How do Transformers Perform In-Context Autoregressive Learning?

Michaël E. Sander [1] **Raja Giryes** [2] **Taiji Suzuki** [3] **Mathieu Blondel** [4] **Gabriel Peyré** [1]

Abstract

Transformers have achieved state-of-the-art performance in language modeling tasks. However, the reasons behind their tremendous success are still unclear. In this paper, towards a better understanding, we train a Transformer model on a simple next token prediction task, where sequences are generated as a first-order autoregressive process $s_{t+1} = W s_t$. We show how a trained Transformer predicts the next token by first learning W in-context, and then applying a prediction mapping. We call the resulting procedure *in-context autoregressive learning*. More precisely, focusing on commuting orthogonal matrices W, we first show that a trained one-layer linear Transformer implements one step of gradient descent for the minimization of an inner objective function when considering augmented tokens. When the tokens are not augmented, we characterize the global minima of a one-layer diagonal linear multi-head Transformer. Importantly, we exhibit orthogonality between heads and show that positional encoding captures trigonometric relations in the data. On the experimental side, we consider the general case of non-commuting orthogonal matrices and generalize our theoretical findings.

1. Introduction

Transformers (Vaswani et al., 2017) have achieved state-of-the-art performance in natural language processing tasks (Devlin et al., 2018). They now serve as the backbone for large language models, such as GPT (Radford et al., 2018; Brown et al., 2020), Chinchilla (Hoffmann et al., 2022), PaLM (Chowdhery et al., 2023), LLama (Touvron et al., 2023) or Mistral (Jiang et al., 2023). These models, which are causal, are trained to predict the next token s_{T+1} given a sequence (also termed as context) $s_{1:T} \coloneqq (s_1, \cdots, s_T)$. An intriguing property of large Transformers is their ability to adapt their computations given the context $s_{1:T}$. In this work, we make a step towards understanding this *in-context learning ability*. More precisely, assuming the tokens satisfy a relation $s_{T+1} = \varphi_W(s_{1:T})$, with W a context-dependent parameter varying with each sequence, we say that a trained Transformer *autoregressively learns* this relation *in-context* if it decomposes its prediction into 2 steps: first, estimating W through an in-context mapping, and then applying a simple prediction mapping, which is equal or closely related to φ_W (see Definition 1).

The goal of this paper is to fully characterize the autoregressive in-context learning process for optimally-trained Transformers. More precisely, building on the work of Von Oswald et al. (2023b), we focus on a simple autoregressive (AR) process of order 1, where each sequence is generated following the recursion $s_{T+1} = \varphi_W(s_{1:T}) \coloneqq W s_T$, and W is a randomly sampled orthogonal matrix, referred to as the *context matrix*. Such a process is illustrated in dimension 3 in Figure 1 for two different matrices W. We investigate the training of a linear Transformer to predict the next token in these AR processes, examining how it estimates W in-context and makes predictions for s_{T+1}. Depending on the input tokens encoding, the in-context mapping can correspond to gradient descent on an inner objective, as suggested by Von Oswald et al. (2023b). Alternatively, the context matrix W might be determined in closed form if the model possesses sufficient expressiveness. This paper investigates both scenarios.

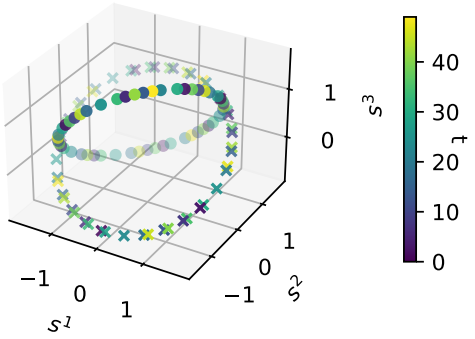

Figure 1. **Illustration of the autoregressive process in $\mathbb{R}^3$.** Dots and crosses correspond to two different orthogonal matrices W.

[1]Ecole Normale Supérieure and CNRS, France [2]Tel Aviv University, Israel [3]University of Tokyo and RIKEN AIP, Japan [4]Google DeepMind. Correspondence to: <michael.sander@ens.fr>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

More precisely, we make the following contributions:

- We begin by reviewing the background and previous works in §2. Then, in §3, we introduce our autoregressive process, which allows us to mathematically formalize the notion of *in-context autoregressive learning*.
- In §4, we demonstrate that if the matrices W commute and the model parameters possess a block structure, then a linear Transformer—trained on augmented tokens as introduced by Von Oswald et al. (2023b)—effectively implements a step of gradient descent on an underlying objective function as in-context mapping.
- In §5, we turn our attention to a one-layer linear attention Transformer that incorporates positional encoding but does not use augmented tokens. We comprehensively characterize the minimizers of the training loss. Notably, these minimizers display an orthogonality property across different heads. This aspect underscores the significance of positional encoding in enabling the Transformer to learn geometric operations between tokens through its in-context mapping. We also study positional-encoding-only attention and show that approximate minimum ℓ_2 norm solutions are favored by the optimization process.
- On the experimental side, in §6, we extend our analysis to the more general case where the context matrices W do not commute. We validate our theoretical findings for both augmented and non-augmented scenarios. Furthermore, we explore how variations in the distribution of the context matrices W affect trained positional encodings and lead to structures resembling those of traditional positional encodings commonly used in Transformers.

Each theoretical result of the paper aims at characterizing the autoregressive in-context learning mechanism for simple models and sequence data. Namely, Propositions 2, 3 and 6 give the structure of the minimizers of the training loss and explicit the corresponding in-context mappings, while Propositions 4, 5 and 7 focus on the optimization process.

Notations. We use lower cases for vectors and upper cases for matrices. $\|.\|$ is the ℓ_2 norm. We denote the transpose and adjoint operators by $^\top$ and *. $O(d)$ (resp $U(d)$) is the orthogonal (resp unitary) manifold, that is $O(d) = \{W \in \mathbb{R}^{d\times d} | W^\top W = I_d\}$ and $U(d) = \{W \in \mathbb{C}^{d\times d} | W^\star W = I_d\}$. The element-wise multiplication is $\odot$. $\langle | \rangle$ is the canonical dot product in $\mathbb{R}^d$, and $\langle \cdot | \cdot \rangle_\mathbb{C}$ the canonical hermitian product in $\mathbb{C}^d$. For $\lambda \in \mathbb{C}^d$, λ^k is the element-wise power k of λ: $(\lambda^k)_i = \lambda_i^k$.

2. Background and previous works

Causal Language Modelling. Language (or sequence) modeling refers to the development of models to predict the likelihood of observing a sequence $(x_1, \dots, x_T)$, where each x_t is called a token, and comes from a finite vocabulary. This can be done by using the chain rule of probability $P(X_1 = x_1, X_2 = x_2, \dots, X_T = x_T) = P(X_1 = x_1) \times P(X_2 = x_2 | X_1 = x_1) \times \dots \times P(X_T = x_T | X_1 = x_1, \dots, X_{T-1} = x_{T-1})$ (Jurafsky & Martin, 2009). Predicting these conditional probabilities can be done using a parametrized model $\mathcal{F}_\theta$ to minimize the loss $L(\mathcal{F}_\theta(x_1, ..., x_{T-1}), x_T)$ across all training samples and sequence length T. In common applications, L is chosen as the cross-entropy loss. In other words, the model is trained to predict the next token sequentially. Such a model is called a causal language model: it cannot access future tokens. Recently, the Transformer has emerged as the model of choice for language modeling.

Transformers. Transformers (Vaswani et al., 2017) process sequences of tokens $(x_1, \dots, x_T)$ of arbitrary length T. In its causal form (Brown et al., 2020; Touvron et al., 2023; Jiang et al., 2023), a Transformer first embeds the tokens to obtain a sequence $(e_1, \dots, e_T)$. It is then composed of a succession of blocks with residual connections (He et al., 2016). Each block is made of the composition of a multi-head self-attention module and a multi-layer perceptron (MLP). Importantly, the latter acts on each token separately, whereas multi-head self-attention mixes tokens, and corresponds to applying vanilla self-attention in parallel (Michel et al., 2019). More precisely, each multi-head self-attention is parametrized by a collection of weight matrices $(W_Q^h, W_K^h, W_V^h, W_O^h)_{1\le h\le H}$ and returns:

$$(\sum_{h=1}^{H} W_O^h \sum_{t'=1}^{t} \mathcal{A}_{t,t'}^h W_V^h e_{t'})_{t\in\{1,\cdots,T\}}, \tag{1}$$

where $\mathcal{A}^h$ is the attention matrix (Bahdanau et al., 2014) and is usually defined as

$$\mathcal{A}_{t,:}^h = \mathrm{softmax}(\langle W_Q^h e_t, W_K^h e_: \rangle),$$

with $\langle \cdot, \cdot \rangle$ a dot product. The sum over t' in (1) stopping at t reflects the causal aspect of the model: the future cannot influence the past. The output at position T is commonly used to predict the next token e_{T+1}. In practice, to help the model encode the relative position of the tokens in the sequence, a positional encoding (PE) is used.

Positional encoding. As described in Kazemnejad et al. (2023), encoding the position in Transformers amounts to defining the dot product $\langle \cdot, \cdot \rangle$ in the attention matrix, using additional (learnable or not) parameters. Popular designs include Absolute PE (Vaswani et al., 2017), Relative PE (Raffel et al., 2020), AliBI (Press et al., 2021), Rotary (Su et al., 2024), and NoPE (Kazemnejad et al., 2023). In this paper, we consider a learnable positional encoding.

Linear attention. In its simplest form, linear attention (Katharopoulos et al., 2020) consists in replacing the softmax in (1) by the identity. More formally, it consists in considering that each coefficient in the attention matrix is $\mathcal{A}^h_{t,t'} = \langle W^h_K e_{t'}, W^h_Q e_t \rangle$. The main practical motivation of linear attention is that it enables faster inference (Katharopoulos et al., 2020; Fournier et al., 2023). Note that even though they are called linear Transformers, the resulting models are non-linear with respect to the input sequence and jointly non-linear with respect to the parameters. From a theoretical perspective, linear attention has become the model of choice to understand the in-context-learning properties of Transformers (Mahankali et al., 2023; Ahn et al., 2023; Zhang et al., 2023).

In-context-Learning in Transformers The seminal work of Brown et al. (2020) reported an in-context-learning phenomenon in Transformer language models: these models can solve few-shot learning problems given examples in-context. Namely, given a sequence $(x_1, f(x_1), x_2, f(x_2), \cdots, x_n)$, a trained Transformer can infer the next output $f(x_n)$ without additional parameter updates. This surprising ability has been the focus of recent research. Some works consider the softmax attention without considering training dynamics (Garg et al., 2022; Akyürek et al., 2022; Li et al., 2023). Other works focus solely on linear attention and characterize the minimizers of the training loss when f is sampled across linear forms on $\mathbb{R}^d$, that is $f(x) = w^\top x$ for some w (Mahankali et al., 2023; Ahn et al., 2023; Zhang et al., 2023). In particular, these works discuss the ability of Transformers to implement optimization algorithms in their forward pass at inference, as empirically suggested by Von Oswald et al. (2023a). Nevertheless, the formulations used by Von Oswald et al. (2023a); Mahankali et al. (2023); Ahn et al. (2023); Zhang et al. (2023) are all based on concatenating the tokens so that the Transformer's input takes the form $\left(\begin{smallmatrix} x_1 & x_2 & \cdots & x_n \\ f(x_1) & f(x_2) & \cdots & 0 \end{smallmatrix}\right) \in \mathbb{R}^{(d+1)\times n}$. However, the necessity for this concatenation limits the impact of these results as there is no guarantee that the Transformer would implement this operation in its first layer. In addition, these works explicitly consider the minimization of an in-context loss, which is different from the next-token prediction loss in causal Transformers. In contrast, our work considers the next-token prediction loss and considers a more general notion of in-context learning, namely *in-context autoregressive learning*, that we describe in the next section.

3. Linear Attention for AR Processes

Token encoding. Building on the framework established by Von Oswald et al. (2023b), we consider a noiseless setting where each sequence begins with an initial token $s_1 = 1_d$. This token acts as a start-of-sentence marker. The subsequent states are generated according to $s_{t+1} = W s_t$, where W is a matrix referred to as the *context matrix*. This matrix is sampled uniformly from a subset $\mathcal{C}_O$ (respectively, $\mathcal{C}_U$) of $O(d)$ (respectively, $U(d)$), and we denote $\mathcal{W}$ as the corresponding distribution: $W \sim \mathcal{W} := \mathcal{U}(\mathcal{C})$. Considering norm-preserving matrices ensures the stability of the AR process, which is crucial to be able to learn from long sequences (i.e. using large T). In this paper, we contrast $\mathcal{C}_O$ and $\mathcal{C}_U$ to showcase how the distribution of in-context parameters W impacts the in-context mapping learned by Transformers. In addition, we have the following.

Remark 1. *If $W_U \in U(d)$, then*

$$W_O := \begin{bmatrix} \mathrm{Real}(W_U) & -\mathrm{Imag}(W_U) \\ \mathrm{Imag}(W_U) & \mathrm{Real}(W_U) \end{bmatrix}$$

is in $O(2d)$ and has pairwise conjugate eigenvalues. W_O is a rotation (because W_O is similar to a 2×2 block diagonal matrix with rotations). Reciprocally, for any rotation W_O of size $2d$ corresponds a unitary matrix W_U of size d by selecting half of the eigenvalues (for instance those with positive imaginary parts).

Therefore, $U(d)$ can be viewed as a subset of $O(2d)$, while $O(d) \subset U(d)$. As such, placing ourselves in $U(d)$ corresponds to a compact way of considering real AR processes in dimension $2d$.

In our analysis, we consider two settings in which the sequence $s_{1:T}$ is mapped to a new sequence $e_{1:T}$. In the *augmented setting* (§4), the tokens are defined as $e_t := (0, s_t, s_{t-1})$, aligning with the setup used by Von Oswald et al. (2023b). In contrast, the *non-augmented setting* (§5) utilizes a simpler definition where the tokens are simply $e_t := s_t$.

Model and training process. We consider a Transformer with linear attention, which includes an optionally trainable positional encoding $P \in \mathbb{R}^{T_{\max} \times T_{\max}}$ for some $T_{\max} \in \mathbb{N}$:

$$\mathcal{A}^h_{t,t'} = P_{t,t'} \langle W^h_Q e_t | W^h_K e_{t'} \rangle. \tag{2}$$

Throughout this paper, we will re-parameterize the model by setting $B^h = W^h_O W^h_V$ and $A^h = W^{h\top}_K W^h_Q$. Note that such an assumption is standard in theoretical studies on the training of Transformers (Mahankali et al., 2023; Zhang et al., 2023; Ahn et al., 2023). The trainable parameters are therefore $\theta = ((A^h, B^h, P))_{1 \le h \le H}$ when the positional encoding is trainable and $\theta = ((A^h, B^h))_{1 \le h \le H}$ otherwise. This defines a mapping $\mathcal{T}_\theta(e_{1:T})$ by selecting a section from some element τ in the output sequence (1). We focus on the population loss, defined as:

$$\ell(\theta) := \sum_{T=2}^{T_{\max}} \mathbb{E}_{W \sim \mathcal{W}} \| \mathcal{T}_\theta(e_{1:T}) - s_{T+1} \|^2, \tag{3}$$

indicating the model's objective to predict s_{T+1} given $e_{1:T}$. It is important to note that both s_{T+1} and $e_{1:T}$ appearing in (3) are computed from a random W and are therefore random variables.

In-context autoregressive learning. Our goal is to theoretically characterize the parameters θ^* that minimize ℓ, discuss the convergence of gradient descent to these minima, and characterize the in-context autoregressive learning of the model. This learning process is defined as the model's ability to learn and adapt within the given context: first by estimating W (or more generally some power of W) using an in-context mapping Γ, then by predicting the next token using a simple mapping ψ. In the context or AR processes, we formalize this procedure in the following definition.

Definition 1 (In-context autoregressive learning)**.** *We say that $\mathcal{T}_{\theta^*}$* learns autoregressively in-context *the AR process $s_{T+1} = Ws_T$ if $\mathcal{T}_{\theta^*}(e_{1:T})$ can be decomposed in two steps: (1) first applying an in-context mapping $\gamma = \Gamma_{\theta^*}(e_{1:T})$, (2) then using a prediction mapping $\mathcal{T}_{\theta^*}(e_{1:T}) = \psi_\gamma(e_{1:T})$. This prediction mapping should be of the form $\psi_\gamma(e_{1:T}) = \gamma s_\tau$ for some shift $\tau \in \{1, \cdots, T\}$. With such a factorization, in-context learning arises when the training loss $\ell(\theta^*)$ is small. This corresponds to having $\Gamma_{\theta^*}(e_{1:T}) \approx W^{T+1-\tau}$ when applied to data $e_{1:T}$ exactly generated by the AR process with matrix W.*

In this work, we will have either $\tau = T$ or $\tau = T - 1$.

Remark 2. *We use the word in-context to make explicit the fact that the matrix W is different for each sequence. As a consequence, attention-based models are particularly well suited to such a task because predicting W involves considering relationships between tokens. In contrast, RNNs perform poorly in this setting precisely because they do not consider interactions between tokens. In fact, in its simplest form, a linear RNN with parameters A and B outputs, for each t: $y_t = \sum_{k=1}^{t} A^{t-k} B W^{k-1} e_1$. It is easy to see that A and B would have to depend on W for y_t to be close to $W^t s_1$, which is impossible because W is different for each sequence.*

To fully characterize the in-context mapping Γ and prediction mapping ψ, we rely on a commutativity assumption.

Assumption 1 (Commutativity)**.** *The matrices W in $\mathcal{C}$ commute. Hence, they are co-diagonalizable in a unitary basis of $\mathbb{C}^{d\times d}$. Up to a change of basis, we therefore suppose $\mathcal{C}_U = \{\mathrm{diag}(\lambda_1, \cdots, \lambda_d), |\lambda_i| = 1\}$ and $\mathcal{C}_O = \{(\lambda_1, \bar{\lambda_1}, \cdots, \lambda_\delta, \bar{\lambda_\delta}), |\lambda_i| = 1\}$, with $d = 2\delta$.*

For conciseness, we only consider pairwise conjugate eigenvalues in $\mathcal{C}_O$. While assumption 1 is a strong one, it is a standard practice in the study of matrix-involved learning problems (Arora et al., 2019). We highlight that, to the best of our knowledge, this is the first work that provides a theoretical characterization of the minima of ℓ. The general problem involving non-commutative matrices is complex, and we leave it for future work. Note that recent studies, such as those by Mahankali et al. (2023); Zhang et al. (2023); Ahn et al. (2023) focus on linear regression problems $x \mapsto w^\top x$, which rewrites $x \mapsto 1_d^\top \mathrm{diag}(w)x$. Therefore, considering diagonal matrices is a natural extension of these approaches to autoregressive settings. Note also that imposing commutativity, while a simplification, represents a practical method of narrowing down the class of models φ_W. Indeed, in high dimension, it becomes necessary to restrict the set $\mathcal{C}$, otherwise, W cannot be accurately estimated when $T < d$. Note that however, in §6, we experimentally consider the general case of non-commuting matrices.

4. In-context mapping with gradient descent

In this section, we consider the augmented tokens $e_t := (0, s_t, s_{t-1})$. We show that under assumption 1 and an additional assumption on the structure of θ at initialization, the minimization of (3) leads to the linear Transformer implementing one-step of gradient descent on an inner objective as its in-context mapping Γ_{θ^*}. The motivation behind this augmented dataset is that the tokens e_t can be computed after a two-head self-attention layer with a $\mathrm{softmax}$. Indeed, we have the following result.

Lemma 1. *The tokens $e_{1:T}$ can be approximated with arbitrary precision given tokens $s_{1:T}$ with a Transformer (1).*

For a proof, see Appendix A.1. We suppose that $\mathcal{W} = \mathcal{U}(\mathcal{C}_U)$, that is we consider unitary matrices. We consider $\mathcal{T}_\theta$ to be a one-head attention layer with skip connection and output the fist d coordinates of the token T. More precisely, one has

$$\mathcal{T}_\theta(e_{1:T}) = \left(e_T + \sum_{t=1}^{T} \langle Ae_T | e_t \rangle_{\mathbb{C}} Be_t \right)_{1:d}. \tag{4}$$

Importantly, we do not consider a positional encoding as the relative position is already stored in each token e_t. Note that we use the hermitian product $\langle | \rangle_{\mathbb{C}}$ as $e_t \in \mathbb{C}^{3d}$.

We have the following result showing the existence of θ_0 such that (4) corresponds to one step of gradient descent on an inner objective.

Proposition 1 (Von Oswald et al. (2023b))**.** *There exists θ_0 such that $\Gamma_{\theta_0}(e_{1:T}) = W_0 - \eta \nabla L(W_0, e_{1:T})$ with*

$$L(W, e_{1:T}) = \frac{1}{2} \sum_{t=1}^{T-1} \|s_{t+1} - Ws_t\|^2, \tag{5}$$

and W_0 is any gradient descent initialization.

We now make the following assumption on the structure of A and B.

Assumption 2. *We parameterize A and B as*

$$A = \begin{pmatrix} 0 & 0 & 0 \\ 0 & A_1 & A_2 \\ 0 & A_3 & A_4 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 0 & B_1 & B_2 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix},$$

with $A_i = a_i I$ and $B_i = b_i I$.

Importantly, while the zero block structure is stable with gradient descent on loss (3), we do impose the non-zero blocks to stay diagonal during training. Note that considering diagonal matrices is a widely used assumption in the topic of linear diagonal networks (Woodworth et al., 2020; Pesme et al., 2021). Note also that we consider the general parametrization for A and B in our experiments in §6.

Under assumption 2, we have the following result, stating that at optimality, Γ_{θ^*} corresponds to Γ_{θ_0} in Proposition 2 with $W_0 = 0$.

Proposition 2 (In-context autoregressive learning with gradient-descent)**.** *Suppose $\mathcal{C} = \mathcal{C}_U$, assumptions 1 and 2. Then loss (3) is minimal for θ^* such that $a_1^* + a_4^* = a_2^* = b_2^* = 0$ and $a_3^* b_1^* = \frac{\sum_{T=2}^{T_{\max}} T}{\sum_{T=2}^{T_{\max}} (T^2+(d-1)T)}$. Furthermore, an optimal in-context mapping Γ_{θ^*} is one step of gradient descent starting from the initialization $\lambda = 0$, with a step size asymptotically equivalent to $\frac{3}{2T_{\max}}$ with respect to $T_{\max}$.*

For a full proof, see Appendix A.3. Proposition 2 demonstrates that a single step of gradient descent constitutes the optimal forward rule for the Transformer $\mathcal{T}_\theta$. This finding aligns with recent research showing that one step of gradient descent is the optimal in-context learner for one layer of self-attention in the context of linear regression (Mahankali et al., 2023; Zhang et al., 2023; Ahn et al., 2023). However, a substantial drawback of considering augmented tokens is that it requires previous layers to form these tokens which—although is possible according to Lemma 1—is a strong assumption. Therefore, in the next section, we consider the non-augmented setting where we do not make strong assumptions about previous layers.

5. In-context mapping as a geometric relation

In this section, we consider the non-augmented tokens where $e_t := s_t$, and a multi-head self-attention model $\mathcal{T}_\theta$:

$$\mathcal{T}_\theta(e_{1:T}) = \sum_{h=1}^{H} \sum_{t=1}^{T} P_{T-1,t} \langle e_t | A^h e_{T-1} \rangle_{\mathbb{C}} B^h e_t, \quad (6)$$

that is we consider $\tau = T-1$, the second to last token in the output, and no residual connections. While not considering the last token in the output is not done in practice, this small modification is necessary to achieve zero population loss. We stress out that *we still mask* the token we want to predict.

We consider a self-attention module with H heads, and we define the following assumption.

Assumption 3. *A^h and B^h are diagonal for all h: $A^h = \mathrm{diag}(a_h)$ and $B^h = \mathrm{diag}(b_h)$ with $(a_h, b_h) \in \mathbb{R}^d \times \mathbb{R}^d$.*

Importantly, we impose the diagonal structure during training. This diagonal aspect reflects the diagonal property of the context matrices. Under assumptions 1 and 3, we have the following result.

Lemma 2. *Suppose assumptions 1 and 3. Writing $a_h = (a_h^1, \cdots, a_h^d)$ and $b_h = (b_h^1, \cdots, b_h^d)$ and letting $\mathtt{A} := (a^1, \cdots, a^d) \in \mathbb{R}^{H \times d}$, and $\mathtt{B} := (b^1, \cdots, b^d)$, one has for an input sequence $e_{1:T} = (1_d, \lambda, \cdots, \lambda^{T-1}) \in \mathbb{R}^{T \times d}$:*

$$\mathcal{T}_\theta(e_{1:T}) = \sum_{t=1}^{T} P_{T-1,t} [\mathtt{B}^\top \mathtt{A}] \lambda^{t-T+1} \odot \lambda^{t-1}.$$

For a full proof, refer to appendix A.4. Note that $\mathtt{A}$ and $\mathtt{B}$ correspond to the concatenation of diagonals $W_K^{h\top} W_Q^h$ and $W_O^h W_V^h$ along heads.

It is easily seen from Lemma 2 that the choice of $P^*_{T-1,t} = \delta_{t=T}$ and $\mathtt{B}^{*\top}\mathtt{A}^* = I_d$ implies $\mathcal{T}_{\theta^*}(e_{1:T}) = \lambda^T$, and therefore $\ell(\theta^*) = 0$. We see that this requires at least d heads. A natural question is whether there are other optimal solutions and how to characterize them. To answer this question, we investigate the case of unitary context matrices before moving to orthogonal ones. We consider these cases separately because they both lead to different in-context mappings. We recall that $U(d)$ can be seen as a subset of $O(2d)$ (see Remark 1).

5.1. Unitary context matrices.

In this case, coefficients in the context matrices are drawn independently. This constrains the possible values for θ^* achieving zero loss. Indeed, we have the following result.

Proposition 3 (Unitary optimal in-context mapping)**.** *Suppose assumptions 1 and 3. Any $\theta^* = (\mathtt{A}^*, \mathtt{B}^*, P^*)$ achieving zero of the loss (3) satisfies $P^*_{T-1,t} = 0$ if $t \neq T$, $P^*_{T-1,T}(\mathtt{B}^{*\top}\mathtt{A}^*)_{ii} = 1$, and $(\mathtt{B}^{*\top}\mathtt{A}^*)_{ij} = 0$ for $i \neq j$. Therefore, one must have $H \geq d$. An optimal in-context mapping satisfies $\Gamma_{\theta^*}(e_{1:T}) = \bar{e}_{T-1} \odot e_T$ and the predictive mapping $\psi_\gamma(e_{1:T}) = \gamma \odot e_T$.*

Proof sketch. At $\ell(\theta^*) = 0$ one has $\mathcal{T}_{\theta^*}(e_{1:T}) = \lambda^T$. We notice that $\mathcal{T}_{\theta^*}(e_{1:T})$ is a polynomial in the λ_i's. Identifying the coefficients leads to the desired results. □

For a full proof, refer to appendix A.5. In particular, for $e_{1:T} = (1_d, \lambda, \cdots, \lambda^{T-1})$, we have $\Gamma_{\theta^*}(e_{1:T}) = \lambda$, and $\psi_{\Gamma_{\theta^*}}(e_{1:T}) = \lambda \odot \lambda^{T-1} = \lambda^T$.

Orthogonality. The equality $(\mathtt{B}^\top \mathtt{A})_{ij} = 0$ for $i \neq j$ corresponds to an orthogonality property between heads. Indeed, to further understand what Proposition 3 implies in

terms of learned model, let's look at the particular case in which $H = d$ and, at optimality, $\mathtt{A}^* = \mathtt{B}^* = I_d$ and $P^*_{T-1,t} = \delta_{t=T}$. Therefore, the positional encoding selects the last token in the input sequence, hence learning the structure of the training data. In parallel, each attention matrix captures a coefficient in λ: $(\mathcal{T}_{\theta^*}(e_{1:T}))_h = \lambda_h \lambda_h^{T-1}$. When there are more than d heads, some heads are useless, and can therefore be pruned. Such a finding can be related to the work of Michel et al. (2019), where the authors experimentally show that some heads can be pruned without significantly affecting the performance of Transformers. Orthogonality in the context of Transformers was also investigated by directly imposing orthogonality between the outputs of each attention head (Lee et al., 2019) or on attention maps (Chen et al., 2022; Zhang et al., 2021). The ability of the positional encoding to recover the spatial structure was already shown by Jelassi et al. (2022), which studies Vision Transformers (Dosovitskiy et al., 2020).

Convergence of gradient descent. Now that we have characterized all the global minima of the loss (3), we can study the convergence of the optimization process. We have the following Proposition, which shows that the population loss (3) writes as a quadratic form in $\mathtt{B}^\top \mathtt{A}$ and P, which enables connections with matrix factorization.

Proposition 4 (Quadratic loss)**.** *Under assumptions 1 and 3, loss (3) reads*

$$\ell(\mathtt{A}, \mathtt{B}, P) = \sum_{T=2}^{T_{\max}} l(\mathtt{B}^\top \mathtt{A}, P_{T-1})$$

with $l(\mathtt{C}, p) = \|p\|_2^2 \|\mathtt{C}\|_F^2 + p_{T-1}^2 S(\mathtt{C}^\top \mathtt{C}) - 2\mathrm{Tr}(\mathtt{C}) p_T + d$, *where S is the sum of all coefficients operator.*

A proof is in Appendix A.6. For an optimal P^* with $P^*_{T-1,t} = \delta_{t=T}$, $\ell(\mathtt{A}, \mathtt{B}, P^*) = (T_{\max} - 1)\|\mathtt{B}^\top \mathtt{A} - I\|_F^2$, for which we can use Theorem 2.2 of Nguegnang et al. (2021) to argue that for almost all initial values, gradient flow on ℓ will converge to a global minimum, that is $\mathtt{B}^\top \mathtt{A} = I_d$. When training is also done on $p_T := P_{T-1,T}$, the loss is then $\ell(\mathtt{A}, \mathtt{B}, P) = \sum_{T=2}^{T_{\max}} \|p_T \mathtt{B}^\top \mathtt{A} - I\|_F^2$. Note that even for $T_{\max} = 2$, convergence of gradient descent in $(\mathtt{A}, \mathtt{B}, p_2)$ on ℓ to a global minimum is an open problem, for which a conjecture (Nguegnang et al., 2021; Achour et al., 2021) states that for almost all initialization, $(\mathtt{A}, \mathtt{B}, p_2)$ will converge to a global minimum of ℓ. We provide evidence for global convergence in Figure 3. Yet, we have the following result in the scalar case $H = d = 1$. Its proof is in Appendix A.7.

Proposition 5. *Consider the loss $\ell(a, b, p) = (pab - 1)^2$. Suppose that at initialization, $|pab - 1| < 1$. Then gradient flow on (a, b, p) converges to a global minimum satisfying $a^* b^* p^* = 1$.*

Role of the softmax**.** Our results rely heavily on the use of linear attention. In fact, we could not find a natural way to express the global minimum of the training loss (3) when a softmax layer was involved, even in dimension $d = 1$. To gain more insight, we conducted an experiment where we trained different models with and without softmax and MLP layers. The results are shown in Figure 9 in Appendix B, where it is clear that in the case of commuting context matrices, using a softmax is incompatible with learning the underlying in-context mapping.

5.2. Orthogonal context matrices.

We now turn to the case where the context matrices are in $\mathcal{C}_O$. We recall that this imposes that the λ_i are pairwise conjugate. Therefore, the dimension d is even, and we write it $d = 2\delta$. The context matrices are therefore rotations. This property changes the optimization landscape and other solutions are possible, as shown in the following Lemma.

Lemma 3. *Suppose assumptions 1, 3. If $P^*_{T-1,T-1} = -1$, $P^*_{T-1,T} = 2$ and 0 otherwise, and $\mathtt{B}^{*\top} \mathtt{A}^* = \frac{1}{2}\mathrm{diag}(J, \cdots, J)$, with $J \in \mathbb{R}^{2\times 2}$ and $J_{ij} = 1$ for all i, j, then $\mathcal{T}_{\theta^*}(e_{1:T}) = \lambda^T$.*

For a full proof, refer to Appendix A.5. In this case, $H = \delta$ heads are sufficient to reach zero population loss. The optimal parameters can be exactly characterized.

Proposition 6 (Orthogonal optimal in-context mapping)**.** *Suppose assumptions 1 and 3. Any $\theta^* = (\mathtt{A}^*, \mathtt{B}^*, P^*)$ with $\ell(\theta^*) = 0$ in (3) satisfies, denoting $\mathtt{C}^* = \mathtt{B}^{*\top} \mathtt{A}^*$ and $p^* = P^*_{T-1}$: $p^*_t = 0$ if $t < T - 1$, $p^*_T \mathtt{C}^*_{i,i} = 1$, $p^*_T \mathtt{C}^*_{2i-1,2i} + (\mathtt{C}^*_{2i-1,2i-1} + \mathtt{C}^*_{2i-1,2i}) p^*_{T-1} = 0$, $p^*_T \mathtt{C}^*_{2i,2i-1} + (\mathtt{C}^*_{2i,2i} + \mathtt{C}^*_{2i,2i-1}) p^*_{T-1} = 0$, $\mathtt{C}^*_{2i-1,j} = \mathtt{C}^*_{2i,j} = 0$ for $j \neq 2i - 1, 2i$. An optimal in-context mapping is then, for $e_t = \lambda^{t-1}$: $\Gamma_{\theta^*}(e_{1:T}) = \lambda^2$, and the corresponding predictive mapping $\psi_{\Gamma_{\theta^*}(e_{1:T})}(e_{1:T}) = \lambda^2 \odot e_{T-1} = \lambda^T$.*

Proof sketch. Similarly to Proposition 3, we identify the coefficients of a polynomial, with careful inspection of terms involving pairwise conjugate contexts $(\lambda_i, \bar{\lambda}_i)$. □

Similarly to Proposition 3, this result indicates an orthogonal property between heads. A closer look at the computation of Γ_{θ^*} reveals that the relation implemented in-context by the Transformer in Proposition 6 is an extension of a known formula in trigonometry: $2\cos\theta R_\theta - I_2 = R_{2\theta}$, with R_θ the rotation of parameter θ in $\mathbb{R}^2$ (see Figure 2). Importantly, when $\delta \leq H < 2\delta = d$, the optimal $\mathtt{C}^*$ in Proposition 6 is of rank δ, which corresponds to Lemma 3. However, when $H \geq d$, full-rank solutions are achievable.

Under the assumptions of Proposition 3, the population loss (3) is also a quadratic form in P and $\mathtt{B}^\top \mathtt{A}$. Similarly to Proposition 4, global convergence results of gradient

descent on such loss function are still an open problem (Achour et al., 2021). We provide experimental evidence for convergence in Figure 3.

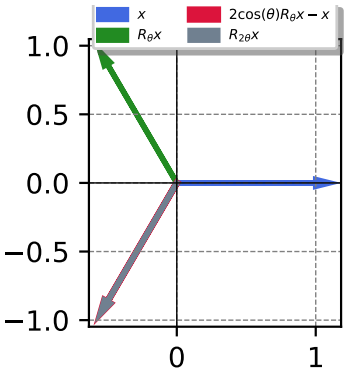

Figure 2. **Trigonometric formula** implemented by the Transformer in-context. The minima of the training loss correspond to implementing, up to multiplying factors: $2\cos\theta R_\theta - I_2 = R_{2\theta}$.

5.3. Positional encoding-only attention.

We end this section by investigating the impact of the context distribution on the trained positional encoding P. For this, we consider a positional encoding-only Transformer, that is, we fix $\mathtt{B}^\top \mathtt{A} = I_d$. In this case, the problem decomposes component-wise, and we only need to consider the $d = 1$ case. We therefore consider the AR process $s_{t+1} = \lambda s_t$ for $|\lambda| = 1$. We break the symmetry of the context distribution: for $\mu \geq 1$ and $\theta \sim \mathcal{U}(0, 2\pi)$, we define $\lambda = e^{i\theta/\mu}$. We denote $\mathcal{W}(\mu)$ as the corresponding distribution. Therefore, we focus on the optimization problem:

$$\min_{p \in \mathbb{R}^T} l(p) := \mathbb{E}_{\lambda \sim \mathcal{W}(\mu)} |\sum_{t=1}^{T} p_t \lambda^{2t-T} - \lambda^T|^2. \qquad (7)$$

Here again, the same proof as for Proposition 3 shows that the optimal positional encoding is $p^* = \delta_{t=T}$, meaning that we predict the next token using the last token in the context. However, depending on μ, (7) can be ill-conditioned.

Proposition 7 (Conditioning)**.** *The Hessian $H \in \mathbb{R}^{T \times T}$ of l in (7) is given by*

$$H_{t,t'} = \frac{\mu}{4\pi(t'-t)} \sin(4(t'-t)\frac{\pi}{\mu}).$$

Denoting $\sigma_1(\mu) \geq \cdots \geq \sigma_T(\mu)$ its eigenvalues, one has $\sigma_1(\mu) \to T$ and $\sigma_{t>1}(\mu) \to 0$ as $\mu \to +\infty$.

Therefore, for large μ, H in Proposition 7 is poorly conditioned. In such a setting, gradient descent, even with a large number of iterations, induces a ℓ_2 regularization (Yao et al., 2007). As an informal consequence, approximate solutions computed by gradient descent significantly deviate from the optimal p^*. As demonstrated experimentally in Figure 6 and §6, the effect of this regularization is a spatial smoothing of the positional encoding, which leads to entirely different in-context mappings Γ, hence showing the effect of the optimization process on the in-context autoregressive learning abilities of Transformers.

6. Experiments

In this section, we illustrate and extend our results through experiments. Our code in Pytorch (Paszke et al., 2017) and JAX (Bradbury et al., 2018) is open-sourced at `https://github.com/michaelsdr/ical`. We use the standard parametrization of Transformers, that is we train on (W_Q, W_K, W_V, W_O).

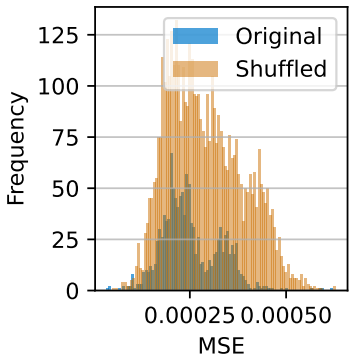

Figure 4. **Histograms** of the mean squared errors (MSE) when fitting an AR process to sequences in D (original, in blue) or D_{shuffle} (shuffled, in orange). We only display MSEs bigger than a threshold of 10^{-12}.

Validation of the token encoding choice. Throughout the paper, we assume that the $s_{1:T}$ are generated following an AR process $s_{t+1} = W s_t$. Even though we acknowledge that the AR process is an overly simplistic model for real-word sentences, we provide empirical justification for using it by showing that such a process better explains real data than random ones. We use the `nltk` package (Bird et al., 2009), and we employ classic literary works, specifically 'Moby Dick' by Herman Melville sourced from Project Gutenberg. We use the tokenizer and word embedding layer of a pre-trained GPT-2 model (Radford et al., 2019), and end up with about 325000 token representations in dimension $d = 1280$, that we reformat in a dataset D of shape (n, T, d), where $T = 5$ (we keep the relative order of each token). We also consider a shuffled counterpart of D where the shuffling is done across the first two dimensions. In other words, we create a dataset D_{shuffle} from a permutation of the tokens of the book.

We then fit AR processes for each sequence in the two datasets using loss (5), which we minimize by solving a linear system. It should be noted that the problem remains non-trivial for some sequences, despite T being significantly smaller than d. This complexity arises because certain sequences might contain identical elements with differing successors or predecessors. We hypothesize that when sequences are shuffled, the number of such inconsistencies increases since the language's structure is lost. This hypothesis is validated in Figure 4, where we display the histograms of the fitting losses when they are bigger than 10^{-12}. There

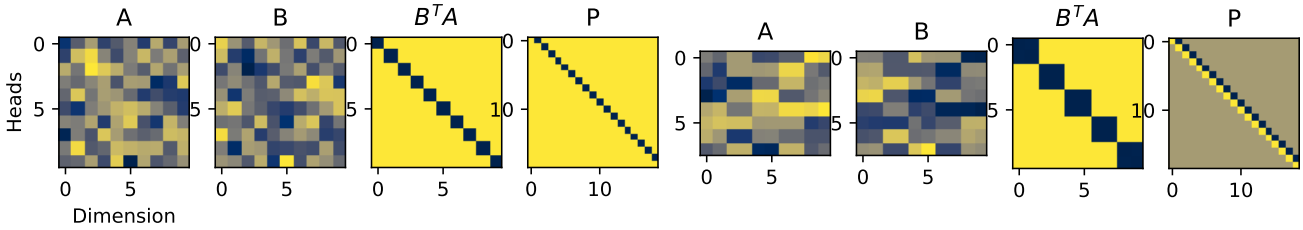

Figure 3. **Matrices $\mathtt{A}$, $\mathtt{B}$, $\mathtt{B}^\top\mathtt{A}$ and P after training model (6) on loss (3) with random initialization**. We take $d = 10$ and $T = 15$. **Left:** Unitary context case with $H = 10$. **Right:** Orthogonal context case, with $H = 8 < d$, which leads to low rank $\mathtt{B}^\top\mathtt{A}$. In both cases, we obtain arbitrarily small final loss. We recover parameters corresponding to our Propositions 3 and 6.

are 4 times more sequences with such an error for the shuffled dataset than for the original. This shows that the AR process is better suited when data present some semantics.

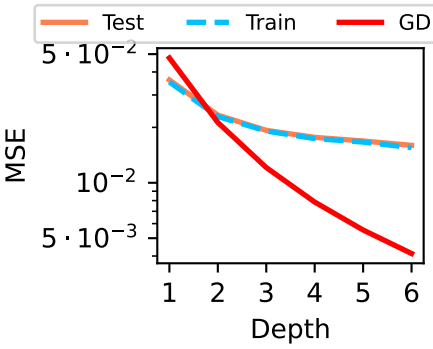

Figure 5. **Evolution** of the MSE with depth L. We compare with L steps of gradient descent on the inner loss (5). At initialization, the MSE is between 1 and 2.

Augmented setting. We investigate whether the results of §4 still hold without assumptions 1 and 2. We consider the model (4) on the augmented tokens $e_t = (0, s_t, s_{t-1})$. We iterate relation (4) with several layers, using layer normalization (Ba et al., 2016). We consider depth values from 1 to 6. We generate a dataset with $n = 2^{14}$ sequences with $T_{\max} = 50$ and $d = 5$ (therefore $e_t \in \mathbb{R}^{15}$) for training. We test using another dataset with 2^{10} sequences of the same shape. We train for 2000 epochs with Adam (Kingma & Ba, 2014) and a learning rate of 5×10^{-3} to minimize the mean squared error (MSE) $\min \ell(\Theta) := \sum_{T=2}^{T_{\max}} \frac{1}{n} \sum_{i=1}^{n} \|\mathcal{T}_\Theta^L(e^i_{1:T}) - s^i_T\|^2$, where $\mathcal{T}_\theta^L$ correspond to L layers of (4) (we apply the forward rule L times, and then consider the section of first d coordinates). We compare the error with L steps of gradient descent on the inner loss (5), with a step size carefully chosen to obtain the fastest decrease. We find out that even though the first Transformer layers are competitive with gradient descent, the latter outperforms the Transformer by order of magnitudes when $L \geq 3$. Results are displayed in Figure 5. The fact that several steps of gradient descent outperform the same number of $\mathcal{T}_\theta$ layers is not surprising, as Proposition 2 does not generalize to more than one layer. In contrast, as shown in Appendix B, a full Transformer with all the bells and whistles as described in Vaswani et al. (2017) (softmax and MLP applied component-wise to each Transformer layer) outperforms gradient descent and has a similar trend, as shown in Figure 10.

Non-Augmented setting. We now investigate whether the results of §5 still hold without assumptions 1 and 3. We consider the model $\mathcal{T}_\theta$ in (6). We parameterize the positional encoding in the linear Transformer equation (6) using the softmax of a positional attention-only similarity cost matrix with learnable parameters $W_{Q_{\text{pos}}}$ and $W_{K_{\text{pos}}}$: $P_{t,t'} = \text{softmax}(\langle W_{Q_{\text{pos}}} p_t | W_{K_{\text{pos}}} p_{t'} \rangle)$, as we found it to stabilize the training process. We use a similar dataset as in the previous section, i.e., a training set with 2^{14} sequences, each with 50 elements of dimension $d = 10$, and we test using another dataset with 2^{10} sequences of the same shape.

We train models $\mathcal{T}_\theta$ in (6) for 200 epochs with different numbers of heads. We use the Adam optimizer with a learning rate of 10^{-2}. Without further modification, we do not observe a significant gain as the number of heads increases. However, when duplicating the data along the dimension axis, that is $e_t := (s_t, s_t)$, we observe a significant improvement, as illustrated in Figure 7. Understanding why duplicating the data leads to a significant improvement is left for future work.

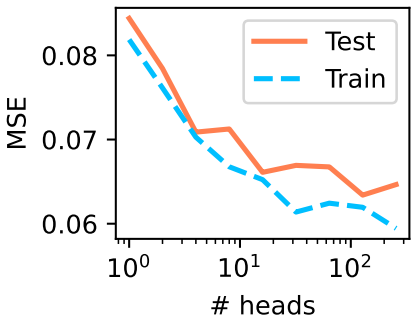

Figure 7. **Evolution** of the MSE with the number of heads. At initialization, the MSE is between 0.35 and 1.

To further relate our experimental findings to our theory, we also exhibit an orthogonality property between heads after training. For this, we take $d = 5$ to ease the visualization, and initialize each parameter equally across heads but add a small perturbation $.05 \times \mathcal{N}(0, 1)$ to ensure different gradient propagation during training. We then train the model and compare the quantity $(\sum_{h=1}^{H} (B_h^\top B_h))_{(i,j)}$ after training and at initialization. Note that it corresponds to a measure of orthogonality of heads. Results are displayed in Figure 8.

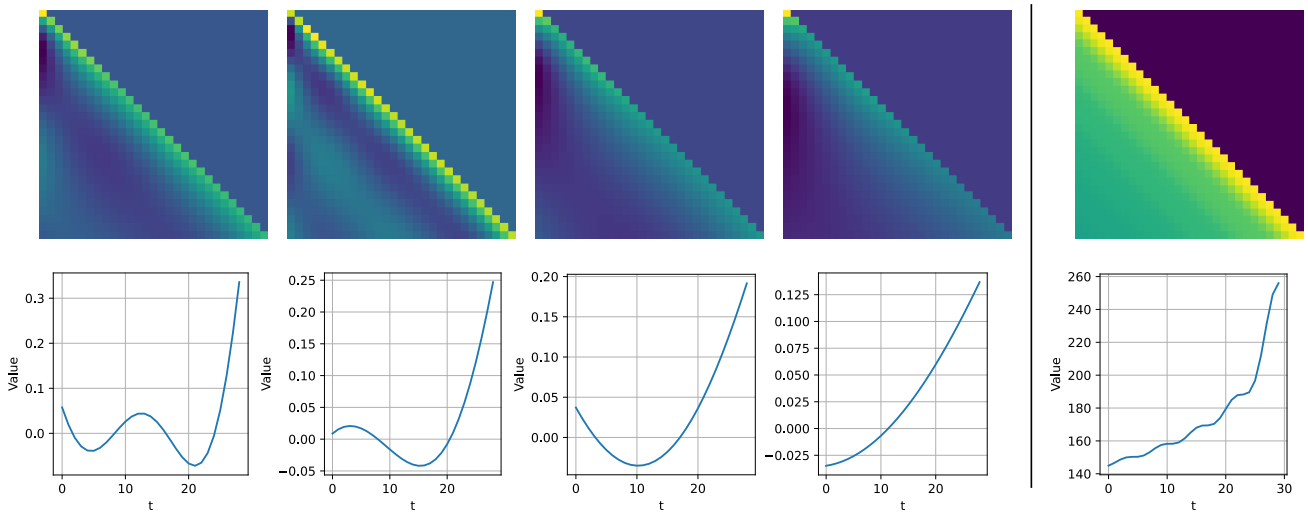

Figure 6. **Left:** Positional encodings after training for $\mu \in \{50, 100, 200, 300\}$. The first raw corresponds to the matrix P, and the second raw to a plot of its last raw. **Right:** Comparison with the cosine absolute positional encoding standardly used in machine translation (Vaswani et al., 2017) (we display $pp^\top$). In both cases, we observe an invariance across diagonals. In addition, for high μ (i.e. small variations of the context), the most recent tokens are more informative, as imposed by the inductive prior of the cosine positional encoding.

We observe that after training, an orthogonality property appears. In addition, as we are duplicating the tokens across dimensions, we can see that heads become specialized in attending to some coordinates across tokens.

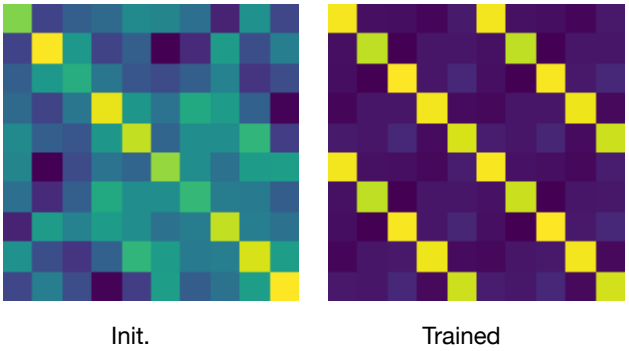

Figure 8. **Matrices** $(\sum_{h=1}^{H}(B_h^\top B_h))_{(i,j)} \in \mathbb{R}^{10\times 10}$ at initialization and after training. The trained parameters lead to an orthogonality between heads, as predicted by our theory.

Change in the context distribution. We consider the setting of §5.3, using the empirical loss counterpart of (7), averaged over $T \in \{2, \cdots, T_{\max}\}$. We generate a dataset with 10^4 examples and $T_{\max} = 30$. We train our positional encoding-only model with gradient descent and stop training (early stopping) when the loss is smaller than 10^{-3}. We initialize $P_{t,t'} = 0$. Results are in Figure 6, where we mask coefficients $P_{T-1,T}$ (which are close to 1 after training) in the display to investigate the behavior of the extra coefficients. We observe that the trained positional encoding exhibits an invariance across diagonals. Importantly, each row P_t has a smooth behavior with t', that we compare to absolute cosine positional encodings (Vaswani et al., 2017).

Conclusion

In this work, we study the in-context autoregressive learning abilities of linear Transformers to learn autoregressive processes of the form $s_{t+1} = Ws_t$. In-context autoregressive learning is decomposed into two steps: estimation of W with an in-context map Γ, followed by a prediction map ψ. Under commutativity and parameter structure assumptions, we first characterized Γ and ψ on augmented tokens, in which case Γ is a step of gradient descent on an inner objective function. We also considered non-augmented tokens and showed that Γ corresponds to a non-trivial geometric relation between tokens, enabled by an orthogonality between trained heads and learnable positional encoding. We also studied positional encoding-only attention and showed that approximate solutions of minimum ℓ_2 norm are favored by the optimization. Moving beyond commutativity assumptions, we extended our theoretical findings to the general case through experiments.

Future work. Investigating the case where $\tau = T$ in the non-augmented setting would lead to approximated in-context mappings, where achieving zero loss is no longer possible. This investigation would provide further insight into the role of positional encoding in estimating W in-context. Another investigation left for future work is to consider the case of non-commuting context matrices and to relate the computation of a Transformer to a proxy for a gradient flow for estimating W in-context, using the connection between Transformers and gradient flows (Sander et al., 2022).

Impact Statement

This paper is mainly theoretical and its goal is to advance our understanding of Transformers.

Acknowledgments

The work of M. Sander and G. Peyré was supported by the French government under the management of Agence Nationale de la Recherche as part of the "Investissements d'avenir" program, reference ANR-19-P3IA-0001 (PRAIRIE 3IA Institute). T. Suzuki was partially supported by JSPS KAKENHI (24K02905) and JST CREST (JPMJCR2115, JPMJCR2015). MS thanks Scott Pesme and Francisco Andrade for fruitful discussions.

References

Achour, E. M., Malgouyres, F., and Gerchinovitz, S. The loss landscape of deep linear neural networks: a second-order analysis. *arXiv preprint arXiv:2107.13289*, 2021.

Ahn, K., Cheng, X., Daneshmand, H., and Sra, S. Transformers learn to implement preconditioned gradient descent for in-context learning. *arXiv preprint arXiv:2306.00297*, 2023.

Akyürek, E., Schuurmans, D., Andreas, J., Ma, T., and Zhou, D. What learning algorithm is in-context learning? investigations with linear models. *arXiv preprint arXiv:2211.15661*, 2022.

Arora, S., Cohen, N., Hu, W., and Luo, Y. Implicit regularization in deep matrix factorization. *Advances in Neural Information Processing Systems*, 32, 2019.

Ba, J. L., Kiros, J. R., and Hinton, G. E. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Bird, S., Klein, E., and Loper, E. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.", 2009.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

Chen, T., Zhang, Z., Cheng, Y., Awadallah, A., and Wang, Z. The principle of diversity: Training stronger vision transformers calls for reducing all levels of redundancy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12020–12030, 2022.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Fournier, Q., Caron, G. M., and Aloise, D. A practical survey on faster and lighter transformers. *ACM Computing Surveys*, 55(14s):1–40, 2023.

Garg, S., Tsipras, D., Liang, P. S., and Valiant, G. What can transformers learn in-context? a case study of simple function classes. *Advances in Neural Information Processing Systems*, 35:30583–30598, 2022.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., Casas, D. d. L., Hendricks, L. A., Welbl, J., Clark, A., et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Jelassi, S., Sander, M., and Li, Y. Vision transformers provably learn spatial structure. *Advances in Neural Information Processing Systems*, 35:37822–37836, 2022.

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., Casas, D. d. l., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Jurafsky, D. and Martin, J. H. *Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition*. Pearson Prentice Hall, 2009.

Katharopoulos, A., Vyas, A., Pappas, N., and Fleuret, F. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pp. 5156–5165. PMLR, 2020.

Kato, T. *Perturbation theory for linear operators*, volume 132. Springer Science & Business Media, 2013.

Kazemnejad, A., Padhi, I., Ramamurthy, K. N., Das, P., and Reddy, S. The impact of positional encoding on length generalization in transformers. *arXiv preprint arXiv:2305.19466*, 2023.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Lee, M., Lee, J., Jang, H. J., Kim, B., Chang, W., and Hwang, K. Orthogonality constrained multi-head attention for keyword spotting. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 86–92. IEEE, 2019.

Li, Y., Ildiz, M. E., Papailiopoulos, D., and Oymak, S. Transformers as algorithms: Generalization and stability in in-context learning. In *International Conference on Machine Learning*, pp. 19565–19594. PMLR, 2023.

Mahankali, A., Hashimoto, T. B., and Ma, T. One step of gradient descent is provably the optimal in-context learner with one layer of linear self-attention. *arXiv preprint arXiv:2307.03576*, 2023.

Michel, P., Levy, O., and Neubig, G. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32, 2019.

Nguegnang, G. M., Rauhut, H., and Terstiege, U. Convergence of gradient descent for learning linear neural networks. *arXiv preprint arXiv:2108.02040*, 2021.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

Pesme, S., Pillaud-Vivien, L., and Flammarion, N. Implicit bias of sgd for diagonal linear networks: a provable benefit of stochasticity. *Advances in Neural Information Processing Systems*, 34:29218–29230, 2021.

Press, O., Smith, N. A., and Lewis, M. Train short, test long: Attention with linear biases enables input length extrapolation. *arXiv preprint arXiv:2108.12409*, 2021.

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. Improving language understanding by generative pre-training. 2018.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. 2019.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

Sander, M. E., Ablin, P., Blondel, M., and Peyré, G. Sinkformers: Transformers with doubly stochastic attention. In *International Conference on Artificial Intelligence and Statistics*, pp. 3515–3530. PMLR, 2022.

Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., and Liu, Y. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Von Oswald, J., Niklasson, E., Randazzo, E., Sacramento, J., Mordvintsev, A., Zhmoginov, A., and Vladymyrov, M. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, pp. 35151–35174. PMLR, 2023a.

Von Oswald, J., Niklasson, E., Schlegel, M., Kobayashi, S., Zucchet, N., Scherrer, N., Miller, N., Sandler, M., y Arcas, B. A., Vladymyrov, M., Pascanu, R., and Sacramento, J. Uncovering mesa-optimization algorithms in transformers, 2023b.

Woodworth, B., Gunasekar, S., Lee, J. D., Moroshko, E., Savarese, P., Golan, I., Soudry, D., and Srebro, N. Kernel and rich regimes in overparametrized models. In *Conference on Learning Theory*, pp. 3635–3673. PMLR, 2020.

Yao, Y., Rosasco, L., and Caponnetto, A. On early stopping in gradient descent learning. *Constructive Approximation*, 26:289–315, 2007.

Zhang, A., Chan, A., Tay, Y., Fu, J., Wang, S., Zhang, S., Shao, H., Yao, S., and Lee, R. K.-W. On orthogonality constraints for transformers. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, volume 2, pp. 375–382. Association for Computational Linguistics, 2021.

Zhang, R., Frei, S., and Bartlett, P. L. Trained transformers learn linear models in-context. *arXiv preprint arXiv:2306.09927*, 2023.

A. Proofs

In what follows, as we consider complex numbers, we use the hermitian product over $\mathbb{C}^d$, that is $\langle \alpha, \beta \rangle := \beta^\star \alpha = \sum_i \alpha_i \bar{\beta}_i$.

We denote $\mathcal{U}^d = \{\lambda \in \mathbb{C}^d | |\lambda_i| = 1 \forall i \in \{1, \cdots, d\}\}$.

A.1. Proof of Lemma 1.

Proof. Let, for $s \in \mathbb{R}^d$, $W_V^1 s := (0, s, 0) \in \mathbb{R}^{3d}$ and $W_V^2 s := (0, 0, s)$.

We now simply consider a positional attention-only model, that is, for $h \in \{1, 2\}$:

$$\mathcal{A}^h_{t,:} = \mathrm{softmax}(P^h_{t,:}).$$

We can choose the positional encodings P^1 and P^2 such that $A^1_{t,t'} \simeq \delta_{t'=t}$ and $A^2_{t,t'} \simeq \delta_{t'=t-1}$.

Then

$$\sum_{h=1}^{2} \sum_{t'=1}^{t} \mathcal{A}^h_{t,t'} W_V^h s_{t'} \simeq (0, s_t, s_{t-1}) = e_t.$$

□

A.2. Proof of Proposition 1.

Proof. We briefly recall the reasoning as presented in Von Oswald et al. (2023b), and consider the case $W_0 = 0$ for simplicity. See Von Oswald et al. (2023b) for a full proof. Let

$$A = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & I_d & 0 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 0 & \eta I_d & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}. \tag{8}$$

Then the section vector of the first d coordinates in (4) is $\eta \sum_{t=1}^{T} s_t s_{t-1}^\top s_T$.

The gradient of L at $W_0 = 0$ is:

$$\nabla_W L(0, e_{1:T}) = -\sum_{t=1}^{T-1} (s_{t+1}) s_t^\top.$$

Therefore, (4) corresponds to a single step of gradient descent starting from $W_0 = 0$. □

A.3. Proof of Proposition 2.

In what follows, the sums over t are from $t = 1$ to T and the sums over j are from $j = 1$ to d.

We first consider the following Lemma.

Lemma 4. *Under assumptions 1 and 2, the loss of the linear Transformer writes:*

$$\ell((a_i), (b_i)) = \mathbb{E}_\lambda \sum_{T=2}^{T_{\max}} \sum_{i=1}^{d} | \sum_{t,j,\alpha \in \mathcal{A}, \beta \in \mathcal{B}} c_{\alpha,\beta} \lambda_j^{T-t-\alpha} \lambda_i^{t-1+\beta} - \lambda_i^T |^2 \tag{9}$$

with $\mathcal{A} = \{-1, 0, 1\}$ and $\mathcal{B} = \{-1, 0\}$. We have $c_{\alpha,\beta} = u_\alpha v_\beta$ for $u_0 = a_1 + a_4$, $u_1 = a_2$, $u_{-1} = a_3$, $v_0 = b_1$ and $v_{-1} = b_2$.

Proof. One has $Ae_T = (0, A_1 s_T + A_2 s_{T-1}, A_3 s_T + A_4 s_{T-1})$. Therefore one has $\langle Ae_T, e_t \rangle = s_t^\star A_1 s_T + s_t^\star A_2 s_{T-1} + s_{t-1}^\star A_3 s_T + s_{t-1}^\star A_4 s_{T-1}$. Since $(Be_t)_{1:d} = B_1 s_t + B_2 s_{t-1}$, one obtains through (4):

$$\mathcal{T}_\theta(e_{1:T}) = \sum_t (a_1 s_t^\star s_T + a_2 s_t^\star s_{T-1} + a_3 s_{t-1}^\star s_T + a_4 s_{t-1}^\star s_{T-1})(b_1 s_t + b_2 s_{t-1}).$$

Developing, we obtain

$$\mathcal{T}_\theta(e_{1:T})_i = \sum_{t,j,\alpha\in\mathcal{A}} u_\alpha \lambda_j^{T-t-\alpha} \sum_{\beta\in\mathcal{B}} v_\beta \lambda_i^{t-1+\beta} = \sum_{t,j,\alpha\in\mathcal{A},\beta\in\mathcal{B}} c_{\alpha,\beta}\lambda_j^{T-t-\alpha}\lambda_i^{t-1+\beta},$$

which implies the result. □

Using the notations of Lemma 4, Proposition 2 now writes as follows.

Proposition 8 (In-context autoregressive learning with gradient-descent.)**.** *Suppose $\mathcal{C} = \mathcal{C}_U$, assumptions 1 and 2. Then loss (9) is minimal for $c_{\alpha,\beta} = 0$ if $(\alpha,\beta) \neq (-1,0)$ and $c_{-1,0} = \frac{\sum_{T=2}^{T_{\max}} T}{\sum_{T=2}^{T_{\max}}(T^2+(d-1)T)}$. Therefore, the optimal in-context map Γ_{θ^*} is one step of gradient descent starting from the initialization $\lambda = 0$, with a step size asymptotically equivalent to $\frac{3}{2T_{\max}}$ with respect to $T_{\max}$.*

Proof. We develop the term in the sum in (9):

$$\begin{aligned}
&|\sum_{t,j,\alpha\in\mathcal{A},\beta\in\mathcal{B}} c_{\alpha,\beta}\lambda_j^{T-t-\alpha}\lambda_i^{t+\beta-1} - \lambda_i^T|^2 \\
&= \sum_{t_1,j_1,\alpha_1,\beta_1,t_2,j_2,\alpha_2,\beta_2} c_{\alpha_1,\beta_1}c_{\alpha_2,\beta_2}\lambda_{j_1}^{T-t_1-\alpha_1}\lambda_{j_2}^{-T+t_2+\alpha_2}\lambda_i^{t_1-t_2+\beta_1-\beta_2} \\
&- 2\mathrm{Real}(\sum_{t,j,\alpha\in\mathcal{A},\beta\in\mathcal{B}} c_{\alpha,\beta}\lambda_j^{T-t-\alpha}\lambda_i^{t+\beta-1}\lambda_i^{-T}) + 1.
\end{aligned}$$

We now need to compute the expectations of the first two terms in the sum. Because $\mathbb{E}_\lambda(\lambda_i^k) = \delta_{k=0}$ and the λ_i are i.i.d., most terms will be zeros.

- For the first term, one needs to look at the different possible values for (j_1, j_2, i) to calculate $\mathbb{E}_\lambda(\lambda_{j_1}^{T-t_1-\alpha_1}\lambda_{j_2}^{-T+t_2+\alpha_2}\lambda_i^{t_1-t_2+\beta_1-\beta_2})$.

  - 1) If $j_1 = j_2 = i$, it is $\mathbb{E}_\lambda(\lambda_i^{\alpha_2-\alpha_1+\beta_1-\beta_2}) = \delta_{\alpha_1-\alpha_2=\beta_1-\beta_2}$.

  - 2) If $j_1 = j_2 := j \neq i$, it is $\mathbb{E}_\lambda(\lambda_j^{t_2-t_1+\alpha_2-\alpha_1})\mathbb{E}_\lambda(\lambda_i^{t_1-t_2+\beta_1-\beta_2}) = \delta_{t_2-t_1=\alpha_1-\alpha_2=\beta_1-\beta_2}$.

  - 3) If $j_1 \neq j_2$, $i \neq j_1$, $i \neq j_2$, then $\mathbb{E}_\lambda(\lambda_{j_1}^{T-t_1-\alpha_1}\lambda_{j_2}^{-T+t_2+\alpha_2}\lambda_i^{t_1-t_2+\beta_1-\beta_2}) = \delta_{t_1-t_2=\beta_2-\beta_1,T-t_1-\alpha_1=0,T-t_2-\alpha_2=0}$.

  - 4) If $j_1 \neq j_2$ and $i = j_1$, the expectation is $\delta_{T-\alpha_1-t_2+\beta_1-\beta_2=0,T=t_2+\alpha_2}$, and similarly when $j_1 \neq j_2$ and $i = j_2$.

  As a consequence, we see that all the terms that do not satisfy $\alpha_1 - \alpha_2 = \beta_1 - \beta_2$ will lead to 0 expectation, which therefore implies that the first term writes:

$$\mathbb{E}_\lambda(\sum_{t_1,j_1,\alpha_1,\beta_1,t_2,j_2,\alpha_2,\beta_2,\alpha_1-\alpha_2=\beta_1-\beta_2} c_{\alpha_1,\beta_1}c_{\alpha_2,\beta_2}\lambda_{j_1}^{T-t_1-\alpha_1}\lambda_{j_2}^{-T+t_2+\alpha_2}\lambda_i^{t_1-t_2+\beta_1-\beta_2}).$$

- For the second term, we have

$$\sum_{t,j,\alpha\in\mathcal{A},\beta\in\mathcal{B}} c_{\alpha,\beta}\lambda_j^{T-t-\alpha}\lambda_i^{t+\beta-1}\lambda_i^{-T} = \sum_{t,\alpha,\beta} c_{\alpha,\beta}\sum_j(\lambda_j^{T-t-\alpha}\lambda_i^{t+\beta-1-T}).$$

  - When $i \neq j$, the expectation of $\lambda_j^{T-t-\alpha}\lambda_i^{t+\beta-1-T}$ is not 0 if and only if $t+\alpha = T$ and $T = t+\beta-1$. Given that $\mathcal{A} = \{-1,0,1\}$ and $\mathcal{B} = \{-1,0\}$, this implies $\alpha = \beta - 1$ and therefore $\alpha = -1$ and $\beta = 0$. But then we have $t = T+1$ which is not possible.

  - When $i = j$, the expectation of $\lambda_j^{T-t-\alpha}\lambda_i^{t+\beta-1-T} = \lambda_i^{\beta-\alpha-1}$ is not 0 if and only if $\beta = \alpha+1$, that is $\beta = 0$ and $\alpha = -1$.

  Therefore, one has

$$\mathbb{E}_\lambda(\sum_{t,\alpha,\beta} c_{\alpha,\beta}\sum_j(\lambda_j^{T-t-\alpha}\lambda_i^{t+\beta-1-T})) = \sum_t c_{-1,0} = Tc_{-1,0}$$

  and the second term is $-2Tc_{-1,0}$.

Back to the full expectation, isolating the term in $c_{-1,0}^2$ from the first term, we get

$$\mathbb{E}_\lambda|\sum_{t,j,\alpha\in\mathcal{A},\beta\in\mathcal{B}} c_{\alpha,\beta}\lambda_j^{T-1-t-\alpha}\lambda_i^{t+\beta-1} - \lambda_i^T|^2 = \tag{10}$$

$$\mathbb{E}_\lambda(\sum_{t_1,j_1,\alpha_1,\beta_1,t_2,j_2,\alpha_2,\beta_2,\alpha_1-\alpha_2=\beta_1-\beta_2,(\alpha_1,\alpha_2,\beta_1,\beta_2)\neq(-1,-1,0,0)} c_{\alpha_1,\beta_1}c_{\alpha_2,\beta_2}\lambda_{j_1}^{T-t_1-\alpha_1}\lambda_{j_2}^{-T+t_2+\alpha_2}\lambda_i^{t_1-t_2+\beta_1-\beta_2})$$

$$+K_T c_{-1,0}^2 - 2Tc_{-1,0} + 1,$$

for some constant K_T.

We now examine the terms in the sum within the expectation. We want to show that $E_1 = E_2$ where

$$E_1 \coloneqq \{(\alpha_1,\alpha_2,\beta_1,\beta_2)|\alpha_1-\alpha_2=\beta_1-\beta_2,(\alpha_1,\alpha_2,\beta_1,\beta_2)\neq(-1,-1,0,0)\}$$

and

$$E_2 \coloneqq \{(\alpha_1,\alpha_2,\beta_1,\beta_2)|\alpha_1-\alpha_2=\beta_1-\beta_2,(\alpha_1,\beta_1)\neq(-1,0),(\alpha_2,\beta_2)\neq(-1,0)\}.$$

We already have $E_2 \subset E_1$. If $(\alpha_1,\alpha_2,\beta_1,\beta_2) \in E_1\backslash E_2$, then either $(\alpha_1,\beta_1) = (-1,0)$ or $(\alpha_2,\beta_2) = (-1,0)$. If $(\alpha_1,\beta_1) = (-1,0)$, since $\alpha_1-\alpha_2 = \beta_1-\beta_2$, then $\alpha_2 = \beta_2 - 1$, which necessarily implies $\beta_2 = 0$ and $\alpha_2 = -1$, which contradicts the fact that $(\alpha_1,\alpha_2,\beta_1,\beta_2) \in E_1$. Similarly, if $(\alpha_2,\beta_2) = (-1,0)$ and $\alpha_1-\alpha_2 = \beta_1-\beta_2$, then $\beta_1 = 0$ and $\alpha_1 = -1$. Therefore, $E_1 = E_2$, and

$$\mathbb{E}_\lambda(\sum_{t_1,j_1,t_2,j_2,(\alpha_1,\alpha_2,\beta_1,\beta_2)\in E_1} c_{\alpha_1,\beta_1}c_{\alpha_2,\beta_2}\lambda_{j_1}^{T-t_1-\alpha_1}\lambda_{j_2}^{-T+t_2+\alpha_2}\lambda_i^{t_1-t_2+\beta_1-\beta_2}) =$$

$$\mathbb{E}_\lambda(\sum_{t_1,j_1,t_2,j_2,(\alpha_1,\alpha_2,\beta_1,\beta_2)\in E_2} c_{\alpha_1,\beta_1}c_{\alpha_2,\beta_2}\lambda_{j_1}^{T-t_1-\alpha_1}\lambda_{j_2}^{-T+t_2+\alpha_2}\lambda_i^{t_1-t_2+\beta_1-\beta_2}) =$$

$$\mathbb{E}_\lambda(|\sum_{t,j,\alpha\in\mathcal{A},\beta\in\mathcal{B},(\alpha,\beta)\neq(-1,0)} c_{\alpha,\beta}\lambda_j^{T-t-\alpha}\lambda_i^{t+\beta-1}|^2) \geq 0.$$

We are interested in the minimum of (10). The minimum of $K_T c_{-1,0}^2 - 2Tc_{-1,0} + 1$ is reached for $c_{-1,0} \neq 0$. We also just showed that the first term is non-negative, so the minimum will be reached when (almost surely in $\lambda \sim \mathcal{U}(\mathcal{C}_U)$)

$$\sum_{t,j,\alpha\in\mathcal{A},\beta\in\mathcal{B},(\alpha,\beta)\neq(-1,0)} c_{\alpha,\beta}\lambda_j^{T-t-\alpha}\lambda_i^{t+\beta-1} = 0.$$

In particular, the terms corresponding to monomials in λ_i are

$$\sum_{t,\alpha\in\mathcal{A},\beta\in\mathcal{B},(\alpha,\beta)\neq(-1,0)} c_{\alpha,\beta}\lambda_i^{T-\alpha+\beta-1} = (T\sum_{\alpha\in\mathcal{A},\beta\in\mathcal{B},(\alpha,\beta)\neq(-1,0)} c_{\alpha,\beta})\lambda_i^{T-\alpha+\beta-1} = 0.$$

Therefore, identifying the coefficients of this polynomial gives $c_{1,-1} = c_{-1,-1} + c_{0,0} = c_{1,0} + c_{0,-1} = 0$. We want to show that this implies $v_{-1} = u_1 = u_0 = 0$.

If $v_{-1} \neq 0$, then because $c_{1,-1} = 0$, we have $u_1 = 0$. From $c_{1,0} + c_{0,-1} = 0$, it follows $c_{0,-1} = 0$ and therefore $u_0 = 0$. From $c_{-1,-1} + c_{0,0} = 0$, this implies $c_{-1,-1} = 0$ and thus $u_{-1} = 0$, which contradicts $c_{-1,0} \neq 0$. Therefore, $v_{-1} = 0$.

Now, if $u_1 \neq 0$, then $v_{-1} = 0$, and from $c_{-1,-1} + c_{0,0} = 0$, we have $c_{0,0} = 0$. But because $c_{-1,0} \neq 0$, we have $u_0 = 0$, which combined with $c_{1,0} + c_{0,-1} = 0$ implies $c_{1,0} = 0$, which is impossible because $v_0 \neq 0$. Therefore, $u_1 = 0$.

So $u_1 = v_{-1} = 0$, and $c_{0,0} = 0$, so that $u_0 = 0$. This shows that loss (9) is minimal for $c_{\alpha,\beta} = 0$ if $(\alpha,\beta) \neq (-1,0)$.

Last, we need to calculate the constant

$$K_T \coloneqq \mathbb{E}_\lambda(\sum_{t_1,j_1,t_2,j_2} \lambda_{j_1}^{T-t_1+1}\lambda_{j_2}^{-T+t_2-1}\lambda_i^{t_1-t_2}).$$

Back to the different possible values for (j_1, j_2, i) analyzed above, we get non-zeros when $j_1 = j_2 = i$ and when $j_1 = j_2 \neq i$. In cases 3) and 4) we obtain as a necessary condition for non-zero expectation: $t_2 = T + 1$ or $t_1 = T + 1$; which is not possible. Therefore,

$$K_T = (\sum_{j_1=j_2=i} \sum_{t_1,t_2} 1) + (\sum_{j_1=j_2\neq i} \sum_{t_1=t_2} 1) = T^2 + (d-1)T.$$

We now denote $\eta = a_3 b_1 = c_{-1,0}$. Replacing the zero terms in loss (9), we obtain

$$d \sum_{T=2}^{T_{\max}} \left(\eta^2 (T^2 + (d-1)T) - 2\eta T + 1\right),$$

for which the argmin is given by

$$\eta^* = \frac{\sum_{T=2}^{T_{\max}} T}{\sum_{T=2}^{T_{\max}} (T^2 + (d-1)T)} \sim \frac{3}{2T_{\max}} \quad \text{as} \quad T_{\max} \to +\infty.$$

Finally at optimality,

$$\mathcal{T}_{\theta^*}(e_{1:T}) = \sum_t \eta^* s^\star_{t-1} s_T s_t = \eta^* (\sum_t s_t s^\star_{t-1}) s_T = -\eta^* \nabla_W L(0, e_{1:T}) s_T = \Gamma_{\theta^*}(e_{1:T}) s_T$$

with $\Gamma_{\theta^*}(e_{1:T}) \coloneqq -\eta^* \nabla_W L(0, e_{1:T})$. □

A.4. Proof of Lemma 2.

Proof. For a given input sequence $e_{1:T} = s_{1:T} = (s, \lambda \odot s, \cdots, \lambda^{T-1} \odot s)$, with context $\lambda \in \mathcal{U}^d$, one has

$$\mathcal{T}_\theta(e_{1:T}) = \sum_{t=1}^{T} P_{T-1,t} \sum_{h=1}^{H} \langle \lambda^{t-1} \odot s, a_h \odot \lambda^{T-2} \odot s \rangle b_h \odot \lambda^{t-1} \odot s.$$

We have

$$(\langle \lambda^{t-1} \odot s, a_h \odot \lambda^{T-2} \odot s \rangle b_h \odot \lambda^{t-1} \odot s)_i = \sum_{j=1}^{d} a_h^j \lambda_j^{2-T} \lambda_j^{t-1} b_h^i \lambda_i^{t-1}.$$

Since we precisely have

$$([\mathtt{B}^\top \mathtt{A}] \lambda^{t-T+1} \odot \lambda^{t-1})_i = \sum_{j=1}^{d} \sum_{h=1}^{H} b_h^i a_h^j \lambda_j^{t-T+1} \lambda_i^{t-1},$$

this gives us the desired result.

□

Remark. In fact, looking at the above proof, considering arbitrary real values for the vector $s_0 = s$, one can absorb the terms in s_j^2 in each a_h^j and the terms in s_i in each b_h^i, since these are learnable parameters. Therefore, our results can be adapted to any arbitrary initial value s_0.

A.5. Proof of Proposition 3.

Proof. Denote $\mathtt{C} = \mathtt{B}^\top \mathtt{A}$. Let us suppose that we have an optimal solution θ such that $l(\theta) = 0$. Therefore, for almost all $\lambda \in \mathcal{U}^d$ and $s_t = \lambda^{t-1}$, one has $\mathcal{T}_\theta(s_{1:T}) = \lambda^T$. Then, $\forall i \in \{1, \cdots, d\}$:

$$\sum_{t=1}^{T} P_{T-1,t} \sum_{j=1}^{d} \mathtt{C}_{ij} \lambda_j^{t-T+1} \lambda_i^{t-1} = \lambda_i^T.$$

By identifying the coefficients of the polynomial in the λ_i's, we see that one must have for all $T \geq 2$ that $P_{T-1,t} = 0$ if $t \neq T$, and for all $1 \leq i \leq d$, $p_{T-1,T} \mathtt{C}_{ii} = 1$, and $\mathtt{C}_{ij} = 0$ for $i \neq j$.

An optimal in-context mapping is then obtained by considering the forward rule of $\mathcal{T}_{\theta^*}$ for optimal parameters. One gets:

$$\mathcal{T}_{\theta^*}(e_{1:T}) = (\bar{e}_{T-1} \odot e_T) \odot (e_T).$$

□

A.6. Proof of Proposition 4.

Proof. The loss writes

$$\ell(\theta) = \sum_{T=2}^{T_{\max}} \mathbb{E}_{\text{diag}\,\lambda \sim \mathcal{W}} \| \sum_{t=1}^{T} P_{T-1,t} \mathtt{C} \lambda^{t-T+1} \odot \lambda^{t-1} - \lambda^T \|^2.$$

We compute the expectation of each term in the sum by first developing it.

One has

$$\| \sum_{t=1}^{T} P_{T-1,t} \mathtt{C} \lambda^{t-T+1} \odot \lambda^{t-1} - \lambda^T \|^2 = \sum_{t,t'} P_{T-1,t} P_{T-1,t'} \sum_{i,j,k} \mathtt{C}_{ij} \mathtt{C}_{ik} \lambda_j^{T-t-1} \lambda_k^{t'-T+1} \lambda_i^{t'-t}$$

$$-2\text{Real}(\sum_t P_{T-1,t} \sum_{i,j} \mathtt{C}_{ij} \lambda_j^{T-t-1} \lambda_i^{T-t+1}) + d.$$

Looking at the expectation of the first term

$$\sum_{t,t'} P_{T-1,t} P_{T-1,t'} \sum_{i,j,k} \mathtt{C}_{ij} \mathtt{C}_{ik} \mathbb{E}(\lambda_j^{T-t-1} \lambda_k^{t'-T+1} \lambda_i^{t'-t}),$$

we see that one has to calculate for t, t', i, j, k

$$\mathbb{E}(\lambda_j^{T-t-1} \lambda_k^{t'-T+1} \lambda_i^{t'-t}).$$

When $j \neq k$, it is $\delta_{t'=t=T-1}$. When $j = k$ it is $\delta_{t'=t}$.

Therefore

$$\sum_{t,t'} P_{T-1,t} P_{T-1,t'} \sum_{i,j,k} \mathtt{C}_{ij} \mathtt{C}_{ik} \mathbb{E}(\lambda_j^{T-t-1} \lambda_k^{t'-T+1} \lambda_i^{t'-t}) = \sum_{t=t'} P_{T-1,t}^2 \sum_{i,j} \mathtt{C}_{ij}^2 + P_{T-1,T-1}^2 \sum_{i,j,k} \mathtt{C}_{i,j} \mathtt{C}_{i,k} =$$

$$\|P_{T-1}\|^2 \|\mathtt{C}\|_F^2 + P_{T-1,T-1}^2 S(\mathtt{C}^\top \mathtt{C}).$$

Similarly, because $\mathbb{E}(\lambda_j^{T-t-1} \lambda_i^{T-t+1}) = \delta_{t=T,i=j}$, the second term is

$$-2\mathbb{E}(\text{Real}(\sum_t P_{T-1,t} \sum_{i,j} \mathtt{C}_{ij} \lambda_j^{T-t-1} \lambda_i^{T-t+1})) = -2P_{T-1,T} \text{Tr}(\mathtt{C}).$$

This concludes the proof. □

A.7. Proof of Proposition 5.

Proof. Let us denote $a(t)$, $b(t)$ and $p(t)$ the functions defined by the gradient flow on loss ℓ, that is $\dot{a} = -\nabla_a \ell(a, b, p)$ and similarly for b and p. We start with the result from Nguegnang et al. (2021); Achour et al. (2021), which states that $a(t)$, $b(t)$, and $p(t)$ are bounded and converge to a stationary point (a^*, b^*, p^*) as $t \to \infty$:

$$\lim_{t \to \infty} (a(t), b(t), p(t)) = (a^*, b^*, p^*).$$

The possible stationary points are either global or non-global minima. The conditions for these are:

- Global minimum if $p^*a^*b^* = 1$
- Non-global minimum if $a^*b^* = 0$, $a^*p^* = 0$ and $b^*p^* = 0$ and therefore

$$p^*a^*b^* = 0.$$

If the system were to converge to a non-global minimum, we would therefore have

$$\ell(+\infty) = |p^*a^*b^* - 1| = |0 - 1| = 1 > \ell(0)$$

This is a contradiction because the energy should have decreased over time.

As a result, the only remaining possibility for the stationary points is that they must satisfy $p^*a^*b^* = 1$. Therefore, the functions $a(t)$, $b(t)$, and $p(t)$ must converge to a global minimum. □

A.8. Proof of Lemma 3.

Proof. For the announced parameters, the problem simply decomposes into sub-problems in dimension 2. Indeed, we have for all $i \in \{1, \dots, \delta\}$:

$$(\mathcal{T}_{\theta^*}(e_{1:T}))_{2i-1} = -\frac{1}{2}(\lambda_{2i-1}^0 + \bar{\lambda}_{2i-1}^0)\lambda_{2i-1}^{T-2} + (\lambda_{2i-1}^1 + \bar{\lambda}_{2i-1}^1)\lambda_{2i-1}^{T-1} = \lambda_{2i-1}^{T-2}(-1 + \lambda_{2i-1}^2 + 1) = \lambda_{2i-1}^T.$$

Similarly,

$$(\mathcal{T}_{\theta^*}(e_{1:T}))_{2i} = \lambda_{2i}^T.$$

□

A.9. Proof of Proposition 6.

Proof. The proof is similar to Proposition A.5, by regrouping each λ_i with $\bar{\lambda}_i$ and identifying coefficients in two polynomials. More precisely, one must have

$$\sum_{t=1}^{T} P_{T-1,t} \sum_{j=1}^{d} \mathtt{C}_{2i-1,j} \lambda_j^{t-T+1} \lambda_{2i-1}^{t-1} = \lambda_{2i-1}^T.$$

Therefore, isolating terms in λ_{2i-1} (recall that $\lambda_{2i} = 1/\lambda_{2i-1}$), and developing, we get, noting $p \coloneqq P_{T-1}$:

$$\mathtt{C}_{2i-1,2i-1}(\sum_{t<T-1} p_t \lambda_{2i-1}^{2t-T} + p_{T-1}\lambda_{2i-1}^{T-2} + p_T \lambda_{2i-1}^T) + \sum_t p_t \mathtt{C}_{2i-1,2i} \lambda_{2i-1}^{T-2} = \lambda_{2i-1}^T.$$

Identifying gives $(\mathtt{C}_{2i-1,2i-1} + \mathtt{C}_{2i-1,2i})p_{T-1} + \mathtt{C}_{2i-1,2i}p_T = 0$, $\mathtt{C}_{2i-1,2i-1}p_T = 1$ and $p_{t<T-1} = 0$.

Similarly, on the conjugates, $(\mathtt{C}_{2i,2i} + \mathtt{C}_{2i,2i-1})p_{T-1} + \mathtt{C}_{2i,2i-1}p_T = 0$, $\mathtt{C}_{2i,2i}p_T = 1$.

Identifying the other terms gives $C_{2i,j} = C_{2i-1,j} = 0$ for $j \neq 2i$ and $j \neq 2i-1$. □

Interpretation. Up to rescaling, the relation $p_T C_{i,i} = 1$ gives $p_T = C_{i,i} = 1$, and therefore

$$C_{2i-1,2i} + (1 + C_{2i-1,2i})p_{T-1} = C_{2i,2i-1} + (1 + C_{2i,2i-1})p_{T-1} = 0, \tag{11}$$

which gives $C_{2i-1,2i} = C_{2i,2i-1}$. Therefore, C is symmetric and has the form $\mathrm{diag}(J_b, \dots, J_b)$ with $J_b = ((1, b), (b, 1))$ and $b = -p_{T-1}/(1 + p_{T-1})$. If $H < d$, C cannot be full rank, and therefore necessarily $C_{2i,2i-1} = 1$ or -1. But $C_{2i,2i-1} = -1$ is impossible given (11). We then recover Lemma 3.

A.10. Proof of Proposition 7

Proof. One has that the second order term in the quadratic form (7) writes

$$\langle p|Hp\rangle = \sum_{t,t'} p_t p_{t'} \mathbb{E}_{\lambda\sim\mathcal{W}(\mu)} \lambda^{2(t'-t)}.$$

We can calculate this expectation in closed form. Writing $\alpha = \frac{2\pi}{\mu}$, it gives

$$\mathbb{E}_{\lambda\sim\mathcal{W}(\mu)} \lambda^{2(t'-t)} = \frac{1}{\alpha}\int_0^{\alpha} e^{2i\theta(t'-t)} d\theta,$$

which gives $\frac{i}{2\alpha(t'-t)}[1 - e^{2i(t'-t)\alpha}]$ if $t \neq t'$ and 1 otherwise. Since $\langle p|Hp\rangle$ is real, we can identify the real parts so that

$$H_{t,t'} = (\mu/(4\pi(t'-t)))\sin(4(t'-t)\frac{\pi}{\mu}).$$

We now turn to the eigenvalues of H. H is a smooth function of α:

$$H : [0, 2\pi] \to S_T,$$

where S_T are the symmetric matrices of $\mathbb{R}^{T\times T}$. Using Th. 5.2 from Kato (2013), we know that the eigenvalues of H can be parametrized as continuous functions $\nu_1(\alpha) \geq \nu_2(\alpha) \geq \cdots \nu_T(\alpha)$. Since for $\alpha = 0$, the eigenvalues of H are $T, 0, \cdots 0$, and recalling that $\mu = \frac{2\pi}{\alpha}$, we obtain the result.

□

B. Additional Experiments

Effect of the addition of $\mathrm{softmax}$ **layers and MLP layers on the trained solutions.** We consider the unitary context matrix setup of section 5. As mentionned, we could not find a natural way to express the global minimum of the training loss when a $\mathrm{softmax}$ layer was involved, even in dimension 1. To get more insight, we conducted an additional experiment where we trained different models with and without $\mathrm{softmax}$ and MLP layers. We used $d = 1$, and $T = 10$, and a hidden dimension of 32 in the MLP. Results are shown in Figure 9, where it is clear that in the case of commuting context matrices, using a $\mathrm{softmax}$ is incompatible with learning the underlying in-context mapping.

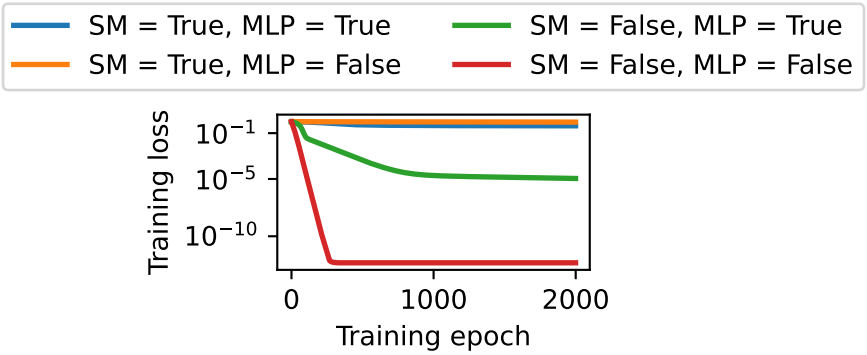

Figure 9. Training loss with training epoch for different configurations of the Transformer's architecture, when using a $\mathrm{softmax}$ (SM) and an MLP or not.

Transformer with all the bells and whistles and gradient descent. The experiment displayed in Figure 5 shows that linear attention fails to compete with gradient descent when there are more than 2 layers. In contrast, a full Transformer with all the bells and whistles as described in Vaswani et al. (2017) ($\mathrm{softmax}$ and MLP applied component-wise to each transformer layer) outperforms gradient descent and has a similar trend, as shown in Figure 10. The training procedure and the dataset are identical to those described in the *Augmented setting* paragraph of Section 6.

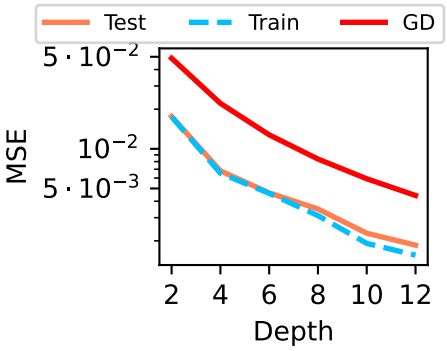

Figure 10. **Evolution** of the mean squared errors (MSE) with depth L for a vanilla Transformer (Vaswani et al., 2017). We compare with L steps of gradient descent (GD) on the inner loss (5). At initialization, the MSE is between 1 and 2. .