IS YOUR LLM A SEQUENCE MODEL ON THE TRAINING HISTORY? THE ORIGINS AND CONSEQUENCES OF ANTICIPATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent work showed that, in cyclic fine-tuning settings, Transformers exhibit striking *anticipatory recovery*: the loss on an upcoming document decreases *before* that document is revisited, even though models hold no explicit memory of training history. Despite early efforts to explain this behavior, our understanding of anticipation remains limited. We first show that anticipation is broader than previously demonstrated: beyond individual-documents, Transformers anticipate upcoming data domains, such as English and French, and the behavior extends to pre-training, even to models 5000 times smaller. Second, we show that anticipation can be interpreted as sequence model-like behavior. Anticipating the next domain leads to lower training loss, and this loss advantage realizes a substantial fraction of that achievable by sequence models that explicitly learn class order. To a limited extent, anticipation transfers even to second-order structures, where the next domain can only be predicted from the previous two. Third, we offer a potential explanation of anticipation as gradient alignment developing through an implicit bias of optimization in cyclically ordered training. The resulting picture helps account for the surprising optimizer dependence of anticipation: at matched held-out loss, AdamW develops strong anticipation, while Muon does not anticipate. Finally, we show that structured training order can affect what the model learns, and can act as a source of representational bias during optimization.

1 INTRODUCTION

Modern language models are trained under the assumption that the order in which training documents are presented (the training order or training history) does not matter. Yet recent work suggests that Transformers (Vaswani et al., 2017) can retain information about the training order in their activations despite having no explicit memory of past updates (Krasheninnikov et al., 2026). Yang et al. (2024) further showed that, in cyclic fine-tuning, where documents are revisited in deterministic cycles, models exhibit striking anticipatory recovery: the loss on a document decreases *before* that document is revisited, indicating that the model anticipates it.

The phenomenon of anticipation is surprising because it is consistent with behavior we expect of *sequence models* over the training stream: models that use information about previous training states to predict what comes next. At first sight, this behavior resembles a model exploiting temporal information about its own training trajectory. Such behavior is broadly relevant to questions of *training process awareness* or situational awareness (Berglund et al., 2023): if models can infer or exploit regularities in when and how they are trained, these regularities may constitute an additional, potentially unintended information channel. This motivates us to understand the extent and origins of anticipation.

To understand how far this behavior extends beyond repeated examples, we move from individual documents to *data domains*, which we will also refer to as classes. This lets us ask whether models can become preferentially adapted not merely to a particular upcoming example, but to an upcoming data distribution. Concretely, we arrange mixture data in class-homogeneous batches, and train on each class for several steps before moving to another class, hereby creating class bursts. Similar forms of predictable temporal structure naturally arise in continual learning (Chen & Liu, 2018; Kirkpatrick et al., 2017; McCloskey & Cohen, 1989), curriculum learning (Bengio et al., 2009), and reinforcement learning (Sutton & Barto, 2018). This setting allows us to reproduce anticipation even in 5000 times smaller models. Our contributions are:

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

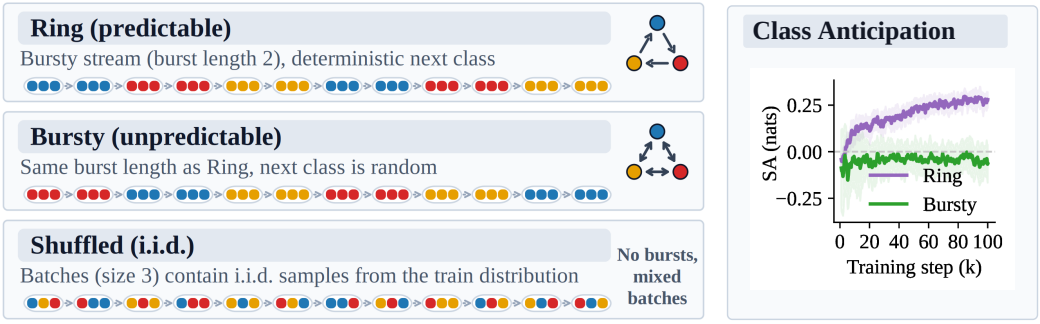


Figure 1: **Surprising class anticipation during training.** *(Left) Experimental setup.* In Ring and Bursty settings, the model is trained in bursts – multiple updates in a row on data from the same single class – meaning the training history is not i.i.d. Transition structure between bursts is defined by a Markov chain (see transition diagrams on the right). Colors encode classes, dots make up batches (groups) that make up the training streams. We show 3 classes and a burst length of 2 for readability, experiments use 10 classes and geometric bursts with mean 32. *(Right) Anticipation measured with Successor Advantage (SA) over time*. Roughly, SA measures how much the eval loss on the actual successor class is lower than that on the other non-current classes. In the Ring training regime SA keeps increasing, meaning the **model learns to anticipate which class is coming next** before being trained on that class, despite no explicit memory of the training stream.

1. **Anticipation extends beyond repeated examples.** We show that Transformers trained on structured, bursty data streams arranged in predictable orders anticipate upcoming data domains, preferentially adapting to the next class before observing it. This effect arises in pre-training and at substantially smaller model scales than shown in prior work (Section 2, Section 5).
2. **Anticipation yields a training-loss advantage and exhibits sequence-model-like behavior.** We show that predictable training order can lower training loss, and that Transformers realize a substantial fraction of the advantage available under explicit class conditioning. Moreover, models exhibit weaker but significant anticipation under second-order transition structure, where the next class cannot be predicted from the current class alone. (Section 3).
3. **We identify an optimization bias that promotes anticipation.** Our analysis of bursty mini-batch training shows that deterministic class transitions align gradients of temporally adjacent classes. We connect this mechanism to anticipation and use it to study its strong dependence on optimizer geometry, in particular why AdamW strongly anticipates while Muon does not. We also use our theory to disable anticipation via explicit regularization (Section 4).
4. **Training order can reshape learned relationships between data distributions.** We show that cyclic order alters relative gradient relationships between classes, sometimes overriding their a priori similarities. This suggests that training order itself can act as a source of inductive bias during optimization. (Section 6)

2 Experimental setup and metrics

Prior work studied anticipatory recovery in cyclic document fine-tuning (Yang et al., 2024). We instead study bursty pre-training streams, where class distributions repeat and recur in a structured way. We use *anticipation* to refer to the general phenomenon, *anticipatory recovery* (Yang et al., 2024) for the metric used in prior work, and *Successor Advantage* (SA) as our primary metric.

Training order versions. We focus on a type of structured training schedule we refer to as "bursty training". This means organizing the training stream into short sequences, "bursts" of same-class batches. The length of each burst is geometrically distributed. Within a bursty training, we can organize the classes cyclically. We refer to this as the `Ring` configuration, which, after the burst ends, deterministically transitions to the (unique) successor class. In the `Bursty` configuration, the order of classes in bursty training is unpredictable. `Shuffled` is our fully i.i.d. baseline with mixed-class batches (and no bursts). Figure 1 (left) illustrates our setup. More details are in Section B.4.

While the anticipatory recovery metric measures the loss trajectory of a repeated document, we seek a quantity that emphasizes adaptation to the structure of the training stream itself. We therefore define

Successor Advantage (SA), which measures whether the model assigns preferentially lower loss to the upcoming class relative to other non-current classes.

Definition 1 (Successor Advantage (SA)). *Let $\mathcal{D}_{eval}$ be a held-out dataset drawn i.i.d. from $p(x) = \sum_{c \in C} p(x \mid c)p(c)$, let $R := |C|$ be the number of classes with $R \geq 3$, and let $\mathcal{D}_c \subset \mathcal{D}_{eval}$ be the examples of class c, with empirical class loss $\widehat{L}_c(\theta) = \frac{1}{|\mathcal{D}_c|} \sum_{x \in \mathcal{D}_c} \ell(\theta, x)$. At step t of training, let c_t be the class whose batch the model has most recently been trained on, and n_t the upcoming class after the current one. The successor advantage at step t is*

$$\mathrm{SA}_t = \frac{1}{R-2} \sum_{k \notin \{c_t, n_t\}} \widehat{L}_k(\theta_t) - \widehat{L}_{n_t}(\theta_t).$$

Experiments use a corrected version of SA to remove effects of differing class difficulty (Section B).

Experiments. Our main experiments use a ∼20M parameter decoder-only Transformer (Vaswani et al., 2017) model trained with AdamW and the standard next-token prediction objective for 100k steps on 10-language Wikipedia data. Batch size is 64, and there are on average 32 gradient steps per class burst (so on average each burst has 2048 samples). Unless stated otherwise, the context length is 8 tokens (excluding BOS and EOS) and class orderings vary with the random seed. See Section B for details and explanation. Figure 1 (Right) shows increasing SA in `Ring` training, whereas `Bursty` (unpredictable) does not anticipate. Figure 8 in Section A in the Appendix shows that high SA correlates with increased anticipatory recovery.

3 Anticipation decreases the training loss

Positive SA in Figure 1 indicates that the model is preferentially adapted to the future training distribution, rather than just tracking the class currently being optimized. However, positive SA does not by itself explain why optimization should produce anticipation. It could simply arise as an incidental consequence of cyclic training or representational overlap. The following proposition shows that, under bursty training, adaptation to the future training distribution can provide a direct optimization advantage.

Proposition 1. *Let T denote the sequence length, assume that classes have uniform marginals ($p(c) = \frac{1}{R}$) and that the class c is identifiable from the prefix, i.e. $p(c|x) \approx 1$. The best possible average loss improvement over `Shuffled` training via anticipation is $\log R/T$. This is achieved by sequence models explicitly modeling class order, i.e. $p_{\text{sequence model}}(x) = p(x \mid c : x \in c)$.*

This result reframes anticipation as more than a curious training artifact: *model are incentivized to anticipate, because this allows them to decrease the training loss further*. The proposition also gives us a way to test to what extent do models behave like sequence models. The proof is in Section C.1.

Comparison to sequence models. Figure 2 shows that `Ring` training realizes about 30% of the possible anticipatory loss advantage (the difference between the `Shuffled` and Theoretical limit), achievable by sequence models explicitly learning class order. To test whether the rest of possible anticipatory loss advantage is absent due to limitations of our empirical setup (such as model size, training horizon, etc.), we estimated an *empirical limit* of anticipation, computed as follows: during training, models received the true class index prepended to each training sample, artificially "conditioning" the model on the class in-context. These class cue tokens were excluded from training loss computation. The experimental setup was otherwise identical. The training loss of this oracle model was taken as an empirical estimate of the best possible anticipatory loss advantage achievable with the specific model architecture and training horizon we considered. Our transformer realized 40% of this advantage. Details are in Section D.1. We note that anticipatory advantage on the training loss comes at the expense of larger validation loss. This is not surprising, as models that anticipate adapted themselves heavily to the training history, while the validation distribution is a mixture, which creates a distribution shift for these models.

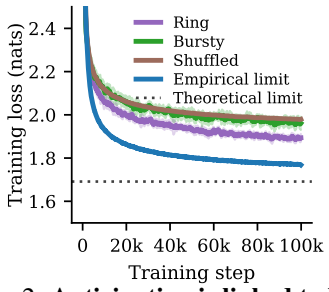


Figure 2: **Anticipation is linked to lower training loss.** Lower than "Shuffled" indicates sequence model-like behavior.

Anticipation replicates with second-order transitions (Section D.3). To further support the sequence model interpretation, we showcase anticipation, albeit smaller-scale, in second-order settings – where the next class is inferrable only from the pair (previous class, current class).

The transition table is a Latin square. We test a 5-class (Figure 3) and 6-class version (Section D.3),which have 25 and 36 states, respectively. For any fixed current class, each possible next class appears exactly once across predecessor classes. This makes the first-order marginal uninformative: knowing only the current class gives a uniform distribution over next classes. A model can do better than chance only if training has left it with usable information about the previous class too. Averaged across 16 seeds, corrected SA is positive: $+0.0412 \pm 0.0081$, demonstrating small but significant anticipation.

The above experiment demonstrates that transformers can partially behave like sequence models over the training stream even beyond simple first-order transition structure.

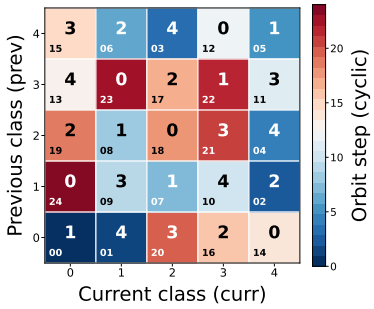


Figure 3: **Latin-squared 2nd order transition function.** x and y axis labels show the individual state index. Colors and small-sized numbers in each cell show the index of state pairs (previous, current). Large-sized digits denote the next class from that cell pair. The orbit starts from the bottom left.

4 Anticipation as the implicit bias of bursty mini-batch training

Section 2 established that Transformers can utilize predictable temporal patterns in their training stream, similarly to sequence models. We next ask how gradient-based training gives rise to this behavior. A key clue is anticipation's strong optimizer dependence (Table 6): under the same `Ring` schedule, AdamW anticipates strongly, whereas SGD and Muon do not. In Section 4.1, we show that ordered bursty updates induce a local bias toward aligning the gradients of temporally adjacent classes. Whether this bias translates into anticipation depends on the geometry of the optimizer update. In Sections 4.2 and 4.3, we validate our theory with experimental results on the optimiser-dependence of anticipation. We also use our theory to switch off anticipation via explicit regularization.

Optimizer	$\mathrm{SA}^{\pm\mathrm{std}}$
SGD	$-0.0096^{\pm 0.0026}$
AdamW	$+\mathbf{0.2730}^{\pm \mathbf{0.0194}}$
AdamW (no momentum)	$+\mathbf{0.1846}^{\pm \mathbf{0.0170}}$
Muon	$-0.0104^{\pm 0.0417}$

Table 1: **The effect of the optimizer on anticipation.** In this training setup, only AdamW anticipates (positive SA). Shuffle-corrected SA is averaged over evaluations in the last 25% of training and reported as mean $\pm$ across-seed standard deviation (12 seeds per setting). A more detailed comparison is in Table 6 in the Appendix.

4.1 Implicit bias theory

To describe how anticipation emerges, we analyze the training dynamics of bursty training. Our results are valid for SGD and optimizers with fixed preconditioner (including an approximation of Adam), and provide informal insights for general optimizers, such as Muon.

Why study the implicit bias? Our analysis builds on a view of SGD in which stochastic mini-batch updates are compared to full-batch gradient descent on the same data. The difference between the two trajectories can contain systematic drift terms. In the standard i.i.d. setting, such drift terms have been called implicit biases or implicit regularizers, and were used in arguments as to why SGD generalizes (Roberts, 2021b; Smith et al., 2021a). Explicit regularization with these drift terms even demonstrated some control on generalization (Geiping et al., 2022). We ask what this decomposition becomes when the mini-batches are no longer i.i.d., but instead arrive in class-homogeneous bursts.

We identify a local implicit bias of bursty mini-batch training that promotes gradient alignment between temporally adjacent classes. We then show that the `Ring` schedule turns this into a systematic pressure toward successor-specific alignment, and connect this geometry to changes in successor advantage under an optimizer update. We summarize the results here. Precise statements, assumptions, and proofs are given in Section C.2.

Result 1: bursty SGD updates promote gradient alignment (Theorem C.1). Consider two consecutive mini-batches drawn from classes c and $n = s(c)$ ("current" and "next", where $s(c)$ stands for the successor of class c), with population gradients $G^{(c)}$ and $G^{(n)}$ and Hessians $H^{(c)}$ and $H^{(n)}$. We compare two sequential SGD mini-batch updates with two full-batch GD updates in the style of Roberts (2021b). This reveals that in bursty training, SGD drifts from the GD trajectory in a way described by three leading-order drift terms:

$$T_1 \propto -\nabla \operatorname{tr} \Sigma_{within}, \qquad T_2 \propto -\nabla \left\| G^{(c)} - G^{(n)} \right\|^2, \qquad T_3 \propto -\left(H^{(c)} G^{(n)} - H^{(n)} G^{(c)} \right). \tag{1}$$

T_1 is a gradient variance term previously identified as the implicit bias of i.i.d. SGD, contributing to generalisation (Roberts, 2021a). T_3 is an order-sensitive term describing *curvature interaction*, and is closely related to implicit biases previously studied in sequential and continual learning (Dherin, 2023). Its effect on anticipation is unclear. The most interesting and important ingredient for anticipation is T_2: it can be interpreted as an *alignment pressure*, pushing the gradients of consecutive classes together. Our theorem applies to SGD, but an **analogous decomposition holds for optimizers with a fixed preconditioner (see Section C.2).**

To quantify whether this alignment is specifically directed toward the successor in the `Ring`, define the successor-contrast vector $w_c(\theta) := G^{(s(c))} - \frac{1}{R-2} \sum_{k \notin \{c, s(c)\}} G^{(k)}$, and the *Successor-Gradient Alignment* (SGA):

$$\mathrm{SGA}(\theta) := \frac{1}{R} \sum_{c \in C} \left\langle w_c(\theta), G^{(c)}(\theta) \right\rangle. \tag{2}$$

SGA measures how aligned a class gradient is to its successor versus to the other non-current classes.

Result 2: predictable order can create SGA in a Ring shape (Theorem C.2) The local alignment effect above occurs for any transition $c \to n$. What distinguishes `Ring` from `Bursty` is what transitions are possible at each step. Averaging the local drift over the two transition distributions shows that the T_2 contribution of Ring relative to Bursty points in the direction of increasing SGA. The corresponding T_3 contribution in general cannot be related to anticipation, and may reinforce or oppose T_2.

Hence cyclic ordering supplies a systematic pressure toward aligning each class with its successor, while curvature can interfere with the accumulation of this structure. This already suggests a route by which optimizer geometry can affect anticipation.

Result 3: gradient alignment increases anticipation (Theorem C.3) Finally, we connect successor-gradient alignment (SGA) to our measure of anticipation, the successor advantage (SA). Let $\mathrm{SA}_c(\theta)$ denote the population analogue of the SA (Definition 1), for a model currently trained on class c. For an optimizer whose update on class c is $\theta_c^+ = \theta - \eta U_c(\theta)$, the first-order change in successor advantage is

$$\mathrm{SA}_c(\theta_c^+) - \mathrm{SA}_c(\theta) = \eta \left\langle w_c(\theta), U_c(\theta) \right\rangle + O(\eta^2). \tag{3}$$

This motivates the *effective SGA* $\mathrm{SGA}_{\mathrm{eff}}(\theta) := \frac{1}{R} \sum_c \left\langle w_c(\theta), U_c(\theta) \right\rangle$, which measures successor alignment in the direction of the optimizer's actual update. For SGD, $\mathrm{SGA}_{\mathrm{eff}} = \mathrm{SGA}$. For fixed preconditioning, $\mathrm{SGA}_{\mathrm{eff}}$ reduces to SGA measured in the corresponding preconditioned geometry.

Interpretation. Our implicit bias decomposition (Theorem C.1) connects three perspectives: (1) the classical implicit-regularization view of SGD through T_1, (2) sequential interference in continual learning through T_3, and (3) anticipation through the *alignment pressure* term T_2. Together, our results suggest the following mechanism of anticipation. Bursty sequential updates induce a local pressure to align temporally adjacent class gradients (Theorem C.1, Theorem C.2). The predictable `Ring` schedule directs this pressure consistently toward each class's successor. While we do not have

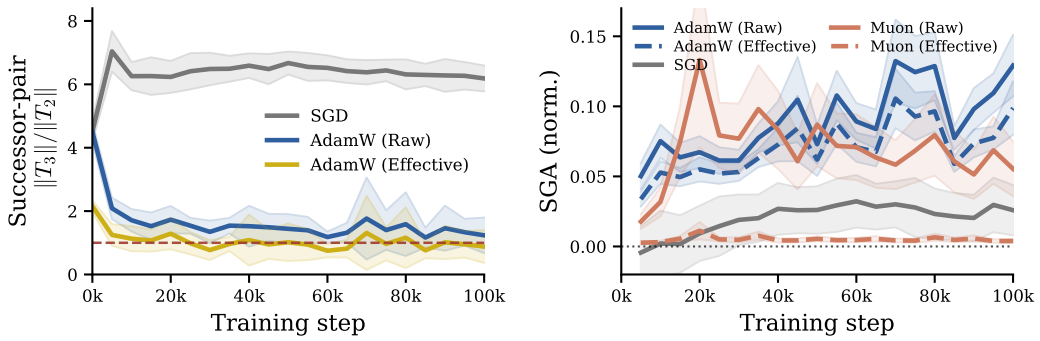


Figure 5: **Gradient geometry of optimizers.** *(Left) AdamW enables anticipation through persistent second moment scaling.* AdamW enters a low interference regime, as demonstrated by low $\|T_3\|/\|T_2\|$ ratio and high SGA. Preconditioning further reduces the ratio. In SGD, T_3 interferes. *(Right) Muon's update direction suppresses high raw SGA.* Results are averaged over 12 seeds.

formal results that prove accumulation of this local bias over training, we supply an informal argument in Remark 1. Nevertheless, Theorem C.3 tells us that **this geometry can produce anticipation only insofar as it is preserved by the optimizer's update direction.** We test these predictions across optimizers in the next section.

4.2 Optimizer dependence through gradient geometry

In this section, we validate our theory with experimental results on the optimizer-dependence of anticipation, focusing first on SGD and AdamW, and then on Muon.

SGD and AdamW. For SGD, Theorem C.1 and Theorem C.2 apply directly. They identify two competing contributions to the anticipatory implicit bias: the *alignment pressure* term T_2 and the *curvature interaction* T_3. Since T_2 provides the successor-alignment pressure while T_3 has no definite effect, whether anticipation becomes prominent depends in part on their relative magnitude.

AdamW is not exactly covered by our theory, since its preconditioner changes over training and it normally includes first-moment momentum. However, our experiments suggest that **the fixed-preconditioner approximation of AdamW captures the regime relevant for anticipation**. Figure 4 shows that anticipation increases as β_2 grows and saturates in the long-timescale regime around $\beta_2 = 0.999$, where the preconditioner changes slowly. In addition, substantial anticipation remains when $\beta_1 = 0$ (no momentum). This motivates approximating AdamW locally by a fixed preconditioned update $\theta \leftarrow \theta - \eta P g$, for which the corresponding terms are $T_{2,P}$ and $T_{3,P}$.

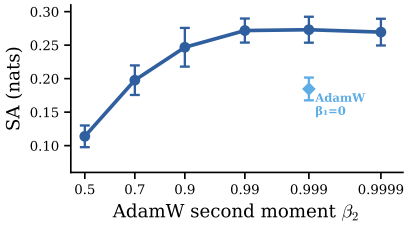


Figure 4: Ablating β_1 (momentum) and β_2 in AdamW motivates our fixed preconditioner approximation to AdamW

The resulting geometry differs markedly between SGD and AdamW. Figure 5 (left) plots the norm ratios $\|T_3\|/\|T_2\|$ (raw) and $\|T_{3,P}\|/\|T_{2,P}\|$ (effective, for AdamW). Throughout SGD training, the curvature term's norm is substantially larger than the alignment term's, suggesting that **curvature interference from T_3 opposes anticipation (T_2) in SGD** (see also Figure 11 in the Appendix). For AdamW, the raw and effective ratios were computed from the same training trajectory. The unpreconditioned ratio is already considerably smaller than SGD's, and incorporating AdamW's preconditioner reduces the ratio further. We also tested whether AdamW's anticipation may be due to memory accumulating in the optimizer state. However, resetting the optimizer state at every burst transition produced a similar SA trajectory (Figure 12), showing that persistent optimizer memory across bursts is not necessary for anticipation. Together with our $\|T_3\|/\|T_2\|$ analysis, this supports the interpretation that **AdamW's coordinatewise preconditioning encourages anticipation** by reducing curvature interference relative to the alignment pressure.

Muon. Theorem C.1 and Theorem C.2 do not apply to Muon, whose update is a nonlinear transformation of the gradient and not a fixed linear preconditioning. We can nevertheless measure the raw SGA of the states reached by Muon. By Theorem C.3, this quantity describes how strongly an infinitesimal SGD step would locally increase successor advantage (SA). High raw SGA therefore means that the gradient's geometry already has successor aligned structure. Surprisingly, Muon reaches states with raw SGA comparable to AdamW (Figure 5, right), despite not exhibiting anticipation. Thus, Muon's absence of anticipation cannot simply be attributed to the learned gradient geometry lacking successor-aligned structure.

We hypothesize instead that this structure is mitigated by Muon's Newton-Schulz (NS) transformation. To test this, we construct a heuristic effective SGA ($\mathrm{SGA}_{\mathrm{eff}}$) by applying the NS transformation to the class gradients before measuring SGA. Figure 5 (right) shows that this addition substantially reduces SGA, consistent with the interpretation that **Newton-Schulz suppresses anticipation-relevant structure that is present in the raw gradient geometry**. This is possible because raw SGA characterizes alignment with the *gradient* direction, whereas Muon does not follow that direction directly. Its nonlinear orthogonalization can reweight or rotate the gradient components that carry the successor-alignment signal. Therefore, a parameter state may be locally favorable for anticipation under SGD while producing no corresponding effect under the Muon update. Details of the raw and effective SGA estimators are given in Appendix Section D.5. Since gradient and update magnitudes differ substantially across optimizers, we report normalized versions of these statistics.

Cyclic trajectory of model weights. We further illustrate this geometric perspective in Section D.2. We plot PCA projections of the flattened weights of a small model immediately before each class transition in the cycle. **The projected trajectory forms a cyclical orbit**, consistent with the successor gradient alignment (SGA) geometry described above.

4.3 Controlling Anticipation

If, as the theory predicts, anticipation is indeed caused by the T_2 alignment pressure of Theorem C.1, then we should be able to manipulate it directly, similarly to how Geiping et al. (2022) controlled generalization via an implicit bias regularizer. Theorem C.1 says that at a transition $c \to n$, bursty training implicitly minimises $\|G^{(c)} - G^{(n)}\|^2$. On the `Ring`, the next class is always $s(c)$, so we can add this term to the loss as an *explicit regularizer*:

$$\mathcal{L}(\theta) = \mathcal{L}_{\mathrm{task}}(\theta) + \lambda\, \widehat{D}\big(c, s(c)\big), \tag{4}$$

where $\widehat{D}$ estimates $\|G^{(c)} - G^{(s(c))}\|^2$ for the current class c of the `Ring`, from per-class mini-batch gradients (implementation details are in Section D.6). The theory then makes the following predictions. With $\lambda < 0$, gradient ascent on the successor distance opposes the implicit T_2 drift. If this drift is what accumulates into anticipation, a sufficiently strong $\lambda < 0$ should destroy SA. With small $\lambda > 0$, the regularizer pushes in the same direction as the schedule, so SA should be preserved or increase.

Estimating the T_2 regularizer. We found estimating $\|G^{(c)} - G^{(s(c))}\|^2$ from mini-batches tricky. This is likely because the plug-in estimator $\widehat{D}_{\mathrm{plug}} = \|\hat{g}^{(c)} - \hat{g}^{(s(c))}\|^2$ on fresh mini-batch gradients $\hat{g}$ is biased upward by the within-class variance, $\mathbb{E}[\widehat{D}_{\mathrm{plug}}] = \|G^{(c)} - G^{(s(c))}\|^2 + \frac{1}{b}\big(\operatorname{tr}\Sigma^{(c)} + \operatorname{tr}\Sigma^{(s(c))}\big)$. Minimizing this also minimizes a variance term (similar to that in Theorem C.1), which at large λ may be cheaper to satisfy by suppressing gradient noise than by aligning class-mean gradients. For $\lambda > 0$, we therefore also use an unbiased estimator, $\widehat{D}_{\mathrm{EMA}} = \big\langle \hat{g}^{(c)} - \hat{g}^{(s(c))},\ \bar{G}^{(c)} - \bar{G}^{(s(c))} \big\rangle$. $\widehat{D}_{\mathrm{EMA}}$ is the inner product against exponential-moving-average estimates $\bar{G}$ of the class-mean gradients. Because the EMA estimate is (nearly) independent of the fresh mini-batch, the contribution of the variance term, and hence the estimator's bias is reduced. Further details are in Section D.6.

Results. Figure 6 shows SA on the standard `Ring` configuration (AdamW, 12 seeds per curve) as a function of λ. Negative λ switches anticipation off: at $\lambda \le -0.003$ the final SA drops from the baseline 0.274 to zero and below, while eval loss remains identical. Small positive λ with the unbiased estimator leaves anticipation intact (0.279 and 0.28 at $\lambda = 0.03, 0.06$), as does the plug-in estimator at $\lambda = 0.03$ (0.247). Larger positive regularization degrades SA: $\lambda = 0.1$ has high seed variance, $\lambda \ge 0.3$ collapses to zero).

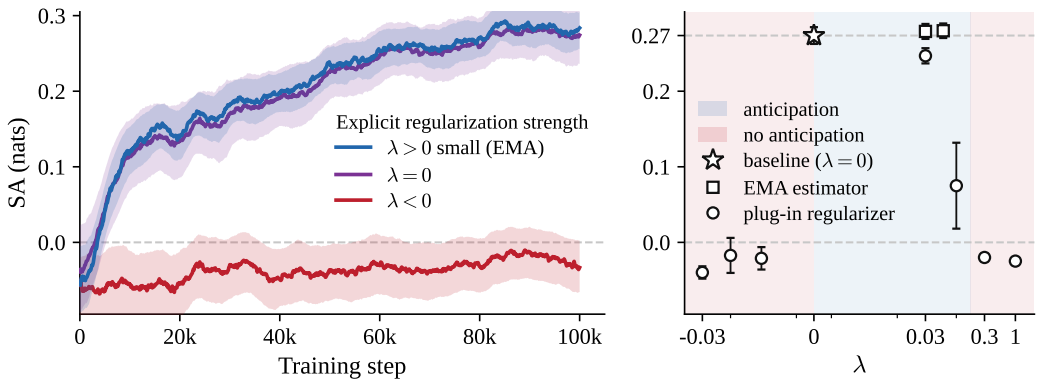


Figure 6: **Explicit T_2 regularization partly controls anticipation on the Ring.** *(Left)* SA over training for the unregularized Ring ($\lambda = 0$), small positive λ with the unbiased EMA estimator (pooled $0.03/0.06$), and negative λ (pooled $-0.003/-0.01/-0.03$). Maximizing $\|G^{(c)} - G^{(n)}\|^2$ from T_2 abolishes anticipation, minimizing it preserves the baseline level. *(Right)* Final-quarter SA against regularization strength (λ). Strong regularization with the biased plug-in estimator also destroys SA, while the EMA estimator keeps SA for small positive λ. We report 12 seeds per curve, error bars are SEM (left) and 95% CIs of the final-quarter SA (right).

Interpretation. Together with Section 4.2, these results are consistent with our theoretical findings: **the same $\|G^{(c)} - G^{(n)}\|^2$ term from T_2 the `Ring` schedule minimizes implicitly can be maximized explicitly, and doing so removes anticipation from the `Ring`, without changing the training order.** We report that we tried adding anticipation into `Bursty` schedules using explicit regularization. This wasn't successful, likely because of the difficulties of constructing unbiased estimators of T_2.

5 How far does anticipation extend?

The implicit bias interpretation identifies a mechanism by which anticipation can emerge under cyclic bursty training – but how general and sequence-model-like is this behavior in practice? Here, beyond optimizers, we study anticipation across transition structures, scales, and training configurations. Section D contains further results.

In Section D.4, we find small, but present **anticipation in image modeling**, suggesting that the behavior is not specific to the text domain. Ablating the **Expected number of gradient steps per burst (Figure 13)**, we observe increasing anticipation up to an expected 32 steps per class, consistent with Yang et al. (2024). This is in line with Theorem C.1 as more gradient steps allow class-gradient alignment to exercise its effect more. However, increasing the number of steps further is detrimental, likely because adapting to the current class becomes the advantageous behavior. This is shown by `Bursty`'s training loss approaching `Ring`'s. We find that larger **batch size (Figure 15)** generally helps anticipation, though the difference between 64 and 256 in SA is negligible. Larger batch size allows for better gradient mean estimates, helping class alignment. In addition, T_1 in Theorem C.1 decreases with increasing batch size, hence any possible interference from this term is reduced. Varying the **context length (Figure 14),** we find that, contrary to Yang et al. (2024) who used individual documents instead of classes, longer context lengths hurt anticipation, likely because the class becomes easily identifiable from a prefix, hence any anticipation advantage persists only for a few tokens. The theoretical room for anticipation also decreases. Finally, increasing **model size (Figure 16)** increases the anticipation effect. While Yang et al. (2024) only showed anticipation for model scales starting at 160M parameters, we showcase non-negligible anticipation for as small as 0.19M parameter models.

6 Consequences of anticipatory inductive bias

So far, we have treated anticipation as a behavioral phenomenon: evidence that models can exploit temporal structure in their training stream. *But does this temporal structure also shape what the*

model learns? To demonstrate the extent to which anticipatory inductive bias shapes training, we investigate relative inter-class gradient alignments under fixed `Ring` orders.

We measure inter-class relationships using gradient cosine similarity. This captures local cross-domain transfer, because aligned gradients mean that an update improving one domain also tends to improve the other. The gradient normalization in cosine similarity isolates this directional compatibility from class-specific gradient magnitudes.

Figure 7 shows excess cosine similarities after 200k training steps under the displayed `Ring` order. For each current class i and comparison class k, cell (i, k) reports $\cos(g^{(i)}, g^{(k)}) - \frac{1}{R-2}\sum_{j \notin \{i,k\}} \cos(g^{(i)}, g^{(j)})$, the alignment of class i with class k relative to all remaining classes. When $k = s(i)$, this is precisely a per-class SGA term computed on unit-normalized gradients. See Appendix Section D.8 for details.

After training, relative alignments can depart sharply from a priori class similarities and instead reflect the training order, as seen in the red successor cells. We highlight the surprising example of Spanish becoming more aligned with its successor Finnish than with the a priori more similar Portuguese. This demonstrates that training order can intervene on the relationships between classes learned by the model.

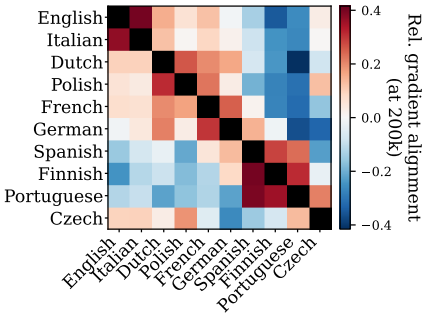


Figure 7: **Relative gradient cosine similarities** at the end of training (200k steps), when using the fixed Ring order matching the order of the figure labels (up to bottom, left to right).

7 Related Work

Our work builds most directly on Yang et al. (2024), who showed anticipatory recovery in cyclic document fine-tuning. We extend this phenomenon to bursty pre-training over recurring data domains (or classes) and study its optimization mechanism. More broadly, structured and non-i.i.d. training orders have been studied through random reshuffling, without-replacement optimization, and their induced optimization trajectories (Gürbüzbalaban et al., 2015; Mishchenko et al., 2020; Beneventano, 2024). Our analysis is also related to work on SGD's implicit regularization (Smith et al., 2021a; Roberts, 2021b). Particularly close is Dherin (2023), who derives local order-dependent biases from consecutive updates on different tasks. Continual-learning work has likewise emphasized gradient interference, transfer, and the effect of task order (Riemer et al., 2019; Bell & Lawrence, 2022; Li & Hiratani, 2025). Instead, our setting considers cyclically recurring classes and asks how predictable transitions induce gradient alignments between consecutive classes. Finally, our framing is related to work on implicit meta-learning (Krasheninnikov et al., 2024; 2026) and broadly to situational awareness in language models (Berglund et al., 2023; Laine et al., 2024). We discuss related work in more detail in Section A.

8 Discussion

In this work, we showed that Transformers anticipate the upcoming classes in cyclic bursty pre-training, and this behavior can be interpreted as sequence-model-like. We provided a possible explanation for this phenomenon by identifying an implicit bias of bursty mini-batch training, that aligns adjacent class gradients. Our results suggest that training order is not an easy-to-ignore optimization detail. Cyclic bursty training promotes alignment between temporally adjacent classes, reshaping relationships between them. As anticipation strengthens with scale, increasingly weak temporal structure may become sufficient to induce such effects beyond explicitly cyclic settings. Understanding the inductive biases induced by training order is therefore important for understanding how language models acquire associations during training.

While our work narrows the gap between anticipation and ordinary optimization dynamics, we do not resolve whether models ever explicitly represent or reason about their own training history. Understanding when training-history dependence reflects implicit optimization bias versus genuine awareness of the training process remains an open problem.

AI USE STATEMENT

In this work, we used generative AI tools for assisting the development of mathematical claims in multiple back and forth conversations. These include discussing mathematical ideas, cleaning up and verifying proofs with multiple LLMs. All mathematical statements were checked manually and pieced together into the final theorems by the authors. We used AI to propose or refine hypotheses collaboratively and provide feedback on research methodology or experiments. We also used coding assistants to implement methods, support qualitative and thematic data analysis, and interpret results. We have not used generative AI tools for generating synthetic datasets, assist with translation and cleaning datasets, and for formatting references. Additionally, we used generative AI tools for creating or modifying scientific figures or images (with reviewed code scripts), suggesting experimental parameters, creating or editing software code, creating short draft parts of a research paper, summarizing or analysing existing literature, discovering research topics or identifying gaps, brainstorming, sourcing/searching for information, editing a research paper to improve readability, identifying relevant literature, formatting references, suggesting a structure for a research paper. We have reviewed all AI-assisted work. We checked LLM-generated research ideas for potential plagiarism through a manual literature survey. Most of these research ideas were minor, along the lines of experimental methodology, so they weren't at risk of plagiarism. LLM-generated code was tested and verified in code reviews. LLM-generated drafts were only used as first drafts, and were edited extensively. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Our appendix contains all details and assumptions of the theoretical results. We also provide all hyperparameters and descriptions of the analyses we ran. We commit to releasing our codebase upon acceptance.

REFERENCES

Kwangjun Ahn, Chulhee Yun, and Suvrit Sra. SGD with shuffling: Optimal rates without component convexity and large epoch requirements. *Advances in Neural Information Processing Systems*, 33: 17526–17535, 2020.

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision*, pp. 139–154, 2018.

Samuel J. Bell and Neil D. Lawrence. The effect of task ordering in continual learning, 2022. URL https://arxiv.org/abs/2205.13323.

Pierfrancesco Beneventano. On the trajectories of sgd without replacement, 2024. URL https://arxiv.org/abs/2312.16143.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pp. 41–48, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605585161. doi: 10.1145/1553374.1553380. URL https://doi.org/10.1145/1553374.1553380.

Lukas Berglund, Asa Stickland, Mikita Balesni, Max Kaufmann, Meg Tong, Tomasz Korbak, Daniel Kokotajlo, and Owain Evans. Taken out of context: On measuring situational awareness in llms, 09 2023.

Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019.

Zhiyuan Chen and Bing Liu. *Lifelong Machine Learning*, volume 1. Springer, 2018.

Benoit Dherin. Implicit biases in multitask and continual learningfrom a backward error analysis perspective. In *NeurIPS 2023 Workshop on Mathematics of Modern Machine Learning*, 2023. URL https://openreview.net/forum?id=vXkC6AOupO.

Wikimedia Foundation. Wikimedia downloads, 2023. URL https://dumps.wikimedia.org.

Shivam Garg, Dimitris Tsipras, Percy Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes, 2023. URL https://arxiv.org/abs/2208.01066.

Jonas Geiping, Micah Goldblum, Phil Pope, Michael Moeller, and Tom Goldstein. Stochastic training is not necessary for generalization. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=ZBESeIUB5k.

M Gurbuzbalaban, Asu Ozdaglar, and Pablo A Parrilo. Convergence rate of incremental gradient and incremental newton methods. *SIAM Journal on Optimization*, 29(4):2542–2565, 2019.

Mert Gürbüzbalaban, Asu Ozdaglar, and Pablo Parrilo. Why random reshuffling beats stochastic gradient descent. *Mathematical Programming*, 186, 10 2015. doi: 10.1007/s10107-019-01440-w.

Jeff Z. HaoChen, Colin Wei, Jason Lee, and Tengyu Ma. Shape matters: Understanding the implicit bias of the noise covariance. In Mikhail Belkin and Samory Kpotufe (eds.), *Proceedings of Thirty Fourth Conference on Learning Theory*, volume 134 of *Proceedings of Machine Learning Research*, pp. 2315–2357. PMLR, 15–19 Aug 2021. URL https://proceedings.mlr.press/v134/haochen21a.html.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114 (13):3521–3526, 2017.

Dmitrii Krasheninnikov, Egor Krasheninnikov, Bruno Mlodozeniec, Tegan Maharaj, and David Krueger. Implicit meta-learning may lead language models to trust more reliable sources. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Dmitrii Krasheninnikov, Richard E. Turner, and David Krueger. Fresh in memory: Training-order recency is linearly encoded in language model activations. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=Tn6famjSxN.

Rudolf Laine, Bilal Chughtai, Jan Betley, Kaivalya Hariharan, Mikita Balesni, Jérémy Scheurer, Marius Hobbhahn, Alexander Meinke, and Owain Evans. Me, myself, and AI: The situational awareness dataset (SAD) for LLMs. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=UnWhcpIyUC.

Quentin Lhoest, Albert Villanova del Moral, Patrick von Platen, Thomas Wolf, Mario Šaško, Yacine Jernite, and Abhishek Thakur. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 175–184, 2021. URL https://aclanthology.org.

Ziyan Li and Naoki Hiratani. Optimal task order for continual learning of multiple tasks. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=145So0OrGC.

Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of Learning and Motivation*, volume 24, pp. 109–165. Elsevier, 1989.

Konstantin Mishchenko, Ahmed Khaled, and Peter Richtárik. Random reshuffling: Simple analysis with vast improvements. *Advances in Neural Information Processing Systems*, 33:17309–17320, 2020.

Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. iCaRL: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2001–2010, 2017.

Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=B1gTShAct7.

Daniel A Roberts. Sgd implicitly regularizes generalization error. *arXiv preprint arXiv:2104.04874*, 2021a.

Daniel A. Roberts. Sgd implicitly regularizes generalization error, 2021b. URL https://arxiv.org/abs/2104.04874.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Itay Safran and Ohad Shamir. How good is SGD with random shuffling? In *Conference on Learning Theory*, pp. 3250–3284. PMLR, 2020.

Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, pp. 4548–4557. PMLR, 2018.

Samuel L Smith, Benoit Dherin, David Barrett, and Soham De. On the origin of implicit regularization in stochastic gradient descent. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=rq_Qr0c1Hyo.

Samuel L Smith, Benoit Dherin, David GT Barrett, and Soham De. On the origin of implicit regularization in stochastic gradient descent. *arXiv preprint arXiv:2101.12176*, 2021b.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018. URL http://incompleteideas.net/book/the-book-2nd.html.

Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural networks for machine learning, 2012.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022.

Lijie Xu, Shuang Qiu, Binhang Yuan, Jiawei Jiang, Cedric Renggli, Shaoduo Gan, Kaan Kara, Guoliang Li, Ji Liu, Wentao Wu, et al. Stochastic gradient descent without full data shuffle. *arXiv preprint arXiv:2206.05830*, 2022.

Yanlai Yang, Matt Jones, Michael C. Mozer, and Mengye Ren. Reawakening knowledge: Anticipatory recovery from catastrophic interference via structured training. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 82438–82464. Curran Associates, Inc., 2024. doi: 10.52202/079017-2621. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/9628b97f74cd8f83ad12b631378b81e2-Paper-Conference.pdf.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pp. 3987–3995. PMLR, 2017.

Liu Ziyin, Kangqiao Liu, Takashi Mori, and Masahito Ueda. Strength of minibatch noise in SGD. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=uorVGbWV5sw.

A Extended Related Work

Emergent adaptation and implicit meta-learning. Recent work has interpreted Transformers as systems capable of implicit adaptation and meta-learning (Krasheninnikov et al., 2024). LLMs have been shown to exhibit emergent capabilities that appear only beyond certain scales (Wei et al., 2022). Separately, recent work showed that Transformer activations linearly encode recency information about when knowledge was acquired during training Krasheninnikov et al. (2026), suggesting that optimization history may leave persistent traces in model representations. Our work complements these findings by studying transformers as partial sequence models over the training stream itself. Unlike standard in-context learning (Garg et al., 2023), where adaptation occurs within a forward pass over tokens, anticipation emerges through the optimization trajectory across training steps. Broadly speaking, anticipation is related to questions of situational awareness or training process awareness in LLMs, whereby models can represent and exploit information about their own circumstances (Berglund et al., 2023; Laine et al., 2024). Specifically, anticipation studies a narrower form of exploiting information: regularities in the model's training stream.

Implicit biases of SGD and optimizer dynamics. A growing literature studies stochastic gradient descent not merely as an optimization algorithm, but as a source of implicit regularization and inductive bias. Prior analyses have connected SGD noise to flat minima, generalization, and gradient covariance structure (Roberts, 2021b; Smith et al., 2021a; HaoChen et al., 2021; Ziyin et al., 2022), but mostly focused on i.i.d. batches, whereas we consider bursty training. In this category, the closest to us is Dherin (2023), who uses backward error analysis to characterize the local implicit bias of consecutive updates on *different tasks*. They identify an order-dependent term that's essentially identical to our T_3 in Theorem C.1. However, we frame the implicit bias in comparison to a matched full-batch baseline (GD), which allows us to additionally identify our gradient alignment term (T_2). Our further results also describe how this local bias changes under different transition schedules, and that it's connected to anticipation. Implicit regularizers were found to aid generalization even when replaced by an explicit regularizer in gradient descent (Geiping et al., 2022).

Continual learning and catastrophic interference. Continual learning provides a perspective on sequentially non-i.i.d. training complementary to ours. Catastrophic interference occurs when learning new tasks degrades previously acquired behavior (McCloskey & Cohen, 1989), motivating replay- and regularization-based methods (Rebuffi et al., 2017; Rolnick et al., 2019; Chaudhry et al., 2019; Kirkpatrick et al., 2017; Zenke et al., 2017; Aljundi et al., 2018; Chen & Liu, 2018; Serra et al., 2018). Riemer et al. (2019) characterizes transfer and interference through gradient alignment and explicitly optimizes for gradients that facilitate rather than conflict across experiences. Other work shows that the order of tasks itself can substantially alter forgetting and generalization (Bell & Lawrence, 2022; Li & Hiratani, 2025). Our setting differs in that tasks recur cyclically instead of arriving once. We ask how a predictable cyclic order induces persistent successor-specific gradient relationships.

Structured training streams and ordered optimization. Prior work has studied optimization under ordered or reshuffled data streams, including random reshuffling SGD, without-replacement sampling, and cyclic training regimes, primarily through their convergence rates (Gürbüzbalaban et al., 2015; Gurbuzbalaban et al., 2019; Mishchenko et al., 2020; Ahn et al., 2020; Safran & Shamir, 2020; Xu et al., 2022). More recent work has also characterized how without-replacement sampling changes the optimization trajectory and induces implicit regularization (Beneventano, 2024). Yang et al. (2024) recently demonstrated that large language models trained in cyclic fine-tuning settings exhibit anticipatory recovery, whereby the loss on an upcoming document decreases before the document is revisited. Our work extends this line of research from repeated-document fine-tuning to bursty pre-training over classes, and argues that structured training streams induce a systematic sequential inductive bias. Rather than studying only convergence or forgetting, we focus on how temporal structure in the training order reshapes learned relationships between classes and representations.

Relationship to anticipatory recovery (Yang et al., 2024). While anticipatory recovery measures the loss trajectory of a repeated document, we instead wanted to measure adaptation to the structure of the training stream itself. This was motivated by our framing of anticipation as sequence model-like non-i.i.d. behavior (see Proposition 1). In Figure 8, we show that our setup also reproduces

anticipation when measured with the anticipatory recovery metric: the evaluation loss of a given class distribution starts decreasing before that class is revisited again.

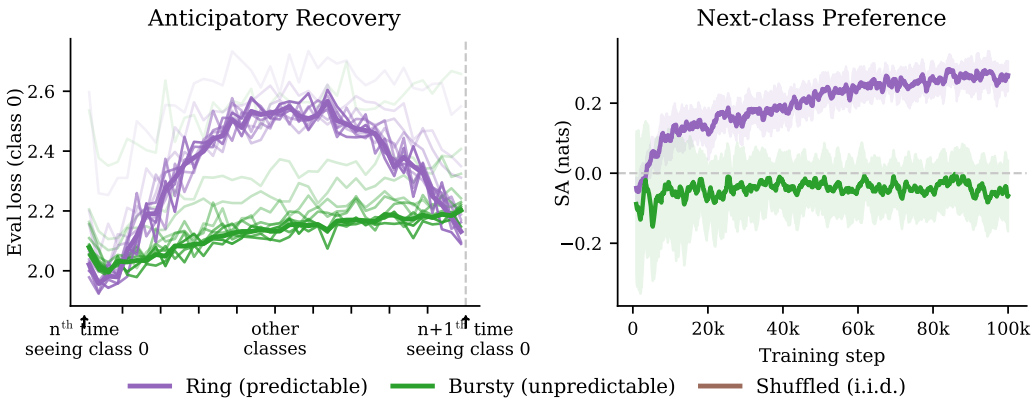


Figure 8: **Comparison of anticipatory recovery (Yang et al., 2024) and anticipation as next-class preference.** *(Left) Reproduced Anticipatory recovery over cycle progress for class* 0. The thick curves show eval loss, averaged over cycles in the last half of training. Lighter curves show 10% training windows, lighter means earlier. Much like in Yang et al. (2024), the Ring configuration recovers most of its forgetting during the cycle *before seeing the class again*, while the Bursty setting doesn't. *(Right) Next-class Preference measured with Successor Advantage (SA) over time*, with ± 1 std intervals. Ring SA keeps increasing, while Bursty stays below zero. Values are averaged over 12 seeds.

B Details on the experimental setup

B.1 Dataset

We use byte-level text from the `wikimedia/wikipedia` dataset (Foundation, 2023) on Hugging Face Datasets (Lhoest et al., 2021), across ten languages: English, Dutch, German, French, Spanish, Portuguese, Italian, Polish, Czech, and Finnish. Each language constitutes one class. We use 1,000,000 training samples and 256 evaluation samples per language class. Text is chunked into contiguous segments of 8 tokens, except in the context length ablation, where we additionally use segments of 32, 64, 128 and 256 tokens. The default choice of sequence length was motivated by Proposition 1: which shows that, for a fixed number of classes, the amount of achievable anticipatory advantage in the training loss scales inversely with the context length. Short sequence lengths therefore provide a setting where anticipation is highly incentivized, allowing us to isolate the behavior easily. The model input additionally prepends a BOS token and appends an EOS token, yielding a default input length of 10. Text is encoded as UTF-8 bytes; each byte value is used directly as a token ID, with no subword tokenization. The vocabulary size is 258 (256 token values plus BOS and EOS).

In the `Ring` and `Bursty` setups, data is sampled from a Markov chain over classes, with transition probability $1/\tau$, with default $\tau = 2048$. This means that burst lengths are geometrically distributed, and on average, 2048 samples (organized in batches) are seen from a class before a transition.

B.2 Model architectures

We train GPT-2 style transformers (Vaswani et al., 2017) of several scales. All models share the same byte-level vocabulary of size 258 and input length 10 (excluding the "Empirical limits of anticipation" experiment and the context length ablation). Architecture details are given in Table 2.

Parameters	Hidden dim (H)	Layers (L)	Heads (A)	Intermediate dim (I)
0.19M	48	6	2	144
1.31M	128	6	4	384
5.2M	256	6	8	768
20.6M	512	6	8	1536
51.9M	576	12	8	1728

Table 2: **Model configurations.** All models use the same byte-level vocabulary (size 258) and input length 10. Parameter counts include embedding layers.

B.3 Training hyperparameters

All experiments use AdamW with weight decay 0.1, AMP mixed precision, and a warmup of 1,000 steps. "Gradient steps per class" refers to the expected number of gradient updates taken while the `Ring` chain remains in a single class, equal to τ/B where τ is the mean burst length and B is the batch size. Hyperparameters for the default configuration and each ablation (except for the optimizer ablation, whose hyperparamaters are reported in Table 6 and discussed in Section 4.2) are summarised in Table 3.

Experiment	Batch size	Context len	Grad steps	Total steps	Optimizer	LR	Wd
Default (Figure 1)	64	8	32	100k	AdamW	5×10^{-5}	0.1
Gradient steps ablation (Figure 13)	64	8	8 16 32 64 128	100k	AdamW	5×10^{-5}	0.1
Context length ablation (Figure 14)	64	8 32 128 256	32	100k	AdamW	5×10^{-5}, 1×10^{-4} 2×10^{-4}, 3×10^{-4}	0.1
Batch size ablation (Figure 15)	4 16 64 256	8	32	100k	AdamW	1.25×10^{-5}, 2.5×10^{-5}, 5×10^{-5}, 1×10^{-4}	0.1
Model size ablation (Figure 16)	64	8	32	100k	AdamW	1×10^{-4}, 1×10^{-4}, 5×10^{-5}, 5×10^{-5}, 5×10^{-5}	0.1

Table 3: **Training hyperparameters for each experiment.** Grad steps should be understood as *expected* number of grad steps as burst lengths are geometrically distributed. All runs use weight decay 0.1 and a 1,000-step linear warmup and then constant learning rate. AdamW uses default $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1e-8$. In multi-row ablation cells, LR and batch size values correspond to the listed context lengths or model sizes top to bottom.

B.4 Training protocols

We train under three protocols. See Figure 1 (left) for a visual explanation of these orderings.

- **Ring.** Training examples are drawn from a 10-class Markov chain. Classes are arranged in a fixed cyclic order, which, unless stated otherwise, depends on the random seed used for the experiment. After each batch the chain stays in the current class with probability $p_{\text{stay}} = 1 - B/\tau$, giving geometrically distributed burst lengths with mean τ, where τ equals the batch size times the number

of gradient steps per class. When a transition occurs, the chain deterministically moves to the next class in the `Ring`.

- **Bursty.** The chain has the same geometric burst structure as `Ring` (same τ and p_{stay}), but transitions are drawn uniformly over all other classes rather than following a fixed cyclic order. This condition controls for the burst structure while removing any predictable successor identity.
- **Shuffled.** Training examples are drawn uniformly at random from the mixture of all classes, with no temporal structure. This is the standard i.i.d. training baseline.

B.5 Evaluation metrics

Successor Advantage. Successor Advantage is defined in Definition 1 as:

$$\mathrm{SA}_t = \frac{1}{R-2} \sum_{n \notin \{c_t, n_t\}} \widehat{L}_n(\theta_t) - \widehat{L}_{n_t}(\theta_t).$$

In practice, we compute a *corrected variant*, which removes the bias of some successor classes being inherently easier to predict than others, which could cause accidental positive SA in `Ring` setups. Since `Bursty` doesn't have deterministic successor, shuffle-correction is not necessary. Hence for `Bursty`, raw SA is reported, where the next class is taken to be the realization of the random drawn next class. The corrected SA for `Ring` is the SA relative to a paired `Shuffled` model: $\mathrm{SA}_{\text{corrected},t} = \mathrm{SA}_{\text{Ring},t} - \mathrm{SA}_{\text{Shuffled},t}$. For both `Ring` and `Shuffled`, SA computation uses the `Ring`'s successor and current labels. The paired `Shuffled` model is trained on a uniformly random permutation of the identical token stream seen by the `Ring` model, so both models see the same multiset of examples. Only the temporal ordering differs. This ensures that any "accidental" SA in the `Shuffled` model, arising from inherent differences in class difficulty or vocabulary rather than sequential structure, is removed. In practice, the shuffle correction only marginally changes the SA. All SA values are averaged over the last 25% of training steps, with evaluation losses recorded every 100 steps.

C Proofs

C.1 Theoretical limit of anticipatory advantage

Setup. Let $x = (x_1, x_2, ..., x_T)$ denote a data sequence, where each token x_i is drawn from a fixed token dictionary $\mathcal{V}$. Let D_{train} denote a dataset of N such samples drawn i.i.d. from some mixture distribution over classes $c : p(x) = \sum_{c \in C} p(x \mid c)p(c)$. For simplicity, assume that classes are discrete, and have equal marginal probability. Autoregressive language models approximate $p(x)$ by next-token prediction, by fitting a distribution $p_M(x_{i+1} \mid x_{1:i}; \theta)$, parametrized by θ. Crucially, the architecture is memoriless: it does not explicitly model the training order $x^1, x^2, \ldots x^{|D|}$.

Sequence models. Anticipation is a property of *sequence models*, which approximate $p(x^i \mid x^1, x^2, \ldots, x^{i-1})$. In the setup we consider, the training order cycles through the classes c, hence $p(x^i \mid x^1, x^2, \ldots, x^{i-1}) = p(x^i \mid c : x_i \in c)$.

Proposition 1. *Let T denote the sequence length, assume that classes have uniform marginals ($p(c) = \frac{1}{R}$) and that the class c is identifiable from the prefix, i.e. $p(c|x) \approx 1$. The best possible average loss improvement over `Shuffled` training via anticipation is $\log R/T$. This is achieved by sequence models explicitly modeling class order, i.e. $p_{\text{sequence model}}(x) = p(x \mid c : x \in c)$.*

Proof. The per-token cross-entropy loss for a model $q(x)$ is

$$\mathbb{E}_{x \sim p(x)}\Big[-\frac{1}{T} \sum_{i=1}^{T} \log q(x_i | x_1, x_2, \ldots, x_{i-1}) \Big] = \mathbb{E}_{x \sim p(x)}\Big[-\frac{1}{T} \log q(x) \Big].$$

Training on shuffled batches targets $p(x)$, hence the best achievable cross-entropy loss is $\ell_{\text{shuffled}}(x) = \mathbb{E}_{x \sim p(x)}\Big[-\frac{1}{T} \log p(x) \Big]$. Perfect sequence anticipation means that the model learns $p(x|c)$, wherever

$x \in c$. Hence the best achievable loss is $\ell_{\text{anticipatory}}(x) = \mathbb{E}_{c,x\sim p(c,x)}\Big[-\frac{1}{T}\log p(x \mid c)\Big]$. From Bayes's rule, we have

$$p(x \mid c) = \frac{p(x)p(c \mid x)}{p(c)} \approx \frac{p(x)}{p(c)},$$

because we assumed that c is identifiable from x, $p(c|x) \approx 1$. Hence the best possible loss improvement is

$$\begin{aligned}\ell_{\text{shuffled}}(x) - \ell_{\text{anticipatory}}(x) &= \mathbb{E}_{c,x\sim p(c,x)}\Big[\frac{1}{T}\Big[-\log p(x) + \log p(x) - \log p(c)\Big]\Big] \\ &\approx \frac{\mathbb{E}_{c\sim p(c)}[-\log p(c)]}{T}.\end{aligned}$$

For $p(c) = \frac{1}{R}$, this simplifies to $\frac{\log R}{T}$. □

C.2 Anticipation as the implicit bias of bursty mini-batch training

Our theoretical results are structured as follows: we first derive the implicit bias of SGD in bursty settings (Theorem C.1), showing that at class transitions, cyclic mini-batch updates create a consistent pressure toward alignment of temporally adjacent class gradients. We then show that this alignment survives averaging over the training schedule (Theorem C.2). We then relate this to successor advantage, our measure of anticipation (Theorem C.3). Our Theorem C.1 builds on Roberts (2021b). A similar approach is described in Dherin (2023), but considers a different baseline and hence does not identify the gradient alignment implicit bias. A related gradient-alignment mechanism was hypothesized informally by Krasheninnikov et al. (2024) and Nichol et al. (2018). Definitions and proofs are in Section C.2.

C.2.1 Setup

We study two sequential SGD updates on batches B_{current} and B_{next}, starting from θ_0, and ask how the construction of mini-batches shapes optimization trajectory compared to full-batch gradient descent. For readability, we will use the shorthands B_c and B_n. We are interested in the below two settings:

- `Shuffled` training: B_c and B_n are i.i.d. draws from the data mixture $p(x)$.
- `Bursty` training: B_c and B_n each only contain samples from a single class c_{current} and c_{next}, respectively.

Roberts (2021b)'s analysis assumes the shuffled setting. Both settings take two sequential gradient steps of size η on batches of size $N/2$, and we compare this two-step SGD trajectory to the equivalent two full-batch GD steps on $B_c \cup B_n$, isolating the effect of sequential updating.

Notation Let $p(x)$ be a data-generating distribution, which can be written as a mixture $p(x) = \sum_{c\in C} p(x \mid c)p(c)$ over a finite set of classes C, $R := |C|$. Let $\ell(\theta, x)$ be a twice continuously differentiable loss function with parameter vector $\theta \in \mathbb{R}^d$, and define the per-example gradient and Hessian:

$$g_i(\theta, x) = \partial\ell(\theta, x)/\partial\theta_i, \qquad h_{ij}(\theta, x) = \partial^2\ell(\theta, x)/\partial\theta_i\partial\theta_j.$$

For each class c, define the population class loss and its gradient, Hessian and gradient covariance

$$L_c(\theta) = \mathbb{E}_{x\sim p(\cdot|c)}[\ell(\theta, x)], \qquad G^{(c)} = \nabla L_c, \qquad H^{(c)} = \nabla^2 L_c, \qquad \Sigma^{(c)}_{ij}(\theta) = \mathrm{Cov}_{x\sim p(\cdot|c)}(g_i, g_j),$$

so that $G^{(c)}_i = \mathbb{E}_{x\sim p(\cdot|c)}[g_i(\theta, x)]$ and $H^{(c)}_{ij} = \mathbb{E}_{x\sim p(\cdot|c)}[h_{ij}(\theta, x)]$. Under $p(x)$, define the full-data gradient mean and covariance

$$G^{(\text{full})}_i(\theta) = \mathbb{E}_{x\sim p}[g_i(\theta, x)] = \sum_{c\in C} p(c)G^{(c)}_i, \qquad \Sigma^{(\text{full})}_{ij}(\theta) = \mathrm{Cov}_{x\sim p}(g_i, g_j) = \Sigma_{\text{within}} + \Sigma_{\text{between}}.$$

Further, for batch B, define the sample mean gradient and Hessian as

$$g^{(B_c)}_i(\theta) = \frac{1}{|B|}\sum_{x\in B} g_i(\theta, x), \qquad h^{(B_c)}_{ij}(\theta) = \frac{1}{|B|}\sum_{x\in B} h_{ij}(\theta, x).$$

Let the change in parameter θ^i after updates on two SGD batches B_c, B_n be denoted as $\delta\theta_2^{i(B_c,B_n)}$, and the change after two gradient descent updates on the full data $B_c \cup B_n$ be $\delta\theta_2^{i(B_c\cup B_n,B_c\cup B_n)}$. Then the difference in gradient δg_i satisfies:

$$\delta\theta_2^{i(B_c\cup B_n,B_c\cup B_n)} - \delta\theta_2^{i(B_c,B_n)} = -\eta\,\delta g_i.$$

We rely on the following regularity assumptions.

Assumptions

(i) **Finite second moments**: $\operatorname{tr}\Sigma^{(c)} < \infty$ for every class c;

(ii) **Independence of minibatches** B_c and B_n.

(iii) **Second-order smoothness.** The loss $\ell(\theta, x)$ is twice continuously differentiable with respect to θ, and its Hessian is locally Lipschitz in a neighborhood of the optimization trajectory. This ensures that the Taylor expansions used below are valid up to $O(\eta^2)$ remainder terms;

(iv) **Differentiation under the expectation.** For every class distribution considered, the functions $g_i(\theta, x)$, $h_{ij}(\theta, x)$, and the products $g_i(\theta, x)g_j(\theta, x)$ are differentiable in a neighborhood of the current parameter θ, and their derivatives are dominated by integrable functions independent of θ. This ensures, by dominated convergence, that derivatives may be exchanged with expectations.

(v) **Approximately equal within-class gradient variance:** $\operatorname{tr}\Sigma^{(c)} \approx \operatorname{tr}\Sigma^{(n)}$

(vi) **Matched schedules** (for Theorem C.2 only) `Ring` and `Bursty` training have uniform class marginals and identical burst-length laws, so they differ only in the conditional law of the next class given a class transition.

We note that assumption (v) is not integral, and is only made to simplify the final formula.

C.2.2 Formal statements of the core results

Theorem C.1 (Local drift of bursty training)**.** *Let $B_c \overset{\text{i.i.d.}}{\sim} p(\cdot \mid c)$ and $B_n \overset{\text{i.i.d.}}{\sim} p(\cdot \mid n)$ be two mini-batches of size $N/2$, and assume $\operatorname{tr}\Sigma^{(c)} \approx \operatorname{tr}\Sigma^{(n)} = \operatorname{tr}\Sigma_{\text{within}}$. Then, to leading order in η, with expectation taken over the mini-batch randomness,*

$$\mathbb{E}[\delta g_i^{\text{bursty}}] = \underbrace{-\frac{\eta}{2N}\,\partial_i \operatorname{tr}\Sigma_{\text{within}}}_{T_1:\ \text{variance regularization}} \underbrace{-\frac{\eta}{8}\,\partial_i\|G^{(c)} - G^{(n)}\|^2}_{T_2:\ \text{alignment pressure}} \underbrace{-\frac{\eta}{2}\Big(G_j^{(n)}H_{ij}^{(c)} - G_j^{(c)}H_{ij}^{(n)}\Big)}_{T_3:\ \text{curvature interaction}}.$$

If instead both trajectories take preconditioned steps $\theta \leftarrow \theta - \eta P g$ with a fixed symmetric positive definite P, the same holds with

$$\mathbb{E}[\delta g^{\text{bursty}}] = -\frac{\eta}{2N}\,P\nabla\operatorname{tr}(P\Sigma_{\text{within}}) - \frac{\eta}{8}\,P\nabla\|G^{(c)} - G^{(n)}\|_P^2 - \frac{\eta}{2}\,P\Big(H^{(c)}PG^{(n)} - H^{(n)}PG^{(c)}\Big),$$

where $\|x\|_P^2 := x^\top P x$.

Interpretation. Theorem C.1 shows that bursty SGD (including preconditioned steps) is locally equivalent, to leading order, to GD on an objective augmented by three additional terms. T_1 is the shuffled-SGD mechanism of Roberts (2021b) restricted to within-class variance, hence it does not couple different classes. T_2 is a symmetric force pulling $G^{(c)}$ and $G^{(n)}$ together. T_3 is antisymmetric and measures how a step on one class reshapes the landscape seen by the other. T_3 is the only term sensitive to the *order* of the two updates.

Theorem C.1 describes one transition, which can look exactly the same in `Ring` and `Bursty` training. what differentiates them is their training schedules, which determine which transitions occur and how often.

Definition 2 (Bursty and Ring schedules)**.** *The `Bursty` and `Ring` schedules are probability distributions $\pi(n \mid c) : C \times C \mapsto [0, 1]$ that put mass onto possible next classes n given the current class c. We have $\pi^{\text{Bursty}}(n \mid c) = \mathbf{1}\{n \neq c\}/(R-1)$ and $\pi^{\text{Ring}}(n \mid c) = \mathbf{1}\{n = s(c)\}$.*

Finally, we will need the concept of Successor-Gradient Alignment (SGA).

Definition 3 (Successor-Gradient Alignment, SGA)**.** *Let $s(c)$ be the successor of class c in the* `Ring` *order, and define the* successor-contrast vector $w_c(\theta) := G^{(s(c))} - \frac{1}{R-2}\sum_{n\notin\{c,s(c)\}} G^{(n)}$*, the successor's gradient in excess of the average gradient of the other non-current classes. Then the SGA is*

$$\mathrm{SGA}(\theta) := \frac{1}{R}\sum_{c\in C}\big\langle w_c(\theta),\, G^{(c)}\big\rangle. \tag{5}$$

For preconditioned steps with a symmetric positive definite P we have $\mathrm{SGA}_P(\theta) := \frac{1}{R}\sum_{c\in C}\big\langle w_c(\theta),\, PG^{(c)}\big\rangle$, which is SGA *computed in the P-inner product $\langle x, y\rangle_P = x^\top Py$.*

SGA is the amount by which a class's gradient is aligned with its successor's *in excess of* its alignment with the other non-current classes.

Theorem C.2 (Ring training can increase SGA)**.** *Let $R \geq 3$ and $\gamma_R = \frac{R-2}{R-1}$, and let* `Ring` *and* `Bursty` *training have matched class marginals and burst statistics (assumption (vi) of Section C.3). For any schedule π, write $\mathcal{K}^\pi_i(\theta) = \mathbb{E}_\pi\big[G^{(n)}_j H^{(c)}_{ij} - G^{(c)}_j H^{(n)}_{ij}\big]$ for the schedule-averaged T_3 (with c uniform) and $\Delta\mathcal{K} = \mathcal{K}^{\mathrm{Ring}} - \mathcal{K}^{\mathrm{Bursty}}$. Then, to leading order in η, the expected drift of Theorem C.1 at a class transition, averaged over which transition occurs, exceeds that of the* `Bursty` *schedule by*

$$\underbrace{\frac{\eta\,\gamma_R}{4}\nabla\mathrm{SGA}(\theta)}_{\text{from } T_2} \;-\; \underbrace{\frac{\eta}{2}\Delta\mathcal{K}(\theta)}_{\text{from } T_3} + O(\eta^2).$$

For preconditioned steps with fixed P the same holds with SGA *replaced by* SGA_P*, ∇ by $P\nabla$, and $\mathcal{K}^\pi$ by $P\mathcal{K}^\pi_P = \mathbb{E}_\pi\big[H^{(c)}PG^{(n)} - H^{(n)}PG^{(c)}\big]$.*

To connect gradient alignment to the quantity we measure, the SA, let

$$\mathrm{SA}_c(\theta) := \frac{1}{R-2}\sum_{n\notin\{c,s(c)\}} L_n(\theta) - L_{s(c)}(\theta)$$

be the population successor advantage during a burst on class c, i.e. Definition 1 with expected losses L_n in place of held-out sample means.

Theorem C.3 (Effective SGA increases SA)**.** *Let $R \geq 3$ and let the class losses be twice continuously differentiable with locally bounded Hessians. Let $U_c(\theta)$ denote the optimizer's update direction for a step on class c, conditional on its current optimizer state, i.e. $\theta^+_c = \theta - \eta U_c(\theta)$. Then*

$$\mathrm{SA}_c(\theta^+_c) - \mathrm{SA}_c(\theta) = \eta\,\big\langle w_c(\theta),\, U_c(\theta)\big\rangle + O\big(\eta^2\|U_c(\theta)\|^2\big),$$

and hence averaging over the current class gives

$$\frac{1}{R}\sum_c\Big[\mathrm{SA}_c(\theta^+_c) - \mathrm{SA}_c(\theta)\Big] = \eta\,\mathrm{SGA}_{\mathrm{eff}}(\theta) + O(\eta^2), \qquad \mathrm{SGA}_{\mathrm{eff}}(\theta) := \frac{1}{R}\sum_c\big\langle w_c(\theta), U_c(\theta)\big\rangle.$$

For gradient descent, $U_c = G^{(c)}$ and $\mathrm{SGA}_{\mathrm{eff}} = \mathrm{SGA}$; for a fixed symmetric positive definite preconditioner P, $U_c = PG^{(c)}$ and $\mathrm{SGA}_{\mathrm{eff}} = \mathrm{SGA}_P$.

Interpretation. Theorem C.2 says that, compared to `Bursty` training, the `Ring` schedule induces a consistent, strictly non-negative pressure increasing SGA at every class transition. The second, curvature interaction term T_3 is generally not related to anticipation. Hence the theory predicts that anticipation depends on curvature geometry and thus on the optimizer, but does not by itself say in which direction. Theorem C.3 describes how gradient alignment reaches successor advantage: each update on the current class changes SA, to first order, by η times the alignment of that update with the successor-contrast vector w_c. Averaging over the cycle, this rate is $\eta\,\mathrm{SGA}_{\mathrm{eff}}$. Theorem C.3 holds for any U_c, for example U_c produced via nonlinear gradient transformations, or dependence on optimizer state. Together, the two theorems say that `Ring` training raises SGA, and that a larger $\mathrm{SGA}_{\mathrm{eff}}$ makes every burst raise the SA more.

Remark 1 (Accumulation of Local Implicit Bias)**.** *Theorem C.2 is a statement about one transition, while our experimental results (Figure 1) track the evolution of anticipation over* $100k$ *steps. We*

hypothesize that the former accumulates into the latter because the `Ring`*'s successor map is* deterministic*: the transition drift difference is a fixed function of θ and is applied with the same ordering at every transition, so contributions add consistently and grow with the number of bursts. In contrast, in* `Bursty` *training, the successor is re-drawn each burst randomly, hence local alignments cancel each other.*

A common technique in the literature to explain the accumulation of local bias is "SGD minimises a modified loss" type arguments (Smith et al., 2021b). However, we believe that these are not applicable here: T_3 is not in general a gradient field (the antisymmetric part of its Jacobian is $2[H^{(c)}, H^{(n)}]$, so it has a potential only where the class Hessians commute). We therefore work at the level of transitions. This choice is in accordance with Dherin (2023), who also identify T_3 as a local bias in a slightly different setting.

The above theorems are applicable to SGD and updates with fixed preconditioning. This regime applies to AdamW approximately: its preconditioner is not fixed but varies slowly when $\beta_2 \approx 1$. Our theory does not cover momentum, but Figure 4 (right) shows momentum is not necessary for anticipation. In Muon, the update direction is not a linear image of the gradient, but an orthogonalised step. Therefore, Theorem C.1 and Theorem C.2 don't apply to Muon. However, Theorem C.3 tells us that high $\mathrm{SGA}_{\mathrm{eff}}$ for any update direction U_c, including Muon's is predictive of anticipation. Hence for Muon, our theory produces a test: high $\mathrm{SGA}_{\mathrm{eff}}$ should mean anticipation, but we provide no explanation of how the `Ring` schedule might raise $\mathrm{SGA}_{\mathrm{eff}}$.

C.2.3 Proof of Theorem C.1: local drift of bursty training

Our proof builds on the shuffled-SGD implicit bias result from Roberts (2021b). We first replicate the statement and proof in our notation, and then extend it to bursty training.

Theorem C.4 (Implicit Bias of Shuffled SGD (Roberts, 2021b))**.** *Let $B_{current}$ and B_{next} be two independent minibatches of size $N/2$, drawn i.i.d. from $p(x)$. Then, to leading order in η,*

$$\mathbb{E}[\delta g_i^{shuffled}] = -\frac{\eta}{2N}\,\partial_i \operatorname{tr}\Sigma(\theta)$$

where the expectation is taken over the randomness in the minibatches B_c and B_n. Equivalently, the expected SGD update differs from the corresponding GD update by a drift term proportional to the gradient of the trace of the gradient covariance:

$$\mathbb{E}[\delta g^{shuffled}] = -\frac{\eta}{2N}\,\nabla_\theta \operatorname{tr}\Sigma(\theta).$$

The proof requires the following Lemma that our bursty result will rely on as well.

Lemma 1. *When comparing GD to SGD for any two batches B_c and B_n, we have*

$$\delta g_i = -\frac{\eta}{8}\left[\partial_i \|g^{(B_c)} - g^{(B_n)}\|^2 + 4\big(g_j^{(B_n)} h_{ij}^{(B_c)} - g_j^{(B_c)} h_{ij}^{(B_n)}\big)\right] + O(\eta^2).$$

Proof. Let the initial parameter be θ_0. As in Section 3 of Roberts (2021b), consider two subsequent SGD updates with minibatches B_c and B_n, each of size $N/2$:

$$\theta_1^{i(B_c)} = \theta_0^i - \eta g_i^{(B_c)}(\theta_0), \qquad \theta_2^{i(B_c,B_n)} = \theta_1^{i(B_c)} - \eta g_i^{(B_n)}(\theta_1^{(B_c)}). \tag{6}$$

We compare this to the equivalent of two full-data GD updates using the union batch $B_c \cup B_n$ of size N:

$$\theta_1^{i(B_c\cup B_n)} = \theta_0^i - \eta g_i^{(B_c\cup B_n)}(\theta_0), \qquad \theta_2^{i(B_c\cup B_n, B_c\cup B_n)} = \theta_1^{i(B_c\cup B_n)} - \eta g_i^{(B_c\cup B_n)}(\theta_1^{(B_c\cup B_n)}). \tag{7}$$

We now expand both trajectories to second order in η. Substituting the first equation of equation 6 into the second gives

$$\theta_2^{i(B_c,B_n)} = \theta_0^i - \eta g_i^{(B_c)}(\theta_0) - \eta g_i^{(B_n)}(\theta_{1(B_c)}).$$

A one-step Taylor expansion of the last gradient around θ_0 gives:

$$g_i^{(B_n)}(\theta_1^{(B_c)}) = g_i^{(B_n)}(\theta_0) + \partial_j g_i^{(B_n)}(\theta_0)\,(\theta_1^{j(B_c)} - \theta_0^j) + O(\eta^2).$$

Since $\theta_1^{j(B_c)} - \theta_0^j \approx -\eta g_j^{(B_c)}(\theta_0)$ and $\partial_j g_i^{(B_n)} = h_{ij}^{(B_n)}$ for small enough η, this becomes

$$g_i^{(B_n)}(\theta_1^{(B_c)}) = g_i^{(B_n)}(\theta_0) - \eta h_{ij}^{(B_n)}(\theta_0) g_j^{(B_c)}(\theta_0) + O(\eta^2).$$

Hence

$$\theta_2^{i(B_c,B_n)} - \theta_0^i = -\eta\big(g_i^{(B_c)} + g_i^{(B_n)}\big) + \eta^2 h_{ij}^{(B_n)} g_j^{(B_c)} + O(\eta^3), \tag{8}$$

where from now on, all gradients and Hessians are evaluated at θ_0 unless explicitly stated otherwise. Now, we expand of the one-step GD iterate on $B_c \cup B_n$. Similarly,

$$\theta_2^{i(B_c\cup B_n, B_c\cup B_n)} = \theta_0^i - \eta g_i^{(B_c\cup B_n)}(\theta_0) - \eta g_i^{(B_c\cup B_n)}(\theta_1^{(B_c\cup B_n)}).$$

Taylor expanding the second gradient around θ_0 gives

$$g_i^{(B_c\cup B_n)}(\theta_1^{(B_c\cup B_n)}) = g_i^{(B_c\cup B_n)} - \eta h_{ij}^{(B_c\cup B_n)} g_j^{(B_c\cup B_n)} + O(\eta^2),$$

and therefore

$$\theta_2^{i(B_c\cup B_n, B_c\cup B_n)} - \theta_0^i = -2\eta g_i^{(B_c\cup B_n)} + \eta^2 h_{ij}^{(B_c\cup B_n)} g_j^{(B_c\cup B_n)} + O(\eta^3). \tag{9}$$

Because $|B_c| = |B_n| = N/2$, the union-batch gradient and Hessian are simple averages:

$$g_i^{(B_c\cup B_n)} = \frac{1}{2}\big(g_i^{(B_c)} + g_i^{(B_n)}\big), \qquad h_{ij}^{(B_c\cup B_n)} = \frac{1}{2}\big(h_{ij}^{(B_c)} + h_{ij}^{(B_n)}\big).$$

Substituting these identities into equation 9 yields

$$\theta_2^{i(B_c\cup B_n, B_c\cup B_n)} - \theta_0^i = -\eta\big(g_i^{(B_c)} + g_i^{(B_n)}\big) + \frac{\eta^2}{4}\big(h_{ij}^{(B_c)} + h_{ij}^{(B_n)}\big)\big(g_j^{(B_c)} + g_j^{(B_n)}\big) + O(\eta^3). \tag{10}$$

Since $\big(\theta_2^{i(B_c\cup B_n, B_c\cup B_n)} - \theta_0^i\big) - \big(\theta_2^{i(B_c,B_n)} - \theta_0^i\big) \equiv -\eta\,\delta g_i$, Subtracting equation 8 from equation 10, the $O(\eta)$ terms cancel, and we obtain

$$-\eta\,\delta g_i = \eta^2\left[\frac{1}{4}(h_{ij}^{(B_c)} + h_{ij}^{(B_n)})(g_j^{(B_c)} + g_j^{(B_n)}) - h_{ij}^{(B_n)} g_j^{(B_c)}\right] + O(\eta^3). \tag{11}$$

Expanding the bracket and using the identity $h_{ij} g_j = \partial_i\big(\frac{1}{2}\|g\|^2\big)$, separately to the batch gradients $g^{(B_c)}$ and $g^{(B_n)}$, we get

$$\delta g_i = -\frac{\eta}{8}\partial_i\Big(\|g^{(B_c)}\|^2 + \|g^{(B_n)}\|^2\Big) - \frac{\eta}{4}\Big(h_{ij}^{(B_c)} g_j^{(B_n)} - 3h_{ij}^{(B_n)} g_j^{(B_c)}\Big) + O(\eta^2). \tag{12}$$

Adding and subtracting $2g^{(B_c)} \cdot g^{(B_n)}$ inside the derivative gives:

$$\|g^{(B_c)}\|^2 + \|g^{(B_n)}\|^2 = \|g^{(B_c)} - g^{(B_n)}\|^2 + 2g^{(B_c)} \cdot g^{(B_n)}.$$

Since $\partial_i\big(g^{(B_c)} \cdot g^{(B_n)}\big) = h_{ij}^{(B_c)} g_j^{(B_n)} + h_{ij}^{(B_n)} g_j^{(B_c)}$, equation equation 12 becomes

$$\begin{aligned}\delta g_i &= -\frac{\eta}{8}\partial_i\|g^{(B_c)} - g^{(B_n)}\|^2 - \frac{\eta}{4}\partial_i\big(g^{(B_c)} \cdot g^{(B_n)}\big) - \frac{\eta}{4}\Big(h_{ij}^{(B_c)} g_j^{(B_n)} - 3h_{ij}^{(B_n)} g_j^{(B_c)}\Big) + O(\eta^2) \\ &= -\frac{\eta}{8}\partial_i\|g^{(B_c)} - g^{(B_n)}\|^2 - \frac{\eta}{2}\Big(g_j^{(B_n)} h_{ij}^{(B_c)} - g_j^{(B_c)} h_{ij}^{(B_n)}\Big) + O(\eta^2).\end{aligned} \tag{13}$$

Equivalently,

$$\delta g_i = -\frac{\eta}{8}\Big[\partial_i\|g^{(B_c)} - g^{(B_n)}\|^2 + 4\big(g_j^{(B_n)} h_{ij}^{(B_c)} - g_j^{(B_c)} h_{ij}^{(B_n)}\big)\Big] + O(\eta^2). \tag{14}$$

□

We now prove the implicit bias result for *shuffled* SGD.

Proof. of Theorem C.4. B_c and B_n are i.i.d. draws from the full mixture $p(x)$. For a minibatch of size $N/2$, the batch gradients satisfy:

$$\mathbb{E}[g_i^{(B_c)}] = G_i^{(\text{full})}, \qquad \text{Cov}(g_i^{(B_c)}, g_j^{(B_c)}) = \frac{2}{N}\Sigma_{ij}^{(\text{full})},$$

and the same for $g^{(B_n)}$. Since in the shuffled case, B_c and B_n are independent and identically distributed,

$$\mathbb{E}[g^{(B_c)} \cdot g^{(B_n)}] = \mathbb{E}[g^{(B_c)}] \cdot \mathbb{E}[g^{(B_n)}] = \|G^{(\text{full})}\|^2.$$

Also,

$$\mathbb{E}\Big[\|g^{(B_c)}\|^2\Big] = \|G^{(\text{full})}\|^2 + \frac{2}{N}\,\text{tr}\,\Sigma^{(\text{full})}.$$

Hence

$$\mathbb{E}\Big[\|g^{(B_c)} - g^{(B_n)}\|^2\Big] = \mathbb{E}\Big[\|g^{(B_c)}\|^2\Big] + \mathbb{E}\Big[\|g^{(B_n)}\|^2\Big] - 2\mathbb{E}\Big[g^{(B_c)} \cdot g^{(B_n)}\Big] = \frac{4}{N}\,\text{tr}\,\Sigma^{(\text{full})}.$$

For the antisymmetric term in equation 14, independence and identical distribution imply

$$\mathbb{E}\Big[g_j^{(B_n)} h_{ij}^{(B_c)}\Big] = \mathbb{E}[g_j^{(B_n)}]\mathbb{E}[h_{ij}^{(B_c)}] = G_j^{(\text{full})} H_{ij}^{(\text{full})},$$

and similarly $\mathbb{E}\Big[g_j^{(B_c)} h_{ij}^{(B_n)}\Big] = G_j^{(\text{full})} H_{ij}^{(\text{full})}$, so $\mathbb{E}\Big[g_j^{(B_n)} h_{ij}^{(B_c)} - g_j^{(B_c)} h_{ij}^{(B_n)}\Big] = 0$. Taking expectation in equation 14 and commuting ∂_i with expectation (justified by assumption (iv)), we conclude

$$\begin{aligned}
\mathbb{E}[\delta g_i^{\text{shuffled}}] &= -\frac{\eta}{8}\,\partial_i \mathbb{E}\Big[\|g^{(B_c)} - g^{(B_n)}\|^2\Big] + O(\eta^2) \\
&= -\frac{\eta}{8}\,\partial_i \left(\frac{4}{N}\,\text{tr}\,\Sigma^{(\text{full})}\right) + O(\eta^2) \\
&= -\frac{\eta}{2N}\,\partial_i\,\text{tr}\,\Sigma^{(\text{full})} + O(\eta^2). \qquad (15)
\end{aligned}$$

This corresponds to an isotropic bias toward regions of low gradient variance across the full dataset. □

We can now prove Theorem C.1. We first prove the version without preconditioning.

Proof of Theorem C.1 (without preconditioning). We start from Lemma 1:

$$\mathbb{E}[\delta g_i] = -\frac{\eta}{8}\,\partial_i\,\mathbb{E}\Big[\|g^{(B_c)} - g^{(B_n)}\|^2\Big] - \frac{\eta}{2}\,\mathbb{E}\Big[g_j^{(B_n)} h_{ij}^{(B_c)} - g_j^{(B_c)} h_{ij}^{(B_n)}\Big] + O(\eta^2). \qquad (16)$$

In the bursty setting, the two expectations in equation 16 are no longer symmetric under $B_c \leftrightarrow B_n$, because the two batches are drawn from different class-conditional distributions. Recall the class-conditional means and covariances

$$G_i^{(c)} = \mathbb{E}_{x\sim p(\cdot|c)}[g_i(\theta, x)], \qquad \Sigma_{ij}^{(c)} = \text{Cov}_{x\sim p(\cdot|c)}(g_i, g_j),$$

and similarly for class n (Section C.2.1). Since B_c is a minibatch of size $N/2$ drawn i.i.d. from $p(\cdot \mid c)$, its batch gradient satisfies

$$\mathbb{E}[g_i^{(B_c)}] = G_i^{(c)}, \qquad \text{Cov}(g_i^{(B_c)}, g_j^{(B_c)}) = \frac{2}{N}\Sigma_{ij}^{(c)}.$$

Hence

$$\mathbb{E}\Big[\|g^{(B_c)}\|^2\Big] = \|G^{(c)}\|^2 + \frac{2}{N}\,\text{tr}\,\Sigma^{(c)}. \qquad (17)$$

Similarly,

$$\mathbb{E}\Big[\|g^{(B_n)}\|^2\Big] = \|G^{(n)}\|^2 + \frac{2}{N}\,\text{tr}\,\Sigma^{(n)}. \qquad (18)$$

Because B_c and B_n are independent,

$$\mathbb{E}[g^{(B_c)} \cdot g^{(B_n)}] = \mathbb{E}[g^{(B_c)}] \cdot \mathbb{E}[g^{(B_n)}] = G^{(c)} \cdot G^{(n)}. \qquad (19)$$

Therefore,

$$\begin{aligned}\mathbb{E}\Big[\|g^{(B_c)} - g^{(B_n)}\|^2\Big] &= \mathbb{E}\Big[\|g^{(B_c)}\|^2\Big] + \mathbb{E}\Big[\|g^{(B_n)}\|^2\Big] - 2\,\mathbb{E}[g^{(B_c)} \cdot g^{(B_n)}] \\ &= \|G^{(c)}\|^2 + \frac{2}{N}\operatorname{tr}\Sigma^{(c)} + \|G^{(n)}\|^2 + \frac{2}{N}\operatorname{tr}\Sigma^{(n)} - 2\,G^{(c)} \cdot G^{(n)} \\ &= \|G^{(c)} - G^{(n)}\|^2 + \frac{2}{N}\operatorname{tr}\Sigma^{(c)} + \frac{2}{N}\operatorname{tr}\Sigma^{(n)}. \end{aligned} \tag{20}$$

Due to the assumption $\operatorname{tr}\Sigma^{(c)} \approx \operatorname{tr}\Sigma^{(n)} \approx \operatorname{tr}\Sigma_{\text{within}}$, equation 20 becomes

$$\mathbb{E}\Big[\|g^{(B_c)} - g^{(B_n)}\|^2\Big] = \|G^{(c)} - G^{(n)}\|^2 + \frac{4}{N}\operatorname{tr}\Sigma_{\text{within}}. \tag{21}$$

Substituting equation 21 into the first term of equation 16 gives

$$\begin{aligned}-\frac{\eta}{8}\partial_i\,\mathbb{E}\Big[\|g^{(B_c)} - g^{(B_n)}\|^2\Big] &= -\frac{\eta}{8}\partial_i\left(\|G^{(c)} - G^{(n)}\|^2 + \frac{4}{N}\operatorname{tr}\Sigma_{\text{within}}\right) \\ &= -\frac{\eta}{8}\partial_i\|G^{(c)} - G^{(n)}\|^2 - \frac{\eta}{2N}\partial_i\operatorname{tr}\Sigma_{\text{within}}. \end{aligned} \tag{22}$$

Since B_c and B_n are independent, and $g^{(B_n)}$ depends only on B_n while $h^{(B_c)}$ depends only on B_c, we may factor expectations:

$$\mathbb{E}[g_j^{(B_n)} h_{ij}^{(B_c)}] = \mathbb{E}[g_j^{(B_n)}]\,\mathbb{E}[h_{ij}^{(B_c)}] = G_j^{(n)} H_{ij}^{(c)}. \tag{23}$$

Therefore

$$\mathbb{E}\Big[g_j^{(B_n)} h_{ij}^{(B_c)} - g_j^{(B_c)} h_{ij}^{(B_n)}\Big] = G_j^{(n)} H_{ij}^{(c)} - G_j^{(c)} H_{ij}^{(n)}. \tag{24}$$

Substituting equation 24 into the second term of equation 16 yields

$$-\frac{\eta}{2}\,\mathbb{E}\Big[g_j^{(B_n)} h_{ij}^{(B_c)} - g_j^{(B_c)} h_{ij}^{(B_n)}\Big] = -\frac{\eta}{2}\Big(G_j^{(n)} H_{ij}^{(c)} - G_j^{(c)} H_{ij}^{(n)}\Big). \tag{25}$$

Combining equation 22 and equation 25 in equation 16, we obtain

$$\mathbb{E}[\delta g_i^{\text{bursty}}] = -\frac{\eta}{2N}\,\partial_i \operatorname{tr}\Sigma_{\text{within}} - \frac{\eta}{8}\,\partial_i\|G^{(c)} - G^{(n)}\|^2 - \frac{\eta}{2}\Big(G_j^{(n)} H_{ij}^{(c)} - G_j^{(c)} H_{ij}^{(n)}\Big) + O(\eta^2).$$

□

To prove the remaining, preconditioned part of Theorem C.1, we use a change of coordinates that we establish below.

Lemma 2 (Preconditioning is SGD in $P^{1/2}$-coordinates)**.** *Let P be symmetric positive definite and set $\phi = P^{-1/2}\theta$, $\tilde{\ell}(\phi, x) = \ell(P^{1/2}\phi, x)$, and similarly $\tilde{L}_n(\phi) = L_n(P^{1/2}\phi)$. Then:*

(a) *$\nabla_\phi \tilde{\ell} = P^{1/2}\nabla_\theta \ell$ and $\nabla_\phi^2 \tilde{\ell} = P^{1/2}(\nabla_\theta^2 \ell)P^{1/2}$. Furthermore, the class quantities transform as $\tilde{G}^{(a)} = P^{1/2}G^{(a)}$, $\tilde{H}^{(a)} = P^{1/2}H^{(a)}P^{1/2}$, $\tilde{\Sigma}^{(a)} = P^{1/2}\Sigma^{(a)}P^{1/2}$, and $\operatorname{tr}\tilde{\Sigma}^{(a)} = \operatorname{tr}(P\Sigma^{(a)})$.*

(b) *$\theta' = \theta - \eta P\, g(\theta)$ if and only if $\phi' = \phi - \eta\,\tilde{g}(\phi)$, where $\tilde{g} = \nabla_\phi \tilde{\ell}$ is the mini-batch gradient in ϕ-coordinates. A preconditioned step in θ is a plain gradient step in ϕ.*

(c) *For any differentiable F, $\nabla_\phi\big[F(P^{1/2}\phi)\big] = P^{1/2}\nabla_\theta F$.*

Proof. (a) and (c) are the chain rule for the linear map $\theta = P^{1/2}\phi$, whose Jacobian is the symmetric matrix $P^{1/2}$. Expectations and covariances over x commute with this fixed linear map, which gives the transformation of G, H and Σ. We have $\operatorname{tr}(P^{1/2}\Sigma P^{1/2}) = \operatorname{tr}(P\Sigma)$ by the cyclicity of the trace. For (b), multiply $\theta' = \theta - \eta P g$ by $P^{-1/2}$: $\phi' = \phi - \eta P^{1/2} g = \phi - \eta\tilde{g}$ by (a). □

Proof of the preconditioned clause of Theorem C.1. By Lemma 2(b), both the two preconditioned SGD steps and the two preconditioned GD steps on $B_c \cup B_n$ are plain (S)GD steps in ϕ-coordinates.

Hence the $P = I$ case of the theorem applies to $\tilde{\ell}$ and gives (with tildes denoting ϕ-coordinate quantities):

$$\mathbb{E}[\delta\tilde{g}] = -\frac{\eta}{2N}\nabla_\phi \operatorname{tr}\tilde{\Sigma}_{\text{within}} - \frac{\eta}{8}\nabla_\phi\|\tilde{G}^{(c)} - \tilde{G}^{(n)}\|^2 - \frac{\eta}{2}\big(\tilde{H}^{(c)}\tilde{G}^{(n)} - \tilde{H}^{(n)}\tilde{G}^{(c)}\big) + O(\eta^2). \quad (26)$$

We can reformulate each term with Lemma 2(a) and (c): $\operatorname{tr}\tilde{\Sigma}_{\text{within}} = \operatorname{tr}(P\Sigma_{\text{within}})$; $\|\tilde{G}^{(c)} - \tilde{G}^{(n)}\|^2 = (G^{(c)} - G^{(n)})^\top P(G^{(c)} - G^{(n)}) = \|G^{(c)} - G^{(n)}\|_P^2$; $\nabla_\phi = P^{1/2}\nabla_\theta$; and $\tilde{H}^{(c)}\tilde{G}^{(n)} = P^{1/2}H^{(c)}P^{1/2}P^{1/2}G^{(n)} = P^{1/2}H^{(c)}PG^{(n)}$. Finally, we transform δg. It is defined in Section C.2.3 via the difference of displacements after two steps:

$$\delta\theta_2^{i(B_c\cup B_n, B_c\cup B_n)} - \delta\theta_2^{i(B_c,B_n)} = -\eta\,\delta g_i.$$

Displacements transform as $\theta_2 - \theta_0 = P^{1/2}(\phi_2 - \phi_0)$, so $\delta g = P^{1/2}\delta\tilde{g}$. Multiplying Equation (26) by $P^{1/2}$ gives

$$\mathbb{E}[\delta g] = -\frac{\eta}{2N}\,P\nabla\operatorname{tr}(P\Sigma_{\text{within}}) - \frac{\eta}{8}\,P\nabla\|G^{(c)} - G^{(n)}\|_P^2 - \frac{\eta}{2}\,P\big(H^{(c)}PG^{(n)} - H^{(n)}PG^{(c)}\big) + O(\eta^2),$$

as claimed. Without assumption (v) the same computation gives the variance term as

$$-\tfrac{\eta}{4N}\,P\nabla\big(\operatorname{tr}P\Sigma^{(c)} + \operatorname{tr}P\Sigma^{(n)}\big).$$

□

Interpretation of Theorem C.1. Our theorem shows that three distinct effects appear in the bursty setting.

1. **Variance regularization**: the same gradient noise reduction mechanism as in shuffled SGD, but restricted to within-class variance
2. **Alignment pressure**: a symmetric force that pushes $G^{(c)}(\theta)$ and $G^{(n)}(\theta)$ toward each other.
3. **Curvature interaction**: an antisymmetric term capturing how a gradient step on one class modifies the landscape encountered by the next.

The main candidate for driving anticipation is T_2:$-\frac{\eta}{8}\,\partial_i\|G^{(c)} - G^{(n)}\|^2$. Under cyclic training, that is, repeatedly cycling through $c_{\text{current}}, c_{\text{next}}$ and other classes in fixed order, T_2 is repeatedly applied and encourages alignment of class-conditional gradients. When $G^{(c)}(\theta) \approx G^{(n)}(\theta)$, updates from one class can become beneficial for the next, producing anticipatory recovery. However, the remaining two terms may also impact the overall effect. For example, T_3 introduces curvature-dependent interference, which may reinforce or oppose alignment.

Looking at only two gradient steps, we cannot differentiate between `Bursty` and `Ring` training schedules, where the effects of the individual terms may accumulate differently. Our next result takes into account the training schedules as well.

C.3 Proof of Theorem C.2: Ring training can increase SGA

Theorem C.1 This section proves our result on the difference of the implicit biases of `Ring` and `Bursty` training, which is visible once one averages over the training schedules. Recall $R = |C|$ (Section C.2.1) and assume $R \geq 3$ throughout, so that the control average in Equation (5) is over a nonempty set.

Reduction to a single transition. Within a burst, the two consecutive mini-batches come from the same class, so $G^{(n)} = G^{(c)}$, $H^{(n)} = H^{(c)}$ and $T_2 = T_3 = 0$ in Theorem C.1. This is because all cross-class drift occurs at class transitions. Each class transition can be described by two steps. We therefore condition on a transition. Under Assumption (vi), the current class c is uniform, and the law of the next class given a transition is

$$\pi^{\text{Ring}}(n \mid c) = \mathbf{1}\{n = s(c)\}, \qquad \pi^{\text{Bursty}}(n \mid c) = \frac{\mathbf{1}\{n \neq c\}}{R-1},$$

as stated in Definition 2. Our quantity of interest is $\mathbb{E}_{\pi^{\text{Ring}}}\big[\mathbb{E}[\delta g^{\text{bursty}}]\big] - \mathbb{E}_{\pi^{\text{Bursty}}}\big[\mathbb{E}[\delta g^{\text{bursty}}]\big]$, where the inner expectation is over mini-batch randomness at fixed (c, n) as in Theorem C.1 and the outer one is over the transition.

Proof of Theorem C.2. We prove the theorem in 4 steps. *Step 1: averaging operators.* For a function $f(c, n)$ of the ordered pair, write

$$\mathbb{E}_\pi[f] := \frac{1}{R} \sum_{c \in C} \sum_{n \in C} \pi(n \mid c)\, f(c, n),$$

so that, for the two schedules,

$$\mathbb{E}_{\pi^{\mathrm{Ring}}}[f] = \frac{1}{R} \sum_c f\big(c, s(c)\big), \qquad \mathbb{E}_{\pi^{\mathrm{Bursty}}}[f] = \frac{1}{R(R-1)} \sum_c \sum_{n \neq c} f(c, n).$$

Both are finite averages, hence linear in f and commuting with ∂_i. Both are applied to the drift of Theorem C.1 evaluated at the same current parameter θ, so the $O(\eta^2)$ remainder, being uniform over the finitely many pairs, remains $O(\eta^2)$ after averaging.

Step 2: both schedules have uniform marginals. If $f(c, n) = \varphi(c)$ depends on the current class only, $\mathbb{E}_\pi[f] = \frac{1}{R}\sum_c \sum_n \pi(n|c)\varphi(c) = \frac{1}{R}\sum_c \varphi(c)\,(\sum_n \pi(n|c)) = \frac{1}{R}\sum_c \varphi(c)$ because $\pi(n|c)$ is a probability distribution. If $f(c, n) = \psi(n)$ depends on the next class only, then under the `Ring` $\mathbb{E}_{\pi^{\mathrm{Ring}}}[f] = \frac{1}{R}\sum_c \psi(s(c)) = \frac{1}{R}\sum_a \psi(a)$, because s is a bijection of C and $a = s(c)$ runs over every class once. Under `Bursty`, exchanging the order of summation,

$$\mathbb{E}_{\pi^{\mathrm{Bursty}}}[f] = \frac{1}{R(R-1)} \sum_n \psi(n)\, \#\{c :\, c \neq n\} = \frac{1}{R(R-1)}\,(R-1) \sum_n \psi(n) = \frac{1}{R} \sum_n \psi(n).$$

Hence, by linearity, *every f of the form $\varphi(c) + \psi(n)$ has the same average under both schedules and drops out of the difference* $\mathbb{E}_{\pi^{\mathrm{Ring}}}[f] - \mathbb{E}_{\pi^{\mathrm{Bursty}}}[f]$. This is used three times below.

Step 3: the drift for a fixed pair. For a fixed transition (c, n), Theorem C.1 (in the form Equation (20), i.e. before assumption (v) is used to symmetrise the variance term) gives

$$\mathbb{E}[\delta g_i^{\mathrm{bursty}} \mid c, n] = T_1(c, n) + T_2(c, n) + T_3(c, n) + O(\eta^2),$$

$$T_1(c, n) = -\frac{\eta}{4N}\, \partial_i\big(\operatorname{tr} \Sigma^{(c)} + \operatorname{tr} \Sigma^{(n)}\big),$$
$$T_2(c, n) = -\frac{\eta}{8}\, \partial_i \|G^{(c)} - G^{(n)}\|^2,$$
$$T_3(c, n) = -\frac{\eta}{2}\big(G_j^{(n)} H_{ij}^{(c)} - G_j^{(c)} H_{ij}^{(n)}\big),$$

and we compute $\mathbb{E}_{\pi^{\mathrm{Ring}}} - \mathbb{E}_{\pi^{\mathrm{Bursty}}}$ of each term.

T_1. $T_1(c, n) = -\frac{\eta}{4N}\partial_i \operatorname{tr} \Sigma^{(c)} - \frac{\eta}{4N}\partial_i \operatorname{tr} \Sigma^{(n)}$ is of the form $\varphi(c) + \psi(n)$, so by Step 1 its difference vanishes. In particular the `Ring`-minus-`Bursty` comparison does not require assumption (v).

T_2. Expand $\|G^{(c)} - G^{(n)}\|^2 = \|G^{(c)}\|^2 + \|G^{(n)}\|^2 - 2\langle G^{(c)}, G^{(n)}\rangle$. The two norm terms are of the form $\varphi(c) + \psi(n)$ and vanish from the difference by Step 1, leaving

$$\mathbb{E}_{\pi^{\mathrm{Ring}}} \|G^{(c)} - G^{(n)}\|^2 - \mathbb{E}_{\pi^{\mathrm{Bursty}}} \|G^{(c)} - G^{(n)}\|^2 = -2\Big(\mathbb{E}_{\pi^{\mathrm{Ring}}} \langle G^{(c)}, G^{(n)}\rangle - \mathbb{E}_{\pi^{\mathrm{Bursty}}} \langle G^{(c)}, G^{(n)}\rangle\Big). \tag{27}$$

By Step 1 the right hand side becomes

$$\frac{1}{R} \sum_c \Big[\langle G^{(c)}, G^{(s(c))}\rangle - \frac{1}{R-1} \sum_{n \neq c} \langle G^{(c)}, G^{(n)}\rangle\Big].$$

Since $s(c) \neq c$, the set $\{n : n \neq c\}$ is the disjoint union of $\{s(c)\}$ and $\{n : n \notin \{c, s(c)\}\}$, so for each c the bracket equals

$$\Big(1 - \tfrac{1}{R-1}\Big)\langle G^{(c)}, G^{(s(c))}\rangle - \tfrac{1}{R-1} \sum_{n \notin \{c, s(c)\}} \langle G^{(c)}, G^{(n)}\rangle = \gamma_R \Big[\langle G^{(c)}, G^{(s(c))}\rangle - \frac{1}{R-2} \sum_{n \notin \{c, s(c)\}} \langle G^{(c)}, G^{(n)}\rangle\Big],$$

where the second form factors $\gamma_R = \frac{R-2}{R-1}$ out of both coefficients ($1 - \frac{1}{R-1} = \frac{R-2}{R-1}$ and $\frac{1}{R-1} = \frac{R-2}{R-1} \cdot \frac{1}{R-2}$). Averaging over c, the bracket is exactly the summand of Equation (2), so the inner-product difference is $\gamma_R\, \mathrm{SGA}(\theta)$ and equation 27 is equal to $-2\gamma_R \mathrm{SGA}(\theta)$. Since ∂_i commutes with the finite averages,

$$\mathbb{E}_{\pi^{\mathrm{Ring}}}[T_2] - \mathbb{E}_{\pi^{\mathrm{Bursty}}}[T_2] = -\frac{\eta}{8}\, \partial_i\big(-2\gamma_R \mathrm{SGA}\big) = +\frac{\eta\gamma_R}{4}\, \partial_i \mathrm{SGA}(\theta).$$

T_3. $T_3(c,n) = -\frac{\eta}{2}\big(G_j^{(n)} H_{ij}^{(c)} - G_j^{(c)} H_{ij}^{(n)}\big)$ is bilinear in the pair and does not split as $\varphi(c) + \psi(n)$, so no cancellation occurs; applying the two operators of Step 1 directly,

$$\mathbb{E}_{\pi^{\text{Ring}}}[T_3] = -\frac{\eta}{2}\cdot\frac{1}{R}\sum_c \big(G_j^{(s(c))} H_{ij}^{(c)} - G_j^{(c)} H_{ij}^{(s(c))}\big) = -\frac{\eta}{2}\mathcal{K}_i^{\text{Ring}}, \qquad \mathbb{E}_{\pi^{\text{Bursty}}}[T_3] = -\frac{\eta}{2}\mathcal{K}_i^{\text{Bursty}},$$

so the T_3 difference is $-\frac{\eta}{2}\Delta\mathcal{K}_i$. Adding the three differences gives the displayed drift difference.

Step 4: preconditioning. Repeat Steps 1–3 starting from the preconditioned form of Theorem C.1. T_1 is $-\frac{\eta}{4N}P\nabla(\operatorname{tr} P\Sigma^{(c)} + \operatorname{tr} P\Sigma^{(n)})$, again of the form $\varphi(c) + \psi(n)$, and cancels. For T_2 expand $\|G^{(c)} - G^{(n)}\|_P^2 = \|G^{(c)}\|_P^2 + \|G^{(n)}\|_P^2 - 2\langle G^{(c)}, PG^{(n)}\rangle$. The norms cancel by Step 1, and the previous inner-product computation goes through verbatim with $\langle G^{(c)}, G^{(n)}\rangle$ replaced by the symmetric bilinear form $\langle G^{(c)}, PG^{(n)}\rangle$. This gives the difference $\gamma_R \cdot \frac{1}{R}\sum_c \langle w_c, PG^{(c)}\rangle = \gamma_R \mathrm{SGA}_P$ by Definition 3. Hence the T_2 difference is $-\frac{\eta}{8}P\nabla(-2\gamma_R \mathrm{SGA}_P) = \frac{\eta\gamma_R}{4}P\nabla\mathrm{SGA}_P$. T_3 becomes $-\frac{\eta}{2}P\,\mathbb{E}_\pi[H^{(c)}PG^{(n)} - H^{(n)}PG^{(c)}] = -\frac{\eta}{2}P\mathcal{K}_P^\pi$ under either schedule, so its difference is $-\frac{\eta}{2}P\Delta\mathcal{K}_P$. □

Interpretation of Theorem C.2. Theorem C.2 compares the implicit biases of the `Ring` and `Bursty` schedules. Theorem C.1, which isolates implicit bias with respect to gradient descent. In Theorem C.2, substracting `Bursty`'s expectation from `Ring`'s removes the GD baseline. Our final theorem relates SGA to our metric of anticipation, SA.

C.4 Proof of Theorem C.3: Effective SGA increases SA

Recall from Section C.2.1 that L_n denotes the population loss of class n, $G^{(n)} = \nabla L_n$, $H^{(n)} = \nabla^2 L_n$, and that $w_c = G^{(s(c))} - \frac{1}{R-2}\sum_{n\notin\{c,s(c)\}} G^{(n)}$ is the successor-contrast vector of Definition 3.

Proof of Theorem C.3. SA_c is a fixed linear combination of the class losses, so it is twice continuously differentiable with

$$\nabla\mathrm{SA}_c(\theta) = \frac{1}{R-2}\sum_{n\notin\{c,s(c)\}} G^{(n)}(\theta) - G^{(s(c))}(\theta) = -w_c(\theta),$$
$$\nabla^2\mathrm{SA}_c(\theta) = \frac{1}{R-2}\sum_{n\notin\{c,s(c)\}} H^{(n)}(\theta) - H^{(s(c))}(\theta).$$

By Taylor's theorem with Lagrange remainder, we have that along the segment from θ to $\theta_c^+ = \theta - \eta U_c$, there exists ξ with

$$\mathrm{SA}_c(\theta_c^+) - \mathrm{SA}_c(\theta) = -\eta\,\langle\nabla\mathrm{SA}_c(\theta), U_c\rangle + \tfrac{\eta^2}{2}\,U_c^\top\nabla^2\mathrm{SA}_c(\xi)\,U_c \tag{28}$$
$$= \eta\,\langle w_c(\theta), U_c\rangle + \tfrac{\eta^2}{2}\,U_c^\top\nabla^2\mathrm{SA}_c(\xi)\,U_c. \tag{29}$$

Since the class Hessians are locally bounded, say by $\|H^{(n)}\| \le \Lambda$ on a neighbourhood of θ containing the segment for η small enough, we have that $\|\nabla^2\mathrm{SA}_c(\xi)\| \le 2\Lambda$. Hence $U_c^\top\nabla^2\mathrm{SA}_c(\xi)\,U_c$ is bounded by $\Lambda\eta^2\|U_c\|^2$, which is the stated $O(\eta^2\|U_c\|^2)$. Averaging the identity over the R classes gives $\frac{1}{R}\sum_c[\mathrm{SA}_c(\theta_c^+) - \mathrm{SA}_c(\theta)] = \eta\,\frac{1}{R}\sum_c\langle w_c, U_c\rangle + O(\eta^2) = \eta\,\mathrm{SGA}_{\text{eff}} + O(\eta^2)$. The $O(\eta^2)$ is because the remainder is a finite average of $O(\eta^2)$ terms.

For the two special cases: with $U_c = G^{(c)}$, $\mathrm{SGA}_{\text{eff}} = \frac{1}{R}\sum_c\langle w_c, G^{(c)}\rangle = \mathrm{SGA}$ by Equation (5). With $U_c = PG^{(c)}$ for a fixed symmetric positive definite P, $\mathrm{SGA}_{\text{eff}} = \frac{1}{R}\sum_c\langle w_c, PG^{(c)}\rangle = \mathrm{SGA}_P$ by Definition 3. □

Interpretation of Theorem C.3. Theorem C.3 does not assume anything about U_c, hence the relationship between $\mathrm{SGA}_{\text{eff}}$ and SA holds for any optimiser. The theorem by itself does not tell us whether anticipation emerges, but instead provides a test: if $\mathrm{SGA}_{\text{eff}}$ is large, we should have anticipation.

D FURTHER RESULTS AND ANALYSIS

D.1 DETAILS ON THE *"comparison to sequence models"* EXPERIMENT

In measuring agreement with a sequence model in Section 2, our goal was to test to what extent our trained Transformer M behaves like a sequence model. The only predictable information from the training history was the class of the next batch. Therefore, a perfect sequence model equals $p(x \mid c : x \in c)$, which corresponds to an average train loss of the conditional entropy $H(x|c)$. However, failing to reach this limit can happen due to reasons different from the core mechanism of anticipation; for example, limited model expressivity or limited training duration. Hence, in this experiment, we estimate the maximal anticipation achievable *by the given model and the used training duration*, giving a model- and experiment-dependent empirical limit of achievable anticipation.

Experiment details. We estimate this empirical limit of anticipation by artificially giving class information to the model in-context, creating an in-context oracle baseline. We augment our training and evaluation samples by prepending the (correct) class information, so the sample becomes `BOS @@C@@ {class_id} || {original_data} EOS`, where `class_id` is the numeric class index (0–9), encoded as UTF-8 bytes, `||` are separator tokens, and `{original_data}` is the original data sample. This format was chosen to make such patterns unlikely to accidentally appear in the data. We compute the training and evaluation losses via masking the class cue and separator tokens out, hence the model is not trained to predict the cue and separator. This ensures that the model is trained to match the class index to the data content. At evaluation time, this acts as an implicit conditioning on the correct class. Other than the above changes, our experimental configuration is identical to the default experiment (see Table 3).

Results. The results are shown in Figure 2. The theoretical limit was computed as 1.6913 (see Proposition 1), and the empirical limit at 100k steps is estimated as 1.7682. `Ring` achieves a training loss of 1.8938 ± 0.0175 at 100k, beating `Shuffled` with training loss 1.9791. This corresponds to realizing 29.63% of the achievable theoretical advantage, and 40.43% of the achievable empirical advantage.

D.2 GEOMETRIC TRAJECTORY OF MODEL WEIGHTS

We visualize the trajectory of model weights immediately before class transitions in a `Ring` training run. We use a 0.19M-parameter model with sequence length 8, 32 gradient steps per class, batch size 64, and learning rate 5e-5, trained for 100k steps. In this experiment, we enforce fixed burst lengths for simplicity. We save pre-transition checkpoints at the last 200 class transitions. For each checkpoint, we flatten the model parameters into a single weight vector. To remove slow training drift, we group checkpoints into complete 10-transition `Ring` loops and subtract the mean vector of each loop from its corresponding checkpoints. We then apply PCA to these "centered" weight vectors.

Figure 9 shows the first two principal components. The colored clusters correspond to the transition phase, or `Ring` edge. Their circular organization shows that the model weights enter a reproducible cyclic state across the ordered class transitions, rather than merely drifting monotonically during training. This geometry is is consistent with gradient alignment towards successor classes.

D.3 ANTICIPATION IN SECOND-ORDER SETUPS

We test whether anticipation emerges beyond first order Markov chains.

Experiment details. We implement a second-order transition rule over Wikipedia language classes. Instead of the next class being determined by the current class alone, it is determined by the ordered pair (previous class, current class). The transition table is a Latin square. We test a 5 and 6-class version, which have 25 and 36 states, respectively. The Latin square configurations are reported in Figure 10. For any fixed current class, each possible next class appears exactly once across predecessor classes. This makes the first-order marginal uninformative: knowing only the current class gives a uniform distribution over next classes. A model can only do better than chance if training has left it with usable information about the predecessor that led into the current burst. The tables are

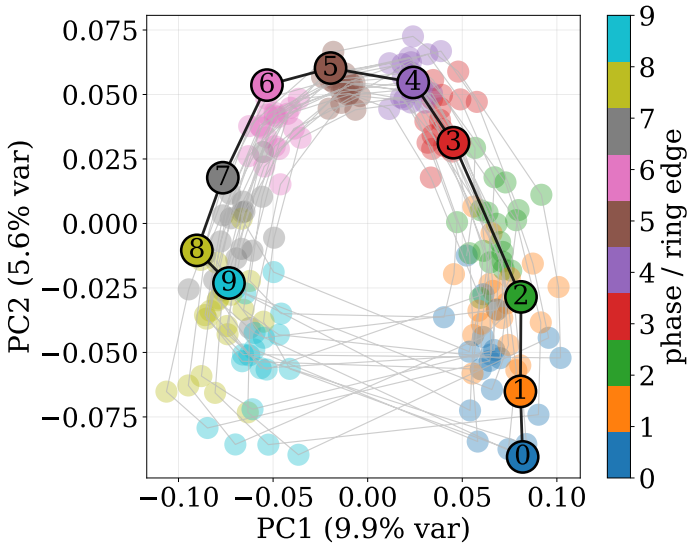


Figure 9: **Geometric structure of model weights.** We projected the flattened model weights recorded just before class transitions. 20 full cycles (200 transitions) are recorded, the last one is highlighted. Before PCA, the mean weight vector in each Ring loop is substracted to remove slow training drift. This removes most of the variance in data. The cyclic organization indicates that model weights enter a reproducible cyclic state across the ordered class transitions.

chosen to form a full orbit over joint states, so the deterministic class-pair dynamics cycle through all class pairs before repeating, giving a second-order equivalent of the original `Ring` configuration.

During a burst, the model sees batches from the current class. Contrary to the first-order experiments, here we enforce switches to happen deterministically, after a set burst length/number of gradient steps on a class (e.g. 32). When a switch happens, the tuple (previous class, current class) determines the next class, and after the switch, the latent state updates to (current class, next class). Self-successors, i.e., when next class is the same as the current class are possible, but these are excluded from Successor Advantage computation because no visible class switch occurs.

SA computation. We use a corrected version of the Successor Advantage metric, similar but not identical to the one in Section B.5. From the raw SA, we subtract a `Bursty` control SA. The `Bursty` control is a training stream that keeps the same burst structure but removes the predictable transition rule. This removes any anticipation effect due to factors not intrinsic to anticipation, such as class difficulty, current-class adaptation, or consequences of long same-class bursts. SA values are averaged over 16 seeds.

Results. Section D.3 reports the conducted experiments and corresponding SA results. The values are more modest than in the first-order setting. However, anticipation emerges in both 5-class and 6-class setups, and the ideal number of gradient steps (32) matches the first-order setup.

D.4 Anticipation in image modeling

We include MNIST experiments as an existence result showing that successor class anticipation is not specific to byte-level language modeling. Yang et al. (2024) showed that anticipatory recovery arises in individual image-cycling.

Experiment details. We train autoregressive image models on MNIST. Each digit class is treated as one class in the Markov chain. Images are downsampled to 14×14, quantized to 8 grayscale levels, and flattened into a sequence of 196 image tokens. We prepend a BOS token and append an EOS token, giving sequence length 198 and vocabulary size 10. The model never receives the digit

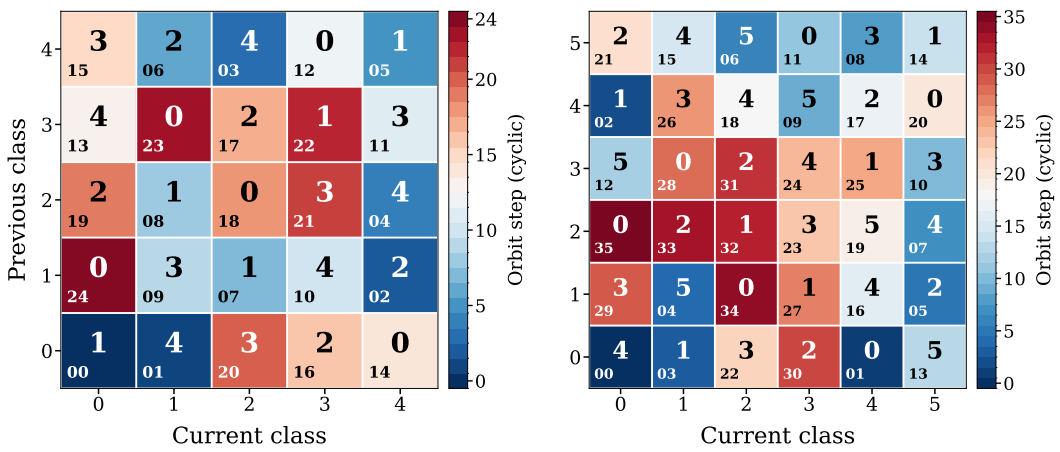


Figure 10: **The latin squared 2nd order transition function.** *(Left) 5-class configuration. (Right) 6-class configuration.* In both matrices, x and y axis labels show the individual state index. Colors and small-sized numbers in each cell show the index of state pairs (previous, current). Large-sized digits denote the next class from that cell pair. The orbit starts from the bottom left.

Number of classes	Gradient steps until switch	SA $\pm$ 1 std
5	16	-0.0053 ± 0.0045
5	**32**	$\mathbf{+0.0412 \pm 0.0081}$
5	64	$+0.0076 \pm 0.0105$
6	16	-0.0065 ± 0.0008
6	**32**	$\mathbf{+0.0152 \pm 0.0028}$
6	64	-0.0048 ± 0.0034

Table 4: **SA in the second-order setting.** Results are computed from 16 seeds per experiment. Both the 5 and 6-class setups display anticipation.

label as input or target, labels are used only to construct the class-structured training stream and to compute per-class evaluation losses.

The training protocols are the same as in Section B. In the `Ring` condition, digit classes are arranged in a seed-dependent cyclic order, and a transition deterministically moves to the next class in the `Ring`. In the `Bursty` condition, burst lengths are matched, but the successor class is uninformative. In the `Shuffled` condition, the same image stream is randomly permuted, removing temporal structure while preserving the multiset of examples.

We use a GPT-style Transformer (Vaswani et al., 2017) with hidden dimension 128, 4 layers, 4 attention heads, and intermediate dimension 512, for approximately 1.05M parameters. Models are trained for 100,000 steps with batch size 64, AdamW, learning rate 10^{-3}, weight decay 0.01, AMP mixed precision, and 1,000 warmup steps. We sweep expected burst lengths $\tau \in \{256, 512, 1024, 2048, 4096\}$ (expected gradient steps per burst in $\{4, 8, 16, 32, 64\}$) using 4-5 seeds per condition. We compute the shuffle-corrected Successor Advantage (SA) as defined in in Section B.5.

Results. Table Table 5 confirms that MNIST shows a small but clear first-order anticipation effect once bursts are sufficiently long: anticipation emerges between $\tau = 1024$ and $\tau = 2048$, corresponding to roughly 16–32 gradient steps per class burst.

These results should be treated as existence results for anticipation in image modeling. The experimental setup was not optimized extensively. The effect is smaller than in the Wikipedia experiments, but it appears under the same qualitative conditions: predictable first-order transitions, sufficiently long bursts, and per-class held-out evaluation.

Expected grad. steps until switch	Ring SA	Bursty SA
4	-0.001 ± 0.001	-0.039 ± 0.037
8	-0.001 ± 0.001	-0.039 ± 0.037
16	$+0.000 \pm 0.002$	-0.039 ± 0.038
32	$\mathbf{+0.013 \pm 0.008}$	-0.036 ± 0.034
64	$\mathbf{+0.028 \pm 0.011}$	-0.035 ± 0.037

Table 5: **Anticipation (positive SA) exists for MNIST image modeling.** Values are mean $\pm$ 1 std across 4-5 seeds. For each seed, SA is first averaged over the last 25% of training.

D.5 Details of the optimizer experiments

To fully report the observed optimizer dependence of anticipation, Table 6 more detailed information, including learning rates found through small hyperparameter sweeps.

Optimizer	LR	WD	$\mathrm{SA}^{\pm\mathrm{std}}$	Training adv	Train loss	Eval loss
SGD	5×10^{-5}	0.0	$-0.0096^{\pm 0.0026}$	-0.0132	$2.498^{\pm 0.024}$	$2.497^{\pm 0.020}$
SGD	5×10^{-5}	0.1	$-0.0010^{\pm 0.0005}$	-0.0672	$3.415^{\pm 0.052}$	$3.380^{\pm 0.046}$
AdamW	5×10^{-5}	0.1	$\mathbf{+0.2730^{\pm 0.0194}}$	$\mathbf{+0.0903}$	$1.895^{\pm 0.017}$	$2.271^{\pm 0.094}$
AdamW ($\beta_1 = 0$)	3×10^{-4}	0.1	$\mathbf{+0.1846^{\pm 0.0170}}$	$\mathbf{+0.1771}$	$1.904^{\pm 0.027}$	$2.525^{\pm 0.086}$
AdagradW	1×10^{-2}	0.0	$\mathbf{+0.2297^{\pm 0.0167}}$	$\mathbf{+0.1502}$	$1.803^{\pm 0.012}$	$2.300^{\pm 0.099}$
Muon	1×10^{-3}	0.1	$-0.0104^{\pm 0.0417}$	-0.0122	$2.103^{\pm 0.023}$	$2.279^{\pm 0.119}$

Table 6: **The effect of the optimizer on anticipation.** Only AdamW and AdagradW anticipate (positive SA) and achieve a positive training advantage in consequence. Shuffle-corrected SA is averaged over evaluations in the last 25% of training and reported as mean $\pm$ across-seed standard deviation (12 seeds per setting). Training advantage is based on proposition 1 and is computed as: Shuffled train loss minus Ring train loss, using late-window train-loss means.

Results. Only AdamW and AdagradW showed anticipation (positive SA) and a positive training advantage. Muon trained well but did not anticipate. For SGD, weight decay resulted in slow training. Even without weight decay, SGD's final training loss is high, hence the lack of anticipation is confounded with early training regimes potentially opposing anticipation. This is not the case with Muon, whose eval loss is competitive with AdamW's. AdagradW's cumulative squared-gradient accumulator can be viewed as a limiting long-timescale second-moment accumulator (Tieleman & Hinton, 2012), and it also produces strong anticipation.

Relationship to Yang et al. (2024). It is important to note that Yang et al. (2024) observed anticipatory recovery for SGD too. An explanation of the discrepancy could be that same-domain (CNN news) individual-documents naturally have more similar Hessians than samples from different languages, and that pretraining and scale further reduces the Hessian mismatch. It is also possible that training SGD longer in our setup would have produced anticipation.

Details on measuring T_3 interference. To test whether optimizer-dependent anticipation is explained by curvature-driven interference, we directly estimated the alignment term (T_2) and the order-dependent interference term (T_3) during SGD and AdamW training, using 12 independent seeds per optimizer. We logged these quantities every 5000 iterations using Hessian-vector products on adjacent class pairs $c_{\mathrm{current}} \to c_{\mathrm{next}}$. For each pair, we computed class gradients and Hessian-vector products with the corresponding class Hessians, giving

$$T_3 = \frac{1}{2} H^{(\mathrm{current})} G^{(\mathrm{next})} - \frac{1}{2} H^{(\mathrm{next})} G^{(\mathrm{current})}.$$

T_2 was computed using the equivalent expanded form of the alignment-gradient term,

$$T_2 = -\frac{1}{4} H^{(\mathrm{current})} \big(G^{(\mathrm{current})} - G^{(\mathrm{next})}\big) + \frac{1}{4} H^{(\mathrm{next})} \big(G^{(\mathrm{current})} - G^{(\mathrm{next})}\big).$$

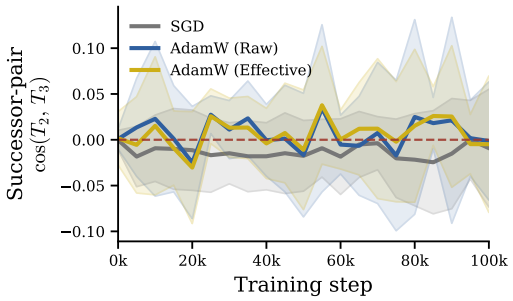


Figure 11: **Cosine similarity evolution of T_2 and T_3 (as flattened vectors)**

For AdamW, we additionally measured $T_{2,P}$ and $T_{3,P}$ (as in Theorem C.1), the optimizer-effective geometry induced by its diagonal second-moment preconditioner $P = \mathrm{diag}\big((\hat{v} + \epsilon)^{-1/2}\big)$, where $\hat{v}$ is AdamW's bias-corrected second-moment accumulator. We therefore computed

$$T_{3,P} = \frac{1}{2} PH^{(\mathrm{current})} PG^{(\mathrm{next})} - \frac{1}{2} PH^{(\mathrm{next})} PG^{(\mathrm{current})},$$

and $T_{2,P}$ analogously. We report the raw interference ratio $\|T_3\|/\|T_2\|$ and, for AdamW, the optimizer-effective ratio $\|T_{3,P}\|/\|T_{2,P}\|$.

In addition to Figure 5, Figure 11 confirms that SGD's T_3 slightly opposes T_2.

Details on measuring SGA and effective SGA. For SGD, AdamW and Muon we estimated all class gradients $G^{(c)}$ as full-parameter gradients of the class loss on a held-out batch of class-c documents (64 each), every 5,000 iterations. We formed the successor-contrast vectors w_c of equation 4.1. For comparability across optimizers, we normalize SGA with the mean squared gradient norm to form our *normalized SGA* estimator:

$$\widetilde{\mathrm{SGA}} = \frac{\mathrm{SGA}}{\frac{1}{R}\sum_c \|G^{(c)}\|^2}.$$

The numerator is unbiased because w_c excludes class c, so every term is an inner product of gradient estimates from independent held-out batches. The *effective SGA* estimator normalises $\mathrm{SGA}_{\mathrm{eff}}$ of Theorem C.3. To compute $\mathrm{SGA}_{\mathrm{eff}}$, we replace $G^{(c)}$ on the right of the inner product by the optimizer's update direction $U_c = \mathcal{T}(G^{(c)})$, where $\mathcal{T}$ is specified below.

$$\widetilde{\mathrm{SGA}}_{\mathrm{eff}} = \frac{\frac{1}{R}\sum_c \langle w_c, \mathcal{T}(G^{(c)})\rangle}{\frac{1}{R}\sum_c \|G^{(c)}\|\,\|\mathcal{T}(G^{(c)})\|}.$$

For SGD, $\mathcal{T}$ is the identity and effective and normalized SGA coincide.

For AdamW, $\mathcal{T}(G) = PG$ with $P = \mathrm{diag}\left(\frac{1}{\sqrt{\hat{v}}+\epsilon}\right)$, where $\hat{v}$ is the bias-corrected second-moment accumulator we read from the optimizer state at each measurement time. The numerator equals SGA_P, the statistic on which the `Ring` drift acts in Theorem C.2.

For Muon, we define $\mathcal{T}$ to reproduce the optimizer's Newton-Schulz orthogonalisation (without momentum) on hidden weight matrices, and to apply AdamW's diagonal preconditioner on the parameters which Muon delegates to its internal AdamW optimizer (embeddings, output parameters, and one-dimensional parameters). Because the Newton-Schulz transformation is nonlinear, this update cannot in general be written as PG for a fixed matrix P. Hence $\mathrm{SGA}_{\mathrm{eff}}$ does not reduce to SGA_P and is instead computed directly from its definition (in Theorem C.2) using the transformed class gradients $\mathcal{T}(G^{(c)})$. We report normalized raw and effective SGA averaged over the late-training window (iterations $\geq$ 75,000).

Resetting AdamW's optimizer state. We also tested whether AdamW's anticipation may be due to memory accumulating in the optimizer state. However, resetting the optimizer state at every burst transition produced a similar SA trajectory (Figure 12), showing that persistent optimizer memory

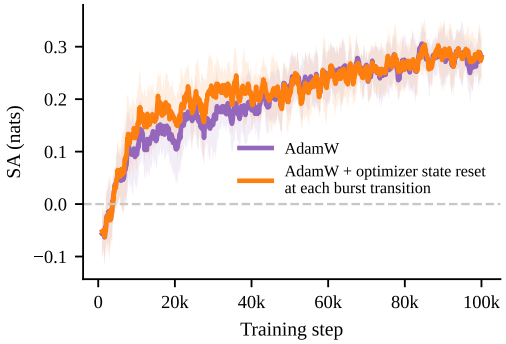


Figure 12: **AdamW's optimization is not due to optimizer state memory.** We reset the optimizer state at every burst transition, and see similar anticipation behaviour, confirming that persistent optimizer memory across bursts is not necessary for anticipation. Values are averaged over 12 seeds.

across bursts is not necessary for anticipation. Together with our $\|T_3\|/\|T_2\|$ analysis, this supports the interpretation that AdamW's coordinatewise preconditioning encourages anticipation by reducing curvature interference relative to the alignment pressure.

D.6 Details of the explicit T_2 regularization

All runs use the default `Ring` configuration of Section B.3 with 12 seeds per experiment, unless noted otherwise.

Regularizer. Every 16 optimizer steps, we form $\widehat{D}(c, s(c))$ for the `Ring`'s current class c from per-class mini-batches of size 64 drawn from the training data, and add $\lambda\widehat{D}$ to the loss. The regularizer is successor-only: gradients of the $R-2$ control classes are computed and logged as monitors but do not enter the loss. Backpropagating through $\widehat{D}$ is second order: it involves Hessian-vector products, e.g. $2(H^{(c)} - H^{(s(c))})(\hat{g}^{(c)} - \hat{g}^{(s(c))})$ for the plug-in estimator. We compute the per-class gradients with `create_graph=True`. The task gradient is clipped to norm 1.0 on its own, after which the λ-scaled regularizer gradient itself is clipped to norm 5.0.

Estimators. The plug-in estimator is $\widehat{D}_{\text{plug}} = \|\hat{g}^{(c)} - \hat{g}^{(s(c))}\|^2$ on fresh mini-batch gradients. The unbiased estimator maintains per-class EMAs $\bar{G}^{(c)}$ (decay 0.99) of the mini-batch class gradients and uses $\widehat{D}_{\text{EMA}} = \langle \hat{g}^{(c)} - \hat{g}^{(s(c))}, \bar{G}^{(c)} - \bar{G}^{(s(c))} \rangle$ with the EMA estimates detached from the graph. The EMA is updated only after the regularizer is formed. We ablate regularization strengths $\lambda \in \{-0.03, -0.01, -0.003, 0.03, 0.1, 0.3, 1.0\}$ with the plug-in estimator and $\lambda \in \{0.03, 0.06\}$ with the EMA estimator. Our baseline is $\lambda = 0$. All experiments use 12 seeds except $\lambda = -0.03$, which uses 4.

D.7 The effect of hyperparameters

We ablate four hyperparameters of the default configuration reported in Table 3: the number of gradient steps per class (Figure 13), batch size (Figure 15), context length (Figure 14), and model size (Figure 16). We highlight two particularly informative trends below.

Batch size. Figure 15 shows that anticipation increases with batch size, before saturating around a batch size of 64. This trend is consistent with Theorem C.1: T_1 scales inversely with batch size and therefore becomes weaker for larger batches. Although T_1 does not directly shape interactions between classes, the additional gradient noise introduced at small batch sizes appears to oppose anticipation.

Context length. Figure 14 shows that the differences between `Ring`, `Bursty`, and `Shuffled` training diminish as context length increases. In our setup, this is likely because the class becomes

identifiable from short prefixes. Once enough tokens have been observed, even non-anticipatory models can infer the current class and predict accordingly, reducing the advantage provided by anticipation. This is consistent with proposition 1, which predicts that the best possible average loss improvement via anticipation scales inversely with sequence length.

Importantly, this is not a limitation of anticipation itself, but of our class-based construction. If the relevant structure in the training stream were not recoverable from short prefixes, anticipation could remain beneficial at longer context lengths. Such setting is studied by Yang et al. (2024), where individual documents rather than broad classes are anticipated. Other similar examples could include anticipation of more document-specific units such as concepts, whose occurrence could become predictable from training order.

We additionally analyze the "forgetting profile" of trained models to study whether adaptation extends beyond the immediate successor class. All ablation figures include forgetting profiles in the right-hand panels. We describe their construction and interpretation below.

Forgetting profiles. The forgetting profile plots show the evaluation loss grouped by relative position in the training stream. For each evaluation, classes are indexed by their distance from the current class, i.e. the class on which the model was most recently trained.

This means that the horizontal axis represents relative, *not* than absolute class identity. In `Ring` training, this means a rotation of the fixed cyclic order. In `Bursty` training, it corresponds to the realized order of classes in a particular run. Reported values are averaged over evaluations from the final 25% of training. The axis wraps around cyclically, so positions "-4" and "5" correspond to the same class.

Across ablations, stronger anticipation generally corresponds to larger differences between the `Ring` and `Bursty` forgetting profiles. Increasing the number of gradient steps, batch size, or model scale causes both `Ring` and `Bursty` models to adapt more strongly to the current class. However, `Bursty` forgetting profiles are typically asymmetric: classes from the recent past (negative positions) experience less forgetting than distant classes.

The immediate successor class in `Bursty` training can sometimes exhibit unusually high loss. This is an artifact of the transition structure: `Bursty` position $+2$-onward can include the "current" class, whereas `Ring` cannot, as it must first cycle through all other classes.

`Ring` training exhibits stronger adaptation to the current class, but also substantial adaptation to future classes, including classes beyond the immediate successor. This adaptation decays approximately with future distance in the cycle, producing a more symmetric forgetting profile. This anticipation of future classes comes at the cost of increased forgetting on classes further away in the cycle.

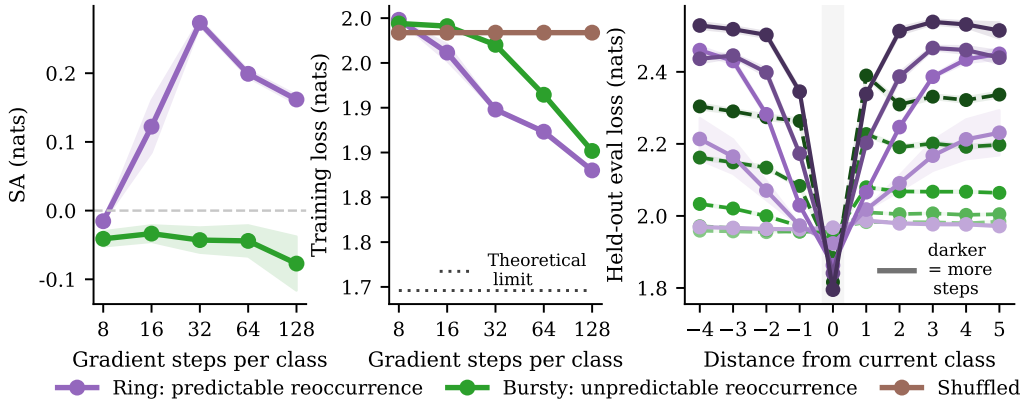


Figure 13: **Effect of the number of gradient steps taken per class.** Reported values are averaged over 12 seeds. *(Left) Effect on Successor Advantage. (Center) Effect on training loss.* Consistent with Yang et al. (2024), we observe increasing anticipation up to 32 gradient steps per class. However, increasing the number of steps further is detrimental, likely because tracking the current class becomes the advantageous behavior: Bursty's training loss approaches Ring's. *(Right) Effect on the forgetting profile.* Higher SA means larger adaptation to successor classes, including non-immediate successors in Ring-trained models. Too many gradient steps cause overadaptation to the current class, which comes with a loss of anticipation.

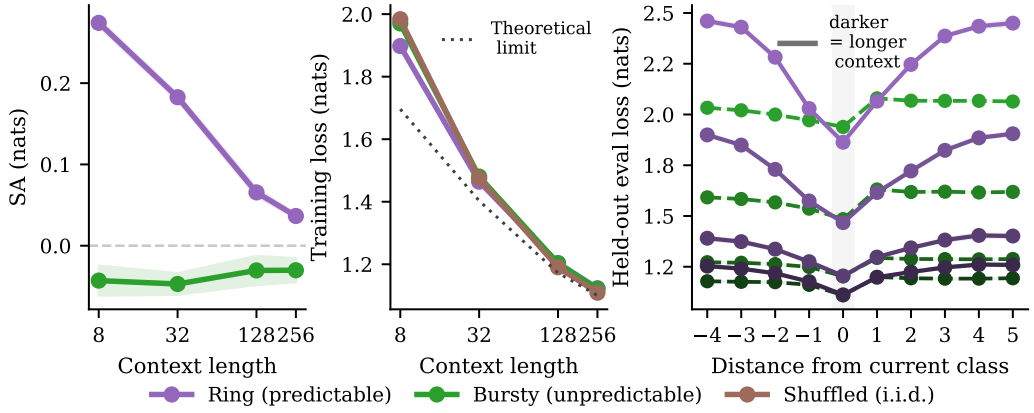


Figure 14: **Effect of the context length.** Reported values are averaged over at least 11 seeds. *(Left) Effect on Successor Advantage. (Center) Effect on training loss.* Contrary to Yang et al. (2024), in our class-based setup, longer context lengths hurt anticipation, likely because the class becomes easily identifiable from the prefix, hence any anticipation advantage persists only for a few tokens. *(Right) Effect on the forgetting profile.* Smaller context lengths in Ring-trained models mean larger adaptation, relative to Bursty, to current and successor classes, including non-immediate successors.

D.8 Details on the gradient-alignment experiments

Raw and directional gradient alignment. The SGA (Definition 3) statistic from our theory is defined on the raw class gradients. It preserves differing class-specific gradient magnitudes, because they determine first-order loss changes. Here, however, our primary question is *whether training order changes the relative directions of the class gradients*. To remove the confound of differing class gradient magnitudes, we first normalize class gradients before computing the SGA. We use this to analyze directional gradient geometry, but subsequently show that the corresponding effect is also present in raw SGA.

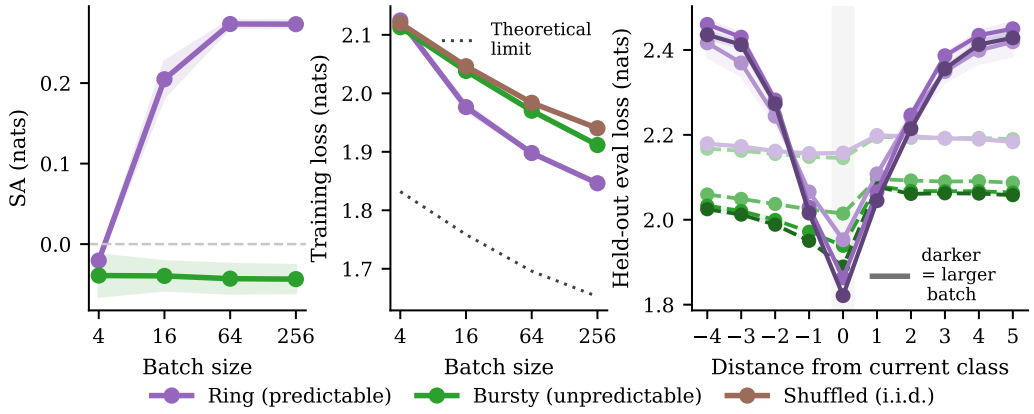


Figure 15: **Effect of the batch size.** Reported values are averaged over 12 seeds. *(Left) Effect on Successor Advantage. (Center) Effect on training loss.* Larger batch size generally helps anticipation, though the difference between batch size 64 and 256 in SA is negligible. *(Right) Effect on the forgetting profile.* Larger batches in Ring-trained models mean larger adaptation, relative to Bursty, to current and successor classes, including non-immediate successors.

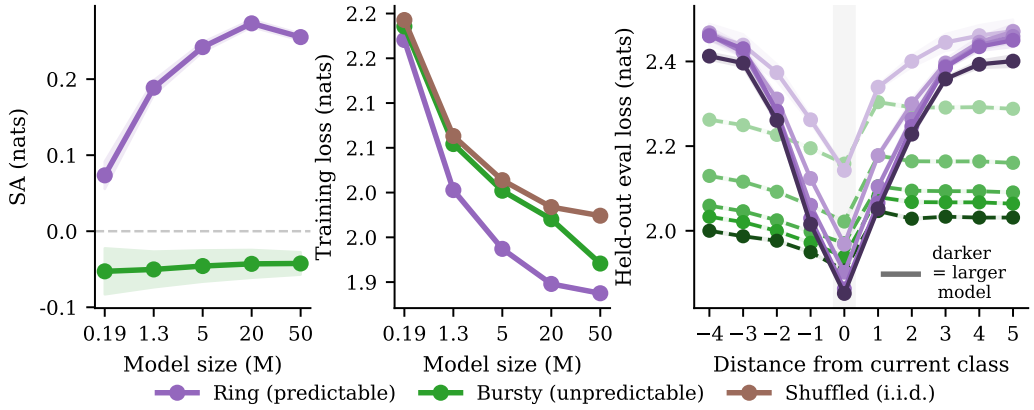


Figure 16: **Effect of the model size.** Reported values are averaged over 12 seeds. *(Left) Effect on Successor Advantage. (Center) Effect on training loss.* Larger model size helps anticipation. *(Right) Effect on the forgetting profile.* Larger Ring-trained models display larger adaptation, relative to Bursty, to current and successor classes, including non-immediate successors.

Let $\widehat{g}^{(c)} = \frac{g^{(c)}}{\|g^{(c)}\|}$ denote the unit-normalized gradient of class c. For every current class i and comparison class k, define its relative gradient alignment as

$$r_{i,k} = \left\langle \widehat{g}^{(i)}, \widehat{g}^{(k)} \right\rangle - \frac{1}{R-2} \sum_{j \notin \{i,k\}} \left\langle \widehat{g}^{(i)}, \widehat{g}^{(j)} \right\rangle .$$

When $k = s(i)$, this is the per-class SGA term evaluated on unit-normalized gradients. We call its average *angular SGA*:

$$\mathrm{SGA}_{\mathrm{ang}} = \frac{1}{R} \sum_{i} r_{i,s(i)}.$$

It's called "angular", because $\left\langle \widehat{g}^{(i)}, \widehat{g}^{(k)} \right\rangle = \cos(g^{(i)}, g^{(k)})$,, so $r_{i,k}$ is the cosine similarity of classes i and k relative to the remaining classes.

Experimental setup. We adapt our default experimental setup and record the per-class gradients at equally spaced checkpoints. We make two modifications. First, the class order in each `Ring`

schedule is held fixed across seed, because we study the effect of *fixed* class orders. Second, we train for 200k instead of 100k steps to make the evolution of the gradient geometry more visible. At each checkpoint, we record gradients for every class, allowing us to evaluate all class pairs rather than only successor pairs.

Measuring the quality of Ring orderings. In order to probe the effect of the specific order of classes chosen, we computed a randomly chosen "`Ring`: original" ordering, a "Maximum A-priori Similarity" and "Minimum A-priori Similarity" ordering of classes as follows. We created an a-priori influence matrix by recording, for each language pair (i, j), excluding the diagonal, the change in the loss of j, while training on i for 500 steps. The matrix is reported in Figure 17. This estimates the early-influence of training on class i on class j. We centered the rows of the matrix by subtracting the row mean. For a given ordering of classes, the sum of influence values corresponding to consecutive classes measures the quality of the ordering, in the sense of how much it respects a-priori class alignments. We then used a Held-Karp dynamic program over subsets to find the minimum- and maximum-weight Hamiltonian cycles in the influence matrix. We call the most optimal ordering `Max Similarity` order and the most suboptimal `Min Similarity` order. `Max Similarity` order achieves a total score of -0.312, meaning that training on a class already decreases the loss of the next class, while `Min Similarity` order achieves $+0.218$, and `Ring` scores 0.051.

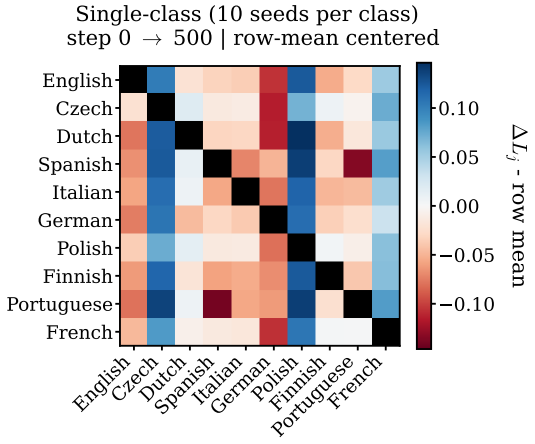


Figure 17: **Matrix of the early-influence of training on class i on class j.** Influences reflect common-sense relationships and origin-based clusters such as Romance languages, Germanic languages, Slavic languages and Uralic languages. Rows are centered via subtracting the row mean, values are computed using 10 seeds.

Results. Figure 18 shows the relative gradient alignments $r_{i,k}$ between languages i, k after 200k training steps, while Figure 19 shows their change over training, $r_{i,k}^{(200\text{k})} - r_{i,k}^{(0)}$. Although the underlying cosine similarity matrix is symmetric, the relative alignments $r_{i,k}$ need not be, because each row is centered using a different set of control classes.

Under all three fixed `Ring` orders: `Ring`, `Max Similarity`, and `Min Similarity`, the successor cells become strongly aligned relative to the remaining classes. By contrast, `Bursty` and `Shuffled` generally preserve existing alignments (cosine similarities have low magnitudes), and only reinforce priori similarities, such as that between Czech and Polish.

Figure 20 summarizes this effect over training. The left panel shows the mean cosine similarity between classes current and next, alongside its average similarity to the control classes. Their difference, shown in the center panel, is angular SGA. The `Max Similarity` order has the largest absolute angular SGA, although it also begins with higher alignment because its order was selected to maximize a priori similarity. The right panel shows that larger angular SGA is associated with larger SA.

Connection to raw SGA. The cosine-based analysis above isolates changes in the relative directions of the class gradients. To connect this directional effect to the quantity identified by our theory

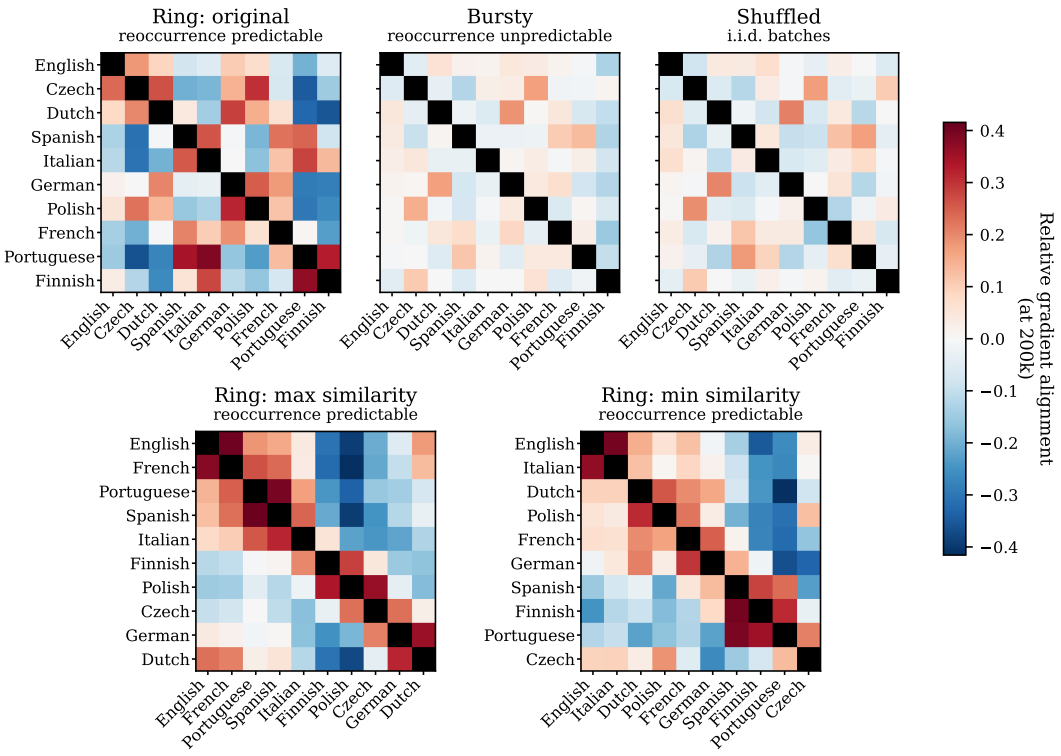


Figure 18: **Heatmaps of the relative gradient alignments (cosine similarities) at the end of training (200k).** Cell (i, k) reports the cosine similarity between the gradients of current class i and comparison class k, relative to the mean alignment between i and the remaining classes. The figure shows that Ring orders (Ring, Max Similarity and Min Similarity) create gradient alignment between consecutive classes, even in the case of very different languages like Spanish and Finnish (e.g. the "Spanish" row on the Min Similarity panel). Values are averaged over 12 seeds.

(Theorem C.2), Figure 21 reports SGA computed directly from the unnormalized class gradients. Raw SGA also increases under the fixed `Ring` orders. Taken together, these experiments show that predictable training order increases the SGA, as identified by our theory, and that this effect persists after removing class-specific gradient magnitudes. Therefore, the increase reflects genuine directional reorganization in the gradient's geometry.

This directional reorganization can be quite striking: `Ring` orders can create alignments between classes that are completely unrelated a-priori, to the extent that a language's gradients may become more aligned to unrelated languages than those most similar to them. An example of this in the "Spanish" row in the `Min Similarity` order in Figure 18 and Figure 19: Spanish becomes more aligned to Finnish than to Portuguese. *This demonstrates that order can have a significant influence on inter-data relationships between learnt by the model.*

We test the impact of these influences on the training and evaluation losses. As expected, Figure 22 (left) shows `Max Similarity` order achieves the lowest train loss due to high anticipation. However, this comes with the highest evaluation loss (center and right panels). We note that the fact that `Ring` orders have high eval loss is not surprising, as they are heavily adapted to the training history, and the evaluation distribution is a mixture, which therefore creates a distribution shift for these models. The fact that `Max Similarity` order performs even worse than other ordered setups may be due to adapting too much to the next class, to the point that its ability to distinguish the current class from the next class degrades. This is visible on the right panel: even though Spanish and Portuguese are similar languages, due to the training order including the transition Portuguese $\rightarrow$ Spanish, their connection is reinforced to the extent that the model struggles to distinguish them.

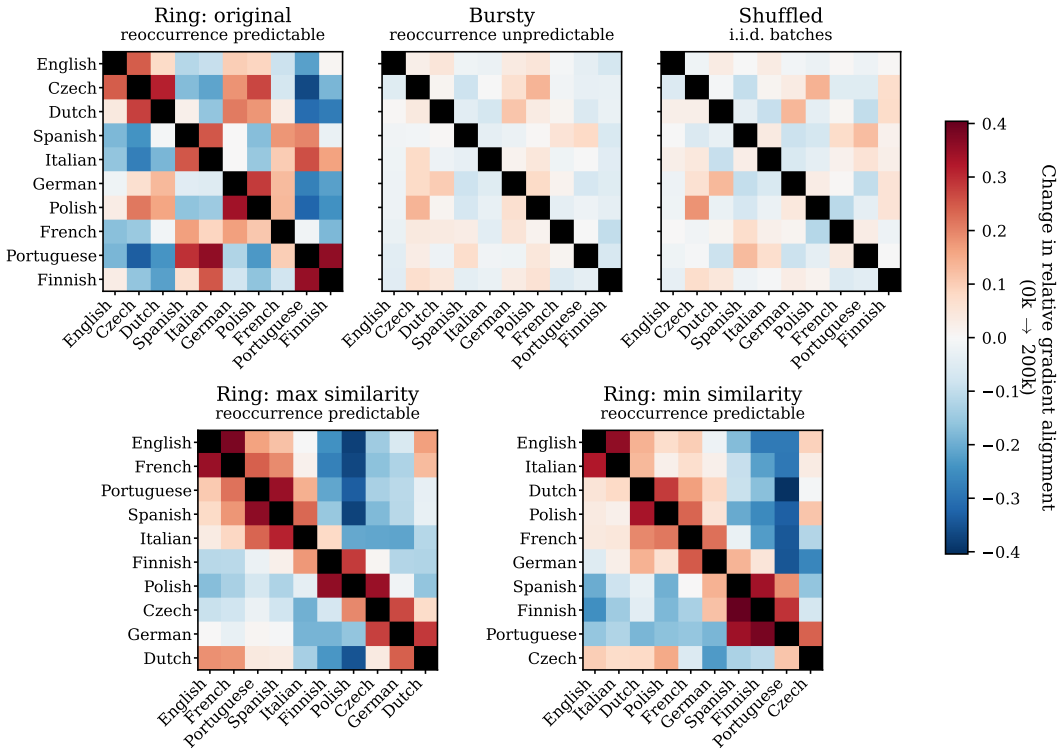


Figure 19: **Heatmaps of the evolution of relative gradient alignments (cosine similarities).** We plot, for each order configuration, the difference of relative gradient alignments between training step 200k and step 0. Relative gradient alignments are computed for each cell (i, k) as the cosine similarity between the gradients of current class i and comparison class k, relative to the mean alignment between i and the remaining classes. The figure shows that Ring orders (Ring, Max Similarity and Min Similarity) create gradient alignment between consecutive classes, even in the case of very different languages like Spanish and Finnish (e.g. the "Spanish" row on the Min Similarity panel). Values are averaged over 12 seeds.

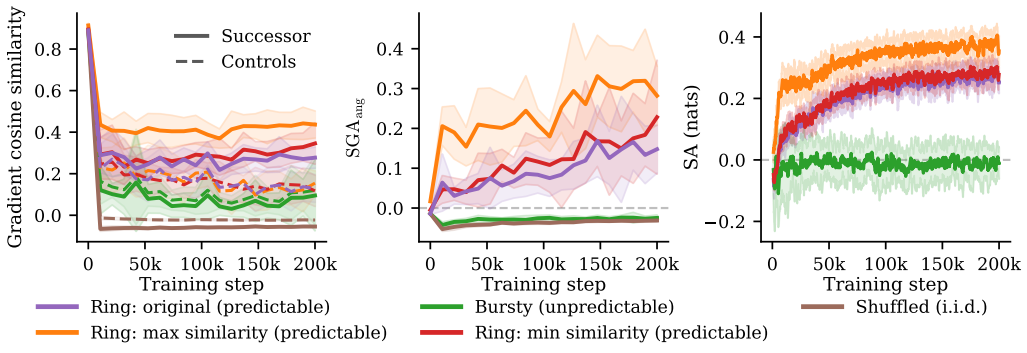


Figure 20: **Evolution of gradient alignment towards successor classes and Anticipation.** *(Left) Gradient cosine similarity between the current and next class, and current and other classes and (Center) their difference.* Max Similarity order achieves largest gradient alignment, while Ring and Min Similarity order are comparable. *(Right) Successor Advantage during training.* Max Similarity order achieves highest anticipation. Standard deviations are computed from 12 seeds.

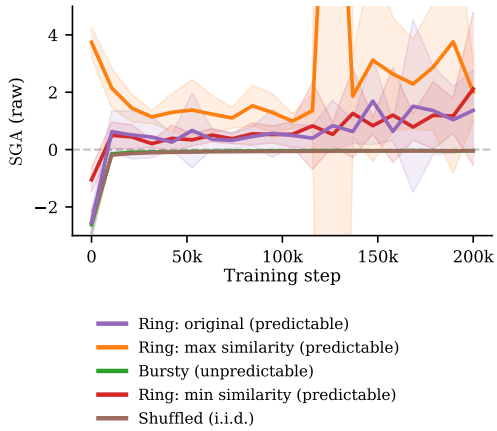


Figure 21: **Evolution of raw successor-gradient alignment (SGA).** Raw SGA is computed from the unnormalized class gradients . The fixed `Ring` schedules increase raw SGA, showing that successor-specific alignment also emerges in the original quantity identified by Theorem C.2. Standard deviations are computed from 12 seeds.

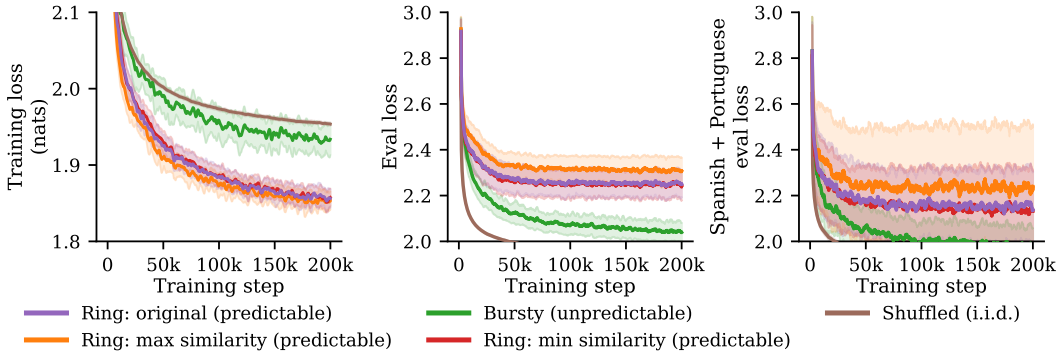


Figure 22: **The effect or class orderings.** *(Left) Effect on the training loss.* Max Similarity order has lowest train loss. *(Center) Effect on eval loss.* Non-random orderings achieve higher eval loss due to the distribution shift of training on a structured order, but evaluated on a class mixture. Max Similarity order achieves high loss even among ring configurations. *(Right) Effect on the eval loss of Spanish and Portuguese documents only.* Max Similarity order still performs worst, likely because too much consecutive class alignment blurs the ability to distinguish these classes. In the Max Similarity order, Spanish directly follows Portuguese. Values are averaged over 12 seeds, losses were smoothed with a 20-point moving average.