypothesis on the MLP (Appendix B.2.6). The carrier is the deepest layer in every size without exception. The late waves occur only in the models that have crossed their transition (6.9% at 84k for 70M, 15.7% at 108k for 160M, 5.2% at 116k for 410M), whereas the 1B and 1.4B maxima sit at 8k and 12k and are early-training residue.

Head-scale and row-scale do not coincide. Within a layer, the head-granularity calibrated readout fades before the row-level population appears. Measured with a symmetric ±32k window around each layer's pocket onset, so that the search is not restricted to the pre-onset side, the median lag from the calibrated trough to row-level onset is +20k steps at 1.4B (four of four layers) and +24k at 410M (eight of eleven layers earlier, three reversed). The horizontal counterpart runs the other way and must be reported alongside: across cells the Spearman correlation between the row-level fraction and the calibrated readout is *positive* (+0.10 overall, +0.34 late),

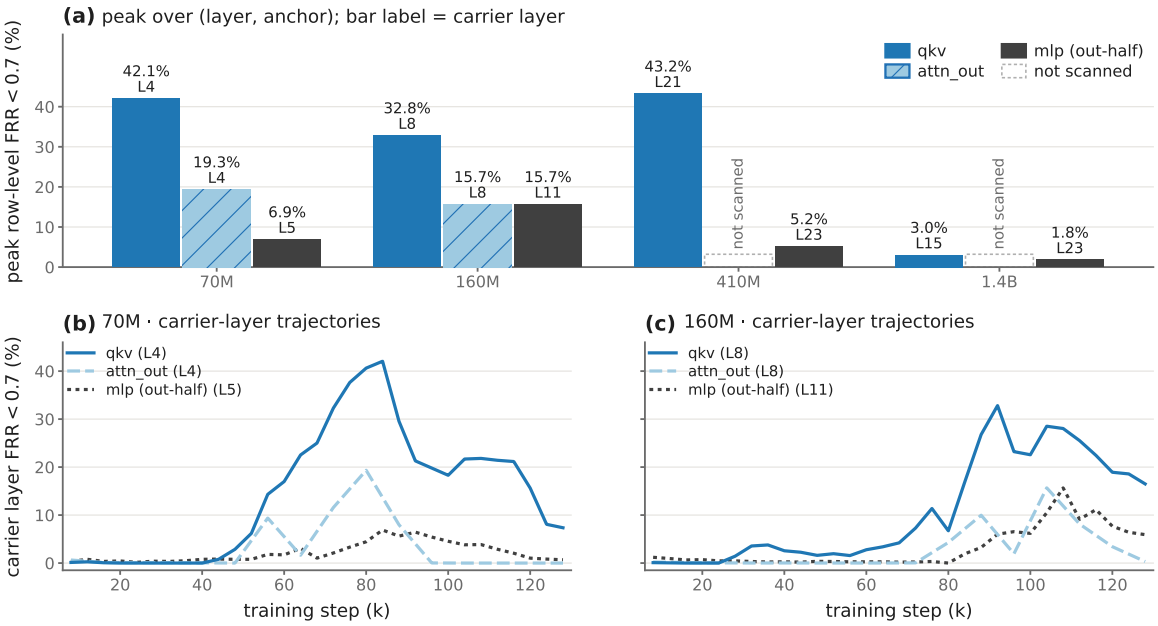


Figure B.23: The same row-level population on three segments. (a) peak fraction of rows with FRR < 0.7, maximized over layer and anchor, for qkv, the attention output projection, and the MLP output half at four sizes, with the bar label giving the carrier layer; row scans of the attention output projection exist for 70M and 160M only, and the missing cells are drawn as explicit markers rather than as gaps. (b) and (c) trajectories of the carrier layer of each segment at 70M and 160M. Two conventions travel with the figure: the attention-output granularity is less clean than qkv's, since groups are flattened on $(d_{\text{model}}, d_{\text{head}})$ with d_{head} as the unit and 70M reads 19.3% at D=64 against 13.1% at D=512; and group size differs between segments, so levels are compared within a segment across sizes rather than across segments. The 1B series is excluded on the grounds registered in Appendix B.4.3.

and the cells where the row-level population emerges have the same median calibrated readout as the rest (0.208 against 0.208). A low calibrated readout therefore does not imply a pocket. What is registered here is a within-layer temporal ordering, not a horizontal association.

Form of the emerging population. At 70M qkv the coordinates carrying the coherent motion rotate continuously: adjacent-anchor membership Jaccard of the comoving family is .14 to .18, while cross-head timing sharing stays at 0.75 to 0.99 throughout. Within a head the family condenses into two or three row blocks, at four to eight times chance adjacency and 20 times at layer 5. In shallow and middle layers the per-head block directions are private (cross-head direction cosine near chance at .035) and only the timing is shared; at layer 5 both are shared (cross-head cosine .82, effective rank 1). The granularity ladder settles the unit, since the one-row and quarter-row distributions nearly coincide, so within the tested ladder the coherent object is row-scale. Pooled over all layers the model-wide row-level fraction is stable across training (3.7%, 4.8%, 4.4% at 24k, 60k, 120k), so the layer-localized waves ride on a small standing population rather than replacing it.

The One Discontinuous Event

Where the ∼20k event sits. Between anchors 19k and 20k at 70M the qkv pocket set collapses from 51 groups to 3 and the layer-level median calibrated readout drops from .354 to .000, while the measured median FRR of the same groups does not move at all (.9959 to .9952, staying near .995 before and after). What moves is the null: the pooled membership-resample q_{05} falls from .9937 to .7201 and stays at .59 to .72 through 25k (Figure B.24). The calibrated readout is a difference between a measurement and its null, so this event is located on the null side of that difference. This is what lies behind the caution in Section 4.3.4 that the fading of head-aligned excess should not be read as a global disappearance of predictable structure.

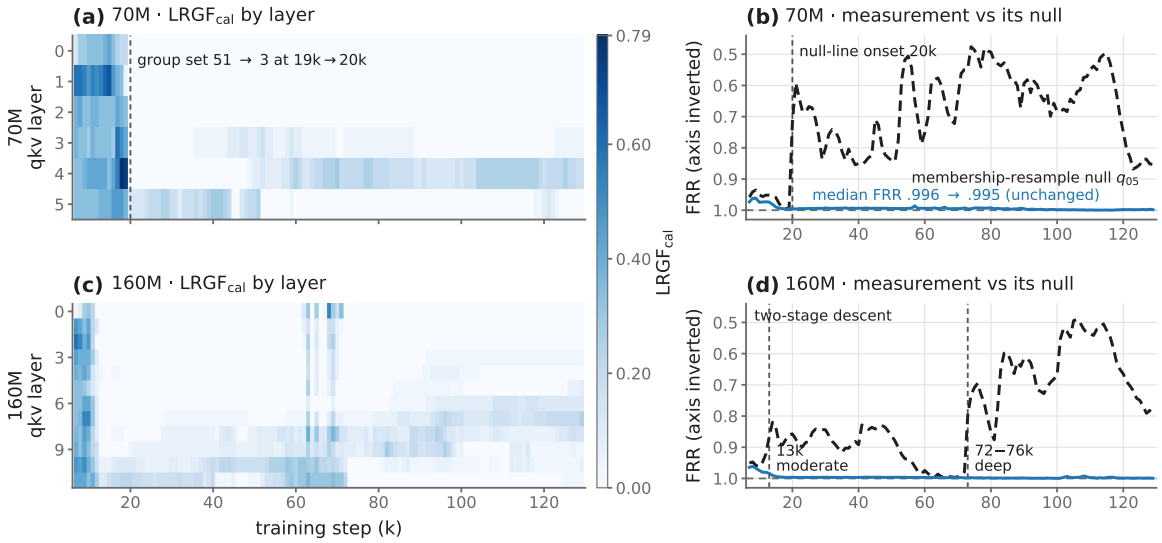


Figure B.24: Where the ∼20k *qkv* event sits, in the measurement or in its null. (a) and (c) layer-level calibrated readout for *qkv* (τ=5, raw) at 70M and 160M on a common scale. (b) and (d) the same lane read as two curves: the measured median FRR of those layers (solid) and the pooled membership-resample q_{05} (dashed), with the FRR axis inverted so that upward means lower FRR and the null pool merged across layers per segment and group size. At 70M the pocket set collapses from 51 groups to 3 between anchors 19k and 20k and the layer-level median calibrated readout falls from .354 to .000, while the measured median FRR of the same groups is unchanged (.9959 to .9952). What moves is the null, whose q_{05} falls from .9937 to .7201 and stays at .59 to .72 through 25k. The calibrated readout is a difference between a measurement and its null, so the event is located on the null side of that difference and is not described here as a change in the predictability of the groups. The 160M null line descends in two stages, a moderate descent at 13k and a deep one over about 72 to 76k, the latter in the neighbourhood of that run's smoothed loss minimum (77.3k); the time co-location with the loss landmark is registered as a correspondence only.

Composition of the null pool. A single-anchor diagnostic at 70M at 25k finds the null FRR distribution long-tailed and bimodal ($q_{10} = .67$ with a jump to $q_{25} = .992$), and across resampled groups the FRR tracks the share of squared update energy carried by the top 1% of coordinates at $r = -.996$. The reading is that about 1% of large-magnitude coordinates move coherently across heads and dominate the resampled groups.

Turnover across the event. The set that survives is not a subset of the one that collapsed (Figure B.25). Jaccard against the final set is near zero before 48k (maximum .061), rises to .5 to .8 over 52 to 70k, and settles at 1 in the tail, while the adjacent-anchor Jaccard after the collapse has median .88: the surviving set is stable from anchor to anchor but shares almost no membership with the pre-collapse population. On the attention output projection the sets never stabilize (adjacent-anchor median .33, versus-final median .09).

A co-located depth reading. An independent lane measures the effective rank of the *qkv* update column space, as participation ratio divided by output dimension, in phase-diagram windows. In the ballistic window both *qkv* and MLP read 0.24 to 0.44 at every depth. In the floor windows the MLP stays at 0.75 to 0.77 (70M) and 0.65 to 0.85 (160M) and shallow *qkv* at 0.59 to 0.70, while deep *qkv* collapses, to .011 and .007 at 70M layers 3 and 4 and to .007 to .030 at 160M layers 6 to 10. Those depths overlap the layers that carry the persistent calibrated band after the event. The two lanes read the same update sequence, so this is registered as a co-located depth pattern between two readouts of one record.

No multi-pass counterpart. The vision runs show no event of this kind. Pocket counts grow monotonically (ViT-mid patch 34 to 192, ResNet 3×3 280 to 903) and the Jaccard against the final set rises steadily from about 0.2 to about 0.8, which is

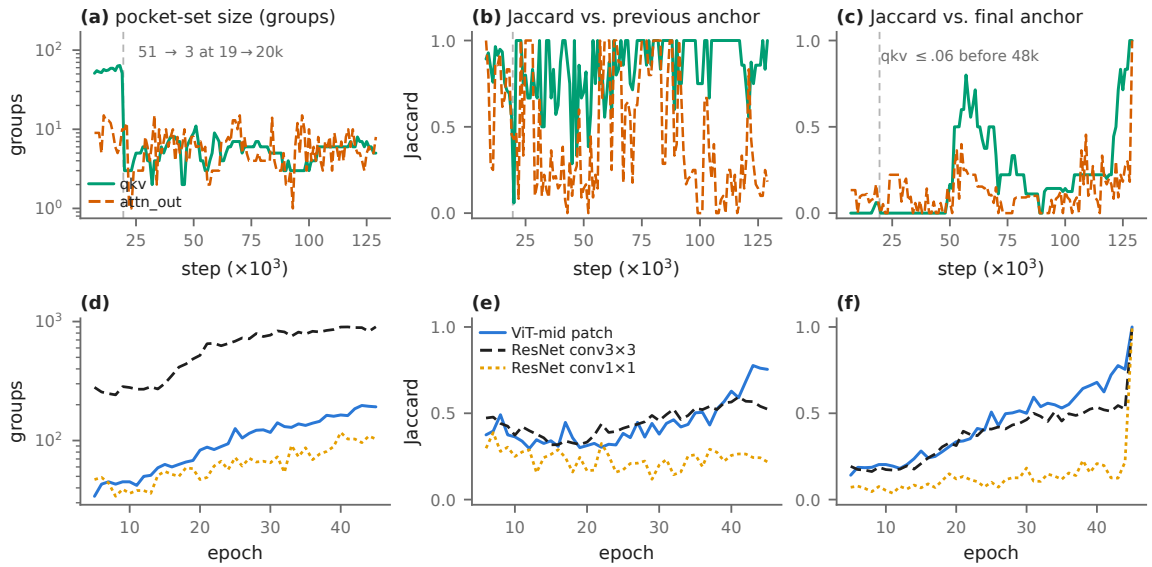


Figure B.25: Turnover of the pocket set itself, at group level, for the one-pass 70M run (top row, dashed vertical at the 19 to 20k anchor pair) and the two multi-pass vision runs (bottom row). Columns: set size on a log scale, Jaccard against the previous anchor, and Jaccard against the final anchor. On *qkv* the set collapses from 51 groups to 3 and then holds 2 to 11 groups, with Jaccard against the final set at or below .06 before 48k, rising to .5 to .8 over 52 to 70k, while the adjacent-anchor Jaccard after the collapse has median .88. The attention output projection never stabilizes. The multi-pass runs show no event of this kind, with counts growing monotonically and the Jaccard against the final set rising steadily, which is cumulative expansion rather than replacement. This panel sets the reading conditions for the two-anchor movement overlaps: because the pocket set changes both size and membership between anchors, an anchor-to-anchor difference in overlap contains both a change in overlap and turnover of the set, and the two are separated only by reporting them side by side.

cumulative expansion rather than replacement; the 1×1 family is the exception, with a versus-final median of .12.

Overlap Registrations

Table B.17 is the registry. Every entry is a measured overlap between pocket membership and an indicator that is not a probe of the subspace-residual family.

Signed norm-change proxy. A signed norm-change proxy $S_i = \sum_t w_i(t)\Delta w_i(t)$ is built from checkpoint differences alone; preconditioning is not removed, and S is not a curvature-based importance measure. Against an equal-cardinality hypergeometric null, *qkv* pocket coordinates overlap the top-K movement set well above the null band (4.0× at 70k, 9.0× at 129k), the ViT-mid magnitude-bin families sit just above their upper bound (.142 to .217 against .125), ResNet 3×3 exceeds its bound by only 1.04 to 1.08×, and 1×1 falls below it (Figure B.26). The attention output projection reverses at late anchors, with its terminal pocket set displaced towards the low end of the signed movement axis.

Because the pocket sets themselves change size and membership over time, the overlap is also measured anchor by anchor (Figure B.27). On *qkv* the ratio hugs the random line before the collapse (0.98 to 1.25 over 7 to 19k, when the set holds 51 to 64 groups) and steps to 3.77 at 20k, staying high for the rest of training (cumulative median 7.5×, recent-window median 9.6× with a maximum of 16.2× at 23k). ViT-mid patch filters sit slightly above the band early and inside it from epoch 23 apart from single-epoch excursions; ResNet 3×3 sits *below* the band over epochs 9 to 16 and above it from epoch 17.

Normalization gain outliers. Coordinates whose normalization gain is a robust-z outlier above 3 at the late anchor have higher median ESA than their non-outlier neighbours in 12 of 13 pretraining tensors and higher median coherence in 11 of 13, with the rank correlation between the outlier score and coherence between .78 and

Indicator pair	Paradigm side	Direction	Key readings	Null convention	Caveat registered
Movement top-K × pocket coordinates	Pythia qkv (70M)	above the null band	precision .195 at a70k (4.0× null .0486) and .312 at a129k (9.0× null .0347); time-resolved ratio steps 0.98 → 3.77 across 19k→20k, then stays high (cumulative median 7.5×, recent-window median 9.6×, max 16.2× at 23k); pocket coordinates sit at family movement quantile p_{50}=.93 at a129k	hypergeometric, equal cardinality (K = number of pocket scalars), with 95% interval	$S_i = \sum_t w_i(t)\Delta w_i(t)$ from checkpoint differences only; AdamW preconditioning not removed. Pocket sets change both membership and size between anchors, so a two-anchor comparison mixes overlap change with set turnover
	Pythia attn_out (70M), late anchors	below the null band on signed-top	precision .093 vs. null .167 at a129k; for anchors ≥100k the cumulative top ratio has median 0.34 and the cumulative bottom ratio 1.90, while both recent-window ratios sit at ≈0.6	same	small sets (1–15 groups) with heavy turnover (adjacent-anchor Jaccard median .33, Jaccard vs. final set median .09); the two time conventions disagree, and both are reported
	CIFAR (ViT-mid, ResNet)	above the upper bound, small margin	ViT-mid magnitude-bin families .142–.217 vs. null .125 (five families, both anchors); ResNet conv3×3 .304/.382 vs. null .282/.367 (1.08×/1.04×); patch-filter inside the band at both anchors	same	ViT-mid patch at a45: the cut is near saturation (K/N=86%), which narrows the null interval mechanically; ResNet conv1×1 falls below the lower bound at a25

Table B.17: Registry of measured overlaps between pocket membership and indicators defined outside the subspace-residual probe family. Each row records the direction, the key readings, the null convention, and the caveats that travel with it. The rows are not treated as evidence for one another, and none of them supports a statement about what pocket membership represents.

Table B.17 (continued)

Indicator pair	Paradigm side	Direction	Key readings	Null convention	Caveat registered
LN outlier coordinates × per-coordinate dynamics	Pythia 70M (13 LN gains); ViT-mid control (15)	co-located	outlier vs. non-outlier median ESA higher in 12/13 tensors (e.g. post5 .992 vs. .854); median coherence higher in 11/13 (e.g. in2 .898 vs. .114); $\rho_S(z, \text{coherence})$ positive in 11/13 (.78–.94), negative for in0 (−.37) and post0 (−.48); cross-layer outlier Jaccard concentrates on adjacent LN pairs (median .023, p_{90} .32, max .56 for post3×post4)	robust $z > 3$ on |gain| at the late anchor; a top-1% control cut reproduces every direction	the final LN is saturated (ESA and coherence = 1.0 on all 512 coordinates, no resolution). ViT-mid gain range is narrow ($\|\text{gain}\| \leq 1.05$) and only 4/105 cross-layer pairs are non-zero. Outlier-set formation times are non-monotone (final LN set size 49 at 88k, 9 at 129k)
M8 low-participation heads × pocket membership	Pythia 70M qkv	co-located	18 heads fall below the pooled-bulk Tukey fence ($\text{pr}_{\text{norm}} < .9047$); 8 of them are layer-4 q-heads; the seven lowest (pr_{norm} .262–.622) are exactly the a70k pocket set {L4:q_h0,1,2,4,5,6,7}	Tukey fence (pooled bulk $Q_1 - 1.5\,\text{IQR}$) on dimension-normalised participation ratio	participation ratio and pocket flag are computed from the same update history, so this is registered as co-location of two readouts of one record, not as two independent measurements
qkv column-space effective rank × calibrated-readout extinction	Pythia 70M, 160M	co-located in depth	floor window: 70M qkv L3/L4 participation ratio per dimension .011/.007 against .59–.66 in shallow qkv layers and .75–.77 in mlp; 160M hump-rise window L6–L10 .007–.030 against .65–.85 in mlp. The same depths carry the persistent LRGF_{cal} band after the 20k step	none (descriptive ratio; no resampling null defined for this quantity)	window boundaries are operational (ballistic $[1\text{k}, 10\text{k})$ plus phase-diagram floor windows), the two readouts share the same ΔW sequence, and fp16 quantisation noise on per-coordinate increments is registered for this lane

Table B.17 (continued)

Indicator pair	Paradigm side	Direction	Key readings	Null convention	Caveat registered
Group-level FRR × ESA × VCS	CIFAR structural cuts (3 architectures × 7 recipes × 5 anchors)	co-elevated	P(VCS>0.4 | FRR<0.7) = 88.4% against a base rate of 38.9% (ViT-mid); P(ESA>0.7 | FRR<0.7) lifts by 3.4× (ViT-mid) to 23.8× (ResNet); all three thresholds are passed by 6.7% / 3.3% / 0.3% of ViT-mid / ViT / ResNet groups	membership-resample null at large D: σ_{null} is .005–.024 for ESA and .011–.047 for VCS, so all three thresholds are chance-immune	the three probes read the same Δw history and are not independent instruments. Their time profiles separate: pooled ESA and VCS shares decay monotonically (18.4%/49.0% at a5 to 0.9%/6.1% at a40) while the FRR share peaks at a10–a20 (3.7/10.0/10.0/5.9/3.4%); learning-rate schedule is not disentangled
Loss-curve events × qkv transition	Pythia 70M, 160M	co-located in time	70M: calibrated qkv readout fades at ∼19.5–20k (layer-level median LRGF_{cal} .354 → .000 across 19k→20k), smoothed loss minimum at 49.7k, main row-level pocket wave at 80–85k. 160M: smoothed loss minimum at 77.3k, null-line deep descent at ∼72–76k	none (event co-location on the time axis)	one run per size; loss minima are read from smoothed curves with a rule-dependent resolution, and the event windows are wide relative to the anchor spacing. Detail in the main-text section on loss co-location

Every row is a registration of a measured overlap between pocket membership and an indicator defined outside the FRR probe family. The rows are not treated as evidence for one another, and no row is used to support a statement about what pocket membership represents. "Above / below the null band" refers to the stated null only. Pythia pocket sets are taken from the S2 panel group-level flag (τ=5, raw, head granularity, same cut and same α); two cross-checks against the independent census agree (qkv at a70k: 7 groups, all layer-4 q-heads; attn_out at a70k: 15 of 48 heads, matching the calibrated share .313 × 48).

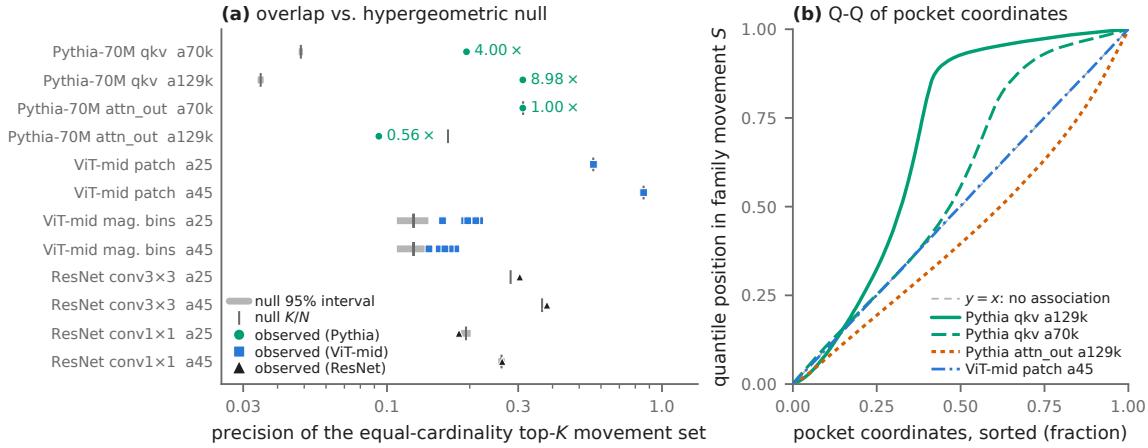


Figure B.26: Overlap between pocket coordinates and the top-K set of the signed norm-change proxy, labelled movement in the panels, registered against an equal-cardinality hypergeometric null. The proxy $S_i = \sum_t w_i(t)\Delta w_i(t)$ is built from checkpoint differences alone; preconditioning is not removed and S is not a curvature-based importance measure. (a) observed precision of the top-K movement set restricted to pocket coordinates, against the null K/N and its 95% interval, with the pretraining rows annotated by the ratio. (b) quantile position of the pocket coordinates inside their family's movement distribution, with the diagonal as the no-association reference; the *qkv* set at 129k sits at median quantile .93 against about .50 for the vision reference, so the whole set is displaced rather than a tail of it. Pocket sets are the group-level flag at τ=5, raw, head granularity. The two anchors carry different pocket sets in both size and membership, so a two-anchor comparison mixes a change in overlap with turnover of the set itself. Each reading is a registration of a measured overlap and is not used to support any statement about what pocket membership represents.

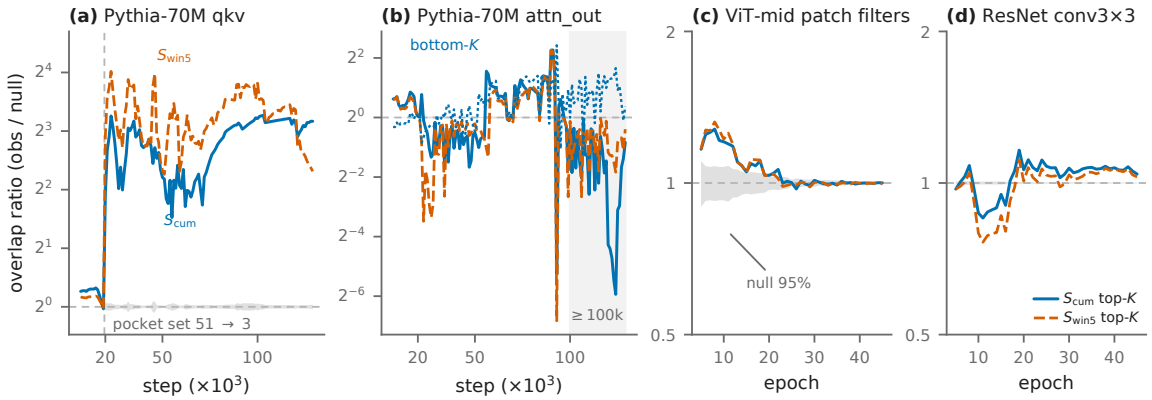


Figure B.27: The same overlap resolved anchor by anchor, with the pocket set re-matched at every anchor. The readout is the ratio of the observed overlap to its hypergeometric expectation, with the grey band giving the null 95% interval. Two time conventions for the movement proxy are reported side by side: one accumulates from the start of training, the other over the last five checkpoint or epoch pairs, matching the probe's history window. (a) *qkv*: the ratio hugs the random line over 7 to 19k, steps to 3.77 at 20k in one anchor, and stays high for the rest of training. (b) the attention output projection, on small sets with heavy turnover, where the two time conventions disagree and both are reported. (c) ViT-mid patch filters sit slightly above the band early and inside it from epoch 23 apart from single-epoch excursions; the late approach to one has a mechanical component, since the cut approaches saturation. (d) ResNet 3×3 sits below the band over epochs 9 to 16 and above it from epoch 17. Every curve is a registration of a measured overlap over time.

.94 in those 11 and negative in the two exceptions (Figure B.28). A top-1% control cut reproduces every direction including the exceptions. Outlier sets overlap across layers mainly between adjacent normalization pairs (median Jaccard .023, p_{90} .32). Their formation times are not monotone: sets grow from early training and first reach half of their final membership between 11k and 101k depending on the tensor. On the ViT-mid control the gain range is narrow and the outlier sets hold at most four coordinates, and two of its three chains read with the opposite sign. The final normalization tensor is saturated, with agreement and coherence both at 1.0 on all 512 coordinates, and carries no resolution.

Low-participation heads. Eighteen *qkv* heads fall below the pooled-bulk fence when participation ratio is normalized by dimension; eight of them are 70M layer-4 q-heads, and the seven lowest are exactly the 70k pocket set. Participation ratio and pocket flag are computed from the same update history, so this is registered as a co-location of two readouts of one record.

Cross-probe overlap at group level. On the vision structural cuts the three group-level probes concentrate on overlapping populations. Conditioning on FRR $<$ 0.7 raises the VCS $>$ 0.4 rate from a base of 38.9% to 88.4% on ViT-mid, and raises the ESA $>$ 0.7 rate by 3.4$\times$ on ViT-mid to 23.8$\times$ on ResNet; all three thresholds are passed simultaneously by 6.7%, 3.3%, and 0.3% of ViT-mid, ViT, and ResNet groups. At these group sizes the thresholds are chance-immune. The three probes read the same update history and are not independent instruments, and their time profiles separate: pooled ESA and VCS shares decay monotonically from 18.4% and 49.0% at anchor 5 to 0.9% and 6.1% at anchor 40, while the FRR share peaks at anchors 10 to 20. The learning-rate schedule is not disentangled from these profiles.

Loss-curve landmarks. The transition markers co-locate in time with loss-curve landmarks at both sizes (70M: calibrated fade near 20k, smoothed loss minimum

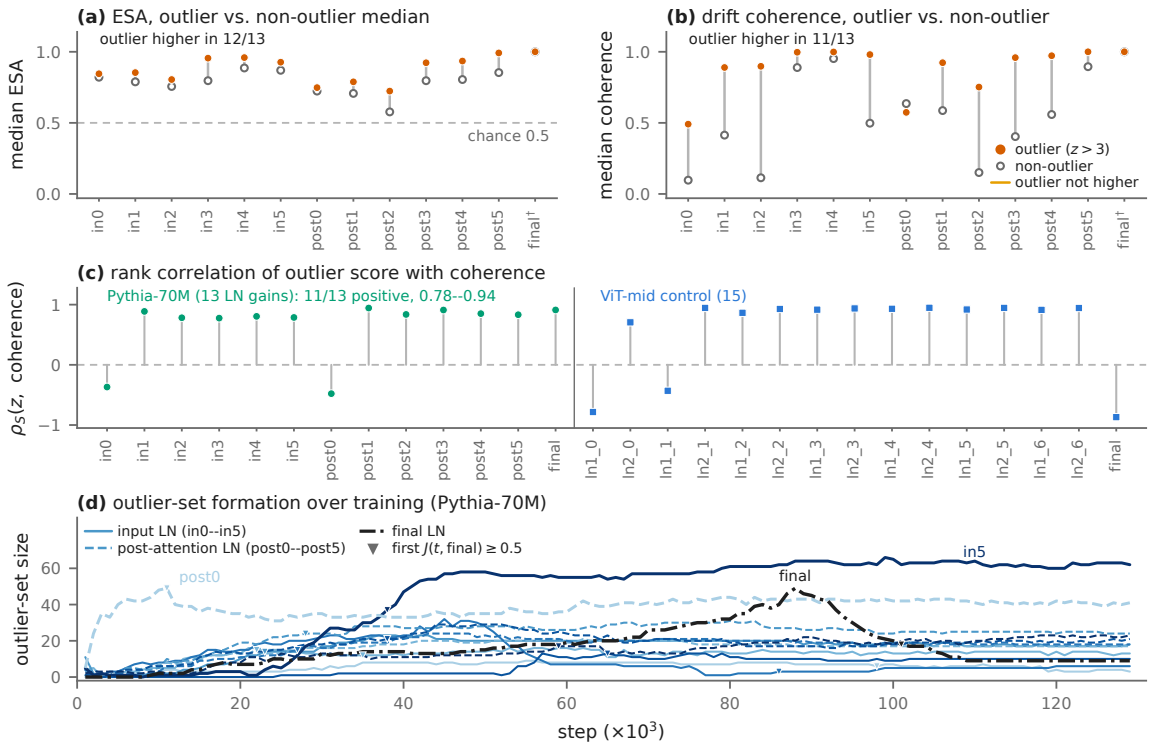


Figure B.28: Normalization gain outliers and per-coordinate dynamics on the same gain trajectories. A coordinate is an outlier when the robust z of its absolute gain at the late anchor exceeds 3. (a) and (b) per-tensor medians for outlier against non-outlier coordinates across the 13 pretraining gain vectors: median ESA is higher for the outliers in 12 of 13 tensors and median drift coherence in 11 of 13, with connectors marking the tensors where it is not. The final normalization tensor is saturated, with agreement and coherence equal to 1.0 on all 512 coordinates, so it carries no resolution and is drawn only for completeness. (c) rank correlation between the outlier score and coherence per tensor, positive in 11 of 13 pretraining tensors and negative in two. On the ViT-mid control one chain reads with the same sign and the others with the opposite sign; its gain range is narrow and its outlier sets hold at most four coordinates. A top-1% control cut reproduces every direction in both regimes.

49.7k, main row-level wave 80 to 85k; 160M: loss minimum 77.3k, deep null-line descent about 72 to 76k). This is one run per size, the loss minima are read from smoothed curves with a rule-dependent resolution, and the event windows are wide relative to the anchor spacing. The co-location is discussed in Section 4.3.4 and is registered here only for completeness.

B.4.3 Pythia Scale Extensions

Section 4.3.4 uses Pythia-70M as its case study and defers the rest to this appendix. Six sizes enter the cross-scale registry, and every statement below is a registration of readings and co-occurrences rather than a directional claim.

Coverage and Protocol

The six sizes are 14M, 31M, 70M, 160M, 410M, and 1.4B. They share the pretraining schedule that makes step numbers comparable: 143,000 optimizer steps, a single cosine decay whose decay length equals the training length so that there is no constant tail, a 1% warmup of 1,430 steps, a floor at 10% of the peak rate, decoupled weight decay 0.1 with normalization parameters and all biases exempt, and a global batch of 1024 sequences of 2048 tokens, or 2,097,152 tokens per step. Peak learning rate is the one recipe quantity that varies with size, from 1.0×10^{-3} at the three smallest to 2.0×10^{-4} at 1.4B. Equal step therefore means equal token count, and the step-indexed comparisons below are token-indexed comparisons as well. They are not matched loss, matched compute, or matched training progress, and they carry no further normalization.

The measurement protocol is identical at every size and follows Appendix B.1.3: 123 anchors over steps 7000–129000, $n_{\text{past}} = 5$, reference Δw_{a-1}, and Gavish–Donoho rank selection on a 256-unit sample, raw and column-normalized FRR in parallel, a per-anchor membership-resample null, and the calibrated readout at $\alpha = 0.05$ reported as excess. Table B.18 collects the per-size readings.

(a) Sizes, coverage, and external landmarks

Model	Parameters total	Parameters non-embed.	$L \times d_{\text{model}}$	S2 anchors	Train-loss minimum in [7k, 129k]	Bulk ESA landmark (τ=5, group p_{50})
Pythia-14M	14,067,712	1,189,888	6×128	123	no public telemetry	—
Pythia-31M	30,494,720	4,739,072	6×256	123	no public telemetry	—
Pythia-70M	70,426,624	18,915,328	6×512	123	step 50,345 (2.828)	min 0.426 @ 98k; local rise peak 0.467 @ 77k
Pythia-160M	162,322,944	85,056,000	12×768	123	step 76,409 (2.465)	min 0.424 @ 68k; sustained rise from 78k to 0.45–0.47
Pythia-410M	405,334,016	302,311,424	24×1024	123	none in window[a]	late rise from ≈85k, peak ≈0.49 @ 110–118k
Pythia-1.4B	1,414,647,808	1,208,602,624	24×2048	123	no verified telemetry[b]	—

Anchors are steps 7000 to 129000 at spacing 1000, identical for every size. Loss minima are recomputed from the public telemetry with a ±1500-step rolling median. [a] The smoothed 410M loss is still descending at the right edge of the anchor domain (step 128,756, 2.175) and reaches 2.173 at step 143k, so no bottoming event occurs inside the window. [b] The local 1.4B telemetry file does not match the released 143k-step cosine schedule, so no loss landmark is quoted. Empty cells mean not read out on the full protocol; they are deliberately not filled from the sampled-preview instrumentation.

(b) Internal transition readings

Model	qkv LRGF_{cal} collapse	Null-line onset	Row pocket onset	Row pocket peak (frac. @ layer/step)	Persistent qkv LRGF_{cal} band
Pythia-14M	43k (17k)[c]	—	—	—	none surviving in tail
Pythia-31M	28k	—	—	—	L5
Pythia-70M	$a19000 \to a20000$	20k	20k	42.1% @ L4/84k	L4 (mid and tail)
Pythia-160M	71k (12k)[c]	13k (+72k deep)	20k	32.8% @ L8/92k	L8–L11 (mid); L6/L7 (tail)
Pythia-410M	25k	92k	48k	43.2% @ L21/116k	L19–L23 (tail); L21–L23 (mid)
Pythia-1.4B	none in window[d]	not in window	not in window[e]	3.0% @ L15/128k	no collapse; band persists

Onsets use persistence criteria rather than visual reading. Null-line onset: $\text{null}_{q05} < 0.90$ for five consecutive anchors. Row pocket onset: peak-layer share of rows with FRR < 0.7 above 5% for three consecutive anchors, on D=512 qkv rows, with row-length invariance checked over $D \in [128, 3072]$ on three sizes. Segment-level collapse onset: one anchor after the last anchor at which the pooled excess reaches half its early-phase mean, declared only when the tail-phase mean has returned to the null level. The 70M collapse is a single-anchor step: head-level layer-mean excess $0.304 \to -0.029$, a change of 0.333 that is 3.7× the next largest in the series, with layers above chance falling from 5 to 1; across it the real head groups are unchanged ($\text{FRR}_{p50} = 0.994 \pm 0.001$) while the null moves ($\text{null}_{q05}$ from 0.994 to 0.67–0.85). [c] Where the series dips and recovers, the first crossing is given in parentheses. [d] 1.4B has a mid-layer extinction window at 95 to 118k, with the calibrated readout at L11/L15/L21/L23 falling to 0.02–0.27 near 96k and recovering from about 120k, but no global collapse. [e] The 1.4B row-level population reaches 3.0% at the last anchor and is still rising there. **The two onset columns are different instruments and must not be merged.** The null-line and row-pocket columns are read on the row-level lane and order with size; the segment-level collapse column is read on the calibrated pooled lane and does not (70M 20k, 410M 25k, 31M 28k, 14M 43k, 160M 71k). Any size-ordering statement names which instrument it uses.

(c) Model excluded from every cross-scale reading

Excluded model	total	non-embed.	$L \times d_{\text{model}}$	Grounds for exclusion
Pythia-1B	1,011,781,632	805,736,448	16×2048	(i) the only model of the suite trained in bfloat16 upstream, following an fp16 loss spike; (ii) depth not monotone in size, at 16 layers between two 24-layer models; (iii) an upstream defect in revision `step116000`, which forces masked windows at 116 to 121k.

Pythia-1B contributes to no row, ordering, or trend in this appendix. Its released checkpoints are stored in fp16 like every other size, so the difference is in the training arithmetic and not in the container of the weights we read; the size of any residual effect on the probes is not measured here. Reasons (i) to (iii) are independent facts and none is evidence for the others.

Table B.18: Cross-scale coverage, external landmarks, internal transition readings, and the excluded Pythia-1B model.

Why Pythia-1B is excluded. The suite contains a seventh model in this range, and it is left out of every cross-scale statement below for three reasons that are separate facts and should not be conflated. First, it is the only model of the suite trained in bfloat16 upstream, following a half-precision loss spike late in its run; the released checkpoints are stored in fp16 like every other size, so the difference is in the training arithmetic and not in the container we read, and the size of any residual effect on the probes is not measured here. Second, its depth is not monotone in size: at 16 layers of width 2048 it is shallower than both the 24×1024 model below it and the 24×2048 model above it in parameter count, and the depth and width reading of Appendix B.4.3 is precisely a statement about depth and width. Third, one of its released revisions carries the upstream defect described in Appendix B.1.3, which forces masked windows across the very region that Appendix B.4.3 discusses. The first reason alone would keep it out of a scale-ordered comparison; the three together make the exclusion unambiguous. One cost of the exclusion is recorded where it falls due, in Appendix B.4.3.

Reading discipline. The calibrated readout detects within-group coherent motion against a membership-redrawn null whose threshold adapts to the local geometry. Horizontal comparisons of it *between segments* are therefore downgraded, and only three comparison types are used: time shape at fixed grouping, cross-size comparison under one grouping convention, and between-layer comparison inside one segment. Sampled-preview instrumentation exists for the larger sizes and no number here is taken from it; where a full readout does not exist the cell is left empty rather than filled.

The Auxiliary–Bulk Partition Holds at Every Size

Figure B.29 reads the partition of Appendix B.3.3 at all six sizes under three probes, one from each family. The separation is not a tendency that survives aggregation: the median auxiliary series exceeds the median bulk series at every one of the 123

anchors, at all six sizes, under all three probes. The smallest median gaps anywhere in the panel occur at 1.4B late in training, at 0.036 for ESA, 0.351 for $1 - \mathrm{FRR}$, and 0.115 for the DMD direction cosine.

For all six sizes, the interquartile bands of the subspace-residual and predictor-based readouts are disjoint at every anchor. The same holds for ESA at every size except 1.4B, where they overlap on 30% of anchors in the late window. The frozen-coordinate fraction is identically zero here, at every segment and every anchor, so none of this contrast is carried by coordinates that do not move.

Two conventions travel with the figure. Bulk is the strict matrix set, *qkv* together with the attention output projection and the MLP, with both embedding segments dropped. Under the looser alternative that keeps everything not in the vector segment, the bulk means move by at most 1.2×10^{-4} for ESA and $1-\mathrm{FRR}$ but by up to 6.3×10^{-3} for the DMD cosine, which is why the strict rule is fixed throughout this appendix. And the calibrated readout does not enter this figure at all; where it appears later it is reported as excess over its null level.

Segment Division of Labour

Figure B.30 puts the four segments of every size on one axis under two readouts, and the two readouts do not agree about how much structure there is to see.

Under ESA the three matrix segments are nearly indistinguishable, and they become more so as the model grows: the median spread between them narrows monotonically from 0.048 at 14M to 0.011 at 1.4B. The vector segment stays above all three by at least 0.034 at every anchor and every size. Read through coordinate signs alone, the bulk of a larger model looks more internally uniform, not less.

Under the calibrated readout the same segments separate clearly, and *qkv* carries a distinct time shape: high early, then a sustained collapse to the null level. The collapse onsets are 20k at 70M, 25k at 410M, 28k at 31M, 43k at 14M, and 71k at 160M, with no sustained collapse at 1.4B inside the window. That ordering is *not*

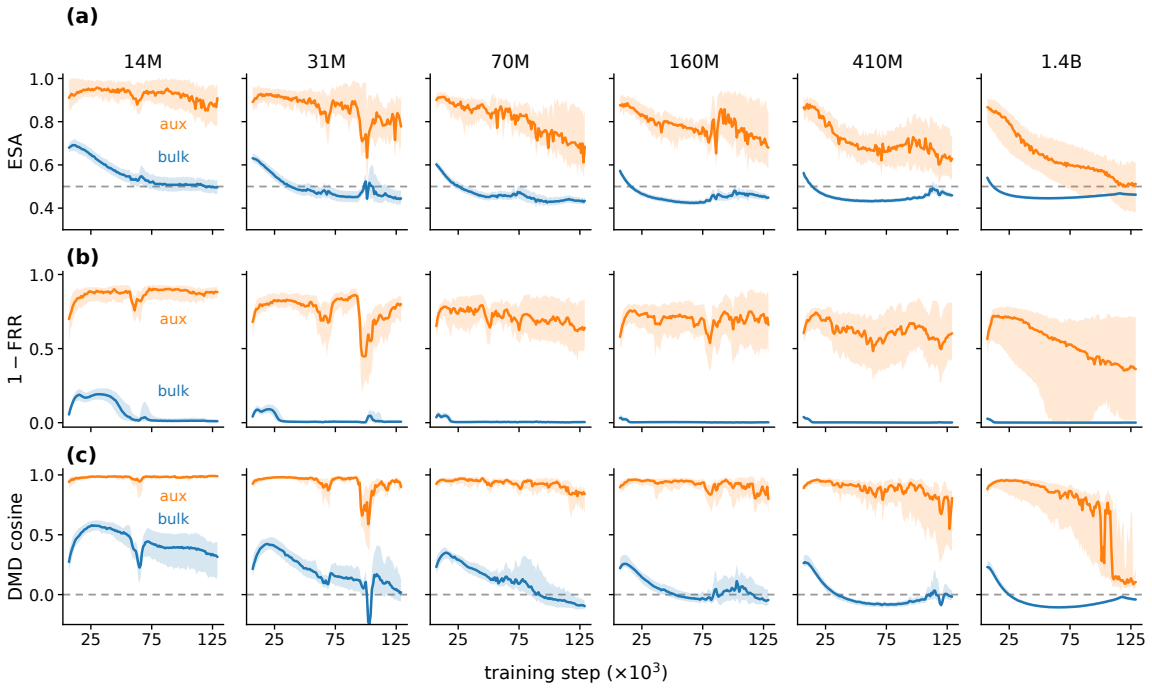


Figure B.29: Auxiliary and bulk coordinates stay separated at every size and every anchor. Rows: (a) ESA, (b) $1 - \mathrm{FRR}$, (c) DMD direction cosine; columns: the six sizes. The solid line is the group median over the pooled group set and the shaded band the interquartile range; orange is auxiliary, the vector segment, and blue is bulk. Grey dashed references mark 0.5 in (a) and 0 in (c). All panels use one readout protocol at τ=5 over the 123 shared anchors, and because the suite trains every size with the same 2M-token batch, a matched step grid is also a matched token grid. Horizontal comparisons between sizes are therefore at matched steps and tokens, and not at matched loss, compute, or training progress. The median separation holds at every anchor at all six sizes under all three probes; the smallest median gaps are 0.036, 0.351, and 0.115 for the three rows, all at 1.4B late in training. Interquartile ranges are disjoint at every anchor for (b) and (c) at all sizes, and for (a) at every size except 1.4B, where they overlap on 30% of anchors. Bulk is the strict matrix set, with both embedding segments dropped; under the looser alternative the bulk means move by at most 1.2×10^{-4} for (a) and (b) but by up to 6.3×10^{-3} for (c), which is why the strict rule is used throughout. Pythia-1B is excluded here and in every other cross-scale panel, on the three independent grounds of Table B.18(c).

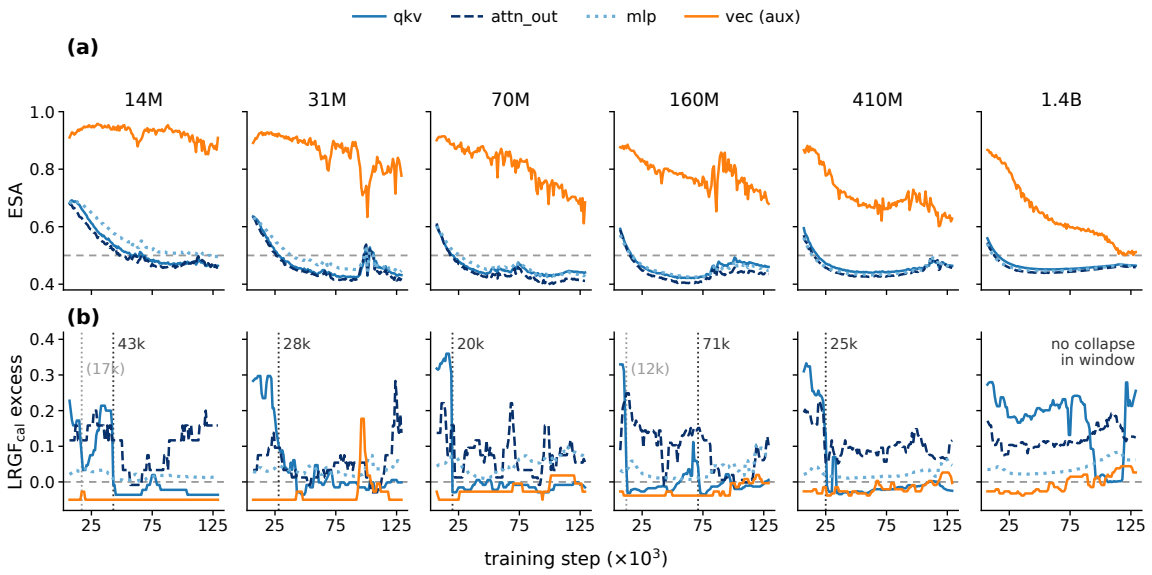


Figure B.30: Segment division of labour across sizes: ESA barely separates the matrix segments while the calibrated readout does. Columns are the six sizes. (a) ESA group median per segment. (b) pooled calibrated excess per segment, drawn as a five-anchor centred rolling median, with the onset markers computed on the unsmoothed series. The three matrix segments share a blue system (*qkv* solid, attention output dashed, MLP dotted) and the vector segment is orange; grey dashed references mark 0.5 in (a) and the null level 0 in (b), where the floor at -0.05 is the zero-fraction case. The dark dotted line is the sustained collapse onset of the *qkv* series, and where the series dips and recovers the first crossing is given in grey parentheses. Onsets are 20k (70M), 25k (410M), 28k (31M), 43k (14M), and 71k (160M), with no sustained collapse at 1.4B, so the ordering is not monotone in parameter count under either crossing. In (a) the three matrix segments stay within a median of 0.048, 0.038, 0.027, 0.021, 0.014, and 0.011 of one another in ascending size order, so the spread between matrix segments shrinks monotonically with size, while the vector segment stays above all of them by at least 0.034 at every anchor and size.

monotone in parameter count, under either of the two crossing conventions recorded in Table B.18.

This is where the two instruments must be kept apart. The collapse just described is read on the pooled calibrated lane. The null-line and row-level pocket onsets of Appendix B.4.2 are read on the row-level lane, and those do order with size. A size-ordering statement about this system is only meaningful once it names which of the two it is about.

The corresponding segment-resolved pocket populations, including the finding that the attention output projection carries the same wave as *qkv* at row granularity and that only the output half of the MLP carries signal, are reported in Appendix B.4.2 and are not repeated here.

Where the Calibrated Signal Survives

Figure B.31 resolves the *qkv* calibrated readout by layer and step at all six sizes on one colour scale, with white at the null level.

Early in training the signal is broad, covering 5/6, 6/6, 6/6, 10/12, 23/24, and 23/24 layers at the six sizes in ascending order. It then contracts onto a small persistent band whose absolute layer index moves deeper as the model deepens: in the middle phase the band is layer 5 at 14M and 31M, layer 4 at 70M, layers 8 to 11 at 160M, and layers 21 to 23 at 410M. In the tail phase nothing survives at 14M, while the band sits at layer 5 (31M), layer 4 (70M), layers 6 to 7 (160M), and layers 19 to 23 (410M). In relative depth the band sits in the deepest quarter of every model that contracts at all.

1.4B does not contract inside the window: 22 of its 24 layers are still above the null in the tail phase. It shows instead an extinction window near 96 to 118k in which the raw fraction drops to nearly zero in most layers, with only the deepest block holding 0.15 to 0.44, followed by a recovery from about 120k to the highest values of the run.

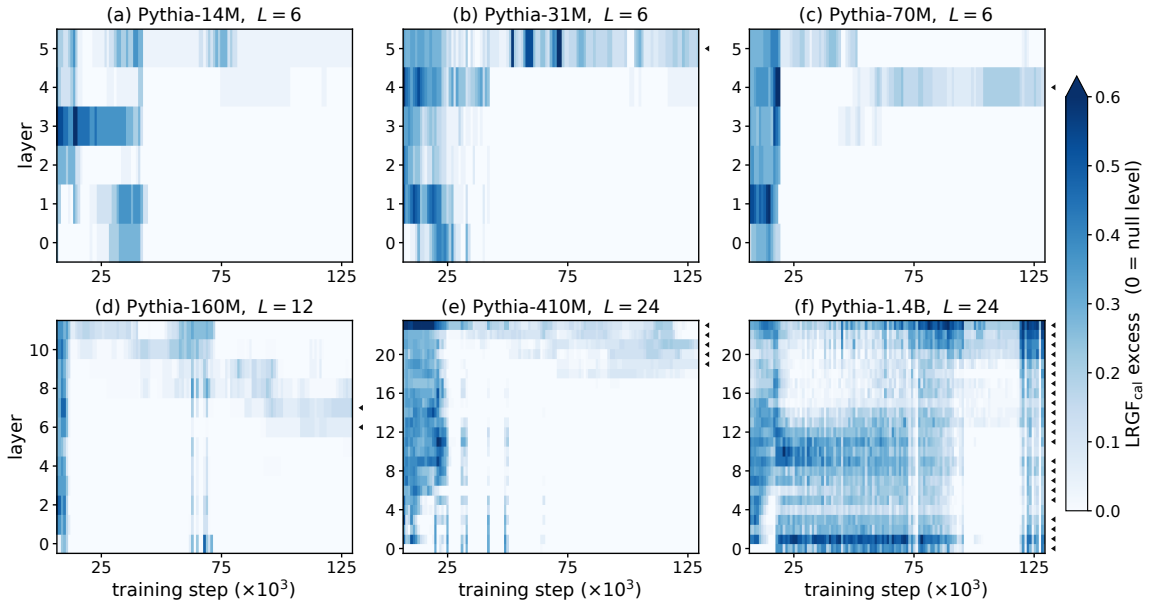


Figure B.31: Where the qkv calibrated signal survives: layer against step at six sizes on one colour scale. Each panel is one model, rows are transformer blocks with layer 0 at the bottom, and columns are the 123 shared anchors. The probe is reported as excess over its α=0.05 null level, so 0 is exactly the null and the raw pocket fraction is the excess plus 0.05; the colour ramp is anchored with white at 0, every cell at or below the null renders white, and the ramp is clipped at 0.60, which affects 0.21% of cells. Ticks on the right edge mark the layers whose tail-phase mean stays above the null. Early in training the signal covers 5/6, 6/6, 6/6, 10/12, 23/24, and 23/24 layers in ascending size order, and then contracts onto a persistent band whose absolute layer index moves deeper as the model deepens. 1.4B does not contract inside the window: 22 of 24 layers are still above the null in the tail phase, and it shows instead an extinction window near 96 to 118k followed by a recovery from about 120k. The calibrated readout is identical under both bulk rules, since the vector segment does not enter it.

Depth and width. Parameter count alone does not order these transitions. Among the sizes retained, the one contrast that survives holds depth fixed: at 24 layers the narrower 410M reaches its row-level pocket onset at 48k and its null-line onset at 92k, while the wider 1.4B has reached neither by 128k. Wider at equal depth is later. The complementary contrast, deeper at equal width, rested on the pair that the exclusion of 1B removes, so depth and width are confounded along the suite's own scaling ladder among the sizes retained. The depth half of a two-factor reading is therefore registered as unsupported here rather than asserted.

Co-location Registry for Late Events

This subsection records where the internal readouts and the public loss telemetry land relative to each other on the step axis. Nothing here is a directional claim.

At 70M the smoothed loss reaches its minimum at step 50,345 and rises afterwards, bulk ESA has a local rise peaking at 0.467 near 77k before descending to its global minimum 0.426 at 98k, and the row-level main wave peaks at 84k. At 160M the loss minimum is at 76,409, bulk ESA reaches its global minimum 0.424 at 68k and rises persistently from 78k to a plateau of 0.45 to 0.47, and the pocket peak is at 92k. On both sizes the loss minimum, the ESA turn, and the main pocket wave fall in the same stretch of training.

At 410M that coincidence does not hold, and this is the more informative case. The smoothed loss has no minimum inside the anchor domain: it is still descending at the right edge and remains flat or descending to the end of training. The ESA readout nevertheless shows the same late rise, recomputed on the full-protocol extended panel: bulk group-level ESA (τ=5, group p_{50}) sits on a 0.436–0.441 plateau from 60k to 85k, climbs steadily from about 85k, and peaks at 0.48–0.49 over 111 to 117k. The late ESA rise therefore occurs without a loss-bottoming event, and the co-location seen at the two smaller sizes should not be carried to larger sizes as if it were a general pairing.

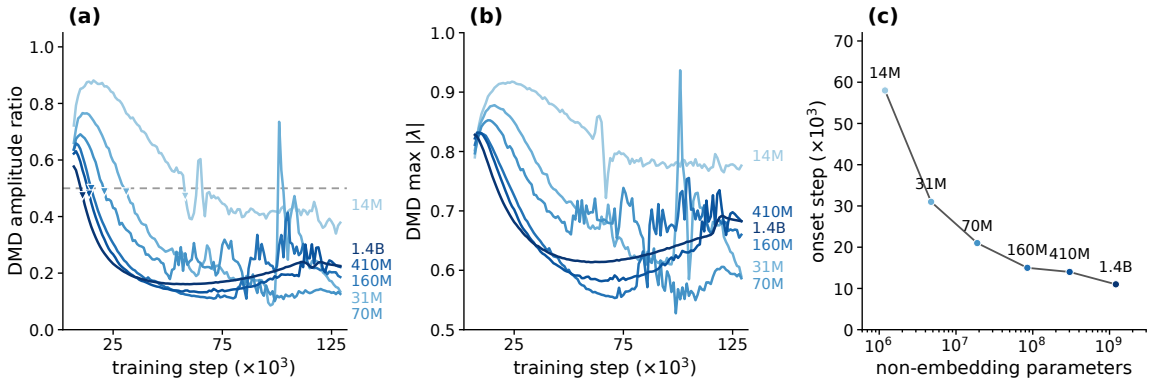


Figure B.32: Internal predictor-family diagnostics on the bulk set across six sizes. (a) median predicted-over-realized step amplitude of the fit; (b) median largest eigenvalue modulus of the fitted operator; (c) the onset step of the amplitude deficit against non-embedding parameter count on a log axis. Line shade encodes size on a single ramp and every curve is additionally labelled at its right end, so colour is never the only encoding. Onset is the first anchor opening five consecutive anchors whose median amplitude ratio is below 0.5, marked by the grey dashed reference in (a). The deficit sets in earlier the larger the model, at 58k, 31k, 21k, 15k, 14k, and 11k in ascending size order, which is the only cross-scale ordering in this appendix that is monotone in parameter count. By the last anchor the bulk median amplitude ratio has fallen to 0.13 to 0.38 at every size while the eigenvalue modulus settles into 0.59 to 0.68, with 14M the exception at 0.78.

One monotone ordering. Figure B.32 reports the internal diagnostics of the predictor-based family on the bulk set. The fitted operator's step amplitude falls progressively below the realized step, and the onset of that deficit is earlier the larger the model: 58k, 31k, 21k, 15k, 14k, and 11k across the six sizes in ascending order. This is the only cross-scale ordering in this appendix that is monotone in parameter count. By the last anchor the bulk median amplitude ratio has fallen to 0.13 to 0.38 at every size while the largest eigenvalue modulus settles into 0.59 to 0.68, with 14M the exception at 0.78: the fitted operator keeps a stable spectrum while the amplitude of the step it predicts falls well below the step actually taken.

The late cluster. Several unrelated readouts move inside a narrow late window. On 70M and 160M the auxiliary DMD amplitude ratio falls to exactly zero and stays there from anchors 118k and 117k respectively, while the bulk series over the same

anchors is unaffected. On 70M a secondary row-level wave appears near 115k. On 410M the ESA peak, the pocket peak at 116k, and an MLP output-half patch in the middle layers all fall in 110 to 120k. Whether this is one phenomenon or several is open. Two constraints are worth recording: the learning-rate schedule has no breakpoint anywhere near it, since the decay is a single cosine with no constant tail, and the auxiliary series that goes to zero does so exactly, which is the signature of a degenerate fit rather than of a small value.

Open Questions

Four items are registered as open rather than resolved.

The late cluster above may be one event or several, and the exact-zero auxiliary amplitude ratio is as consistent with an instrument boundary as with a property of the trajectory. Distinguishing the two requires a readout that does not go through the same fit.

1.4B sits at the edge of the observation window. Its row-level population reaches 3.0% at the last anchor and is still rising, and its calibrated band has not contracted. It is therefore a sample of a transition beginning at the window edge, not of a transition that did or did not happen, and it should not be read as a negative case.

The local 1.4B loss telemetry does not match the released 143,000-step cosine schedule, so no loss landmark is quoted for it and no co-location statement is made at that size.

Excluding 1B removes the only pair of sizes that share a width, so the depth leg of the depth and width reading has no support among the sizes retained. Recovering it would require either a fourth model at 24 layers and width 2048 or an assessment of how much the training-arithmetic difference actually moves these probes.

B.5 Additional Architecture-Conditioned Diagnostics

This section provides additional architecture-conditioned diagnostics supporting Section 4.3.5. They document negative mechanism checks, within-layer heterogeneity, role enrichment, persistence, phase alignment, and representative seed robustness under their stated measurement conventions. In view of Appendix B.2.6, these figures are not interpreted as a grouping-invariant architecture ranking; each result travels with the cut and probe used to produce it.

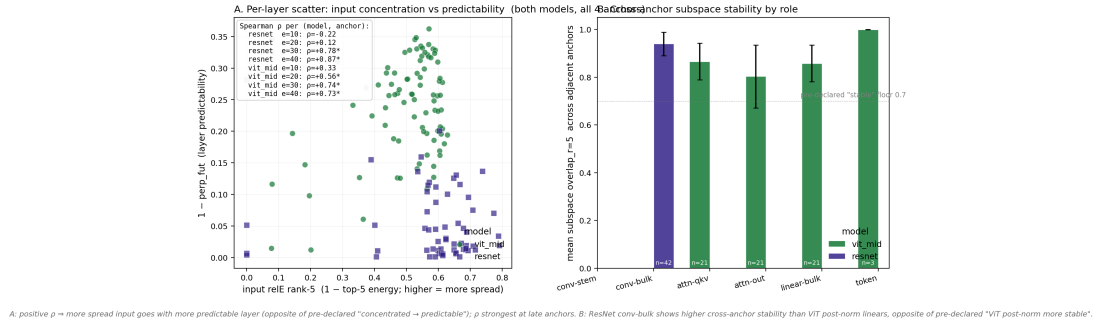


Figure B.33: Input-geometry negative diagnostic. The pre-declared hypothesis that whole-layer input concentration or global input-subspace stability explains predictable pockets is not supported. These quantities measure spatial representation geometry, whereas the architecture-conditioned pocket analysis concerns local temporal update pockets.

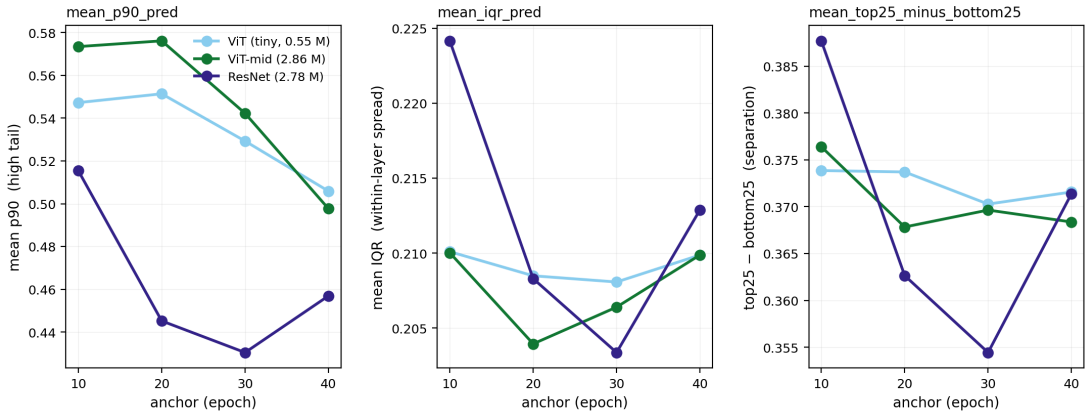


Figure B.34: Within-layer group heterogeneity details. ViT-mid retains a higher high-predictability tail, while spread metrics such as IQR and top-quartile minus bottom-quartile separation are comparable across architectures. This supports a level and persistence interpretation rather than an overclaim that ViT has uniformly larger within-layer separation.

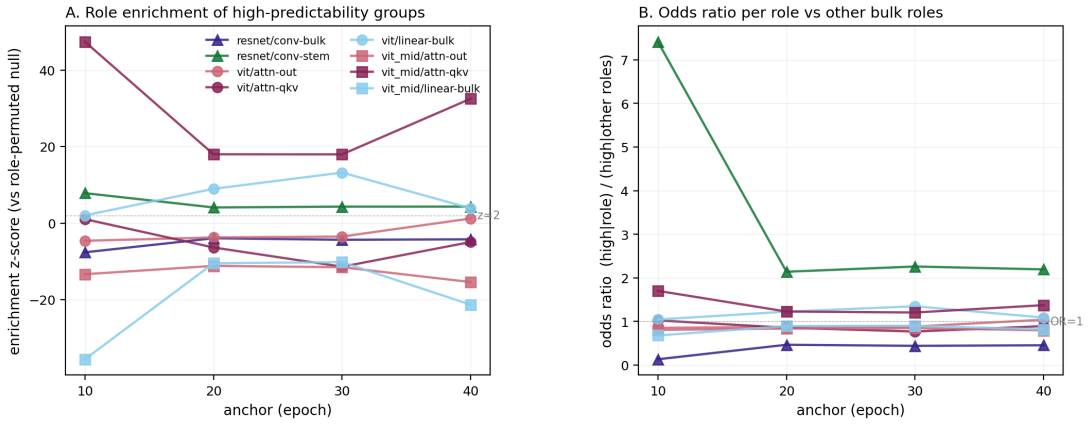


Figure B.35: Full role-enrichment diagnostics. We report role-permutation z-scores and odds ratios for high-predictability groups. Large z-scores reflect both effect size and group count; odds ratios provide a complementary effect-size view. Negative enrichment indicates that high-predictability groups are under-represented in that role relative to other bulk roles.

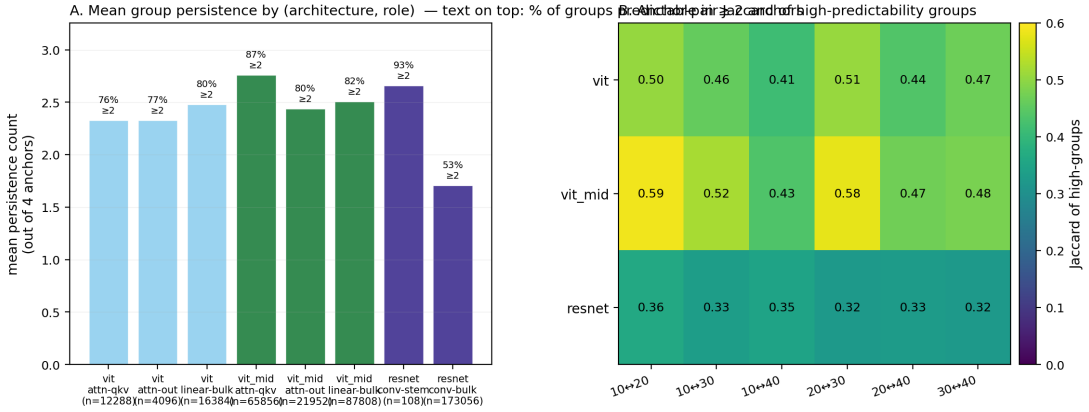


Figure B.36: Pocket identity persistence and anchor-pair Jaccard details. The bar plot expands the role-level persistence summary, and the heatmap reports Jaccard overlap of high-predictability group sets across anchor pairs. The same groups reappear more consistently in ViT-like models than in the large ResNet convolutional bulk.

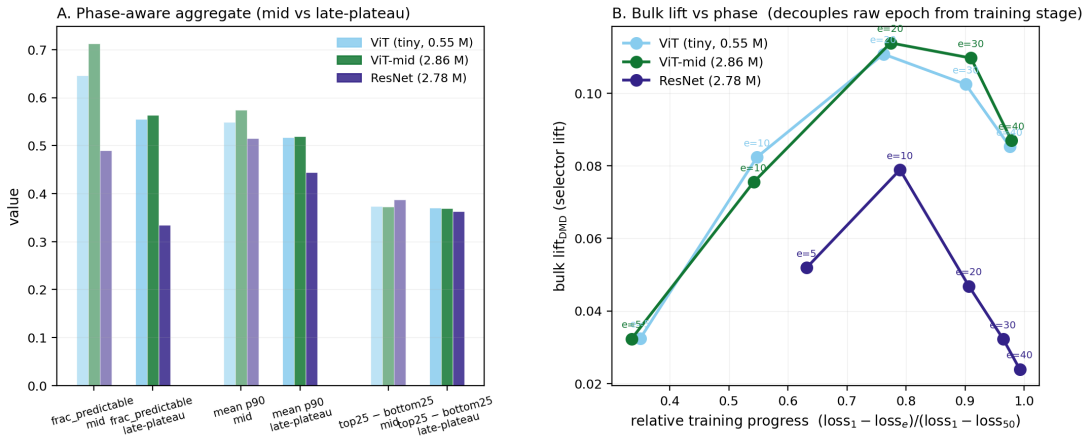


Figure B.37: Phase-aware architecture comparison. The left panel summarizes mid and late-plateau phase statistics; the right panel plots bulk selector lift against relative training progress. This diagnostic addresses the concern that equal raw epochs may correspond to different optimization phases across architectures.

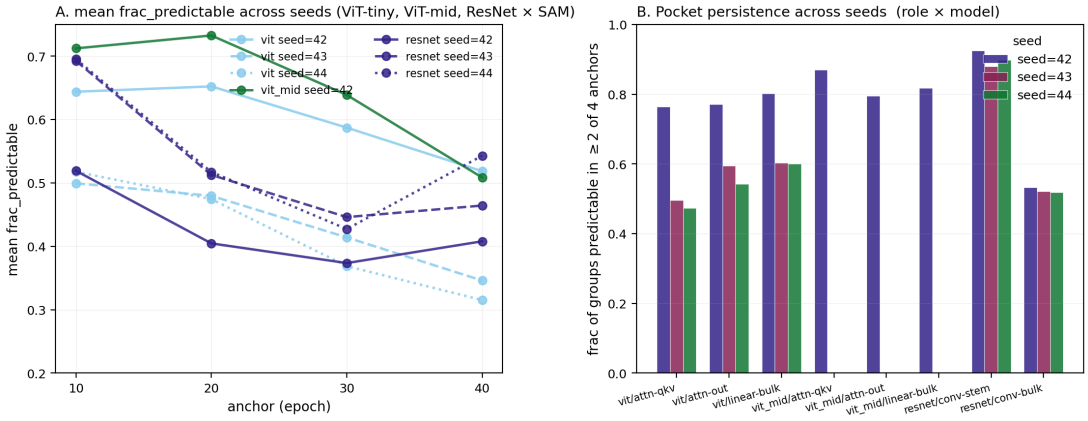


Figure B.38: Representative multi-seed architecture check. ResNet/SAM is highly seed-stable in this subset, while ViT-tiny/SAM shows seed-dependent absolute pocket mass. The role-enrichment and persistence directions remain consistent in the tested seeds. This is a representative check, not an exhaustive multi-seed sweep.

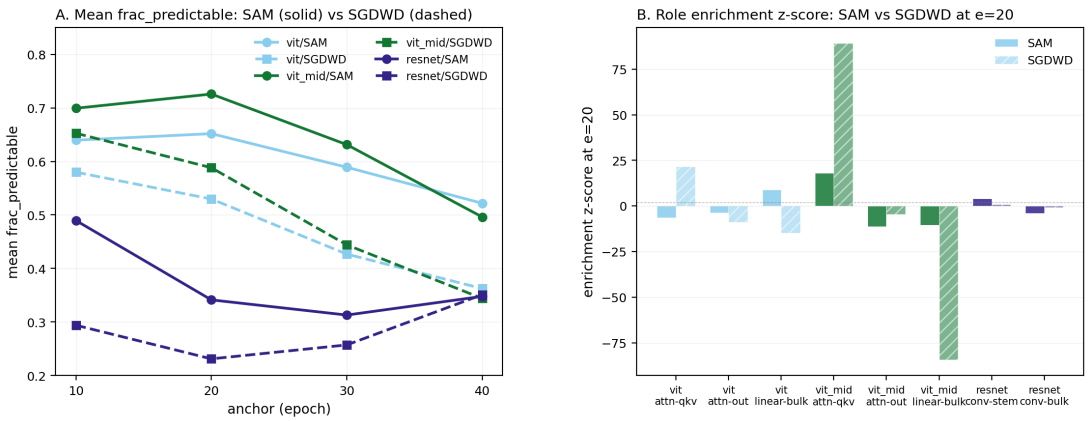


Figure B.39: Auxiliary SGD-wd replication for architecture-conditioned pockets. The architecture-associated role-pocket pattern also appears under SGD-wd, with especially strong attention-`qkv` enrichment in ViT-mid. We treat this as auxiliary robustness for architecture-conditioned predictability and reserve optimizer-mechanism interpretation for the optimizer analysis.

B.6 Additional Optimizer-Conditioned Diagnostics

This section provides supplementary diagnostics for the optimizer-conditioned predictability analysis in Chapter 4. These figures explain the optimizer-conditioned profiles; they are not intended to create a universal optimizer ranking.

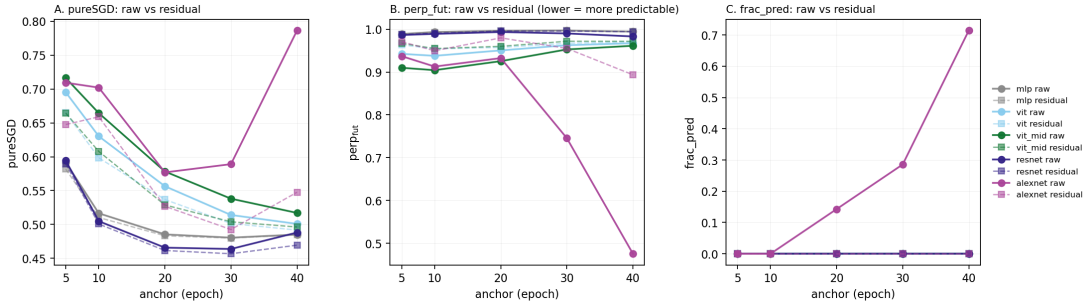


Figure B.40: Full weight-decay residual diagnostic for SGD-wd cells. The solid curves report raw update metrics, while dashed curves remove the instantaneous projection of each update onto $-w_t$. AlexNet/SGD-wd is the strongest shrinkage-drift case: late-anchor `pure_sgd` remains high in the raw update, FRR becomes low, and the fraction of predictable groups rises sharply, but these effects collapse after the shrinkage projection is removed. These residual metrics are diagnostic decompositions, not counterfactual optimizer trajectories.

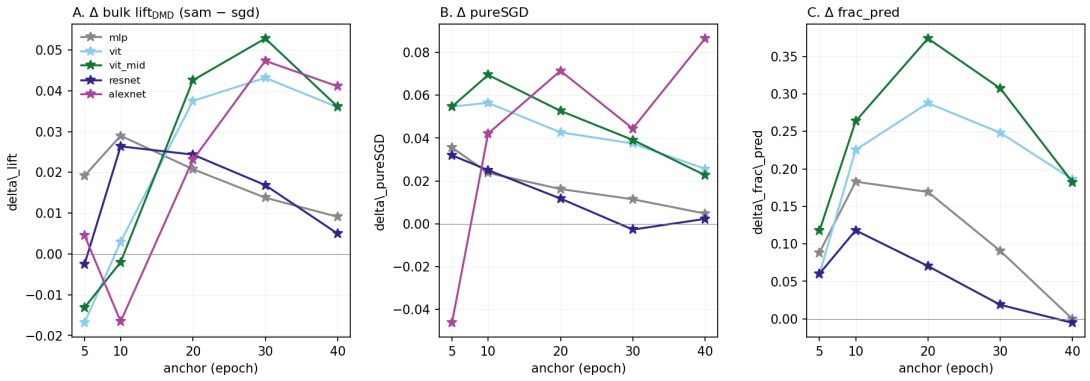


Figure B.41: SAM versus SGD across architectures and anchors. The curves show SAM minus SGD for bulk DMD-selector lift, bulk `pure_sgd`, and bulk fraction of predictable groups. SAM amplifies predictable structure most clearly in ViT-class models, while the effect is weaker or more transient in other architectures. We use this as architecture-dependent optimizer modulation, not as evidence that SAM is universally more predictable.

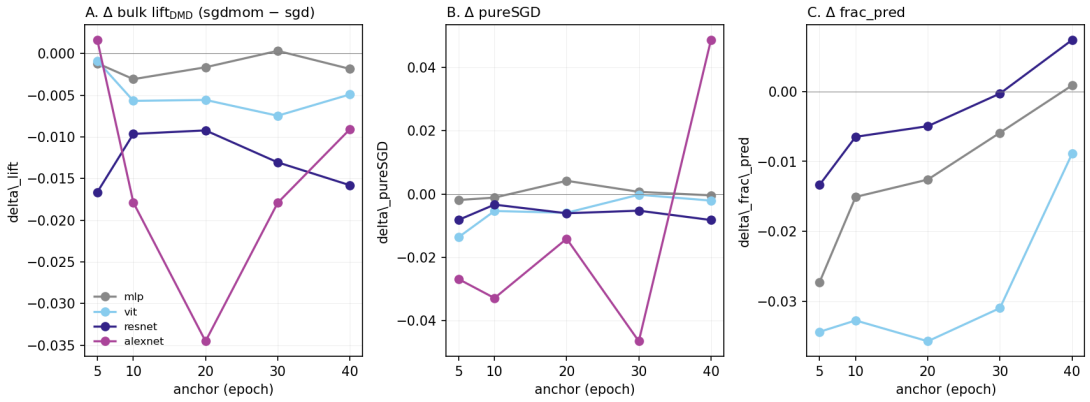


Figure B.42: SGD-momentum versus SGD. Momentum is a natural smoothing mechanism, but under the current bulk probes it does not consistently increase DMD-selector lift or predictable-group mass relative to SGD. We retain this as a cautious supplement because momentum effects are protocol- and optimizer-state-dependent.

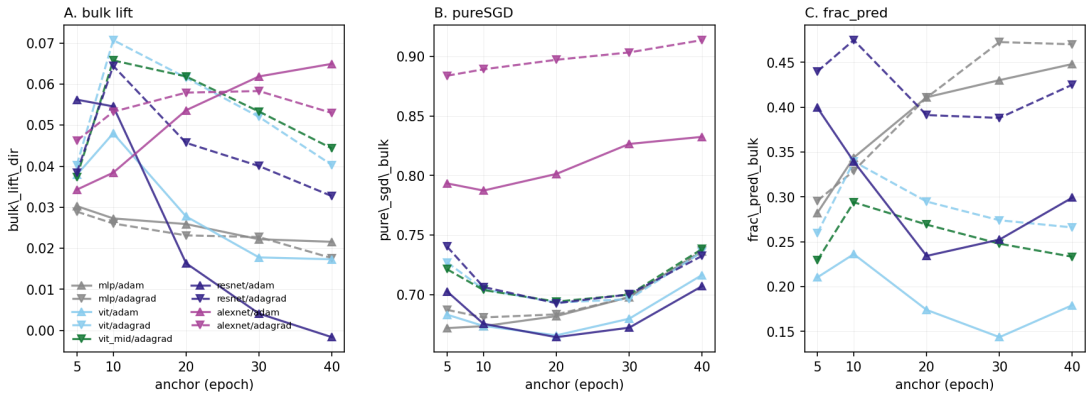


Figure B.43: Adaptive optimizer profiles for Adam and AdaGrad. These plots summarize bulk lift, `pure_sgd`, and predictable-group fraction over anchors. The profiles are architecture-dependent, so we do not use them as a universal adaptive-optimizer mechanism.

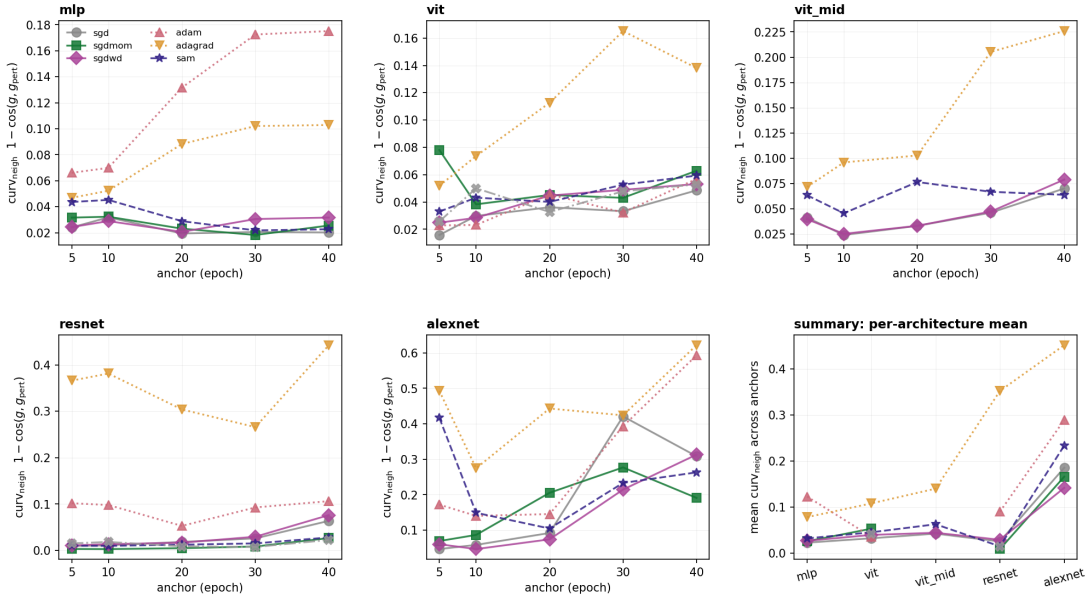


Figure B.44: Neighborhood-curvature diagnostic for adaptive optimizers and other baselines. The metric is $1 - \cos(g(w), g(w + \rho\nabla L/\|\nabla L\|))$ with $\rho = 0.05$ on fixed batches. AdaGrad is an outlier in several architectures. This diagnostic is correlational, not causal. For ResNet and AlexNet, batch-normalization statistics are evaluated in train mode and can shift between the two gradient evaluations, so the metric is partially confounded with the BN-statistic Jacobian. ViT, ViT-mid, and MLP have no batch normalization.

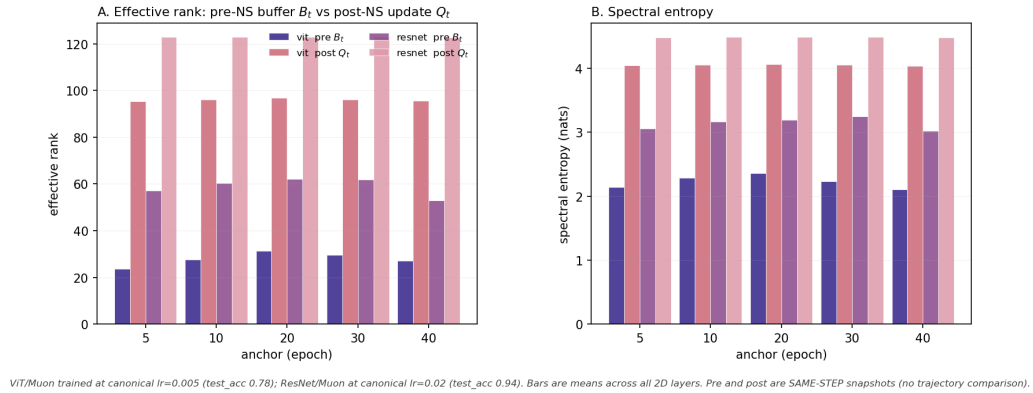


Figure B.45: Muon same-step spectrum sanity check. The pre-Newton–Schulz matrix buffer B_t and post-Newton–Schulz update Q_t are compared at the same optimizer step. The post-Newton–Schulz update has higher effective rank and spectral entropy, as expected from the orthogonalized matrix update. This is a same-step diagnostic only; it is not a pre-trajectory versus post-trajectory comparison.

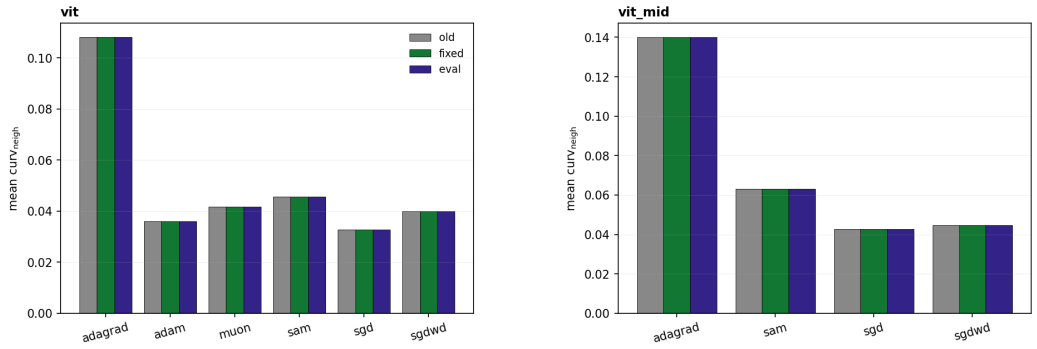


Figure B.46: ViT and ViT-mid neighborhood-curvature sanity check. The original, RNG-fixed, and evaluation-mode versions match because the TinyViT implementation used here has no `nn.Dropout`. This diagnostic is retained only to document that the ViT-class neighborhood-curvature values are not inflated by dropout-mask noise.

Vita

Fanqi Wang received the B.E. degree in Communication Engineering from Xidian University in 2012 and the M.E. degree in Electronics and Communication Engineering from the University of Chinese Academy of Sciences in 2015. She then pursued the Doctor of Philosophy degree in the Department of Electrical Engineering and Computer Science at The University of Tennessee, Knoxville, under the supervision of Dr. Hairong Qi. During her doctoral studies, she was a member of the AICIP Laboratory. Her research included biomedical signal analysis, remote-sensing visual analysis, and deep learning, with her dissertation focusing on predictive training, neural training dynamics, and measurement-guided methods for understanding and accelerating deep neural network optimization.