arXiv:2504.12916v1 [cs.LG] 17 Apr 2025

Exact Learning Dynamics of In-Context Learning in Linear Transformers and Its Application to Non-Linear Transformers

Nischal Mainali
The Edmond and Lily Safra Center for Brain Sciences
Hebrew University of Jerusalem
nischal.mainali@mail.huji.ac.il

Lucas Teixeira
Principles of Intelligent Behavior in Biological and Social Systems (PIBBSS)

Abstract

Transformer models exhibit remarkable in-context learning (ICL), adapting to novel tasks from examples within their context, yet the underlying mechanisms remain largely mysterious. Here, we provide an exact analytical characterization of ICL emergence by deriving the closed-form stochastic gradient descent (SGD) dynamics for a simplified linear transformer performing regression tasks. Our analysis reveals key properties: (1) a natural separation of timescales directly governed by the input data's covariance structure, leading to staged learning; (2) an exact description of how ICL develops, including fixed points corresponding to learned algorithms and conservation laws constraining the dynamics; and (3) surprisingly nonlinear learning behavior despite the model's linearity. We hypothesize this phenomenology extends to non-linear models. To test this, we introduce theory-inspired macroscopic measures (spectral rank dynamics, subspace stability) and use them to provide mechanistic explanations for (1) the sudden emergence of ICL in attention-only networks and (2) delayed generalization (grokking) in modular arithmetic models. Our work offers an exact dynamical model for ICL and theoretically grounded tools for analyzing complex transformer training.

1 Introduction

The transformer architecture [1] has revolutionized machine learning, particularly through its capability for in-context learning (ICL). ICL allows models to adapt their behavior based on examples provided within the input context, obviating the need for task-specific parameter updates [2]. This phenomenon links conceptually to meta-learning, characterized by distinct timescales: slow learning via weight updates during training and fast adaptation based on context during inference [3]. Unlike traditional supervised learning, ICL enables generalization to novel tasks by conditioning on contextual examples [4]. Mechanistically, attention layers can implement adaptive processing analogous to fast weights [5, 6, 3], suggesting that transformers might internally simulate learning algorithms [7, 3, 8]. Despite compelling empirical demonstrations, a full theoretical understanding of how transformers achieve ICL remains unclear.

To address this challenge, our work presents an in-depth analysis of a simplified, mathematically tractable transformer model performing in-context linear regression. Our model incorporates specific simplifications—linear attention [3, 7], a structured input embedding [9, 7], mean-squared error loss, and a continuous-time approximation of learning dynamics—yet retains core aspects of ICL. This

Preprint. Under review.

deliberate simplification allows us to analytically trace how contextual information modulates the model's predictions. By using methods from the study of exactly solvable models, introduced first in deep linear networks [10], we derive a precise characterization of the learning dynamics under stochastic gradient descent (SGD). We analyze the resulting phenomenology, including the emergence of multiple learning timescales dictated by data statistics [10, 11]. This detailed understanding of a simplified system aims to provide basic insights applicable to the interpretation and analysis of more complex transformer models exhibiting ICL. In particular, we believe that this detailed understanding can inform interpretability efforts and evaluation methodologies for more complex, real-world models and argue the point by quantitatively analyzing the properties of model parameters in 1) simple attention only transformer model that exhibit sudden emergence of in context learning during training time, and 2) grokking model that exhibits delayed generalization.

1.1 Related Work

Transformers exhibit remarkable in-context learning (ICL), adapting to new tasks using context examples without explicit parameter updates [2]. This connects to classic idea of meta-learning's dual timescales: slow weight learning and fast context adaptation [3]. Early concepts like fast weights [5] suggested context-based adaptation mechanisms. Recent studies demonstrate that linear self-attention layers can be mathematically equivalent to fast weight update rules [6, 3], enabling Transformers to implement learning algorithms within their forward pass [3].

Several studies confirm that Transformers can encode standard learning algorithms. Akyürek et al. [7] proved by construction that Transformers can implement gradient descent (GD) and ridge regression, finding empirically that trained models mimic these algorithms and converge towards Bayesian optimal predictors for linear tasks. Similarly, Von Oswald et al. [3] demonstrated that Transformers trained on regression problems learn GD internally, with attention updates mapping directly to GD steps, effectively becoming "mesa-optimizers". In parallel work Ahn et al. [8], and Zhang et al. [12] provided theoretical guarantees that standard stochastic gradient descent (SGD) training yields preconditioned GD at the fixed-point of the training dynamics across a range of setups. These findings collectively suggest that standard training procedures can induce Transformers to implicitly learn optimization algorithms. Extending this, Fu et al. [13] found evidence that Transformers might learn higher-order methods resembling Newton's method, explaining superior performance on ill-conditioned problems. Alternative perspectives conceptualize ICL as Bayesian inference, where models infer latent task variables from prompts [14], or as kernel regression, where attention patterns reflect similarity-based weighting akin to kernel methods [15]. These views provide explanations for observed phenomena like prompt sensitivity and the utility of relevant examples [14, 15]. The precise mechanism—gradient-based, Bayesian, or kernel-like—likely depends on model scale, architecture, and training data [7, 3, 15, 14, 8, 12].

At the same time, analyzing exact learning dynamics has a rich history, notably through the study of deep linear networks [10]. Saxe et al. [10] derived exact solutions for these models, revealing nonlinear dynamics and multiple timescales despite linearity. Learning often occurs in stages, marked by plateaus followed by rapid error reduction as different eigenmodes of the data correlation are sequentially learned. Learning timescales are typically eigenvalue-dependent, with dominant modes learned faster [10, 16]. In nonlinear networks, exact solutions are rare, but phenomena like spectral low rank bias indicate structured learning dynamics: models tend to learn low-frequency (smooth) function components before high-frequency details [11]. Such low rank bias is aslo linked to the Neural Tangent Kernel (NTK) spectrum of highly non-linear models, where larger eigenvalues associated with smoother functions lead to faster learning [11, 17], potentially explaining the generalization capabilities of overparameterized models.

Separation of timescales emerges as a common theme across linear dynamics [10], nonlinear spectral bias [11], meta-learning [3] and the abrupt emergence of ICL or induction heads [18] hints at similar timescale separation. Indeed, there have been theoretical results in non-linear transformer establishing distinct phases in the learning dynamics [19, 20] further supporting the idea. Alternative models for ICL like associative memories instead connect ICL to retrieval dynamics [21].

2 Model Setup

2.1 Linear Transformer Architecture

We analyze a single-layer transformer with a single linear attention head. In contrast to standard architectures using softmax attention [1], we employ simplifications common in theoretical studies [3, 7]. The model utilizes two weight matrices, $W^Q \in \mathbb{R}^{D\times D}$ (query/key) and $W^P \in \mathbb{R}^{D\times D}$ (output/projection), and replaces the attention non-linearity with the identity function ($\phi(x) = x$). The layer's computation is defined as (see Appendix A.1-A.3, Eq. (A3)):

$$f(Z) = Z + W^P \left(\frac{ZZ^\top}{N}\right) W^Q Z. \tag{1}$$

Here, $Z \in \mathbb{R}^{D\times(N+1)}$ is the input sequence embedding, and $N + 1$ is the sequence length. The learnable parameters are $\Theta = \{W^P, W^Q\}$. This linear attention model, while simplified, is sufficient for learning in-context linear regression [3].

Relevant parameter submatrices are defined based on the input embedding structure (Appendix A.2). We partition W^Q such that its first d columns form $q = [\ q_1 \mid q_2\ ]^T$, and W^P such that its bottom d rows form $p = [\ p_1 \mid p_2\ ]$. All submatrices q_1, q_2, p_1, p_2 are in $\mathbb{R}^{d\times d}$. More precisely, we have the following parametrization:

$$W^P = \begin{pmatrix} \cdot & \cdot \\ p_1 & p_2 \end{pmatrix} \quad \text{with} \quad p = (p_1 \quad p_2)\,, \tag{2}$$

$$W^Q = \begin{pmatrix} q_1 & \cdot \\ q_2 & \cdot \end{pmatrix} \quad \text{with} \quad q = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix}, \tag{3}$$

2.2 Embedding Structure and Task Setup for In-Context Regression

The model is trained to perform in-context linear regression. Given a prompt containing N example pairs $(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R}^d$, it must predict the output y_q for a query input $x_q \in \mathbb{R}^d$ based on the linear relationship demonstrated in the prompt. This setup extends the $\mathbb{R}^d \to \mathbb{R}^1$ tasks studied in some prior work [7, 9] to the more general vector-valued regression setting $\mathbb{R}^d \to \mathbb{R}^d$.

We employ a structured embedding strategy commonly used in ICL studies [7, 9, 22]. For an input sequence $(x_1, y_1, \ldots, x_N, y_N, x_q)$, the embedding matrix $Z \in \mathbb{R}^{D\times(N+1)}$ (with $D = 2d$) is constructed as:

$$Z = \begin{pmatrix} x_1 & \ldots & x_N & x_q \\ y_1 & \ldots & y_N & \mathbf{0}_d \end{pmatrix}. \tag{4}$$

This embedding structure (Appendix A.2, Eq. (A1)) confers permutation invariance over context examples via the ZZ^T term, distinguishes the query input using the zero vector padding, and implicitly encodes the input-output pairing structure without requiring explicit positional encodings.

2.3 Task Statistics and Training Procedure

We define a generative process for the linear regression tasks. Input data x are sampled from a zero-mean Gaussian distribution $\mathcal{N}(0, \Sigma_x)$. The covariance matrix Σ_x is assumed diagonalizable by an orthogonal matrix U, such that $\Sigma_x = USU^\top$, where $S = \text{diag}(s_1, \ldots, s_d)$ contains the eigenvalues. The corresponding output is generated via $y = Wx$, where W is a task-specific matrix.

We generate a fixed set of P task matrices $\{W^\mu\}_{\mu=1}^P$ prior to training. Each task matrix is constructed as $W^\mu = U\Lambda^\mu U^\top$, where $\Lambda^\mu = \text{diag}(\lambda_{\mu,1}, \ldots, \lambda_{\mu,d})$ has eigenvalues sampled independently from $\mathcal{N}(0, 1)$. Crucially, the *same* orthogonal matrix U is used for both Σ_x and W^μ. This alignment ensures that all relevant matrices commute in the eigenbasis of Σ_x, significantly simplifying the theoretical analysis (see Appendix A.8.2 and A.11).

The model's prediction $\hat{y}$ for query x_q under task μ (derived from the $(N + 1)$-th column of $f(Z^\mu)$) is evaluated against the ground truth $y_q^\mu = W^\mu x_q$ using the Mean Squared Error (MSE) loss function:

$$\mathcal{L}_\mu(\Theta) = \frac{1}{2}\|\hat{y} - y_q^\mu\|_2^2. \tag{5}$$

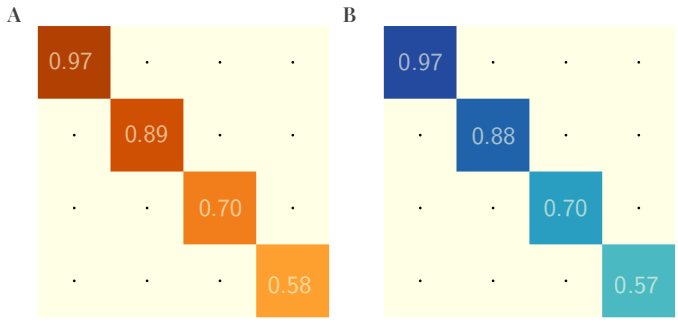

Figure 1: Fixed point parameters: theory vs. simulation. (A) Learned parameters ($p_2 q_1$) in batch-averaged simulation, diagonalized in the input covariance basis. (B) theoretical prediction for the fixed point $p_2 q_1(\infty)$, showing precise quantitative agreement.

The use of MSE is a simplification for analytical tractability compared to typical auto-regressive losses.

Training employs SGD with a small learning rate η. An epoch involves presenting all P tasks, each time sampling fresh data contexts $\{x_i, x_q\}$. Parameter updates occur after processing each task sequentially. The small η assumption justifies approximating the discrete learning process with continuous-time dynamics (see Appendix A.11).

3 Theory

Simplified predictor. The model predicts the query output $\hat{y} \in \mathbb{R}^d$ using the bottom d elements of the final output column $[f(Z)]_{:,N+1}$. As derived in Appendix A.9 (Eq. (A18)), this prediction simplifies to a linear transformation of the query input:

$$\hat{y} = \left(p^T \hat{\Gamma} q\right) x_q, \tag{6}$$

where $\hat{\Gamma} = \frac{ZZ^\top}{N}$ is the empirical covariance of the embedded sequence. The matrix $p^T \hat{\Gamma} q$ effectively encodes the linear function inferred from the context. The MSE loss for task μ (Eq. 5) uses this predictor with the task-specific covariance $\hat{\Gamma}^\mu$:

$$\mathcal{L}_\mu(\Theta) = \frac{1}{2} \left\| \left(p^T \hat{\Gamma}^\mu q\right) x_q - W^\mu x_q \right\|_2^2. \tag{7}$$

We proceed by analyzing the learning dynamics in the continuous-time limit, averaging over the quenched randomness of tasks W^μ and the annealed randomness of data samples x_i, x_q.

Null Parameter Dynamics and Predictor Simplification. A key simplification arises from identifying p_1 and q_2 as "null parameters." Our analysis (Appendix A.10) demonstrates that if initialized at zero, their expected gradients under SGD vanish [12, 8]. This theoretically justifies fixing $p_1 = \mathbf{0}$ and $q_2 = \mathbf{0}$ throughout training. Combined with a large-N approximation for the empirical covariance $\hat{\Gamma}^\mu$ (which neglects the query's contribution, see Appendix A.8.3, Eq. (A11)), the predictor for task μ takes a remarkably simpler form (derived in Appendix A.10, Eq. (A55)):

$$\hat{y}_\mu \approx p_2 W^\mu \hat{\Sigma}_x q_1 x_q, \tag{8}$$

where $\hat{\Sigma}_x = \frac{1}{N} \sum_{i=1}^N x_i x_i^\top$. This simplified predictor, involving only parameters p_2 and q_1, forms the foundation for our subsequent analysis of learning dynamics. Empirically, even with small non-zero initialization, the learning dynamics drives p_1 and q_2 towards zero.

Non-linear Learning Dynamics and Spectral Decoupling. To analyze the dynamics of p_2 and q_1, we assume spectral alignment: parameters $p_2(t), q_1(t)$ diagonalize in the same basis U as the data covariance Σ_x, i.e., $p_2(t) = U \bar{p}_2(t) U^\top$ and $q_1(t) = U \bar{q}_1(t) U^\top$, where $\bar{p}_2, \bar{q}_1$ are diagonal

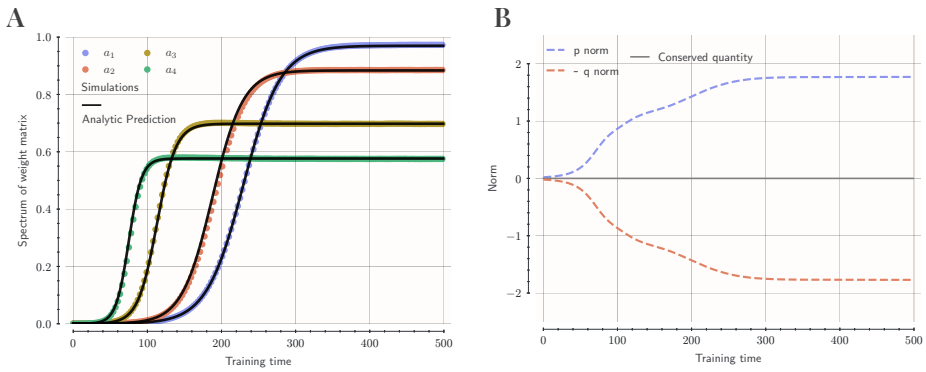

Figure 2: Parameter dynamics: theory vs. simulation. (A) Empirical, batch-avergaed, evolution of the diagonal elements of the learned parameter $\bar{a}(t) = \bar{p}_2(t)\bar{q}_1(t)$ compared with theoretical predictions (Eq. 13), showing excellent agreement. (B) Mirror evolution of two terms in conserved quantity and confirmation of the stability of the conserved quantity $\mathcal{C} = \|\bar{p}_2\|_F^2 - \|\bar{q}_1\|_F^2$.

(Appendix A.11, Eqs. (A68)-(A69)). Averaging over stochasticity and taking the continuous-time limit yields coupled ODEs for the diagonal parameter matrices (Appendix A.11, Eqs. (A80)-(A81)):

$$\frac{\mathrm{d}\bar{p}_2}{\mathrm{d}t} = -\eta P S^2 \left(\bar{p}_2 \bar{q}_1^2 s^\infty(S) - \bar{q}_1\right) \tag{9}$$

$$\frac{\mathrm{d}\bar{q}_1}{\mathrm{d}t} = -\eta P S^2 \left(\bar{q}_1 \bar{p}_2^2 s^\infty(S) - \bar{p}_2\right), \tag{10}$$

where $s^\infty(S) = (\frac{N+1}{N} S + \frac{\mathrm{Tr}(S)}{N} I)$ is a diagonal matrix encoding the influence of context length N and data statistics S. These equations reveal fundamentally non-linear dynamics, despite the linearity of the attention mechanism itself. The learning rates are explicitly modulated by the squared eigenvalues S^2, indicating faster learning along directions of higher input variance. The product structure of terms like $\bar{p}_2\bar{q}_1^2$ leads to cooperative growth from small initial values.

The system converges to a fixed point where $\frac{\mathrm{d}\bar{p}_2}{\mathrm{d}t} = \mathbf{0}$ and $\frac{\mathrm{d}\bar{q}_1}{\mathrm{d}t} = \mathbf{0}$. This occurs when the product of the parameters reaches $\bar{p}_2(\infty)\bar{q}_1(\infty) = [s^\infty(S)]^{-1}$. This fixed point result precisely matches those found in fixed-point analyses of linear transformers trained for in-context regression [8, 12]. Figure 1 confirms this theoretical prediction against simulations. The effective linear transformation learned by the model at convergence is derived by substituting this fixed point into the predictor (Eq. 8):

$$\hat{y}_\mu(\infty) \approx W^\mu \left[\hat{\Sigma}_x \left(\frac{N+1}{N}\Sigma_x + \frac{\mathrm{Tr}(\Sigma_x)}{N} I\right)^{-1}\right] x_q. \tag{11}$$

In the large context limit ($N \to \infty$), assuming $\hat{\Sigma}_x \to \Sigma_x$, the term in brackets approaches I, yielding $\hat{y}_\mu(\infty) \to W^\mu x_q$. This demonstrates that the model learns to perfectly implement the target linear map. The term in brackets acts as a form of data-dependent preconditioning, related to ridge regression or preconditioned gradient descent updates identified in other works [8, 3].

Conserved Quantity and Reduced Degrees of Freedom. The ODE system (Eqs. 9-10) exhibits a conserved quantity: $\mathcal{C} = \|\bar{p}_2\|_F^2 - \|\bar{q}_1\|_F^2$ remains constant throughout training (Figure 2B, Appendix A.12). This conservation law stems from an underlying scaling symmetry in the loss function and implies that the parameters evolve on a lower-dimensional manifold within the full parameter space, constraining their trajectory. This suggests a form of implicit regularization or low-dimensional learning, reminiscent of phenomena observed in other deep learning contexts [23, 24]. The existence of this conserved quantity can also be derived in a more principled way by observing that the loss function $\mathcal{L}(\bar{p}_2, \bar{q}_1)$ is invariant under $\bar{p}_2 \to \eta\bar{p}_2$ and $\bar{q}_1 \to \frac{\bar{q}_1}{\eta}$, and using Noether's theorem.

Analytical Solution for Learning Trajectories and Timescales. The dynamics decouple along the eigenmodes $\alpha = 1, \ldots, d$. Defining the timescale $\tau_\alpha = (\eta P s_\alpha^2)^{-1}$ and the mode-specific fixed point inverse $s_\alpha^\infty = \frac{(N+1)s_\alpha + \mathrm{Tr}(S)}{N}$, the ODEs for the diagonal elements p_α, q_α become (Appendix A.12,

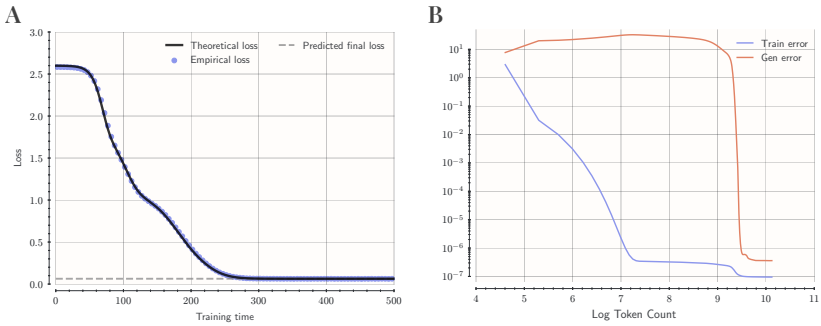

Figure 3: (A) Loss dynamics: theory vs. simulation. The empirical training loss curve closely matches the analytical prediction (Eq. 14), capturing characteristic plateaus and subsequent rapid decreases (cliffs) corresponding to the sequential learning of different spectral modes. (B) Training loss dynamics of a transformer model trained in modular arithmetic task displays qualitative similarity to the training dynamics of linear transformer in (A), and exhibits delayed generalized phenomena called "grokking".

Eqs. (A89)-(A90)):

$$\tau_\alpha \frac{\mathrm{d}p_\alpha}{\mathrm{d}t} = q_\alpha \left(1 - p_\alpha q_\alpha s_\alpha^\infty\right); \quad \tau_\alpha \frac{\mathrm{d}q_\alpha}{\mathrm{d}t} = p_\alpha \left(1 - p_\alpha q_\alpha s_\alpha^\infty\right). \tag{12}$$

If initialized symmetrically ($p_\alpha(0) = q_\alpha(0)$), then $p_\alpha(t) = q_\alpha(t)$ holds for all time. Defining the product $a_\alpha(t) = p_\alpha(t)q_\alpha(t)(= p_\alpha(t)^2)$, its dynamics follow a logistic equation: $\tau_\alpha \frac{\mathrm{d}a_\alpha}{\mathrm{d}t} = 2a_\alpha(1 - a_\alpha s_\alpha^\infty)$. This equation admits an exact analytical solution (Figure 2A, Appendix A.12, Eq. (A91)):

$$a_\alpha(t) = a_\alpha^\infty \cdot \frac{a_\alpha^0}{a_\alpha^0 + (a_\alpha^\infty - a_\alpha^0)\exp(-\frac{2t}{\tau_\alpha})}, \tag{13}$$

where $a_\alpha^\infty = 1/s_\alpha^\infty$ is the stable fixed point for mode α, and a_α^0 is the initial value. The characteristic time t_α^* required to transition from a small initialization ϵ to near its fixed point is approximately $t_\alpha^* \approx \frac{1}{2\lambda P s_\alpha^2} \log\left(\frac{1}{s_\alpha^\infty \epsilon}\right)$. This inverse dependence on s_α^2 establishes timescale separation: modes corresponding to larger input eigenvalues s_α learn significantly faster than those associated with smaller eigenvalues, mirroring the multi-stage learning observed in deep linear networks [10].

Loss Dynamics and Plateaus. Substituting the exact parameter trajectories $a_\alpha(t)$ into the expected loss yields the analytical loss evolution (Figure 3, Appendix A.13, Eq. (A110)):

$$\mathcal{L}(t) = \frac{1}{2}\sum_{\alpha=1}^{d} s_\alpha \left(\frac{s_\alpha}{a_\alpha^\infty} a_\alpha(t)^2 - 2s_\alpha a_\alpha(t) + 1\right). \tag{14}$$

This analytical form precisely captures the shape of the loss curve observed in simulations. The timescale separation inherent in $a_\alpha(t)$ directly translates into structured loss dynamics. The curve often exhibits plateaus, corresponding to periods where faster modes have converged but slower modes are still near their initial values (or slowly escaping saddle points near zero). Subsequent rapid drops ("cliffs") occur when these slower modes eventually learn, significantly reducing the remaining error (Figure 3A). This phenomenology of transformer training dynamics is widely observed empirically, and we show a particular example from a "grokking" Transformer [25] in Figure 3B, that has qualitative similarity to loss dynamics in our simplified model. The initial loss decreases linearly, with a rate proportional to initial parameter values and input covariance eigenvalues. The converged loss at $t \to \infty$ is non-zero due to the inherent randomness of finite context window and is given by $\mathcal{L}(\infty) = \frac{1}{2}\sum_\alpha \frac{s_\alpha(s_\alpha + \mathrm{Tr}(S))}{(N+1)s_\alpha + \mathrm{Tr}(S)}$. The loss dynamics near convergence are dominated by exponential relaxation, governed by the slowest learning modes. The final converged loss $\mathcal{L}(\infty)$ remains non-zero due to finite-N effects related to noise in the empirical covariance estimates (Appendix A.13, Eq. (A114)).

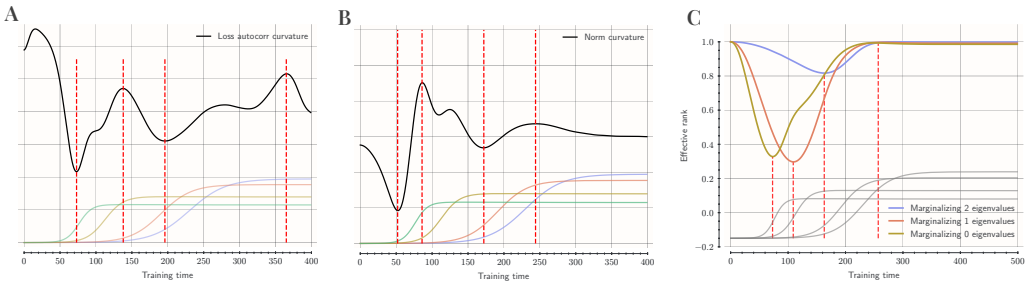

Figure 4: Separation of timescale can be identified in (A) Curvature of loss autocorrelation function, (B) Curvature of parameter norm dynamics, and (C) Marginalized effective rank measure.

4 Applications to Analysis of Non-linear Transformer Models

While our analytical derivations rely on a simplified linear transformer, we hypothesize that key features of the learning dynamics—particularly the separation of timescales dictated by data statistics—may generalize to more complex, non-linear architectures. To investigate this, we extend insights from the solvable model to develop empirical probes for analyzing macroscopic features of learning dynamics in non-linear transformers.

Probing Timescale Separation via Curvature. The solved loss dynamics (Eq. (14)) predict non-uniform learning rates across different spectral modes α, leading to characteristic plateaus followed by cliffs in the loss curve as modes α converge according to their timescales $\tau_\alpha \propto 1/s_\alpha^2$. To empirically detect these points of rapid change in potentially complex loss curves, we examine the curvature of the loss autocorrelation function. High curvature points should correspond to transitions in the learning process. Calculating the second derivative of the loss autocorrelation indeed reveals prominent peaks corresponding to expected mode activations (Figure 4A). Complementarily, we analyze the curvature of the Frobenius norm of the active parameter matrices ($\|p_2(t)\|_F$ or $\|q_1(t)\|_F$). Since the parameters p_α, q_α evolve according to Eq. (12) with mode-dependent rates, their overall norm should also exhibit non-uniform changes. Peaks in the parameter norm curvature align well with those in the loss autocorrelation (Figure 4B), reinforcing that macroscopic observables reflect the underlying timescale separation predicted by the theory.

Tracking Sequential Learning via Effective Rank. The exact solution $a_\alpha(t)$ (Eq. (13)) describes the sequential "activation" or learning of different spectral modes. To track this process empirically in the weight matrices of non-linear models, we utilize the effective rank [26], which measures the dimensionality or concentration of singular values (s_i) of a matrix M:

$$\text{EffRank}(M) = \exp\left(-\sum_i \frac{s_i}{\sum_j s_j} \log\left(\frac{s_i}{\sum_j s_j}\right)\right). \tag{15}$$

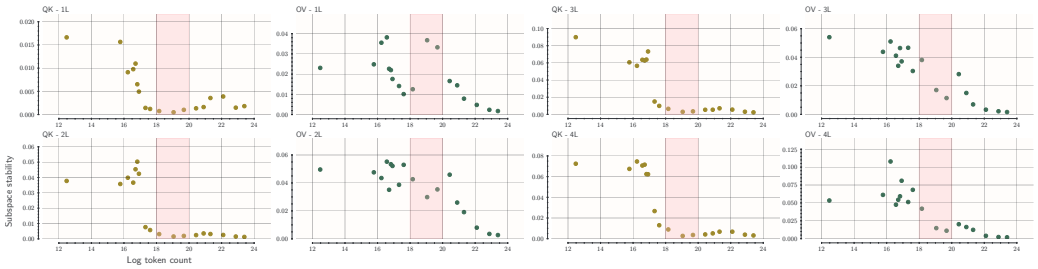

Figure 5: Subspace Stabilization in Attention-Only Models (1-4 Layers). Spectral alignment metric (Subspace Distance, Eq. (16)), averaged across OV and QK matrices, vs. training time. Lower values indicate subspace stability. The highlighted region marks ICL emergence. Subspace directions converge relatively early, preceding or coinciding with ICL emergence, especially in deeper models.

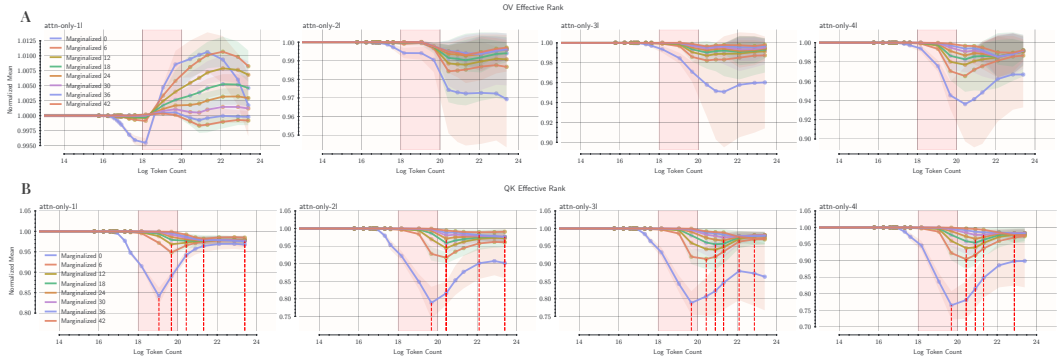

Figure 6: Timescale Separation in Anthropic's Attention-Only Models (1-4 Layers). (A) Marginalized effective rank for Output-Value (OV) matrices. (B) Marginalized effective rank for Query-Key (QK) matrices. Averaged across layers/matrices. Highlighted region: ICL emergence. Note the characteristic dips preceding/coinciding with ICL emergence in deeper models (esp. QK), absent in the 1-layer model.

A lower effective rank implies energy concentration in fewer singular modes. Motivated by the theoretical prediction that dominant modes (large s_α) learn first (small τ_α), we expect the effective rank to initially decrease as the model focuses on these modes, and subsequently increase as subdominant modes are learned. To better visualize this sequential process, we introduce "marginalized effective rank". This involves computing the effective rank repeatedly, each time excluding the singular value component corresponding to the mode assumed to have converged earliest. Dips in these marginalized curves should indicate the time points where specific spectral components undergo significant learning (Figure 4C). As predicted by the sequential learning in Eq. (13), the effective rank initially dips, then recovers, and the marginalized curves reveal staggered learning across modes. We applied these analyses to attention-only non-linear transformers (1-4 layers), similar to those exhibiting ICL capabilities [18]. Figure 6 displays the marginalized effective rank for Output-Value (OV) and Query-Key (QK) matrices. Consistent with the timescale separation hypothesis, models capable of ICL (2+ layers) show characteristic dips in the effective rank curves, particularly for QK matrices, around the time ICL performance emerges (highlighted region). The full set of marginalized curves reveals multiple minima occurring sequentially, supporting the idea derived from our linear model that learning proceeds in stages along different spectral dimensions. In contrast, the 1-layer model, which fails at this ICL task, shows a monotonic increase in OV effective rank, suggesting undifferentiated learning or noise fitting rather than structured, sequential learning. These empirical results suggest that the sequential, mode-dependent learning predicted by our solvable model qualitatively extends to these non-linear settings and correlates with the emergence of functional capabilities like ICL.

Stability of Parameter Subspace. Our theoretical analysis separates the dynamics of parameter magnitudes (related to $a_\alpha = p_\alpha q_\alpha$) from the underlying parameter structure. The decoupled dynamics (Eq. (12)) allow for the possibility that the directions spanned by the singular vectors of weight matrices stabilize faster than the singular values reach their fixed points. To test this, we introduce a "Subspace Distance" metric, measuring the alignment between the subspace spanned by the singular vectors of a weight matrix $M(t)$ at time t and its final state $M(\infty)$:

$$\text{Sub. Dist.}\left[M(t)\right] = \min_{A \in \mathbb{R}^{D \times D}} \left\| AM(t) - M(\infty) \right\| . \tag{16}$$

A smaller distance indicates subspace stabilization. Analyzing the attention-only models (Figure 5), we find that the subspace distance (averaged across OV/QK matrices) decreases and stabilizes relatively early in training, often preceding or coinciding with the emergence of ICL and the primary dips in effective rank. This suggests that the network first identifies the relevant spectral directions and subsequently adjusts the magnitudes along these directions, consistent with a potential separation in learning dynamics for direction and magnitude. This hypothesis is further tested by examining models exhibiting "grokking" in modular arithmetic tasks [25], where generalization is significantly delayed relative to training performance saturation. Figure 3B shows the characteristic delayed drop in test error. Analyzing the OV matrix dynamics, we observe that subspace stabilization, measured by the Subspace Distance (Figure 7C), is markedly delayed. It occurs significantly after the training

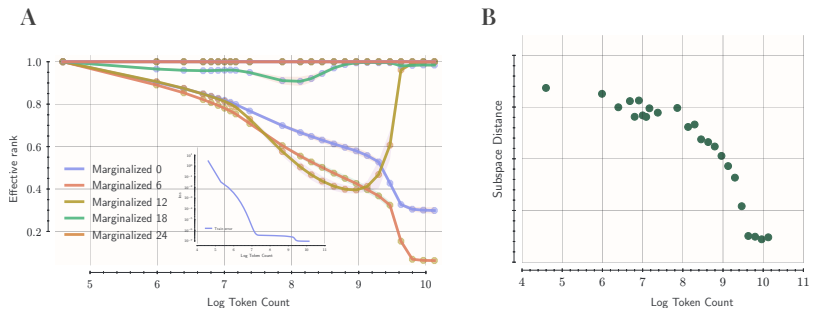

Figure 7: Spectral Analysis of Grokking in Modular Arithmetic. (A) Delayed generalization characteristic of grokking (Train loss drops early, Test loss drops late). (B) Effective rank dynamics (OV matrix). (C) Subspace Distance (OV matrix). Note the late stabilization of the subspace, coinciding with the drop in test error (generalization), long after the training loss has converged.

loss has minimized and coincides closely with the sharp improvement in generalization (test error drop). The effective rank (Figure 7B) also shows dynamics linked to this delayed phase. This finding strongly suggests that grokking, in this context, is linked to a delayed convergence of the parameter subspace, providing a potential mechanistic explanation. Once the correct subspace is identified (albeit late), learning of magnitudes proceeds rapidly, leading to generalization.

In summary, empirical analyses of non-linear attention-based models using measures motivated by our solvable linear model reveal compelling qualitative parallels. Timescale separation, sequential mode-dependent learning, and the distinct role of subspace stabilization appear as potentially robust features extending beyond the linear regime. This suggests that insights from exactly solvable models can offer a valuable lens for interpreting complex learning dynamics in practical transformer architectures.

5 Discussion

By analyzing an exactly solvable linear transformer model, we have derived the precise learning dynamics underlying its acquisition of in-context linear regression capabilities. Our central finding is that learning decomposes into independent dynamics along eigenmodes defined by the input data statistics. This decomposition naturally leads to a separation of timescales, where different modes are learned at rates inversely proportional to the square of the corresponding data eigenvalues ($\tau_\alpha \propto 1/s_\alpha^2$). This sequential learning manifests as distinct stages in parameter evolution and characteristic plateaus and cliffs in the loss function.

The learned computation at the fixed point (Eq. (11)) effectively implements the target linear map W^μ by applying an operator $\hat{\Sigma}_x[s^\infty(\Sigma_x)]^{-1}$ that acts to approximately whiten the input data x_q relative to the context statistics $\hat{\Sigma}_x$. This operator effectively inverts the data covariance modulated by finite-context effects (N), allowing the model to isolate the underlying linear relationship. Broadly, our work demonstrates how analytical techniques from deep linear network theory [10] can be extended to analyze context-dependent computations in transformers, revealing that timescale separation can arise directly from the interaction of attention parameters with data statistics, not just from network depth. Furthermore, the analysis revealed aspects of low-dimensional learning. First, parameter specialization occurs, where certain parameter blocks (p_1, q_2) remain "null" while others (p_2, q_1) actively drive learning. Second, a conservation law constrains the dynamics, reducing the effective dimensionality of the parameter trajectory and highlighting symmetries in the learning landscape. Crucially, the qualitative features predicted by our solvable model—timescale separation, sequential mode learning, subspace stabilization—appear empirically relevant in non-linear attention-only models and grokking scenarios. The alignment between theoretical predictions and empirical observations using measures like effective rank and subspace distance suggests that the core mechanisms identified in the linear model offer valuable insights into the learning processes of more complex transformers.

Finally, this work suggests a principled approach to interpretability and evaluation. The theoretically grounded measures developed here—loss/norm curvature, marginalized effective rank, subspace

distance—can empirically identify critical time points during training associated with significant learning events or potential shifts in model capabilities. Monitoring these metrics could provide a more informed strategy for timing behavioral evaluations, potentially capturing the emergence of new functionalities more effectively than relying solely on overall loss or accuracy measures. This offers a promising direction for bridging theoretical understanding with practical monitoring and analysis of large model training. Several avenues remain for future investigation. Incorporating non-linear attention mechanisms is essential to understand their impact on dynamics and potential deviations from the linear theory. Non-Gaussian data model can give rise to richer dynamics due to non-trivial higher moments and might lead to better predictions on dynamics in real-world models. Analyzing the effects of network depth and multiple attention heads could reveal how scaling influences ICL capabilities and learning complexity. Extending the analysis beyond linear regression to tasks involving non-linear functions or discrete structures is also a critical next step.

Acknowledgement

We thank Jan Hendrick Kirchner for his guidance during the PIBBSS 2023 summer fellowship, and PIBBSS for continued support of the research work.

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, pages 5998–6008.

[2] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, pages 1877–1901.

[3] Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In *Proceedings of the 40th International Conference on Machine Learning (ICML 2023)*, volume 202 of *Proceedings of Machine Learning Research*, pages 34794–34808.

[4] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé de Oliveira Penedones, Karolis J Sidor, Yuntian Wu, Charlie Zhang, Diogo Oliveira, Oleksii Kuchaiev, et al. Evaluating large language models trained on code, .

[5] Jürgen Schmidhuber. Learning to control fast-weight memories: An alternative to dynamic recurrent networks. 4(1):131–139.

[6] Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear transformers are secretly fast weight programmers. In *Proceedings of the 38th International Conference on Machine Learning (ICML 2021)*, volume 139 of *Proceedings of Machine Learning Research*, pages 9355–9366.

[7] Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? Investigations with linear models. In *The Eleventh International Conference on Learning Representations (ICLR 2023)*.

[8] Kwangjun Ahn, Xiang Cheng, Hadi Daneshmand, and Suvrit Sra. Transformers learn to implement preconditioned gradient descent for in-context learning. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, pages 63970–63984.

[9] Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. What can transformers learn in-context? A case study of simple function classes. In *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*, pages 30484–30498.

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

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

[12] Ruiqi Zhang, Spencer Frei, and Peter L Bartlett. Trained Transformers Learn Linear Models In-Context.

[13] Deqing Fu, Tianqi Chen, Robin Jia, and Vatsal Sharan. Transformers learn higher-order optimization methods for in-context learning: A study with linear models. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*.

[14] Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit Bayesian inference. In *The Tenth International Conference on Learning Representations (ICLR 2022)*.

[15] Chi Han, Ziqi Wang, Han Zhao, and Heng Ji. Explaining emergent in-context learning as kernel regression.

[16] Madhu S Advani and Andrew M Saxe. High-dimensional dynamics of generalization error in neural networks.

[17] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems 31 (NeurIPS 2018)*, pages 8571–8580.

[18] Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Benjamin Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. In-context learning and induction heads.

[19] Enric Boix-Adsera, Etai Littwin, Emmanuel Abbe, Samy Bengio, and Joshua Susskind. Transformers learn through gradual rank increase.

[20] Siyu Chen, Heejune Sheen, Tianhao Wang, and Zhuoran Yang. Training Dynamics of Multi-Head Softmax Attention for In-Context Learning: Emergence, Convergence, and Optimality, .

[21] Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Thomas Adler, Lukas Gruber, Markus Holzleitner, Milena Pavlovic, Geir Kjetil Sandve, et al. Hopfield networks is all you need. In *Advances in Neural Information Processing Systems 34 (NeurIPS 2021)*, pages 28866–28878.

[22] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. 21(140):1–67.

[23] Giulio Biroli, Marco Gerace, Florent Krzakala, Marc Mezard, and Pierfrancesco Paga. On the interplay between kernel dynamics and generalization in overparameterized neural networks.

[24] Marco Gerace, Florent Krzakala, Marc Mezard, Pierfrancesco Paga, and Lenka Zdeborová. Disentangling the roles of data, initialization, and optimization for training sentence encoders. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8434–8444.

[25] Neel Nanda, Lawrence Chan, Tom Liberum, Jess Smith, and Stefo Toshniwal. Grokking: Generalization beyond overfitting on small algorithmic datasets.

[26] Olivier Roy and Martin Vetterli. The effective rank: A measure of effective dimensionality. 104 (18):7255–7260.

A Model Setup

A.1 Model Architecture

We consider a single-layer transformer network with a single linear attention head. The attention mechanism employed is purely linear, omitting the softmax normalization typically found in standard attention mechanisms.

A.2 Variables

We define the key variables used in our model as follows:

- **Input Data Pairs:** We are given N pairs of d-dimensional data points $(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R}^d$ for $i = 1, \ldots, N$, and a query point $x_q \in \mathbb{R}^d$.
- **Embedding Dimension:** The embedding dimension is set to $D = 2d$.
- **Input Sequence Matrix:** For an input sequence of N data pairs and a query point $(x_1, y_1, \ldots, x_N, y_n, x_q)$, the input to the transformer is constructed as a matrix $Z \in \mathbb{R}^{D \times (N+1)}$. Specifically, Z is defined as:

$$Z = \begin{bmatrix} x_1 & x_2 & \ldots & x_N & x_q \\ y_1 & y_2 & \ldots & y_N & \mathbf{0}_d \end{bmatrix} \in \mathbb{R}^{D \times (N+1)}, \tag{A1}$$

  where $\mathbf{0}_d \in \mathbb{R}^d$ is a zero vector of dimension d.
- **Weight Matrices:**
  - **Combined Query-Key Weight Matrix:** $W^Q \in \mathbb{R}^{D \times D}$. We define a submatrix $q \in \mathbb{R}^{D \times d}$ comprising the first d columns of W^Q. This submatrix is further partitioned as $q = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix}$, where $q_1, q_2 \in \mathbb{R}^{d \times d}$.
  - **Output Projection Weight Matrix:** $W^P \in \mathbb{R}^{D \times D}$. We define a submatrix $p^T \in \mathbb{R}^{d \times D}$ comprising the bottom d rows of W^P. This submatrix is further partitioned as $p^T = (p_1 \quad p_2)$, where $p_1, p_2 \in \mathbb{R}^{d \times d}$.

A.3 Linear Transformer

The output of the linear attention head, before the residual connection, is given by:

$$O = W^P \left(\frac{ZZ^\top}{N} \right) W^Q Z \in \mathbb{R}^{D \times (N+1)}, \tag{A2}$$

where the attention operation is normalized by N.

The output of the single-layer linear transformer network, denoted as $f(Z) \in \mathbb{R}^{D \times (N+1)}$, is obtained by adding a residual connection:

$$f(Z) = Z + O = Z + W^P \left(\frac{ZZ^\top}{N} \right) W^Q Z \in \mathbb{R}^{D \times (N+1)}. \tag{A3}$$

A.4 Predictor

We are interested in the model's prediction for the query input x_q. This prediction is derived from the $(N+1)$-th column of the transformer network's output, $f(Z)$. Specifically, the prediction $\hat{y} \in \mathbb{R}^d$ is obtained from the bottom d elements of the $(N+1)$-th column of $f(Z)$.

Due to the specific embedding structure defined in Section A.2 and the output reading mechanism, the prediction can be simplified to:

$$\hat{y} = \left(p^\top \hat{\Gamma} q \right) x_q \in \mathbb{R}^d, \tag{A4}$$

where $\hat{\Gamma} = \frac{ZZ^\top}{N} \in \mathbb{R}^{D \times D}$ is the empirical covariance matrix of the input Z. Note that the matrix product $(p^\top \hat{\Gamma} q)$ results in a $d \times d$ matrix. The explicit derivation of this simplification is provided in Section A.9.

A.5 Loss Function

To train the model, we employ the mean squared error (MSE) loss function, defined as:

$$\mathcal{L} = \frac{1}{2}\|\hat{y} - y_q\|_2^2 = \frac{1}{2}\left\|\left(p^\top \hat{\Gamma} q\right) x_q - y_q\right\|_2^2. \tag{A5}$$

This loss function quantifies the discrepancy between the model's prediction $\hat{y}$ and the ground truth y_q.

A.6 Goal

The primary objective is to mathematically analyze this exactly solvable model of a linear transformer for in-context regression. Specifically, we aim to understand the learning dynamics and mechanisms through which the model predicts y_q based on the provided in-context examples $\{(x_i, y_i)\}_{i=1}^N$.

A.7 Key Assumptions

Our analysis relies on the following key assumptions regarding the model architecture and setup:

- **Single-layer, single-head linear transformer.** We consider a simplified transformer architecture with only one layer and a single attention head, both of which are linear.
- **No softmax in attention mechanism.** The linear attention mechanism omits the softmax normalization, resulting in a purely linear operation.
- **Specific embedding mechanism without positional tokens.** We employ a specific input embedding structure as defined in Section A.2, without the use of positional tokens.
- **Prediction focused on the $(N+1)$-th output token.** We focus our analysis on the prediction derived from the $(N+1)$-th output token, corresponding to the query input x_q.

A.8 Task Setup

We aim to train the linear transformer to perform in-context linear regression. To this end, we generate training data by sampling random instances of linear regression problems.

A.8.1 x-data Generation

During training, the input data x is sampled from a multivariate Gaussian distribution with zero mean and a diagonal covariance matrix $\Sigma_x \in \mathbb{R}^{d\times d}$. Formally,

$$x \sim \mathcal{N}\left(\mathbf{0}_d, \Sigma_x\right). \tag{A6}$$

Here, $\mathbf{0}_d \in \mathbb{R}^d$ is a zero-mean vector, and $\Sigma_x = \text{diag}(\sigma_{x,1}^2, \ldots, \sigma_{x,d}^2)$ is a diagonal covariance matrix, where $\sigma_{x,i}^2 > 0$ are the variances along each dimension.

A.8.2 Transformation Matrix W Generation

The transformation matrix $W \in \mathbb{R}^{d\times d}$ is generated to define the linear regression task. First, we sample d eigenvalues $\lambda_1, \ldots, \lambda_d$ from a standard multivariate Gaussian distribution $\mathcal{N}(\mathbf{0}_d, I_d)$. We then construct a diagonal matrix $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_d) \in \mathbb{R}^{d\times d}$. For a fixed orthogonal matrix $V \in \mathbb{R}^{d\times d}$, the transformation matrix W is defined as:

$$W = V\Lambda V^\top \in \mathbb{R}^{d\times d}. \tag{A7}$$

Note that, due to the zero mean of the eigenvalues, the expected value of W is $\mathbb{E}[W] = 0$. Furthermore, $\mathbb{E}[WW^\top] = V\mathbb{E}[\Lambda^2]V^\top = VI_dV^\top = VV^\top$, since $\mathbb{E}[\Lambda^2] = I_d$ when eigenvalues are sampled from standard Gaussian. We consider P independent regression tasks, each defined by a transformation matrix W^μ for $\mu = 1, \ldots, P$. These matrices $\{W^\mu\}_{\mu=1}^P$ are sampled at the beginning of training and remain fixed throughout the training process.

A.8.3 y-data Generation and Empirical Covariance

For each regression task μ defined by W^μ, we generate the corresponding y-data. First, we sample $x_1, \ldots, x_N, x_q \in \mathbb{R}^d$ from the distribution defined in Section A.8.1. Then, for each x_i ($i = 1, \ldots, N$), we generate y_i using the linear transformation $y_i = W^\mu x_i$. For the query point x_q, the ground truth y_q is similarly given by $y_q = W^\mu x_q$.

For a specific task μ, the input sequence matrix Z^μ is constructed as:

$$Z^\mu = \begin{bmatrix} x_1 & x_2 & \ldots & x_N & x_q \\ W^\mu x_1 & W^\mu x_2 & \ldots & W^\mu x_N & \mathbf{0}_d \end{bmatrix} \in \mathbb{R}^{D \times (N+1)}. \tag{A8}$$

The model's prediction is based on the empirical covariance matrix of Z^μ, denoted as $\hat{\Gamma}^\mu = \frac{1}{N} Z^\mu (Z^\mu)^\top \in \mathbb{R}^{D \times D}$. To derive an explicit form for $\hat{\Gamma}^\mu$, we first compute $Z^\mu (Z^\mu)^\top$:

$$\begin{aligned} Z^\mu (Z^\mu)^\top &= \begin{bmatrix} x_1 & \ldots & x_N & x_q \\ W^\mu x_1 & \ldots & W^\mu x_N & \mathbf{0}_d \end{bmatrix} \begin{bmatrix} x_1^\top & (W^\mu x_1)^\top \\ \vdots & \vdots \\ x_N^\top & (W^\mu x_N)^\top \\ x_q^\top & \mathbf{0}_d^\top \end{bmatrix} \\ &= \begin{bmatrix} \sum_{i=1}^N x_i x_i^\top + x_q x_q^\top & \sum_{i=1}^N x_i (W^\mu x_i)^\top \\ \sum_{i=1}^N W^\mu x_i x_i^\top & \sum_{i=1}^N W^\mu x_i (W^\mu x_i)^\top \end{bmatrix} \\ &= \begin{bmatrix} \sum_{i=1}^N x_i x_i^\top + x_q x_q^\top & \sum_{i=1}^N x_i x_i^\top (W^\mu)^\top \\ \sum_{i=1}^N W^\mu x_i x_i^\top & \sum_{i=1}^N W^\mu x_i x_i^\top (W^\mu)^\top \end{bmatrix}. \end{aligned}$$

Thus, the empirical covariance matrix $\hat{\Gamma}^\mu$ is given by:

$$\hat{\Gamma}^\mu = \frac{1}{N} Z^\mu (Z^\mu)^\top = \begin{bmatrix} \frac{1}{N} \sum_{i=1}^N x_i x_i^\top + \frac{1}{N} x_q x_q^\top & \frac{1}{N} \sum_{i=1}^N x_i x_i^\top (W^\mu)^\top \\ \frac{1}{N} \sum_{i=1}^N W^\mu x_i x_i^\top & \frac{1}{N} \sum_{i=1}^N W^\mu x_i x_i^\top (W^\mu)^\top \end{bmatrix} \in \mathbb{R}^{D \times D}. \tag{A9}$$

Let $\hat{\Sigma}_x = \frac{1}{N} \sum_{i=1}^N x_i x_i^\top$ denote the empirical covariance of the training x-data $\{x_i\}_{i=1}^N$. Then, $\hat{\Gamma}^\mu$ can be expressed in block form as:

$$\hat{\Gamma}^\mu = \begin{bmatrix} \hat{\Sigma}_x + \frac{1}{N} x_q x_q^\top & \hat{\Sigma}_x (W^\mu)^\top \\ W^\mu \hat{\Sigma}_x & W^\mu \hat{\Sigma}_x (W^\mu)^\top \end{bmatrix}. \tag{A10}$$

In some subsequent derivations, for simplicity of notation, we might approximate $\hat{\Gamma}^\mu$ by ignoring the term $\frac{1}{N} x_q x_q^\top$ in the top-left block, leading to:

$$\hat{\Gamma}^\mu \approx \begin{bmatrix} \hat{\Sigma}_x & \hat{\Sigma}_x (W^\mu)^\top \\ W^\mu \hat{\Sigma}_x & W^\mu \hat{\Sigma}_x (W^\mu)^\top \end{bmatrix}. \tag{A11}$$

This approximation is valid when N is sufficiently large such that the contribution of the query point x_q to the empirical covariance is negligible compared to the contribution of the in-context examples $\{x_i\}_{i=1}^N$.

A.8.4 Training Setup

We train the model using stochastic gradient descent (SGD) with a learning rate λ. Training proceeds in task-centric epochs. In each epoch, we iterate through the set of P regression tasks $\{W^\mu\}_{\mu=1}^P$. For each task W^μ, we sample a batch of in-context examples $\{(x_i, y_i)\}_{i=1}^N$ and a query point x_q, and compute the loss function $\mathcal{L}^\mu$ as defined in Section A.5. We then update the model parameters using the gradient of $\mathcal{L}^\mu$ with respect to the parameters, accumulated over the batch. A training epoch is completed once all P tasks have been presented to the model. Using batches for gradient updates helps to reduce noise in the training process.

A.9 Simplified Predictor

In this section, we derive the simplified form of the predictor given in Equation (A4). Recall from Equation (A3) that the output of the linear transformer is given by:

$$f(Z) = Z + W^P \left(\frac{Z Z^\top}{N} \right) W^Q Z = Z + W^P \hat{\Gamma} W^Q Z. \tag{A12}$$

We are interested in the prediction $\hat{y}$, which is obtained from the bottom d elements of the $(N+1)$-th column of $f(Z)$.

First, let's consider the $(N+1)$-th column of the input matrix Z, which is given by $\begin{pmatrix} x_q \\ \mathbf{0}_d \end{pmatrix} \in \mathbb{R}^D$. Then, the $(N+1)$-th column of the matrix product $W^Q Z$ is obtained by multiplying W^Q with this column vector:

$$W^Q \begin{pmatrix} x_q \\ \mathbf{0}_d \end{pmatrix} = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} x_q \in \mathbb{R}^D, \tag{A13}$$

where we have used the partition of $W^Q = [q|*]$ and $q = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix}$ as defined in Section A.2.

Next, we consider the term $\hat{\Gamma} W^Q Z = \left(\frac{ZZ^\top}{N}\right) W^Q Z$. The $(N+1)$-th column of this product is obtained by multiplying the matrix $\hat{\Gamma}$ with the $(N+1)$-th column of $W^Q Z$, which we just computed in Equation (A13). Thus, the $(N+1)$-th column of $\hat{\Gamma} W^Q Z$ is:

$$\hat{\Gamma} \left(W^Q \begin{pmatrix} x_q \\ \mathbf{0}_d \end{pmatrix} \right) = \hat{\Gamma} \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} x_q \in \mathbb{R}^D. \tag{A14}$$

Now, we consider the term $W^P \hat{\Gamma} W^Q Z$. The $(N+1)$-th column of this term is obtained by multiplying W^P with the vector in Equation (A14). Using the partition of $W^P = \begin{pmatrix} * \\ p^T \end{pmatrix}$ and $p^T = (p_1 \quad p_2)$ from Section A.2, we have:

$$W^P \left(\hat{\Gamma} \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} x_q \right) = \begin{pmatrix} * \\ p^T \end{pmatrix} \hat{\Gamma} \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} x_q = \begin{pmatrix} * \\ p^T \hat{\Gamma} \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} x_q \end{pmatrix} \in \mathbb{R}^D, \tag{A15}$$

where $*$ denotes the top d rows of the resulting vector, which are not relevant for our prediction.

Finally, we look at the $(N+1)$-th column of $f(Z) = Z + W^P \hat{\Gamma} W^Q Z$. This column is the sum of the $(N+1)$-th column of Z and the $(N+1)$-th column of $W^P \hat{\Gamma} W^Q Z$:

$$\begin{pmatrix} x_q \\ \mathbf{0}_d \end{pmatrix} + W^P \hat{\Gamma} \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} x_q = \begin{pmatrix} x_q \\ \mathbf{0}_d \end{pmatrix} + \begin{pmatrix} * \\ p^T \hat{\Gamma} \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} x_q \end{pmatrix} = \begin{pmatrix} * \\ \mathbf{0}_d + p^T \hat{\Gamma} \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} x_q \end{pmatrix} \in \mathbb{R}^D. \tag{A16}$$

The prediction $\hat{y}$ is the bottom d elements of this resulting column. Thus, we have:

$$\hat{y} = \mathbf{0}_d + p^T \hat{\Gamma} \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} x_q = p^T \hat{\Gamma} \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} x_q = (p_1 \quad p_2) \hat{\Gamma} \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} x_q \in \mathbb{R}^d. \tag{A17}$$

Since $q = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix}$ and $p^T = (p_1 \quad p_2)$, we can rewrite the prediction as:

$$\hat{y} = \left(p^T \hat{\Gamma} q \right) x_q = \left((p_1 \quad p_2) \hat{\Gamma} \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} \right) x_q \in \mathbb{R}^d. \tag{A18}$$

This matches the simplified predictor form given in Equation (A4).

For a specific regression task μ, using the approximated empirical covariance matrix from Equation (A11), $\hat{\Gamma}^\mu \approx \begin{bmatrix} \hat{\Sigma}_x & \hat{\Sigma}_x(W^\mu)^\top \\ W^\mu\hat{\Sigma}_x & W^\mu\hat{\Sigma}_x(W^\mu)^\top \end{bmatrix}$, the prediction can be further expanded as:

$$\begin{aligned}
\hat{y}_\mu &= \left(p^T\hat{\Gamma}^\mu q\right) x_q \\
&= (p_1 \quad p_2) \begin{bmatrix} \hat{\Sigma}_x & \hat{\Sigma}_x(W^\mu)^\top \\ W^\mu\hat{\Sigma}_x & W^\mu\hat{\Sigma}_x(W^\mu)^\top \end{bmatrix} \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} x_q \\
&= (p_1 \quad p_2) \begin{pmatrix} \hat{\Sigma}_x q_1 + \hat{\Sigma}_x(W^\mu)^\top q_2 \\ W^\mu\hat{\Sigma}_x q_1 + W^\mu\hat{\Sigma}_x(W^\mu)^\top q_2 \end{pmatrix} x_q \\
&= \left(p_1(\hat{\Sigma}_x q_1 + \hat{\Sigma}_x(W^\mu)^\top q_2) + p_2(W^\mu\hat{\Sigma}_x q_1 + W^\mu\hat{\Sigma}_x(W^\mu)^\top q_2)\right) x_q \\
&= \left(p_1\hat{\Sigma}_x q_1 + p_1\hat{\Sigma}_x(W^\mu)^\top q_2 + p_2 W^\mu\hat{\Sigma}_x q_1 + p_2 W^\mu\hat{\Sigma}_x(W^\mu)^\top q_2\right) x_q \\
&= \left((p_1\hat{\Sigma}_x + p_2 W^\mu\hat{\Sigma}_x)q_1 + (p_1\hat{\Sigma}_x(W^\mu)^\top + p_2 W^\mu\hat{\Sigma}_x(W^\mu)^\top)q_2\right) x_q \\
&= \left((p_1 + p_2 W^\mu)\hat{\Sigma}_x q_1 + (p_1 + p_2 W^\mu)\hat{\Sigma}_x(W^\mu)^\top q_2\right) x_q \\
&= \left((p_1 + p_2 W^\mu)\hat{\Sigma}_x(q_1 + (W^\mu)^\top q_2)\right) x_q.
\end{aligned}$$

Thus, for task μ, the simplified predictor is given by:

$$\hat{y}_\mu = \left((p_1 + p_2 W^\mu)\hat{\Sigma}_x(q_1 + (W^\mu)^\top q_2)\right) x_q \in \mathbb{R}^d. \tag{A19}$$

When we initialize $p_1 = 0$ and $q_2 = 0$, the predictor for task μ further simplifies to:

$$\hat{y}_\mu = \left(p_2 W^\mu\hat{\Sigma}_x q_1\right) x_q \in \mathbb{R}^d. \tag{A20}$$

This simplified form will be used in the subsequent analysis of the learning dynamics.

A.10 Null Parameter Analysis

In this section, we demonstrate that with specific initial conditions, namely $p_1 = 0$ and $q_2 = 0$, the expected gradients with respect to p_1 and q_2 are zero at the beginning of training. This result justifies setting these parameters to zero throughout the training process, thereby simplifying the predictor.

Recall the loss function for a single task μ is given by:

$$\mathcal{L}^\mu = \frac{1}{2}\|\hat{y}_\mu - y_q^\mu\|_2^2 = \frac{1}{2}\left\|\left(p^T\hat{\Gamma}^\mu q\right) x_q - y_q^\mu\right\|_2^2, \tag{A21}$$

and the simplified predictor is:

$$\hat{y}_\mu = \left(p^T\hat{\Gamma}^\mu q\right) x_q = \left((p_1 \quad p_2)\,\hat{\Gamma}^\mu \begin{pmatrix} q_1 \\ q_2 \end{pmatrix}\right) x_q = \left((p_1 + p_2 W^\mu)\hat{\Sigma}_x(q_1 + (W^\mu)^\top q_2)\right) x_q, \tag{A22}$$

where we have used the approximated empirical covariance matrix $\hat{\Gamma}^\mu \approx \begin{bmatrix} \hat{\Sigma}_x & \hat{\Sigma}_x(W^\mu)^\top \\ W^\mu\hat{\Sigma}_x & W^\mu\hat{\Sigma}_x(W^\mu)^\top \end{bmatrix}$ and expanded the matrix product. For simplicity in gradient derivation, let us define an intermediate term $A^\mu \in \mathbb{R}^d$ as:

$$A^\mu = \hat{\Sigma}_x(q_1 + (W^\mu)^\top q_2)x_q. \tag{A23}$$

Then the predictor can be written as:

$$\hat{y}_\mu = (p_1 + p_2 W^\mu)A^\mu. \tag{A24}$$

And the loss function becomes:

$$\mathcal{L}^\mu = \frac{1}{2}\|(p_1 + p_2 W^\mu)A^\mu - W^\mu x_q\|_2^2. \tag{A25}$$

A.10.1 Gradient with respect to p_1

To compute the gradient of $\mathcal{L}^\mu$ with respect to p_1, we use the chain rule. Let $g(p_1) = (p_1 + p_2 W^\mu)A^\mu - W^\mu x_q$. Then $\mathcal{L}^\mu = \frac{1}{2}\|g(p_1)\|_2^2 = \frac{1}{2} g(p_1)^\top g(p_1)$. The gradient is given by:

$$\frac{\partial \mathcal{L}^\mu}{\partial p_1} = \frac{\partial g(p_1)}{\partial p_1}^\top \frac{\partial \mathcal{L}^\mu}{\partial g(p_1)} = \left(\frac{\partial g(p_1)}{\partial p_1}\right)^\top g(p_1). \tag{A26}$$

We compute the derivative of $g(p_1)$ with respect to p_1. Since $p_1 \in \mathbb{R}^{d\times d}$, we consider the derivative with respect to p_1 as a linear transformation.

$$\frac{\partial g(p_1)}{\partial p_1} = \frac{\partial}{\partial p_1}\left[(p_1 + p_2 W^\mu)A^\mu - W^\mu x_q\right] = \frac{\partial}{\partial p_1}\left[p_1 A^\mu + p_2 W^\mu A^\mu - W^\mu x_q\right] = A^{\mu\top}. \tag{A27}$$

Substituting this and $g(p_1)$ into Equation (A26), we get:

$$\frac{\partial \mathcal{L}^\mu}{\partial p_1} = (A^\mu)^\top \left((p_1 + p_2 W^\mu)A^\mu - W^\mu x_q\right). \tag{A28}$$

Substituting back the expression for A^μ from Equation (A23):

$$\frac{\partial \mathcal{L}^\mu}{\partial p_1} = \left(\hat{\Sigma}_x(q_1 + (W^\mu)^\top q_2)x_q\right)^\top \left((p_1 + p_2 W^\mu)\hat{\Sigma}_x(q_1 + (W^\mu)^\top q_2)x_q - W^\mu x_q\right). \tag{A29}$$

The change in p_1 after one SGD update step, summing over all P tasks, is:

$$\Delta p_1 = -\lambda \sum_{\mu=1}^{P} \left(\hat{\Sigma}_x(q_1 + (W^\mu)^\top q_2)x_q\right)^\top \left((p_1 + p_2 W^\mu)\hat{\Sigma}_x(q_1 + (W^\mu)^\top q_2)x_q - W^\mu x_q\right). \tag{A30}$$

Now, we evaluate the expected update $\mathbb{E}_W[\Delta p_1]$ under the initial conditions $p_1 = 0$ and $q_2 = 0$. Substituting $p_1 = 0$ and $q_2 = 0$ into Equation (A30), we get:

$$\Delta p_1 = -\lambda \sum_{\mu=1}^{P} \left(\hat{\Sigma}_x(q_1 + (W^\mu)^\top 0)x_q\right)^\top \left((0 + p_2 W^\mu)\hat{\Sigma}_x(q_1 + (W^\mu)^\top 0)x_q - W^\mu x_q\right). \tag{A31}$$

Simplifying this expression:

$$\begin{aligned}
\Delta p_1 &= -\lambda \sum_{\mu=1}^{P} \left(\hat{\Sigma}_x q_1 x_q\right)^\top \left(p_2 W^\mu \hat{\Sigma}_x q_1 x_q - W^\mu x_q\right) && \text{(A32)}\\
&= -\lambda \sum_{\mu=1}^{P} \left(x_q^\top q_1^\top \hat{\Sigma}_x^\top\right) \left(p_2 W^\mu \hat{\Sigma}_x q_1 x_q - W^\mu x_q\right) && \text{(A33)}\\
&= -\lambda \sum_{\mu=1}^{P} x_q^\top q_1^\top \hat{\Sigma}_x \left(p_2 W^\mu \hat{\Sigma}_x q_1 x_q - W^\mu x_q\right) && \text{(A34)}\\
&= -\lambda \sum_{\mu=1}^{P} \left(x_q^\top q_1^\top \hat{\Sigma}_x p_2 W^\mu \hat{\Sigma}_x q_1 x_q - x_q^\top q_1^\top \hat{\Sigma}_x W^\mu x_q\right) && \text{(A35)}\\
&= -\lambda \sum_{\mu=1}^{P} x_q^\top q_1^\top \hat{\Sigma}_x p_2 W^\mu \hat{\Sigma}_x q_1 x_q + \lambda \sum_{\mu=1}^{P} x_q^\top q_1^\top \hat{\Sigma}_x W^\mu x_q. && \text{(A36)}
\end{aligned}$$

Taking the expectation of Δp_1 with respect to W, and using the linearity of expectation:

$$\mathbb{E}_W[\Delta p_1] = -\lambda \sum_{\mu=1}^{P} x_q^\top q_1^\top \hat{\Sigma}_x p_2 \mathbb{E}_W[W^\mu] \hat{\Sigma}_x q_1 x_q + \lambda \sum_{\mu=1}^{P} x_q^\top q_1^\top \hat{\Sigma}_x \mathbb{E}_W[W^\mu] x_q. \tag{A37}$$

Since we have $\mathbb{E}_W[W^\mu] = 0$ from Section A.8.2, both terms in Equation (A37) become zero:

$$\mathbb{E}_W[\Delta p_1] = 0. \tag{A38}$$

Thus, the expected gradient of $\mathcal{L}^\mu$ with respect to p_1 is zero when $p_1 = 0$ and $q_2 = 0$.

A.10.2 Gradient with respect to q_2

Now, we compute the gradient of $\mathcal{L}^\mu$ with respect to q_2. Using the chain rule again, we need to compute $\frac{\partial \hat{y}^\mu}{\partial q_2}$. From Equation (A24), $\hat{y}^\mu = (p_1 + p_2 W^\mu) A^\mu$, and from Equation (A23), $A^\mu = \hat{\Sigma}_x (q_1 + (W^\mu)^\top q_2) x_q$. Therefore,

$$\frac{\partial \hat{y}^\mu}{\partial q_2} = \frac{\partial}{\partial q_2} \left[(p_1 + p_2 W^\mu) \hat{\Sigma}_x (q_1 + (W^\mu)^\top q_2) x_q \right] \tag{A39}$$

$$= (p_1 + p_2 W^\mu) \hat{\Sigma}_x \frac{\partial}{\partial q_2} \left[(q_1 + (W^\mu)^\top q_2) x_q \right] \tag{A40}$$

$$= (p_1 + p_2 W^\mu) \hat{\Sigma}_x \frac{\partial}{\partial q_2} \left[q_1 x_q + (W^\mu)^\top q_2 x_q \right] \tag{A41}$$

$$= (p_1 + p_2 W^\mu) \hat{\Sigma}_x (W^\mu)^\top x_q. \tag{A42}$$

The gradient of the loss $\mathcal{L}^\mu$ with respect to q_2 is then:

$$\frac{\partial \mathcal{L}^\mu}{\partial q_2} = \left(\frac{\partial \hat{y}^\mu}{\partial q_2} \right)^\top (\hat{y}^\mu - y_q^\mu) \tag{A43}$$

$$= \left((p_1 + p_2 W^\mu) \hat{\Sigma}_x (W^\mu)^\top x_q \right)^\top \left((p_1 + p_2 W^\mu) \hat{\Sigma}_x (q_1 + (W^\mu)^\top q_2) x_q - W^\mu x_q \right) \tag{A44}$$

$$= x_q^\top (W^\mu) \hat{\Sigma}_x^\top (p_1 + p_2 W^\mu)^\top \left((p_1 + p_2 W^\mu) \hat{\Sigma}_x (q_1 + (W^\mu)^\top q_2) x_q - W^\mu x_q \right) \tag{A45}$$

$$= x_q^\top W^\mu \hat{\Sigma}_x (p_1^\top + (p_2 W^\mu)^\top) \left((p_1 + p_2 W^\mu) \hat{\Sigma}_x (q_1 + (W^\mu)^\top q_2) x_q - W^\mu x_q \right), \tag{A46}$$

assuming $\hat{\Sigma}_x$ and p_1, p_2, W^μ are matrices such that transpose operation distributes linearly. The change in q_2 after one SGD update step, summing over all P tasks, is:

$$\Delta q_2 = -\lambda \sum_{\mu=1}^{P} x_q^\top W^\mu \hat{\Sigma}_x (p_1^\top + (p_2 W^\mu)^\top) \left((p_1 + p_2 W^\mu) \hat{\Sigma}_x (q_1 + (W^\mu)^\top q_2) x_q - W^\mu x_q \right). \tag{A47}$$

Substituting $p_1 = 0$ and $q_2 = 0$ into Equation (A47), we get:

$$\Delta q_2 = -\lambda \sum_{\mu=1}^{P} x_q^\top W^\mu \hat{\Sigma}_x (0^\top + (p_2 W^\mu)^\top) \left((0 + p_2 W^\mu) \hat{\Sigma}_x (q_1 + (W^\mu)^\top 0) x_q - W^\mu x_q \right). \tag{A48}$$

Simplifying this expression:

$$\Delta q_2 = -\lambda \sum_{\mu=1}^{P} x_q^\top W^\mu \hat{\Sigma}_x (p_2 W^\mu)^\top \left(p_2 W^\mu \hat{\Sigma}_x q_1 x_q - W^\mu x_q \right) \tag{A49}$$

$$= -\lambda \sum_{\mu=1}^{P} x_q^\top W^\mu \hat{\Sigma}_x (W^\mu)^\top p_2^\top \left(p_2 W^\mu \hat{\Sigma}_x q_1 x_q - W^\mu x_q \right) \tag{A50}$$

$$= -\lambda \sum_{\mu=1}^{P} \left(x_q^\top W^\mu \hat{\Sigma}_x (W^\mu)^\top p_2^\top p_2 W^\mu \hat{\Sigma}_x q_1 x_q - x_q^\top W^\mu \hat{\Sigma}_x (W^\mu)^\top p_2^\top W^\mu x_q \right) \tag{A51}$$

$$= -\lambda \sum_{\mu=1}^{P} x_q^\top W^\mu \hat{\Sigma}_x (W^\mu)^\top p_2^\top p_2 W^\mu \hat{\Sigma}_x q_1 x_q + \lambda \sum_{\mu=1}^{P} x_q^\top W^\mu \hat{\Sigma}_x (W^\mu)^\top p_2^\top W^\mu x_q. \tag{A52}$$

Taking the expectation of Δq_2 with respect to W:

$$\mathbb{E}_W[\Delta q_2] = -\lambda \sum_{\mu=1}^{P} x_q^\top \mathbb{E}_W[W^\mu \hat{\Sigma}_x (W^\mu)^\top p_2^\top p_2 W^\mu] \hat{\Sigma}_x q_1 x_q + \lambda \sum_{\mu=1}^{P} x_q^\top \mathbb{E}_W[W^\mu \hat{\Sigma}_x (W^\mu)^\top p_2^\top W^\mu] x_q. \tag{A53}$$

Consider the term $\mathbb{E}_W[W^\mu \hat{\Sigma}_x (W^\mu)^\top p_2^\top p_2 W^\mu]$. Since W^μ has zero mean and its elements are sampled from a Gaussian distribution, any term involving the expectation of an odd number of elements of W^μ will be zero. The term $W^\mu \hat{\Sigma}_x (W^\mu)^\top p_2^\top p_2 W^\mu$ involves a product of three instances of W^μ. While $\hat{\Sigma}_x$ and p_2 are constant with respect to the expectation over W^μ, the expectation $\mathbb{E}_W[W^\mu \hat{\Sigma}_x (W^\mu)^\top p_2^\top p_2 W^\mu]$ is a tensor of order 4, and each component of this tensor is an expectation of a product of three elements from W^μ. For a centered Gaussian distribution, all odd moments are zero. Therefore, $\mathbb{E}_W[W^\mu \hat{\Sigma}_x (W^\mu)^\top p_2^\top p_2 W^\mu] = 0$. Similarly, $\mathbb{E}_W[W^\mu \hat{\Sigma}_x (W^\mu)^\top p_2^\top W^\mu] = 0$. Thus,

$$\mathbb{E}_W[\Delta q_2] = 0. \tag{A54}$$

Hence, the expected gradient of $\mathcal{L}^\mu$ with respect to q_2 is also zero when $p_1 = 0$ and $q_2 = 0$.

A.10.3 Simplified Predictor with $p_1 = 0$ and $q_2 = 0$

Given that the expected gradients for p_1 and q_2 are zero at initialization and remain zero in expectation under SGD, we can fix $p_1 = 0$ and $q_2 = 0$ throughout the training. With these null parameters, the predictor from Equation (A19) simplifies to:

$$\hat{y}_\mu = \Big((0 + p_2 W^\mu)\hat{\Sigma}_x (q_1 + (W^\mu)^\top 0)\Big) x_q = \Big(p_2 W^\mu \hat{\Sigma}_x q_1\Big) x_q \in \mathbb{R}^d. \tag{A55}$$

This simplified predictor, which depends only on the parameters p_2 and q_1, will be the focus of our subsequent analysis of the learning dynamics.

A.11 Learning Dynamics of Parameters p_2 and q_1

In this section, we derive the stochastic gradient descent (SGD) update rules for the parameters p_2 and q_1, and then transition to continuous-time dynamics to analyze the learning behavior. We will utilize the simplified predictor $\hat{y}_\mu = (p_2 W^\mu \hat{\Sigma}_x q_1) x_q$ and the loss function $\mathcal{L}^\mu = \frac{1}{2}\|p_2 W^\mu \hat{\Sigma}_x q_1 x_q - W^\mu x_q\|_2^2$, which are valid under the initial conditions $p_1 = 0$ and $q_2 = 0$ maintained throughout training as justified in Section A.10.

A.11.1 SGD Update for p_2

The gradient of the loss function $\mathcal{L}^\mu$ with respect to p_2 for a given task μ is derived as follows. Let $h(p_2) = p_2 W^\mu \hat{\Sigma}_x q_1 x_q - W^\mu x_q$. Then $\mathcal{L}^\mu = \frac{1}{2}\|h(p_2)\|_2^2 = \frac{1}{2} h(p_2)^\top h(p_2)$. The gradient is given by:

$$\nabla_{p_2} \mathcal{L}^\mu = \left(\frac{\partial h(p_2)}{\partial p_2}\right)^\top h(p_2). \tag{A56}$$

We compute the derivative of $h(p_2)$ with respect to p_2:

$$\frac{\partial h(p_2)}{\partial p_2} = \frac{\partial}{\partial p_2}\Big[p_2 W^\mu \hat{\Sigma}_x q_1 x_q - W^\mu x_q\Big] = \frac{\partial}{\partial p_2}\Big[p_2 (W^\mu \hat{\Sigma}_x q_1 x_q) - W^\mu x_q\Big] = x_q^\top q_1^\top \hat{\Sigma}_x (W^\mu)^\top, \tag{A57}$$

since $\hat{\Sigma}_x$ is symmetric. Substituting this into Equation (A56) and using $h(p_2) = p_2 W^\mu \hat{\Sigma}_x q_1 x_q - W^\mu x_q$, we obtain the gradient:

$$\nabla_{p_2} \mathcal{L}^\mu = \Big(x_q^\top q_1^\top \hat{\Sigma}_x (W^\mu)^\top\Big)^\top \Big(p_2 W^\mu \hat{\Sigma}_x q_1 x_q - W^\mu x_q\Big) \tag{A58}$$

$$= (W^\mu \hat{\Sigma}_x q_1 x_q - W^\mu x_q)\Big(x_q^\top q_1^\top \hat{\Sigma}_x (W^\mu)^\top\Big) \tag{A59}$$

$$= \Big(p_2 W^\mu \hat{\Sigma}_x q_1 x_q - W^\mu x_q\Big) x_q^\top q_1^\top \hat{\Sigma}_x (W^\mu)^\top. \tag{A60}$$

The SGD update rule for p_2 for a given task μ is thus:

$$\Delta p_2(\mu) = -\lambda \nabla_{p_2} \mathcal{L}_\mu \tag{A61}$$

$$= -\lambda \Big(p_2 W^\mu \hat{\Sigma}_x q_1 x_q - W^\mu x_q\Big) x_q^\top q_1^\top \hat{\Sigma}_x (W^\mu)^\top. \tag{A62}$$

Summing over all tasks $\mu = 1, \ldots, P$ for one epoch update, we get:

$$\Delta p_2 = \sum_{\mu=1}^{P} \Delta p_2(\mu) \tag{A63}$$

$$= -\lambda \sum_{\mu=1}^{P} \left(p_2 W^\mu \hat{\Sigma}_x q_1 x_q - W^\mu x_q \right) x_q^\top q_1^\top \hat{\Sigma}_x (W^\mu)^\top \tag{A64}$$

$$= -\lambda \sum_{\mu=1}^{P} \left(p_2 W^\mu \hat{\Sigma}_x q_1 x_q x_q^\top q_1^\top \hat{\Sigma}_x (W^\mu)^\top - W^\mu x_q x_q^\top q_1^\top \hat{\Sigma}_x (W^\mu)^\top \right). \tag{A65}$$

A.11.2 Change of Variables and Assumptions for Tractable Dynamics

To simplify the analysis and derive tractable learning dynamics, we introduce a change of variables and make further assumptions. We consider the singular value decomposition of the population covariance matrix Σ_x and the empirical covariance matrix $\hat{\Sigma}_x$ of the input data $\{x_i\}_{i=1}^N$:

$$\Sigma_x = USU^\top, \tag{A66}$$

$$\hat{\Sigma}_x = U\hat{S}U^\top, \tag{A67}$$

where $U \in \mathbb{R}^{d\times d}$ is an orthogonal matrix, $S = \text{diag}(s_1, \ldots, s_d) \in \mathbb{R}^{d\times d}$ is a diagonal matrix of singular values of Σ_x, and $\hat{S} = \text{diag}(\hat{s}_1, \ldots, \hat{s}_d) \in \mathbb{R}^{d\times d}$ is a diagonal matrix related to the empirical singular values. We further assume that the orthogonal matrix V used in the generation of the transformation matrices $W^\mu = V\Lambda^\mu V^\top$ (Equation (A7)) is the same as U, i.e., $V = U$. Thus, we have $W^\mu = U\Lambda^\mu U^\top$.

We also assume that the parameters p_2 and q_1 are initialized and maintain the following form throughout training:

$$p_2 = U\bar{p}_2 U^\top, \tag{A68}$$

$$q_1 = U\bar{q}_1 U^\top, \tag{A69}$$

where $\bar{p}_2$ and $\bar{q}_1$ are diagonal matrices.

Substituting these decompositions into the update rule for Δp_2 (Equation (A65)), and noting that $(W^\mu)^\top = (U\Lambda^\mu U^\top)^\top = U(\Lambda^\mu)^\top U^\top = U\Lambda^\mu U^\top$ since Λ^μ is a diagonal matrix, and similarly $\hat{\Sigma}_x^\top = \hat{\Sigma}_x$, we have:

$$\begin{aligned}\Delta p_2 = -\lambda \sum_{\mu=1}^{P} \Big(& (U\bar{p}_2U^\top)(U\Lambda^\mu U^\top)(U\hat{S}U^\top)(U\bar{q}_1U^\top)x_q x_q^\top (U\bar{q}_1U^\top)^\top (U\hat{S}U^\top)(U\Lambda^\mu U^\top) \\ & -(U\Lambda^\mu U^\top)x_q x_q^\top (U\bar{q}_1U^\top)^\top (U\hat{S}U^\top)(U\Lambda^\mu U^\top)\Big) \\ = -\lambda \sum_{\mu=1}^{P} & \left(U\bar{p}_2\Lambda^\mu \hat{S}\bar{q}_1 U^\top x_q x_q^\top U\bar{q}_1\hat{S}\Lambda^\mu U^\top - U\Lambda^\mu U^\top x_q x_q^\top U\bar{q}_1\hat{S}\Lambda^\mu U^\top \right),\end{aligned}$$

using the orthogonality of U, i.e., $U^\top U = UU^\top = I$. We know that the change in p_2 can also be decomposed as $\Delta p_2 = p_2(t+1) - p_2(t) = U\bar{p}_2(t+1)U^\top - U\bar{p}_2(t)U^\top = U(\bar{p}_2(t+1) - \bar{p}_2(t))U^\top = U\Delta\bar{p}_2U^\top$. Thus, $U\Delta\bar{p}_2U^\top$ is equal to the expression above. Multiplying by $U^\top$ from the left and U from the right, and using orthogonality of U, we get:

$$\begin{aligned}\Delta\bar{p}_2 &= -\lambda \sum_{\mu=1}^{P} U^\top \left(U\bar{p}_2\Lambda^\mu \hat{S}\bar{q}_1 U^\top x_q x_q^\top U\bar{q}_1\hat{S}\Lambda^\mu U^\top - U\Lambda^\mu U^\top x_q x_q^\top U\bar{q}_1\hat{S}\Lambda^\mu U^\top \right) U \\ &= -\lambda \sum_{\mu=1}^{P} \left(\bar{p}_2\Lambda^\mu \hat{S}\bar{q}_1 U^\top x_q x_q^\top U\bar{q}_1\hat{S}\Lambda^\mu - \Lambda^\mu U^\top x_q x_q^\top U\bar{q}_1\hat{S}\Lambda^\mu \right).\end{aligned}$$

Let $\bar{x}_q = U^\top x_q$. Then $x_q = U\bar{x}_q$ and $U^\top x_q x_q^\top U = U^\top (U\bar{x}_q)(U\bar{x}_q)^\top U = U^\top U \bar{x}_q \bar{x}_q^\top U^\top U = \bar{x}_q \bar{x}_q^\top$. Thus,

$$\Delta\bar{p}_2 = -\lambda \sum_{\mu=1}^{P} \left(\bar{p}_2 \Lambda^\mu \hat{S} \bar{q}_1 \bar{x}_q \bar{x}_q^\top \bar{q}_1 \hat{S} \Lambda^\mu - \Lambda^\mu \bar{x}_q \bar{x}_q^\top \bar{q}_1 \hat{S} \Lambda^\mu \right) \tag{A70}$$

$$= -\lambda \sum_{\mu=1}^{P} \left(\bar{p}_2 \Lambda^\mu \hat{S}^2 \bar{q}_1^2 \bar{x}_q \bar{x}_q^\top \Lambda^\mu - \Lambda^{\mu 2} \bar{q}_1 \bar{x}_q \bar{x}_q^\top \hat{S} \right), \tag{A71}$$

since $\hat{S}$ and $\bar{q}_1$ are diagonal matrices and commute.

Replacing the summation over tasks with expectation over Λ, we get the expected update for $\bar{p}_2$:

$$\mathbb{E}_\Lambda[\Delta\bar{p}_2] = -\lambda P \mathbb{E}_\Lambda \left[\bar{p}_2 \Lambda \hat{S}^2 \bar{q}_1^2 \bar{x}_q \bar{x}_q^\top \Lambda - \Lambda^2 \bar{q}_1 \bar{x}_q \bar{x}_q^\top \hat{S} \right]. \tag{A72}$$

Since all matrices in the expression are diagonal, they commute. Thus, we can rewrite Equation (A72) as:

$$\mathbb{E}_\Lambda[\Delta\bar{p}_2] = -\lambda P \left(\bar{p}_2 \hat{S}^2 \bar{q}_1^2 \bar{x}_q \bar{x}_q^\top \mathbb{E}_\Lambda[\Lambda^2] - \mathbb{E}_\Lambda[\Lambda^2] \bar{q}_1 \bar{x}_q \bar{x}_q^\top \hat{S} \right). \tag{A73}$$

Given that the eigenvalues in Λ are sampled from a standard multivariate Gaussian distribution, $\mathbb{E}[\Lambda^2] = I$. Substituting this into Equation (A73):

$$\mathbb{E}_\Lambda[\Delta\bar{p}_2] = -\lambda P \left(\bar{p}_2 \hat{S}^2 \bar{q}_1^2 \bar{x}_q \bar{x}_q^\top - \bar{q}_1 \bar{x}_q \bar{x}_q^\top \hat{S} \right). \tag{A74}$$

Now, taking the expectation over $\bar{x}_q \bar{x}_q^\top$. Since $x_q \sim \mathcal{N}(0, \Sigma_x)$ and $\bar{x}_q = U^\top x_q$, we have $\bar{x}_q \sim \mathcal{N}(0, U^\top \Sigma_x U) = \mathcal{N}(0, S)$. The expectation of $\bar{x}_q \bar{x}_q^\top$ is $\mathbb{E}[\bar{x}_q \bar{x}_q^\top] = \text{Cov}(\bar{x}_q) = S$. Thus,

$$\mathbb{E}_{\Lambda, x_q}[\Delta\bar{p}_2] = -\lambda P \left(\bar{p}_2 \hat{S}^2 \bar{q}_1^2 S - \bar{q}_1 S \hat{S} \right) = -\lambda P S \left(\bar{p}_2 \hat{S}^2 \bar{q}_1^2 - \bar{q}_1 \hat{S} \right). \tag{A75}$$

Finally, we take the expectation over the empirical covariance matrix $\hat{S}$. We know that $N\hat{\Sigma}_x = \sum_{i=1}^N x_i x_i^\top$. If $x_i \sim \mathcal{N}(0, \Sigma_x)$, then $N\hat{\Sigma}_x$ follows a Wishart distribution $W(N, \Sigma_x)$. In our diagonalized basis, $N\hat{S} = U^\top (N\hat{\Sigma}_x) U$ also behaves like a Wishart matrix related to S. For large N, $\hat{S} \approx S$. For expectation, we use the properties of Wishart distribution: $\mathbb{E}[\hat{S}] = S$ and $\mathbb{E}[\hat{S}^2] = \frac{N+1}{N} S^2 + \frac{\text{Tr}(S) S}{N}$. Thus, taking expectation over $\hat{S}$:

$$\mathbb{E}[\Delta\bar{p}_2] = -\lambda P S \left(\bar{p}_2 \bar{q}_1^2 \mathbb{E}[\hat{S}^2] - \bar{q}_1 \mathbb{E}[\hat{S}] \right) \tag{A76}$$

$$= -\lambda P S \left(\bar{p}_2 \bar{q}_1^2 \left(\frac{N+1}{N} S^2 + \frac{\text{Tr}(S) S}{N} \right) - \bar{q}_1 S \right) \tag{A77}$$

$$= -\lambda P S^2 \left(\bar{p}_2 \bar{q}_1^2 \left(\frac{N+1}{N} S + \frac{\text{Tr}(S)}{N} I \right) - \bar{q}_1 I \right). \tag{A78}$$

Due to the symmetry between p_2 and q_1 in the predictor structure and loss function, the derivation for the expected update of $\bar{q}_1$, $\mathbb{E}[\Delta\bar{q}_1]$, will have a similar form with the roles of $\bar{p}_2$ and $\bar{q}_1$ interchanged. Therefore, by symmetry, we can write:

$$\mathbb{E}[\Delta\bar{q}_1] = -\lambda P S^2 \left(\bar{q}_1 \bar{p}_2^2 \left(\frac{N+1}{N} S + \frac{\text{Tr}(S)}{N} I \right) - \bar{p}_2 I \right). \tag{A79}$$

A.11.3 Continuous-Time Dynamics

To analyze the learning dynamics in continuous time, we replace the discrete updates $\mathbb{E}[\Delta\bar{p}_2]$ and $\mathbb{E}[\Delta\bar{q}_1]$ with their continuous-time counterparts $\frac{d\bar{p}_2}{dt}$ and $\frac{d\bar{q}_1}{dt}$, respectively. This leads to the following system of ordinary differential equations describing the learning dynamics:

$$\frac{d\bar{p}_2}{dt} = -\lambda P S^2 \left(\bar{p}_2 \bar{q}_1^2 \left(\frac{N+1}{N} S + \frac{\text{Tr}(S)}{N} I \right) - \bar{q}_1 I \right), \tag{A80}$$

$$\frac{d\bar{q}_1}{dt} = -\lambda P S^2 \left(\bar{q}_1 \bar{p}_2^2 \left(\frac{N+1}{N} S + \frac{\text{Tr}(S)}{N} I \right) - \bar{p}_2 I \right). \tag{A81}$$

Since $\bar{p}_2$, $\bar{q}_1$, and S are diagonal matrices, these matrix equations decouple into d independent systems of differential equations for the diagonal elements. Let s_α be the α-th diagonal element of S, p_α be the α-th diagonal element of $\bar{p}_2$, and q_α be the α-th diagonal element of $\bar{q}_1$. Then, for each dimension $\alpha = 1, \ldots, d$, we have the dynamical equations:

$$\frac{dp_\alpha}{dt} = -\lambda P s_\alpha^2 \left(p_\alpha q_\alpha^2 \left(\frac{N+1}{N} s_\alpha + \frac{\mathrm{Tr}(S)}{N} \right) - q_\alpha \right), \tag{A82}$$

$$\frac{dq_\alpha}{dt} = -\lambda P s_\alpha^2 \left(q_\alpha p_\alpha^2 \left(\frac{N+1}{N} s_\alpha + \frac{\mathrm{Tr}(S)}{N} \right) - p_\alpha \right). \tag{A83}$$

These equations describe the continuous-time learning dynamics of the parameters p_α and q_α for each dimension α.

A.12 Dynamical Equations and Conserved Quantity

In this section, we analyze the continuous-time learning dynamics derived in Equations (A80) and (A81) and identify a conserved quantity associated with these dynamics. We then simplify these equations further by considering the diagonal elements and solve for the learning trajectory.

A.12.1 Conserved Quantity

An important observation regarding the learning dynamics is the existence of a conserved quantity, which remains constant throughout the training process. Let us define a quantity $\mathcal{C}$ as the difference of squared Frobenius norms of $\bar{p}_2$ and $\bar{q}_1$:

$$\mathcal{C} = \|\bar{p}_2\|_F^2 - \|\bar{q}_1\|_F^2 = \mathrm{Tr}(\bar{p}_2^\top \bar{p}_2) - \mathrm{Tr}(\bar{q}_1^\top \bar{q}_1). \tag{A84}$$

Since $\bar{p}_2$ and $\bar{q}_1$ are diagonal matrices, $\|\bar{p}_2\|_F^2 = \sum_\alpha p_\alpha^2$ and $\|\bar{q}_1\|_F^2 = \sum_\alpha q_\alpha^2$, where p_α and q_α are the diagonal elements. Thus, $\mathcal{C} = \sum_\alpha (p_\alpha^2 - q_\alpha^2)$.

To show that $\mathcal{C}$ is conserved, we compute its time derivative $\frac{d\mathcal{C}}{dt}$ using the chain rule and the dynamical equations for $\frac{d\bar{p}_2}{dt}$ and $\frac{d\bar{q}_1}{dt}$ (Equations (A80) and (A81)):

$$\begin{aligned}
\frac{d\mathcal{C}}{dt} &= \frac{d}{dt} \left(\|\bar{p}_2\|_F^2 - \|\bar{q}_1\|_F^2 \right) \\
&= \frac{d}{dt} \mathrm{Tr}(\bar{p}_2^\top \bar{p}_2) - \frac{d}{dt} \mathrm{Tr}(\bar{q}_1^\top \bar{q}_1) \\
&= 2\mathrm{Tr} \left(\bar{p}_2^\top \frac{d\bar{p}_2}{dt} \right) - 2\mathrm{Tr} \left(\bar{q}_1^\top \frac{d\bar{q}_1}{dt} \right) \\
&= 2\mathrm{Tr} \left(\bar{p}_2 \frac{d\bar{p}_2}{dt} \right) - 2\mathrm{Tr} \left(\bar{q}_1 \frac{d\bar{q}_1}{dt} \right),
\end{aligned}$$

since for diagonal matrices $\bar{p}_2^\top = \bar{p}_2$ and $\bar{q}_1^\top = \bar{q}_1$. Substituting the expressions for $\frac{d\bar{p}_2}{dt}$ and $\frac{d\bar{q}_1}{dt}$ from Equations (A80) and (A81):

$$\begin{aligned}
\frac{d\mathcal{C}}{dt} &= 2\mathrm{Tr} \left[\bar{p}_2 \left(-\lambda P S^2 \left(\bar{p}_2 \bar{q}_1^2 \left(\frac{N+1}{N} S + \frac{\mathrm{Tr}(S)}{N} I \right) - \bar{q}_1 I \right) \right) \right] \\
&\quad - 2\mathrm{Tr} \left[\bar{q}_1 \left(-\lambda P S^2 \left(\bar{q}_1 \bar{p}_2^2 \left(\frac{N+1}{N} S + \frac{\mathrm{Tr}(S)}{N} I \right) - \bar{p}_2 I \right) \right) \right] \\
&= -2\lambda P \mathrm{Tr} \left[S^2 \bar{p}_2 \left(\bar{p}_2 \bar{q}_1^2 \left(\frac{N+1}{N} S + \frac{\mathrm{Tr}(S)}{N} I \right) - \bar{q}_1 I \right) \right] \\
&\quad + 2\lambda P \mathrm{Tr} \left[S^2 \bar{q}_1 \left(\bar{q}_1 \bar{p}_2^2 \left(\frac{N+1}{N} S + \frac{\mathrm{Tr}(S)}{N} I \right) - \bar{p}_2 I \right) \right] \\
&= -2\lambda P \mathrm{Tr} \left[S^2 \bar{p}_2^2 \bar{q}_1^2 \left(\frac{N+1}{N} S + \frac{\mathrm{Tr}(S)}{N} I \right) - S^2 \bar{p}_2 \bar{q}_1 I \right] \\
&\quad + 2\lambda P \mathrm{Tr} \left[S^2 \bar{q}_1^2 \bar{p}_2^2 \left(\frac{N+1}{N} S + \frac{\mathrm{Tr}(S)}{N} I \right) - S^2 \bar{q}_1 \bar{p}_2 I \right].
\end{aligned}$$

Since matrix trace is linear and cyclic, and all matrices here are diagonal and thus commute, we have $\mathrm{Tr}[S^2\bar{p}_2^2\bar{q}_1^2(\cdot)] = \mathrm{Tr}[S^2\bar{q}_1^2\bar{p}_2^2(\cdot)]$ and $\mathrm{Tr}[S^2\bar{p}_2\bar{q}_1 I] = \mathrm{Tr}[S^2\bar{q}_1\bar{p}_2 I]$. Therefore, the terms cancel out exactly:

$$\frac{d\mathcal{C}}{dt} = 0.$$

This shows that $\mathcal{C} = \|\bar{p}_2\|_F^2 - \|\bar{q}_1\|_F^2$ is indeed a conserved quantity of the learning dynamics.

A.12.2 Diagonal Element Dynamics and Solution

As noted earlier, the matrix dynamical equations decouple into d independent systems for the diagonal elements. For each dimension $\alpha = 1, \ldots, d$, the dynamical equations are given by Equations (A82) and (A83):

$$\frac{dp_\alpha}{dt} = -\lambda P s_\alpha^2 \left(p_\alpha q_\alpha^2 \left(\frac{N+1}{N} s_\alpha + \frac{\mathrm{Tr}(S)}{N} \right) - q_\alpha \right), \tag{A85}$$

$$\frac{dq_\alpha}{dt} = -\lambda P s_\alpha^2 \left(q_\alpha p_\alpha^2 \left(\frac{N+1}{N} s_\alpha + \frac{\mathrm{Tr}(S)}{N} \right) - p_\alpha \right). \tag{A86}$$

Let us define a term $s_\alpha^\infty = \left(\frac{N+1}{N} s_\alpha + \frac{\mathrm{Tr}(S)}{N} \right)$. Then the dynamical equations can be rewritten as:

$$\frac{dp_\alpha}{dt} = -\lambda P s_\alpha^2 \left(p_\alpha q_\alpha^2 s_\alpha^\infty - q_\alpha \right) = \lambda P s_\alpha^2 q_\alpha \left(1 - p_\alpha q_\alpha s_\alpha^\infty \right), \tag{A87}$$

$$\frac{dq_\alpha}{dt} = -\lambda P s_\alpha^2 \left(q_\alpha p_\alpha^2 s_\alpha^\infty - p_\alpha \right) = \lambda P s_\alpha^2 p_\alpha \left(1 - p_\alpha q_\alpha s_\alpha^\infty \right). \tag{A88}$$

We define an effective timescale for learning in each dimension α as $\tau_\alpha = \frac{1}{\lambda P s_\alpha^2}$. Then, the dynamics become:

$$\tau_\alpha \frac{dp_\alpha}{dt} = q_\alpha \left(1 - p_\alpha q_\alpha s_\alpha^\infty \right), \tag{A89}$$

$$\tau_\alpha \frac{dq_\alpha}{dt} = p_\alpha \left(1 - p_\alpha q_\alpha s_\alpha^\infty \right). \tag{A90}$$

Observe that $\tau_\alpha p_\alpha \frac{dp_\alpha}{dt} = p_\alpha q_\alpha \left(1 - p_\alpha q_\alpha s_\alpha^\infty \right)$ and $\tau_\alpha q_\alpha \frac{dq_\alpha}{dt} = p_\alpha q_\alpha \left(1 - p_\alpha q_\alpha s_\alpha^\infty \right)$. Thus, $\tau_\alpha p_\alpha \frac{dp_\alpha}{dt} = \tau_\alpha q_\alpha \frac{dq_\alpha}{dt}$, which implies $p_\alpha \frac{dp_\alpha}{dt} = q_\alpha \frac{dq_\alpha}{dt}$, or $p_\alpha dp_\alpha = q_\alpha dq_\alpha$. Integrating both sides, we get $\int p_\alpha dp_\alpha = \int q_\alpha dq_\alpha$, leading to $\frac{1}{2}p_\alpha^2 = \frac{1}{2}q_\alpha^2 + C'_\alpha$, or $p_\alpha^2 - q_\alpha^2 = C_\alpha$, where $C_\alpha = 2C'_\alpha$ is a constant of integration. This is the element-wise conserved quantity, consistent with the matrix conserved quantity $\mathcal{C}$. If initialized with $p_\alpha(0) = q_\alpha(0)$, then $C_\alpha = 0$, and $p_\alpha^2(t) = q_\alpha^2(t)$ for all t. Assuming $p_\alpha(t)$ and $q_\alpha(t)$ maintain the same sign (if initialized non-negative, remain non-negative), we can consider $p_\alpha(t) = q_\alpha(t)$.

Let $a_\alpha = p_\alpha q_\alpha = p_\alpha^2 = q_\alpha^2$ (under the assumption $p_\alpha = q_\alpha$). Then, the dynamics for a_α can be derived as:

$$\begin{aligned}
\tau_\alpha \frac{da_\alpha}{dt} &= \tau_\alpha \frac{d}{dt}(p_\alpha q_\alpha) = \tau_\alpha \left(\frac{dp_\alpha}{dt} q_\alpha + p_\alpha \frac{dq_\alpha}{dt} \right) \\
&= q_\alpha \left(q_\alpha \left(1 - p_\alpha q_\alpha s_\alpha^\infty \right) \right) + p_\alpha \left(p_\alpha \left(1 - p_\alpha q_\alpha s_\alpha^\infty \right) \right) \\
&= q_\alpha^2 \left(1 - a_\alpha s_\alpha^\infty \right) + p_\alpha^2 \left(1 - a_\alpha s_\alpha^\infty \right) \\
&= \left(p_\alpha^2 + q_\alpha^2 \right) \left(1 - a_\alpha s_\alpha^\infty \right) = 2a_\alpha \left(1 - a_\alpha s_\alpha^\infty \right),
\end{aligned}$$

under the assumption $p_\alpha = q_\alpha$, so $p_\alpha^2 = q_\alpha^2 = a_\alpha$. The fixed point for a_α is obtained by setting $\frac{da_\alpha}{dt} = 0$, which gives $2a_\alpha \left(1 - a_\alpha s_\alpha^\infty \right) = 0$. Non-trivial fixed point is $a_\alpha^\infty = \frac{1}{s_\alpha^\infty} = \frac{1}{\frac{N+1}{N} s_\alpha + \frac{\mathrm{Tr}(S)}{N}}$.

To solve the differential equation $\tau_\alpha \frac{da_\alpha}{dt} = 2a_\alpha \left(1 - a_\alpha s_\alpha^\infty \right)$, we can separate variables:

$$\int \frac{da_\alpha}{a_\alpha \left(1 - a_\alpha s_\alpha^\infty \right)} = \int \frac{2}{\tau_\alpha} dt.$$

Using partial fraction decomposition, $\frac{1}{a_\alpha(1-a_\alpha s_\alpha^\infty)} = \frac{1}{a_\alpha} + \frac{s_\alpha^\infty}{1-a_\alpha s_\alpha^\infty} = \frac{1}{a_\alpha} - \frac{-s_\alpha^\infty}{1-a_\alpha s_\alpha^\infty}$. Thus,

$$\begin{aligned}
\int \left(\frac{1}{a_\alpha} + \frac{s_\alpha^\infty}{1 - a_\alpha s_\alpha^\infty} \right) da_\alpha &= \frac{2t}{\tau_\alpha} + C_\alpha'' \\
\log|a_\alpha| - \log|1 - a_\alpha s_\alpha^\infty| &= \frac{2t}{\tau_\alpha} + C_\alpha'' \\
\log\left| \frac{a_\alpha}{1 - a_\alpha s_\alpha^\infty} \right| &= \frac{2t}{\tau_\alpha} + C_\alpha'' \\
\frac{a_\alpha}{1 - a_\alpha s_\alpha^\infty} &= e^{\frac{2t}{\tau_\alpha} + C_\alpha''} = e^{C_\alpha''} e^{\frac{2t}{\tau_\alpha}} = C_\alpha''' e^{\frac{2t}{\tau_\alpha}}.
\end{aligned}$$

Let $a_\alpha(0) = a_\alpha^0$. Then $\frac{a_\alpha^0}{1-a_\alpha^0 s_\alpha^\infty} = C_\alpha'''$. Thus, $\frac{a_\alpha}{1-a_\alpha s_\alpha^\infty} = \frac{a_\alpha^0}{1-a_\alpha^0 s_\alpha^\infty} e^{\frac{2t}{\tau_\alpha}}$. Solving for $a_\alpha(t)$:

$$\begin{aligned}
a_\alpha(t) &= \frac{a_\alpha^0}{1 - a_\alpha^0 s_\alpha^\infty} e^{\frac{2t}{\tau_\alpha}} \left(1 - a_\alpha(t) s_\alpha^\infty\right) \\
a_\alpha(t) \left(1 + \frac{a_\alpha^0}{1 - a_\alpha^0 s_\alpha^\infty} e^{\frac{2t}{\tau_\alpha}} s_\alpha^\infty \right) &= \frac{a_\alpha^0}{1 - a_\alpha^0 s_\alpha^\infty} e^{\frac{2t}{\tau_\alpha}} \\
a_\alpha(t) &= \frac{\frac{a_\alpha^0}{1-a_\alpha^0 s_\alpha^\infty} e^{\frac{2t}{\tau_\alpha}}}{1 + \frac{a_\alpha^0 s_\alpha^\infty}{1-a_\alpha^0 s_\alpha^\infty} e^{\frac{2t}{\tau_\alpha}}} = \frac{a_\alpha^0 e^{\frac{2t}{\tau_\alpha}}}{(1 - a_\alpha^0 s_\alpha^\infty) + a_\alpha^0 s_\alpha^\infty e^{\frac{2t}{\tau_\alpha}}} \\
&= \frac{a_\alpha^0}{(1 - a_\alpha^0 s_\alpha^\infty) e^{-\frac{2t}{\tau_\alpha}} + a_\alpha^0 s_\alpha^\infty} = \frac{a_\alpha^0 / s_\alpha^\infty}{((1 - a_\alpha^0 s_\alpha^\infty)/s_\alpha^\infty) e^{-\frac{2t}{\tau_\alpha}} + a_\alpha^0} \\
&= \frac{a_\alpha^0 / s_\alpha^\infty}{(\frac{1}{s_\alpha^\infty} - a_\alpha^0) e^{-\frac{2t}{\tau_\alpha}} + a_\alpha^0} = \frac{a_\alpha^\infty a_\alpha^0}{(a_\alpha^\infty - a_\alpha^0) e^{-\frac{2t}{\tau_\alpha}} + a_\alpha^0} \\
&= a_\alpha^\infty \frac{a_\alpha^0}{a_\alpha^0 + (a_\alpha^\infty - a_\alpha^0) e^{-\frac{2t}{\tau_\alpha}}}.
\end{aligned}$$

Thus, the solution for $a_\alpha(t)$ is:

$$a_\alpha(t) = a_\alpha^\infty \left(\frac{a_\alpha^0}{a_\alpha^0 + (a_\alpha^\infty - a_\alpha^0) \exp\left(-\frac{2t}{\tau_\alpha}\right)} \right). \tag{A91}$$

For small initial values $a_\alpha^0 \approx \epsilon$ and approaching the fixed point $a_\alpha(t) \approx a_\alpha^\infty(1-\epsilon)$, the time t required to reach close to the fixed point can be approximated by solving for t in:

$$\begin{aligned}
\epsilon &\approx \left(a_\alpha^\infty - a_\alpha^0\right) \exp\left(-\frac{2t}{\tau_\alpha}\right) \\
\exp\left(\frac{2t}{\tau_\alpha}\right) &\approx \frac{a_\alpha^\infty - a_\alpha^0}{\epsilon} \approx \frac{a_\alpha^\infty}{\epsilon} = \frac{1}{s_\alpha^\infty \epsilon} \\
\frac{2t}{\tau_\alpha} &\approx \log\left(\frac{1}{s_\alpha^\infty \epsilon}\right) \\
t &\approx \frac{\tau_\alpha}{2} \log\left(\frac{1}{s_\alpha^\infty \epsilon}\right) = \frac{1}{2\lambda P s_\alpha^2} \log\left(\frac{1}{s_\alpha^\infty \epsilon}\right).
\end{aligned}$$

This gives an estimate of the convergence time for each dimension α.

A.13 Loss Analysis

In this section, we analyze the expected loss function of the linear transformer model. We first express the average loss per epoch and then simplify it using the change of variables and assumptions introduced in Section A.11. We will derive the expected loss in terms of the diagonal parameters $a_\alpha(t)$ and analyze the loss behavior as training progresses.

A.13.1 Expected Loss Formulation

The loss function for a single task μ is given by:

$$\mathcal{L}^\mu = \frac{1}{2}\|\hat{y}_\mu - y_q^\mu\|_2^2 = \frac{1}{2}\|p_2 W^\mu \hat{\Sigma}_x q_1 x_q - W^\mu x_q\|_2^2. \tag{A92}$$

The average loss per epoch over all P tasks is:

$$\mathcal{L} = \frac{1}{P}\sum_{\mu=1}^{P}\mathcal{L}^\mu = \frac{1}{2P}\sum_{\mu=1}^{P}\|p_2 W^\mu \hat{\Sigma}_x q_1 x_q - W^\mu x_q\|_2^2. \tag{A93}$$

Replacing the summation with expectation over the distribution of W and x_q, we obtain the expected loss:

$$\mathcal{L} = \frac{1}{2}\mathbb{E}_{W,x_q}\left[\|p_2 W \hat{\Sigma}_x q_1 x_q - W x_q\|_2^2\right]. \tag{A94}$$

Using the change of variables $p_2 = U\bar{p}_2 U^\top$, $q_1 = U\bar{q}_1 U^\top$, $W = U\Lambda U^\top$, $\hat{\Sigma}_x = U\hat{S}U^\top$, and $x_q = U\bar{x}_q$, the expected loss becomes:

$$\begin{aligned}
\mathcal{L} &= \frac{1}{2}\mathbb{E}_{\Lambda,\bar{x}_q}\left[\|(U\bar{p}_2U^\top)(U\Lambda U^\top)(U\hat{S}U^\top)(U\bar{q}_1U^\top)(U\bar{x}_q) - (U\Lambda U^\top)(U\bar{x}_q)\|_2^2\right] && \text{(A95)}\\
&= \frac{1}{2}\mathbb{E}_{\Lambda,\bar{x}_q}\left[\|U\bar{p}_2\Lambda\hat{S}\bar{q}_1\bar{x}_q - U\Lambda\bar{x}_q\|_2^2\right] && \text{(A96)}\\
&= \frac{1}{2}\mathbb{E}_{\Lambda,\bar{x}_q}\left[\|U(\bar{p}_2\Lambda\hat{S}\bar{q}_1\bar{x}_q - \Lambda\bar{x}_q)\|_2^2\right]. && \text{(A97)}
\end{aligned}$$

Since the Euclidean norm is invariant under orthogonal transformations, $\|Uv\|_2 = \|v\|_2$ for any vector v and orthogonal matrix U. Thus, Equation (A97) simplifies to:

$$\mathcal{L} = \frac{1}{2}\mathbb{E}_{\Lambda,\bar{x}_q}\left[\|\bar{p}_2\Lambda\hat{S}\bar{q}_1\bar{x}_q - \Lambda\bar{x}_q\|_2^2\right]. \tag{A98}$$

Under the assumption that $p_\alpha(t) = q_\alpha(t)$, we have $\bar{p}_2 = \bar{q}_1 = \bar{a}^{1/2}$, where $\bar{a} = \text{diag}(a_1, \ldots, a_d)$. Then $\bar{p}_2\bar{q}_1 = \bar{a}$. The expected loss becomes:

$$\mathcal{L} = \frac{1}{2}\mathbb{E}_{\Lambda,\bar{x}_q}\left[\|\bar{a}\Lambda\hat{S}\bar{x}_q - \Lambda\bar{x}_q\|_2^2\right]. \tag{A99}$$

A.13.2 Expansion and Term-wise Expectation

Expanding the squared norm in Equation (A99), we get:

$$\begin{aligned}
\mathcal{L} &= \frac{1}{2}\mathbb{E}_{\Lambda,\bar{x}_q}\left[(\bar{a}\Lambda\hat{S}\bar{x}_q - \Lambda\bar{x}_q)^\top(\bar{a}\Lambda\hat{S}\bar{x}_q - \Lambda\bar{x}_q)\right] && \text{(A100)}\\
&= \frac{1}{2}\mathbb{E}_{\Lambda,\bar{x}_q}\left[(\bar{x}_q^\top\hat{S}\Lambda\bar{a} - \bar{x}_q^\top\Lambda)(\bar{a}\Lambda\hat{S}\bar{x}_q - \Lambda\bar{x}_q)\right] && \text{(A101)}\\
&= \frac{1}{2}\mathbb{E}_{\Lambda,\bar{x}_q}\left[\bar{x}_q^\top\hat{S}\Lambda\bar{a}^2\Lambda\hat{S}\bar{x}_q - \bar{x}_q^\top\hat{S}\Lambda\bar{a}\Lambda\bar{x}_q - \bar{x}_q^\top\Lambda\bar{a}\Lambda\hat{S}\bar{x}_q + \bar{x}_q^\top\Lambda^2\bar{x}_q\right] && \text{(A102)}\\
&= \frac{1}{2}\mathbb{E}_{\Lambda,\bar{x}_q}\left[\bar{x}_q^\top\hat{S}\Lambda\bar{a}^2\Lambda\hat{S}\bar{x}_q - 2\bar{x}_q^\top\hat{S}\Lambda\bar{a}\Lambda\bar{x}_q + \bar{x}_q^\top\Lambda^2\bar{x}_q\right], && \text{(A103)}
\end{aligned}$$

where we used the fact that $\bar{x}_q^\top\hat{S}\Lambda\bar{a}\Lambda\bar{x}_q = (\bar{x}_q^\top\hat{S}\Lambda\bar{a}\Lambda\bar{x}_q)^\top = \bar{x}_q^\top\Lambda^\top\bar{a}^\top\Lambda^\top\hat{S}^\top\bar{x}_q = \bar{x}_q^\top\Lambda\bar{a}\Lambda\hat{S}\bar{x}_q$ since all matrices are diagonal and symmetric. We can analyze each term separately using trace operator and properties of expectation. Let $E_1 = \mathbb{E}[\bar{x}_q^\top\hat{S}\Lambda\bar{a}^2\Lambda\hat{S}\bar{x}_q]$, $E_2 = \mathbb{E}[\bar{x}_q^\top\hat{S}\Lambda\bar{a}\Lambda\bar{x}_q]$, and $E_3 = \mathbb{E}[\bar{x}_q^\top\Lambda^2\bar{x}_q]$. Then $\mathcal{L} = \frac{1}{2}(E_1 - 2E_2 + E_3)$.

For E_1:

$$\begin{aligned}
E_1 &= \mathbb{E}_{\Lambda,\bar{x}_q}[\bar{x}_q^\top\hat{S}\Lambda\bar{a}^2\Lambda\hat{S}\bar{x}_q] = \mathbb{E}_{\Lambda,\bar{x}_q}[\text{Tr}(\bar{x}_q^\top\hat{S}\Lambda\bar{a}^2\Lambda\hat{S}\bar{x}_q)] = \mathbb{E}_{\Lambda,\bar{x}_q}[\text{Tr}(\hat{S}\Lambda\bar{a}^2\Lambda\hat{S}\bar{x}_q\bar{x}_q^\top)]\\
&= \mathbb{E}_\Lambda[\text{Tr}(\hat{S}\Lambda\bar{a}^2\Lambda\hat{S}\mathbb{E}_{\bar{x}_q|\Lambda}[\bar{x}_q\bar{x}_q^\top])] = \mathbb{E}_\Lambda[\text{Tr}(\hat{S}\Lambda\bar{a}^2\Lambda\hat{S}S)],
\end{aligned}$$

since $\mathbb{E}_{\bar{x}_q}[\bar{x}_q\bar{x}_q^\top] = S$ and is independent of Λ. As trace is linear and cyclic, and matrices are diagonal,

$$E_1 = \text{Tr}(\mathbb{E}_\Lambda[\hat{S}\Lambda\bar{a}^2\Lambda\hat{S}]S) = \text{Tr}(\mathbb{E}_\Lambda[\Lambda^2]\hat{S}^2\bar{a}^2S) = \text{Tr}(I\mathbb{E}[\hat{S}^2]\bar{a}^2S) = \text{Tr}(\mathbb{E}[\hat{S}^2]\bar{a}^2S),$$

using $\mathbb{E}[\Lambda^2] = I$ and commutativity. Using $\mathbb{E}[\hat{S}^2] = \frac{N+1}{N}S^2 + \frac{\text{Tr}(S)}{N}S$,

$$E_1 = \text{Tr}\left(\left(\frac{N+1}{N}S^2 + \frac{\text{Tr}(S)}{N}S\right)\bar{a}^2 S\right). \tag{A104}$$

For E_2:

$$\begin{aligned}
E_2 &= \mathbb{E}_{\Lambda,\bar{x}_q}[\bar{x}_q^\top \hat{S}\Lambda\bar{a}\Lambda\bar{x}_q] = \mathbb{E}_{\Lambda,\bar{x}_q}[\text{Tr}(\bar{x}_q^\top \hat{S}\Lambda\bar{a}\Lambda\bar{x}_q)] = \mathbb{E}_{\Lambda,\bar{x}_q}[\text{Tr}(\hat{S}\Lambda\bar{a}\Lambda\bar{x}_q\bar{x}_q^\top)] \\
&= \mathbb{E}_\Lambda[\text{Tr}(\hat{S}\Lambda\bar{a}\Lambda\mathbb{E}_{\bar{x}_q|\Lambda}[\bar{x}_q\bar{x}_q^\top])] = \mathbb{E}_\Lambda[\text{Tr}(\hat{S}\Lambda\bar{a}\Lambda S)] = \text{Tr}(\mathbb{E}_\Lambda[\hat{S}\Lambda\bar{a}\Lambda]S) \\
&= \text{Tr}(\mathbb{E}_\Lambda[\Lambda^2]\hat{S}\bar{a}S) = \text{Tr}(I\mathbb{E}[\hat{S}]\bar{a}S) = \text{Tr}(\mathbb{E}[\hat{S}]\bar{a}S) = \text{Tr}(S\bar{a}S) = \text{Tr}(\bar{a}S^2),
\end{aligned}$$

using $\mathbb{E}[\Lambda^2] = I$ and $\mathbb{E}[\hat{S}] = S$.

$$E_2 = \text{Tr}(\bar{a}S^2). \tag{A105}$$

For E_3:

$$\begin{aligned}
E_3 &= \mathbb{E}_{\Lambda,\bar{x}_q}[\bar{x}_q^\top \Lambda^2\bar{x}_q] = \mathbb{E}_{\Lambda,\bar{x}_q}[\text{Tr}(\bar{x}_q^\top \Lambda^2\bar{x}_q)] = \mathbb{E}_{\Lambda,\bar{x}_q}[\text{Tr}(\Lambda^2\bar{x}_q\bar{x}_q^\top)] = \text{Tr}(\mathbb{E}_{\Lambda,\bar{x}_q}[\Lambda^2\bar{x}_q\bar{x}_q^\top]) \\
&= \text{Tr}(\mathbb{E}_\Lambda[\Lambda^2]\mathbb{E}_{\bar{x}_q}[\bar{x}_q\bar{x}_q^\top]) = \text{Tr}(I \cdot S) = \text{Tr}(S).
\end{aligned}$$

$$E_3 = \text{Tr}(S). \tag{A106}$$

Substituting Equations (A104), (A105), and (A106) into $\mathcal{L} = \frac{1}{2}(E_1 - 2E_2 + E_3)$:

$$\mathcal{L} = \frac{1}{2}\left(\text{Tr}\left(\left(\frac{N+1}{N}S^2 + \frac{\text{Tr}(S)}{N}S\right)\bar{a}^2 S\right) - 2\text{Tr}\left(\bar{a}S^2\right) + \text{Tr}(S)\right) \tag{A107}$$

$$= \frac{1}{2}\sum_\alpha\left(\left(\frac{N+1}{N}s_\alpha^2 + \frac{\text{Tr}(S)}{N}s_\alpha\right)a_\alpha^2 s_\alpha - 2a_\alpha s_\alpha^2 + s_\alpha\right) \tag{A108}$$

$$= \frac{1}{2}\sum_\alpha s_\alpha\left(a_\alpha^2 s_\alpha\left(\frac{N+1}{N}s_\alpha + \frac{\text{Tr}(S)}{N}\right) - 2a_\alpha s_\alpha + 1\right). \tag{A109}$$

Using $a_\alpha^\infty = \frac{1}{\frac{N+1}{N}s_\alpha + \frac{\text{Tr}(S)}{N}}$, or $\frac{1}{a_\alpha^\infty} = \frac{N+1}{N}s_\alpha + \frac{\text{Tr}(S)}{N}$, we can rewrite the loss as:

$$\mathcal{L} = \frac{1}{2}\sum_\alpha s_\alpha\left(\frac{s_\alpha}{a_\alpha^\infty}a_\alpha^2(t) - 2a_\alpha(t)s_\alpha + 1\right). \tag{A110}$$

A.13.3 Loss at Infinity and Initial Loss Behavior

As $t \to \infty$, $a_\alpha(t) \to a_\alpha^\infty$. The loss at infinite time (converged loss) is:

$$\mathcal{L}(\infty) = \frac{1}{2}\sum_\alpha s_\alpha\left(\frac{s_\alpha}{a_\alpha^\infty}(a_\alpha^\infty)^2 - 2a_\alpha^\infty s_\alpha + 1\right) = \frac{1}{2}\sum_\alpha s_\alpha\left(a_\alpha^\infty s_\alpha - 2a_\alpha^\infty s_\alpha + 1\right) \tag{A111}$$

$$= \frac{1}{2}\sum_\alpha s_\alpha\left(1 - a_\alpha^\infty s_\alpha\right) = \frac{1}{2}\sum_\alpha s_\alpha\left(1 - \frac{s_\alpha}{\frac{N+1}{N}s_\alpha + \frac{\text{Tr}(S)}{N}}\right) \tag{A112}$$

$$= \frac{1}{2}\sum_\alpha s_\alpha\left(\frac{\frac{N+1}{N}s_\alpha + \frac{\text{Tr}(S)}{N} - s_\alpha}{\frac{N+1}{N}s_\alpha + \frac{\text{Tr}(S)}{N}}\right) = \frac{1}{2}\sum_\alpha s_\alpha\left(\frac{\frac{1}{N}s_\alpha + \frac{\text{Tr}(S)}{N}}{\frac{N+1}{N}s_\alpha + \frac{\text{Tr}(S)}{N}}\right) \tag{A113}$$

$$= \frac{1}{2}\sum_\alpha \frac{s_\alpha(\frac{1}{N}s_\alpha + \frac{\text{Tr}(S)}{N})}{\frac{N+1}{N}s_\alpha + \frac{\text{Tr}(S)}{N}} = \frac{1}{2N}\sum_\alpha \frac{s_\alpha(s_\alpha + \text{Tr}(S))}{\frac{N+1}{N}s_\alpha + \frac{\text{Tr}(S)}{N}} = \frac{1}{2}\sum_\alpha \frac{s_\alpha(s_\alpha + \text{Tr}(S))}{(N+1)s_\alpha + \text{Tr}(S)}. \tag{A114}$$

To understand the initial behavior of the loss, we compute the time derivative of $\mathcal{L}$ at $t = 0$. Using chain rule and Equation (A109):

$$\begin{aligned}
\frac{d\mathcal{L}}{dt} &= \sum_\alpha \frac{\partial\mathcal{L}}{\partial a_\alpha}\frac{da_\alpha}{dt} = \sum_\alpha \frac{1}{2}s_\alpha\left(2a_\alpha s_\alpha\frac{1}{a_\alpha^\infty} - 2s_\alpha\right)\frac{da_\alpha}{dt} = \sum_\alpha s_\alpha\left(\frac{s_\alpha a_\alpha}{a_\alpha^\infty} - s_\alpha\right)\frac{da_\alpha}{dt} \\
&= \sum_\alpha s_\alpha^2\left(\frac{a_\alpha}{a_\alpha^\infty} - 1\right)\frac{da_\alpha}{dt}.
\end{aligned}$$

Substituting $\frac{da_\alpha}{dt} = \frac{2}{\tau_\alpha} a_\alpha (1 - a_\alpha s_\alpha^\infty)$ from Section A.12:

$$\begin{aligned}
\frac{d\mathcal{L}}{dt} &= \sum_\alpha s_\alpha^2 \left(\frac{a_\alpha}{a_\alpha^\infty} - 1 \right) \left(\frac{2}{\tau_\alpha} a_\alpha (1 - a_\alpha s_\alpha^\infty) \right) = \sum_\alpha \frac{2 s_\alpha^2}{\tau_\alpha} \left(\frac{a_\alpha}{a_\alpha^\infty} - 1 \right) a_\alpha (1 - a_\alpha s_\alpha^\infty) \\
&= \sum_\alpha \frac{2 s_\alpha^2}{\tau_\alpha} \left(\frac{a_\alpha}{a_\alpha^\infty} - 1 \right) a_\alpha \left(1 - \frac{a_\alpha}{a_\alpha^\infty} \right) \qquad \text{(since } a_\alpha^\infty = 1/s_\alpha^\infty) \\
&= - \sum_\alpha \frac{2 s_\alpha^2}{\tau_\alpha} a_\alpha \left(\frac{a_\alpha}{a_\alpha^\infty} - 1 \right)^2 .
\end{aligned}$$

At $t = 0$, we have $a_\alpha = a_\alpha^0$. Thus, the initial rate of change of loss is:

$$\frac{d\mathcal{L}}{dt}\Big|_{t=0} = -2 \sum_\alpha \frac{s_\alpha^2}{\tau_\alpha} a_\alpha^0 \left(\frac{a_\alpha^0}{a_\alpha^\infty} - 1 \right)^2 = -2 \sum_\alpha P \lambda s_\alpha^4 a_\alpha^0 \left(\frac{a_\alpha^0 - a_\alpha^\infty}{a_\alpha^\infty} \right)^2 . \tag{A115}$$

If we assume small initial parameters $a_\alpha^0 \ll a_\alpha^\infty$, then $\left(\frac{a_\alpha^0 - a_\alpha^\infty}{a_\alpha^\infty} \right)^2 \approx 1$. In this case,

$$\frac{d\mathcal{L}}{dt}\Big|_{t=0} \approx -2 \sum_\alpha P \lambda s_\alpha^4 a_\alpha^0 = -2 \sum_\alpha \frac{s_\alpha^2 a_\alpha^0}{\tau_\alpha}. \tag{A116}$$

Since $P, \lambda, s_\alpha^2, a_\alpha^0, \tau_\alpha$ are positive, the initial derivative is negative, indicating that the loss function initially decreases. The rate of decrease is proportional to the initial parameter values a_α^0 and the eigenvalues s_α^2 of the covariance matrix.