TRANSFORMER OPERATORS PERFORM IN-CONTEXT LEARNING BY OPERATOR GRADIENT DESCENT

Abhiti Mishra*, **Yash Patel***, **& Ambuj Tewari**
Department of Statistics
University of Michigan
Ann Arbor, MI 48104, USA
{abhiti,yppatel,tewaria}@umich.edu

ABSTRACT

Neural operator surrogate models are becoming increasingly popular for material design, where they are used to rapidly evaluate candidate designs. Such surrogate models, however, most commonly either amortize the flow map across initial conditions for a fixed design parameter or vice versa. Towards this end, recent interest has emerged in meta-operator learning to allow for simultaneous variation of both, in which a meta-learner is trained to approximate the flow map across design parameters and then fine-tuned for the particular system of interest. Increasing interest is being placed on performing such fine-tuning via in-context learning. While impressive empirical performance has been achieved with function in-context learning, a mechanistic explanation of how such a behavior emerges has yet to be provided. We, thus, here demonstrate that functional in-context learning, when performed by transformer operators, is achieved by gradient descent in an operator RKHS.

1 INTRODUCTION

The evaluation of material designs often requires the solution of PDEs to compute properties of interest, such as elasticity or fracture Arruda & Boyce (1993); Gent (1996); Humphrey (2003). Exploration of the design space then becomes limited by the computational cost of solving such PDEs, which must be performed using numerical solvers Wilson et al. (2022); Winkel et al. (2012); Arber et al. (2015); Tskhakaya et al. (2007); Ning et al. (2023). In particular, numerical solution techniques fail to amortize computational cost across problem instances, rendering them overly expensive Biegler et al. (2003); De los Reyes (2015). For this reason, machine learning methods, known as "neural operators," have become increasingly popular avenues of investigation Jafarzadeh et al. (2024); Oommen et al. (2024); You et al. (2022). Such approaches amortize the cost of solution across either initial conditions or "system parameters," which are parametric quantities that characterize the PDE and are often the free parameters of design loops, such as the conductivity κ of a material in the heat equation (Hsu, 1994; Challis & Guest, 2009; Dunning & Kim, 2015).

Current methods in the space of operator learning, however, only allow for amortization over one of these two modalities. That is, methods learning operator flow maps across initial conditions fix the system parameters and those across system parameters the initial conditions Li et al. (2020b;c;a); Du et al. (2023); Liu et al. (2023). Neural operators are fundamentally unsuited to learning flow maps across multiple system parameters, as the true solution operator then varies across samples. Towards this end, significant work has emerged on neural operator meta-learning. In one branch of this work, PDE foundation models are pre-trained and subsequently fine-tuned on the particular systems of interest Cao et al. (2024a); Shen et al. (2024); Herde et al. (2024); Liu et al. (2024b). The need to perform fine-tuning, however, degrades the computational benefits of surrogate modeling.

In a seemingly unrelated vein, the fine-tuning of LLMs is a similarly remarkably expensive task, owing to their tremendous parameter counts Minaee et al. (2024). It has there been observed that LLMs, and transformers more broadly, display remarkable capacities to be fine-tuned in-context,

*Equal contributions

in which "training pairs" are placed into the context window with no weights of the model being updated Dong et al. (2022). For this reason, much interest has emerged in extending the standard transformer architecture to enable in-context learning in the context of operator learning Cao et al. (2024b); Yang et al. (2023); Meng et al. (2025); Yang & Osher (2024). Despite significant empirical validation, such operator in-context learning have yet to be mechanistically understood.

Recent theoretical work for standard, finite-dimensional transformers suggest that they are capable of performing in-context learning by implicitly implementing optimization algorithms in their forward passes Akyürek et al. (2022); Ahn et al. (2023); Von Oswald et al. (2023); Cheng et al. (2023). We, therefore, herein extend this line of theoretical inquiry to characterize the generalized transformer architectures that have been leveraged for in-context operator learning Calvello et al. (2024). In particular, we prove the generalized transformers leveraged for in-context operator learning do so by performing gradient descent in a Reproducing Kernel Hilbert Space over operators.

2 BACKGROUND

2.1 NEURAL OPERATORS

Neural operator methods, while more broadly applicable, most often seek to amortize the solution of a spatiotemporal PDE. Such PDEs are formally described by spatial fields that evolve over time, namely some $u : \Omega \times [0, \infty) \to \mathbb{R}$, where $x \in \Omega \subset \mathbb{R}^d$ are spatial coordinates and $t \in [0, \infty)$ is a time coordinate. While the more abstract operator learning framework can be formulated as seeking to learn a map $\widehat{\mathcal{G}} : \mathcal{A} \to \mathcal{U}$ between two function spaces $\mathcal{A}$ and $\mathcal{U}$, we are most often interested in learning time-rollout maps, in which the input and output function spaces are identical. In such cases, it is assumed a dataset of the form $\mathcal{D} := \{(u_i^{(0)}, u_i^{(T)})\}$ is available, where $u^{(0)} : \Omega \to \mathbb{R}$ is the initial condition, for which there exists some true operator $\mathcal{G}$ such that, for all i, $u_i^{(T)} = \mathcal{G}(u_i^{(0)})$.

While many different learning-based approaches have been proposed to solve this operator learning problem, they all can be abstractly framed as seeking

$$\min_{\widehat{\mathcal{G}}} ||\widehat{\mathcal{G}} - \mathcal{G}||^2_{\mathcal{L}^2(\mathcal{U},\mathcal{U})} = \int_{\mathcal{U}} ||\widehat{\mathcal{G}}(u_0) - \mathcal{G}(u_0)||^2_{\mathcal{U}} \, du_0. \tag{1}$$

Notably, in such works, it is often assumed that the operator of interest is mapping between two sufficiently smooth spaces, on which the PDE is well-defined Kovachki et al. (2024); Boullé & Townsend (2023). That is, most often $\mathcal{A} = \mathcal{U} = \mathcal{H}^s(\Omega)$, where $\mathcal{H}^s(\Omega) := \{f : ||f||_{\mathcal{H}^s(\Omega)} < \infty\}$ is the standard s-Sobolev space, in which

$$||f||_{\mathcal{H}^s(\Omega)} := \sum_{|\alpha| \leq k : \alpha \in \mathbb{N}^d} ||D^\alpha f||_{\mathcal{L}^2(\Omega)}.$$

s is the smoothness parameter as determined by the PDE, and α is the condensed notation for the corresponding mixed partials, i.e. $\alpha = (\alpha_1, ..., \alpha_d)$ corresponds to $D^\alpha := \partial_{x_1}^{\alpha_1} ... \partial_{x_d}^{\alpha_d}$.

Within this broad family of operator learning, the most widely employed classes are the Deep Operator Networks (DeepONets) Lu et al. (2021; 2019); Wang et al. (2021); Kopaničáková & Karniadakis (2025) and Fourier Neural Operators (FNOs) Li et al. (2020b;c;a); Bonev et al. (2023). FNOs are parameterized as a sequence of layers of linear operators, given as kernel integral transforms, with standard intermediate ReLU nonlinearities. Formally, a single layer is then given by

$$(\mathcal{G}_\ell u)(x) = \sigma\left(\int_\Omega k_\ell(x, y)u(y)dy\right) = \sigma\left(\mathcal{F}^{-1}(R_\ell \odot \mathcal{F}(u))\right), \tag{2}$$

where the latter equivalence follows from the standard convolution theorem under the assumption the learned kernel is translation invariant, i.e. $k_\ell(x, y) = k_\ell(x - y)$, where $\mathcal{F}$ denotes the Fourier transform. Due to improved computational efficiency, therefore, the learning of the kernel is done in Fourier rather than real space, where the kernel is learned up to some fixed truncation point $k_{\max}$.

2.2 TRANSFORMERS AND IN-CONTEXT LEARNING

The attention mechanism is parameterized by $\theta := \{W_k, W_q, W_v\}$, where $W_k \in \mathbb{R}^{d_k \times d}$, $W_q \in \mathbb{R}^{d_q \times d}$, and $W_v \in \mathbb{R}^{d_v \times d}$ for sequential data $X \in \mathbb{R}^{d \times T}$ Vaswani (2017). In a generalized form,

$$\text{Attn}(X) := (W_v X) MH\left((W_q X), (W_k X)\right), \tag{3}$$

where $M \in \mathbb{R}^{T\times T}$ is a masking matrix and $H : \mathbb{R}^{d_q\times T} \times \mathbb{R}^{d_k\times T} \to \mathbb{R}^{T\times T}$ is a nonlinear transform of key-query similarity measures, where $[H(Q,K)]_{i,j} = h(Q^{(i)}, K^{(j)})$ with $Q^{(i)} \in \mathbb{R}^{d_q}$ and $K^{(j)} \in \mathbb{R}^{d_k}$. In practice, most often $d_k = d_q$ and $h := \mathrm{softmax}$ Cheng et al. (2023). Transformer architectures are then simply repeated compositions of such attention blocks with residual connections and feedforward and normalization layers.

We additionally highlight the extension of the attention mechanism proposed in Calvello et al. (2024) that permits its use for operator learning; such an extension is referred to as "continuum attention." In particular, in place of $x_i \in \mathbb{R}^d$, we assume $x_i \in \mathcal{X}$ for some Hilbert space $\mathcal{X}$; notably, the presentation of Calvello et al. (2024) was particular to $\mathcal{X} = \mathcal{F}(\Omega;\mathbb{R}^d)$, i.e. the set of all functions $x : \Omega \to \mathbb{R}^d$. The natural generalization of the attention mechanism for function inputs then replaces the W_k, W_q, and W_v matrices with corresponding linear operators, denoted as $\mathcal{W}_k : \mathcal{X} \to \mathcal{K}$, $\mathcal{W}_q : \mathcal{X} \to \mathcal{Q}$, and $\mathcal{W}_v : \mathcal{X} \to \mathcal{V}$, where $\mathcal{X}, \mathcal{K}, \mathcal{Q}$, and $\mathcal{V}$ are Hilbert spaces. In practice, such operators are implemented as kernel integral transforms in the way of FNOs, as

$$\mathcal{W}_q x = \mathcal{F}^{-1}(R_q \odot \mathcal{F}x), \tag{4}$$

where R_q is the Fourier parameterization of the query kernel, with R_k and R_v similarly defined for the key and value kernels. Thus, the continuum attention mechanism assumes $\mathcal{K} = \mathcal{Q}$ and defines

$$\mathrm{ContAttn}(X) := (\mathcal{W}_v X) M \,\mathrm{softmax}\left((\mathcal{W}_q X), (\mathcal{W}_k X)\right) \tag{5}$$

where the interpretation of M remains the same as that in Equation (3). In particular, while the key, query, and value operators need to be generalized, the resulting attention weights matrix still lies in $\mathbb{R}^{T\times T}$, meaning the M component remains identical to its form in the finite-dimensional case.

Transformer architectures, most notably in LLMs, have been observed to exhibit an unexpected behavior known as "in-context learning" (ICL), in which they perform few-shot learning without any explicit parameter updates but merely by having training examples in their context windows. We follow the conventions of Cheng et al. (2023) in formalizing the ICL phenomenon. We suppose the dataset $\mathcal{D} := \{(X_i, y_i)\}_{i=1}^n$ on which the transformer was trained has samples of the form

$$X_i = \begin{bmatrix} x_i^{(1)} & x_i^{(2)} & \dots & x_i^{(T)} & x_i^{(T+1)} \\ y_i^{(1)} & y_i^{(2)} & \dots & y_i^{(T)} & 0 \end{bmatrix} \qquad y_i = y_i^{(T+1)}. \tag{6}$$

We then suppose $y_i^{(t)} = f_i(x_i^{(t)})$, where $f_i \neq f_j$ for $i \neq j$. This is the critical difference in the in-context learning setting versus typical learning settings: in the standard setting, $f_i = f$ is fixed across samples and necessarily matches the f' present at test time, meaning the goal for the learner is to merely learn such an f. In in-context learning, however, the learning algorithm must be capable of "learning" an unseen f' at inference time without parameter updates.

2.3 Operator Meta-Learning

Significant interest has emerged in meta-operator learning for spatiotemporal PDEs, in which a single network maps from initial conditions to final states across system specifications Wang et al. (2022); Zhang (2024); Sun et al. (2024b); Liu et al. (2024b); Cao et al. (2024b); Chakraborty et al. (2022); Sun et al. (2024a); Yang et al. (2023). Formally, unlike the traditional setting discussed in Section 2.1, here the dataset consists of pairs $\mathcal{D} := \{(u_0^{(i)}, u_T^{(i)})\}_{i=1}^N$, for which the true operator can vary across samples, i.e. $\mathcal{G}_i$ satisfies $u_i^{(T)} = \mathcal{G}_i(u_i^{(0)})$ but $\mathcal{G}_i \neq \mathcal{G}_j$ for $i \neq j$. The goal is to then, given only a limited number of training samples $\mathcal{D}' := \{(u_0', u_T')\}_{i=1}^{N'}$ with $N' \ll N$ for a fixed, potentially unseen operator $\mathcal{G}'$, learn an approximation $\widehat{\mathcal{G}}' \approx \mathcal{G}'$. Most often, this is done by pre-training a meta-learner $\mathcal{G}_{\mathrm{ML}}$ on $\mathcal{D}$ and then fine-tuning $\mathcal{G}_{\mathrm{ML}}$ on $\mathcal{D}'$ to arrive at $\mathcal{G}'$.

There are two primary settings in which this approach is often employed. In the first, meta-learning is sought across different parametric PDEs, i.e. where sample i may be the solution of the Navier Stokes equation and sample j of the shallow-water equations; meta-learners in this context are referred to as "foundation models" Sun et al. (2024a); Herde et al. (2024); Zhang (2024); Sun et al. (2024b); Liu et al. (2024a); McCabe et al. (2024). In the second setup, pre-training is instead sought over a fixed parametric PDE, where meta-learning happens across some system parameter, i.e. where samples i and j are both solutions of the Navier Stokes equation but for different Reynolds numbers $\nu_i \neq \nu_j$ Cao et al. (2024b); Yang et al. (2023); Meng et al. (2025); Yang & Osher (2024).

Building off of the observed ICL of LLMs and transformers more broadly, the fine-tuning of these meta-learners too can be performed via explicit weight modification or via in-context learning. Approaches in the first setup nearly all require fine tuning, likely owing to the greater variability of solution operators across PDEs. In the latter, however, in-context learning is the dominant paradigm, although some works still perform explicit weight modification Rahman et al. (2024). In fact, an entire offshoot of meta-learning known as in-context operator networks (ICONs) has spawned from this approach Cao et al. (2024b); Yang et al. (2023); Meng et al. (2025); Yang & Osher (2024).

Recent works in this vein have demonstrated notable empirical performance leveraging bespoke transformer architectures built atop the continuum attention from Equation (5) Cao et al. (2024b); Alkin et al. (2024); Cao et al. (2025). Similar to Equation (6), it is assumed that a pre-training dataset of the form $\mathcal{D} := \{(U_i^{(0:T-1)}, U_i^{(T)})\}$ is available, where $U_i^{(0:T)} := [u_i^{(0)}, u_i^{(\Delta T)}, ..., u_i^{((T-1)\Delta T)}]$, with $u_i^{(t)} \in \mathcal{L}^2(\Omega)$. Notably, such a setup is equivalent to having $n = T/\Delta T$ training pairs $\{(u_i^{((t-1)\Delta T)}, u_i^{t\Delta T})\}_{t=1}^n$. It is further assumed that, for each sample i, there is a true, deterministic solution operator $\mathcal{G}_i \in \mathcal{O}$ that maps from the spatial field at some time t to its state at some later $t + \Delta T$. The in-context learning goal, therefore, is, given a new sequence of $(U^{(0:T-1)})'$ generated by some unseen operator $\mathcal{G}'$, predict $(U^{(T)})'$. Such behavior has been empirically displayed by proposed methods but has yet to be mechanistically understood.

2.4 Related Works

We are interested herein in providing a mechanistic explanation of the in-context learning empirically exhibited by continuum attention-based ICONs. Such mechanistic explanations have been studied extensively for finite-dimensional transformer architectures Akyürek et al. (2022); Garg et al. (2022); Dai et al. (2022). Most relevant in this line of work is that of Cheng et al. (2023). Loosely speaking, they demonstrated that, if a kernel $\kappa(\cdot, \cdot)$ is defined with $\mathbb{H}$ denoting its associated RKHS and a transformer is then defined with a specific W_k, W_q, W_v and $h(q, k) = \kappa(q, k)$ from Equation (3), that an inference pass through such a layer is equivalent to a single step of functional gradient descent in $\mathbb{H}$. They additionally demonstrated such a learned predictor is Bayes optimal under particular circumstances. These results are formally provided in Appendix A. Notably, however, these results require non-trivial changes to be generalized to function inputs.

While previous works have yet to formally provide a generalized mechanistic explanation, a recent work Cole et al. (2024) began a line of inquiry into formally characterizing the functional ICL phenomenon exhibited by continuum attention. This work, however, studied a fundamentally distinct aspect of functional ICL than that studied herein, characterizing its sample complexity and studying its resulting generalization, specifically in the restricted setting of linear elliptic PDEs.

3 Method

3.1 In-Context Learning Formalism

We now consider a generalization of the continuum attention, paralleling that of Equation (3), allowing for more general key-query similarity measures. In particular, instead of restricting $\mathcal{Q} = \mathcal{K}$ as required for Equation (5) to be well-defined, we allow $H : \mathcal{Q}^{n+1} \times \mathcal{K}^{n+1} \to (\mathcal{L}(\mathcal{V}))^{(n+1)\times(n+1)}$, where $\mathcal{L}(\mathcal{V})$ denotes the set of bounded linear operators from $\mathcal{V}$ to $\mathcal{V}$. Notably, this subsumes softmax if for $c \in \mathbb{R}$, $c \in \mathcal{L}(V)$ is understood in its natural sense to define cf_v for $f_v \in \mathcal{V}$ as $(cf_v)(x) := cf_v(x)$, from which we can view $\mathrm{softmax} : \mathcal{Q} \times \mathcal{K} \to \mathcal{L}(\mathcal{V})$. Notably, H acts component-wise as before, i.e. $[H(Q, K)]_{i,j} = h(Q^{(i)}, K^{(j)})$ for $h : \mathcal{Q} \times \mathcal{K} \to \mathcal{L}(\mathcal{V})$.

If $\mathcal{V} = \mathcal{X}$, an L-layer transformer operator $\mathcal{T} : \mathcal{X} \to \mathcal{X}$ consisting of generalized continuum attention layers with residual connections is well-defined as $\mathcal{T} := \mathcal{T}_L \circ ... \circ \mathcal{T}_0$, where

$$X_{\ell+1} = \mathcal{T}_\ell(X_\ell) := X_\ell + \left(H(\mathcal{W}_{q,\ell}X_\ell, \mathcal{W}_{k,\ell}X_\ell)(\mathcal{W}_{v,\ell}X_\ell M)^T\right)^T. \tag{7}$$

Following the formalization established in Cole et al. (2024), we seek to learn in-context a map $\mathcal{X} \to \mathcal{Y}$, for which we construct a context window consisting of training pairs $z^{(i)} \in \mathcal{Z} = \mathcal{X} \otimes \mathcal{Y}$ as

$$Z_0 = \begin{pmatrix} f^{(1)} & f^{(2)} & \dots & f^{(n)} & f^{(n+1)} \\ u^{(1)} & u^{(2)} & \dots & u^{(n)} & 0 \end{pmatrix} \in \begin{pmatrix} \mathcal{X}^{n+1} \\ \mathcal{Y}^{n+1} \end{pmatrix},$$

where $\{(f^{(i)}, u^{(i)})\}_{i=1}^n$ are n pairs of input-output functions, $z^{(i)} = (f^{(i)}, u^{(i)})$, and $\mathcal{T} : \mathcal{Z}^{n+1} \to \mathcal{Z}^{n+1}$. We predict $u^{(n+1)}$ as $-[\mathcal{T}(z)]_{2,n+1}$. While the typical use case assumes $(f^{(i)}, u^{(i)}) = (u^{((i-1)\Delta t)}, u^{(i\Delta t)})$ for some time increment Δt, we present our results in this more abstract light, as they hold for more general applications. Notably, as in the typical formalization, the key and query operators and nonlinearity operator are assumed to only act upon the space of input functions. That is, denoting $X_0 = [f^{(1)}, f^{(2)}, \ldots, f^{(n)}, f^{(n+1)}] \in \mathcal{X}^{n+1}$ as the vector of input functions and $X_\ell \in \mathcal{X}^{n+1}$ as the first row of the matrix Z_ℓ,

$$Z_{\ell+1} = Z_\ell + \left(\widetilde{H}(\mathcal{W}_{q,\ell}X_\ell, \mathcal{W}_{k,\ell}X_\ell)(\mathcal{W}_{v,\ell}Z_\ell M)^T\right)^T. \tag{8}$$

Notably, the layer ℓ query and key block operators $\mathcal{W}_{q,\ell} : \mathcal{X}^{n+1} \to \mathcal{Q}^{n+1}$ and $\mathcal{W}_{k,\ell} : \mathcal{X}^{n+1} \to \mathcal{K}^{n+1}$ only act on the restricted input space, whereas $\mathcal{W}_{v,\ell} : \mathcal{Z}^{n+1} \to \mathcal{Z}^{n+1}$ acts on the full space. The mask block operator matrix $M \in \mathcal{O}^{2\times 2}$ is set as $\begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix}$. We also denote

$$\mathcal{T}_\ell(f; (\mathcal{W}_v, \mathcal{W}_q, \mathcal{W}_k)|z^{(1)}, \ldots, z^{(n)}) := Z_\ell \tag{9}$$

obtained from setting the last entry of the first row of Z_0 as f. Note that when the input passes through an attention layer as defined in 8, we are assuming that multiplication makes sense in the space $\mathcal{Y}$. This is a stronger structure imposed on $\mathcal{Y}$ than just a Hilbert Space. Usual choices of $\mathcal{Y}$, like $\mathcal{L}^2(\mathbb{R}^d)$ have a natural multiplicative structure, namely, pointwise multiplication of functions.

We state the formal theorem that the transformer operator architecture *can* perform in-context learning by implementing operator gradient descent below. The formalism used for operators and function-valued RKHS has been adopted from Kadri et al. (2016). We defer relevant definitions and the full proof of the theorem to Appendix C. We, however, here highlight novelties of the proof strategy in demonstrating this result. In particular, the explicit calculation of the gradient descent expression over operator spaces is not as directly evident as it is over finite-dimensional vector spaces. In addition, we had to invoke a generalized form of the Representer Theorem as provided in Stepaniants (2023) in place of the more classical statement leveraged in Cheng et al. (2023).

Theorem 3.1. *Let $\kappa : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ be an arbitrary operator-valued kernel. Let $\mathcal{O}$ denote the operator RKHS induced by κ. We are given in-context examples of the form $(f^{(i)}, u^{(i)})$ for $i = 1, \ldots, n$. The empirical loss functional $L : \mathcal{O} \to \mathbb{R}$ is given by*

$$L(O) = \sum_{i=1}^{n} \|u^{(i)} - Of^{(i)}\|_{\mathcal{Y}}^2.$$

Let $O_0 = \mathbf{0}$ and let O_ℓ denote the operator obtained from the l-th operator-valued gradient descent sequence of L with respect to $\|\cdot\|_{\mathcal{O}}$ as defined in Equation (16)*. Then there exist scalar step sizes $r'_0, \ldots, r'_k$ such that the following holds-*

Let $[\widetilde{H}(U, W)]_{i,j} = \kappa(u^{(i)}, w^{(j)})$. Let $\mathcal{W}_{v,\ell} = \begin{pmatrix} 0 & 0 \\ 0 & -r'_\ell I \end{pmatrix}$, $\mathcal{W}_{q,\ell} = I$, $\mathcal{W}_{k,\ell} = I$. Then for any $f^{(n+1)}$, the transformer operator's prediction for $u^{(n+1)}$ at each layer ℓ matches the prediction of the operator-valued gradient descent sequence at step ℓ, that is

$$[\mathcal{T}_\ell(f; (\mathcal{W}_v, \mathcal{W}_q, \mathcal{W}_k)|z^{(1)}, \ldots, z^{(n)})]_{2,n+1} = -O_\ell f.$$

4 Discussion

In this paper, we provided a mechanistic explanation of the in-context learning phenomenon exploited in continuum transformer-based operator networks. Moving forward in this work, we would like to extend more of the in-context learning results from Cheng et al. (2023) to the functional setting. In particular, we would like to demonstrate that the specified key, query, and value operators giving rise to Theorem 3.1 are stationary points of in-context loss so as to demonstrate that the transformer operator, over the course of training, may recover the operators necessary to perform operator valued gradient descent. We additionally wish to empirically validate the theoretical claim demonstrated herein and those that we wish to prove subsequently.

REFERENCES

Kwangjun Ahn, Xiang Cheng, Hadi Daneshmand, and Suvrit Sra. Transformers learn to implement preconditioned gradient descent for in-context learning. *Advances in Neural Information Processing Systems*, 36:45614–45650, 2023.

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. *arXiv preprint arXiv:2211.15661*, 2022.

Benedikt Alkin, Andreas Fürst, Simon Schmid, Lukas Gruber, Markus Holzleitner, and Johannes Brandstetter. Universal physics transformers: A framework for efficiently scaling neural operators. *Advances in Neural Information Processing Systems*, 37:25152–25194, 2024.

TD Arber, Keith Bennett, CS Brady, A Lawrence-Douglas, MG Ramsay, Nathan John Sircombe, Paddy Gillies, RG Evans, Holger Schmitz, AR Bell, et al. Contemporary particle-in-cell approach to laser-plasma modelling. *Plasma Physics and Controlled Fusion*, 57(11):113001, 2015.

Ellen M Arruda and Mary C Boyce. A three-dimensional constitutive model for the large stretch behavior of rubber elastic materials. *Journal of the Mechanics and Physics of Solids*, 41(2): 389–412, 1993.

Lorenz T Biegler, Omar Ghattas, Matthias Heinkenschloss, and Bart van Bloemen Waanders. Large-scale pde-constrained optimization: an introduction. In *Large-scale PDE-constrained optimization*, pp. 3–13. Springer, 2003.

Boris Bonev, Thorsten Kurth, Christian Hundt, Jaideep Pathak, Maximilian Baust, Karthik Kashinath, and Anima Anandkumar. Spherical fourier neural operators: Learning stable dynamics on the sphere. *arXiv preprint arXiv:2306.03838*, 2023.

Nicolas Boullé and Alex Townsend. A mathematical guide to operator learning. *arXiv preprint arXiv:2312.14688*, 2023.

Edoardo Calvello, Nikola B Kovachki, Matthew E Levine, and Andrew M Stuart. Continuum attention for neural operators. *arXiv preprint arXiv:2406.06486*, 2024.

Shuhao Cao, Francesco Brarda, Ruipeng Li, and Yuanzhe Xi. Spectral-refiner: Fine-tuning of accurate spatiotemporal neural operator for turbulent flows. *arXiv preprint arXiv:2405.17211*, 2024a.

Shuhao Cao, Francesco Brarda, Ruipeng Li, and Yuanzhe Xi. Spectral-refiner: Accurate fine-tuning of spatiotemporal fourier neural operator for turbulent flows. In *The Thirteenth International Conference on Learning Representations*, 2025.

Yadi Cao, Yuxuan Liu, Liu Yang, Rose Yu, Hayden Schaeffer, and Stanley Osher. Vicon: Vision in-context operator networks for multi-physics fluid dynamics prediction. *arXiv preprint arXiv:2411.16063*, 2024b.

Ayan Chakraborty, Cosmin Anitescu, Xiaoying Zhuang, and Timon Rabczuk. Domain adaptation based transfer learning approach for solving pdes on complex geometries. *Engineering with Computers*, 38(5):4569–4588, 2022.

Vivien J Challis and James K Guest. Level set topology optimization of fluids in stokes flow. *International journal for numerical methods in engineering*, 79(10):1284–1308, 2009.

Xiang Cheng, Yuxin Chen, and Suvrit Sra. Transformers implement functional gradient descent to learn non-linear functions in context. *arXiv preprint arXiv:2312.06528*, 2023.

Frank Cole, Yulong Lu, Riley O'Neill, and Tianhao Zhang. Provable in-context learning of linear systems and linear elliptic pdes with transformers. *arXiv preprint arXiv:2409.12293*, 2024.

Damai Dai, Yutao Sun, Li Dong, Yaru Hao, Shuming Ma, Zhifang Sui, and Furu Wei. Why can gpt learn in-context? language models implicitly perform gradient descent as meta-optimizers. *arXiv preprint arXiv:2212.10559*, 2022.

Juan Carlos De los Reyes. *Numerical PDE-constrained optimization*. Springer, 2015.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Tianyu Liu, et al. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*, 2022.

Yiheng Du, Nithin Chalapathi, and Aditi Krishnapriyan. Neural spectral methods: Self-supervised learning in the spectral domain. *arXiv preprint arXiv:2312.05225*, 2023.

Peter D Dunning and H Alicia Kim. Introducing the sequential linear programming level-set method for topology optimization. *Structural and Multidisciplinary Optimization*, 51:631–643, 2015.

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

Alan N Gent. A new constitutive relation for rubber. *Rubber chemistry and technology*, 69(1): 59–61, 1996.

Maximilian Herde, Bogdan Raonić, Tobias Rohner, Roger Käppeli, Roberto Molinaro, Emmanuel de Bézenac, and Siddhartha Mishra. Poseidon: Efficient foundation models for pdes. *arXiv preprint arXiv:2405.19101*, 2024.

Yeh-Liang Hsu. A review of structural shape optimization. *Computers in Industry*, 25(1):3–13, 1994.

Jay D Humphrey. Continuum biomechanics of soft biological tissues. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 459(2029):3–46, 2003.

Siavash Jafarzadeh, Stewart Silling, Ning Liu, Zhongqiang Zhang, and Yue Yu. Peridynamic neural operators: A data-driven nonlocal constitutive model for complex material responses. *Computer Methods in Applied Mechanics and Engineering*, 425:116914, 2024.

Hachem Kadri, Emmanuel Duflos, Philippe Preux, Stéphane Canu, Alain Rakotomamonjy, and Julien Audiffren. Operator-valued kernels for learning from functional response data. *Journal of Machine Learning Research*, 17(20):1–54, 2016. URL `http://jmlr.org/papers/v17/11-315.html`.

Alena Kopaničáková and George Em Karniadakis. Deeponet based preconditioning strategies for solving parametric linear systems of equations. *SIAM Journal on Scientific Computing*, 47(1): C151–C181, 2025.

Nikola B Kovachki, Samuel Lanthaler, and Andrew M Stuart. Operator learning: Algorithms and analysis. *arXiv preprint arXiv:2402.15715*, 2024.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020a.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Graph kernel network for partial differential equations. *arXiv preprint arXiv:2003.03485*, 2020b.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Andrew Stuart, Kaushik Bhattacharya, and Anima Anandkumar. Multipole graph neural operator for parametric partial differential equations. *Advances in Neural Information Processing Systems*, 33:6755–6766, 2020c.

Yuqiu Liu, Jingxuan Xu, Mauricio Soroco, Yunchao Wei, and Wuyang Chen. Data-efficient inference of neural fluid fields via sciml foundation model. *arXiv preprint arXiv:2412.13897*, 2024a.

Yuxuan Liu, Jingmin Sun, Xinjie He, Griffin Pinney, Zecheng Zhang, and Hayden Schaeffer. Prosefd: A multimodal pde foundation model for learning multiple operators for forecasting fluid dynamics. *arXiv preprint arXiv:2409.09811*, 2024b.

Ziyuan Liu, Yuhang Wu, Daniel Zhengyu Huang, Hong Zhang, Xu Qian, and Songhe Song. Spfno: Spectral operator learning for pdes with dirichlet and neumann boundary conditions. *arXiv preprint arXiv:2312.06980*, 2023.

Lu Lu, Pengzhan Jin, and George Em Karniadakis. Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *arXiv preprint arXiv:1910.03193*, 2019.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature machine intelligence*, 3(3):218–229, 2021.

Michael McCabe, Régaldo-Saint Blancard, Liam Parker, Ruben Ohana, Miles Cranmer, Alberto Bietti, Michael Eickenberg, Siavash Golkar, Geraud Krawezik, Francois Lanusse, et al. Multiple physics pretraining for spatiotemporal surrogate models. *Advances in Neural Information Processing Systems*, 37:119301–119335, 2024.

Tingwei Meng, Moritz Voss, Nils Detering, Giulio Farolfi, Stanley Osher, and Georg Menz. In-context operator learning for linear propagator models. *arXiv preprint arXiv:2501.15106*, 2025.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. Large language models: A survey. *arXiv preprint arXiv:2402.06196*, 2024.

Jianguo Ning, Ziyan Jin, and Xiangzhao Xu. A multigrid partition coupled eulerian–lagrangian method for fluid–solid interaction problems. *Physics of Fluids*, 35(9), 2023.

Vivek Oommen, Khemraj Shukla, Saaketh Desai, Rémi Dingreville, and George Em Karniadakis. Rethinking materials simulations: Blending direct numerical simulations with neural operators. *npj Computational Materials*, 10(1):145, 2024.

Md Ashiqur Rahman, Robert Joseph George, Mogab Elleithy, Daniel Leibovici, Zongyi Li, Boris Bonev, Colin White, Julius Berner, Raymond A Yeh, Jean Kossaifi, et al. Pretraining codomain attention neural operators for solving multiphysics pdes. *Advances in Neural Information Processing Systems*, 37:104035–104064, 2024.

Junhong Shen, Tanya Marwah, and Ameet Talwalkar. Ups: Towards foundation models for pde solving via cross-modal adaptation. *arXiv preprint arXiv:2403.07187*, 2024.

George Stepaniants. Learning partial differential equations in reproducing kernel hilbert spaces. *Journal of Machine Learning Research*, 24(86):1–72, 2023. URL `http://jmlr.org/papers/v24/21-1363.html`.

Jingmin Sun, Yuxuan Liu, Zecheng Zhang, and Hayden Schaeffer. Towards a foundation model for partial differential equations: Multi-operator learning and extrapolation. *arXiv preprint arXiv:2404.12355*, 2024a.

Jingmin Sun, Zecheng Zhang, and Hayden Schaeffer. Lemon: Learning to learn multi-operator networks. *arXiv preprint arXiv:2408.16168*, 2024b.

David Tskhakaya, Konstantin Matyash, Ralf Schneider, and Francesco Taccogna. The particle-in-cell method. *Contributions to Plasma Physics*, 47(8-9):563–594, 2007.

A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, pp. 35151–35174. PMLR, 2023.

Hengjie Wang, Robert Planas, Aparna Chandramowlishwaran, and Ramin Bostanabad. Mosaic flows: A transferable deep learning framework for solving pdes on unseen domains. *Computer Methods in Applied Mechanics and Engineering*, 389:114424, 2022.

Sifan Wang, Hanwen Wang, and Paris Perdikaris. Learning the solution operator of parametric partial differential equations with physics-informed deeponets. *Science advances*, 7(40):eabi8605, 2021.

Leighton Wilson, Robert Krasny, and Tyler Luchko. Accelerating the 3d reference interaction site model theory of molecular solvation with treecode summation and cut-offs. *Journal of Computational Chemistry*, 43(18):1251–1270, 2022.

Mathias Winkel, Robert Speck, Helge Hübner, Lukas Arnold, Rolf Krause, and Paul Gibbon. A massively parallel, multi-disciplinary barnes–hut tree code for extreme-scale n-body simulations. *Computer physics communications*, 183(4):880–889, 2012.

Liu Yang and Stanley J Osher. Pde generalization of in-context operator networks: A study on 1d scalar nonlinear conservation laws. *Journal of Computational Physics*, 519:113379, 2024.

Liu Yang, Siting Liu, Tingwei Meng, and Stanley J Osher. In-context operator learning with data prompts for differential equation problems. *Proceedings of the National Academy of Sciences*, 120(39):e2310142120, 2023.

Huaiqian You, Quinn Zhang, Colton J Ross, Chung-Hao Lee, and Yue Yu. Learning deep implicit fourier neural operators (ifnos) with applications to heterogeneous material modeling. *Computer Methods in Applied Mechanics and Engineering*, 398:115296, 2022.

Zecheng Zhang. Modno: Multi-operator learning with distributed neural operators. *Computer Methods in Applied Mechanics and Engineering*, 431:117229, 2024.

A RKHS FUNCTIONAL GRADIENT DESCENT THEOREMS

We provide here the precise statements of the relevant results from Cheng et al. (2023).

Proposition A.1. *(Proposition 1 from Cheng et al. (2023)) Let $\mathcal{K}$ be an arbitrary kernel. Let $\mathcal{H}$ denote the Reproducing Kernel Hilbert space induced by $\mathcal{K}$. Let $\mathbf{z}^{(i)} = (x^{(i)}, y^{(i)})$ for $i = 1, \ldots, n$ be an arbitrary set of in-context examples. Denote the empirical loss functional by*

$$L(f) := \sum_{i=1}^{n} \left(f(x^{(i)}) - y^{(i)} \right)^2. \tag{10}$$

Let $f_0 = 0$ and let f_ℓ denote the gradient descent sequence of L with respect to $\|\cdot\|_{\mathcal{H}}$, as defined in (3.1). Then there exist scalars stepsizes $r_0, \ldots, r_k$ such that the following holds:

Let $\widetilde{H}$ be the function defined as

$$\widetilde{H}(U, W)_{i,j} := \mathcal{K}(U^{(i)}, W^{(j)}), \tag{11}$$

where $U^{(i)}$ and $W^{(j)}$ denote the ith column of U and W respectively. Let

$$V_\ell = \begin{bmatrix} 0 & 0 \\ 0 & -r_\ell \end{bmatrix}, \quad B_\ell = I_{d\times d}, \quad C_\ell = I_{d\times d}. \tag{12}$$

Then for any $x := x^{(n+1)}$, the Transformer's prediction for $y^{(n+1)}$ at each layer ℓ matches the prediction of the functional gradient sequence (3.1) at step ℓ, i.e., for all $\ell = 0, \ldots, k$,

$$\mathcal{T}_\ell(x; (V, B, C)|z^{(1)}, \ldots, z^{(n)}) = -f_\ell(x). \tag{13}$$

Proposition A.2. *(Proposition 2 from Cheng et al. (2023)) Let*

$$X = \begin{bmatrix} x^{(1)}, & \ldots, & x^{(n+1)} \end{bmatrix}, \quad Y = \begin{bmatrix} y^{(1)}, & \ldots, & y^{(n+1)} \end{bmatrix}. \tag{14}$$

Let $\mathcal{K} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ be a kernel. Assume that $Y|X$ is drawn from the $\mathcal{K}$ Gaussian Process. Let the attention activation

$$\widetilde{H}(U, W)_{ij} := \mathcal{K}(U^{(i)}, W^{(j)}), \tag{15}$$

and consider the functional gradient descent construction in Proposition 1. Then, as the number of layers $\ell \to \infty$, the Transformer's prediction for $y^{(n+1)}$ at layer ℓ (2.4) approaches the Bayes (optimal) estimator that minimizes the in-context loss (2.5).

B GRADIENT DESCENT IN OPERATOR SPACE

We start by some defining notation that we will use in the next sections. We denote by $\mathcal{X} = \{x : D_X \to \mathbb{R}\}$ and $\mathcal{Y} = \{y : D_Y \to \mathbb{R}\}$ the separable Hilbert spaces in which our input and output functions lie in respectively. We denote by $\mathcal{C}(\mathcal{X}, \mathcal{Y})$ the space of continuous operators from $\mathcal{X}$ to $\mathcal{Y}$. Let $\mathcal{L}(\mathcal{Y})$ denote the set of bounded linear operators from $\mathcal{Y}$ to $\mathcal{Y}$.

We begin by defining gradient descent in operator space. Let $\mathcal{O}$ denote a space of bounded operators from $\mathcal{X}$ to $\mathcal{Y}$ equipped with the operator norm $\|\cdot\|_{\mathcal{O}}$. Let $L : \mathcal{O} \to \mathbb{R}$ denote a loss function. The gradient descent of L is defined as the sequence

$$O_{\ell+1} = O_\ell - r_\ell \nabla L(O_\ell) \tag{16}$$

where

$$\nabla L(O) = \operatorname*{arg\,min}_{G\in\mathcal{O}, \|G\|_{\mathcal{O}}=1} \frac{d}{dt} L(O + tG)\Big|_{t=0}.$$

Suppose we have n input-output function pairs as $f^{(1)}, \ldots, f^{(n)} \in \mathcal{X}$ and $u^{(1)}, \ldots, u^{(n)} \in \mathcal{Y}$ and we define L as the weighted empirical least-squares loss

$$L(O) = \sum_{i=1}^{n} \|u^{(i)} - O f^{(i)}\|_{\mathcal{Y}}^2.$$

Then $\nabla L(O)$ takes the form

$$\nabla L(O) = \underset{G \in \mathcal{O}, \|G\|_{\mathcal{O}}=1}{\arg\min} \frac{d}{dt} \sum_{i=1}^{n} \|u^{(i)} - (O + tG)f^{(i)}\|_{\mathcal{Y}}^2 \Big|_{t=0}. \tag{17}$$

For simplification, suppose that we denote by G^* the steepest descent direction. Then the method of Lagrange multipliers states that there exists some λ for which the problem in Equation (17) is equivalent to

$$G^* = \underset{G \in \mathcal{O}}{\arg\min} \frac{d}{dt} \sum_{i=1}^{n} \|u^{(i)} - (O + tG)f^{(i)}\|_{\mathcal{Y}}^2 \Big|_{t=0} + \lambda \|G\|_{\mathcal{B}(\mathcal{X},\mathcal{Y})}^2 \tag{18}$$

$$= \underset{G \in \mathcal{O}}{\arg\min} \sum_{i=1}^{n} 2\langle u^{(i)} - Of^{(i)}, Gf^{(i)} \rangle_{\mathcal{Y}} + \lambda \|G\|_{\mathcal{B}(\mathcal{X},\mathcal{Y})}^2. \tag{19}$$

The second line can be calculated by thinking of the loss function as a composition of functions $L = L_2 \circ L_1$, $L_1 : \mathbb{R} \to \mathcal{Y}$ which takes

$$L_1(t) = u^{(i)} - (O + tG)f^{(i)}$$

and $L_2 : \mathcal{Y} \to \mathbb{R}$ where

$$L_2(y) = \langle y, y \rangle.$$

Then $L_1'(t)(s) = Gu^{(i)}$ and $L_2'(y)(h) = 2\langle y, h \rangle$. We have

$$\begin{aligned}(L_2 \circ L_1(t))'(s) &= L_2'(L_1(t)) \circ L_1'(t)(s) \\ &= L_2'(u^{(i)} - (O + tG)f^{(i)})(Gf^{(i)}) \\ &= 2\langle u^{(i)} - (O + tG)f^{(i)}, Gf^{(i)} \rangle_{\mathcal{Y}}.\end{aligned}$$

Evaluating the derivative at $t = 0$ gives the desired expression

B.1 Gradient Descent in Operator RKHS

We now introduce the RKHS framework on our space of operators by using an operator-valued kernel. The following definitions were posed in Kadri et al. (2016) (Section 4, Definitions 3 and 5).

Definition B.1. Operator-valued Kernel An operator-valued kernel is a function $\kappa : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ such that

(i) κ is Hermitian, that is, for all $f_1, f_2 \in \mathcal{X}, \kappa(f_1, f_2) = \kappa(f_2, f_1)^*$ where * denotes the adjoint operator.

(ii) κ is positive definite on $\mathcal{X}$ if it is Hermitian and for every $n \in \mathbb{N}$ and all $(f_i, u_i) \in \mathcal{X} \times \mathcal{Y}\ \forall\ i = 1, 2, \ldots, n$, the matrix with (i, j)-th entry $\langle \kappa(f_i, f_j)u_i, u_j \rangle$ is a positive definite matrix.

Definition B.2. Operator RKHS Let $\mathcal{O}$ be a Hilbert space of operators $O : \mathcal{X} \to \mathcal{Y}$, equipped with an inner product $\langle \cdot, \cdot \rangle$. We call $\mathcal{O}$ an operator RKHS if there exists an operator-valued kernel $L : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ such that

(i) The function $g \to \kappa(f, g)u$ for $\mathcal{X}$ belongs to the space $\mathcal{O}$ for all $f \in \mathcal{X}, u \in \mathcal{Y}$.

(ii) κ satisfies the reproducing kernel property:

$$\langle O\kappa(f, \cdot)u \rangle_{\mathcal{O}} = \langle Of, u \rangle_{\mathcal{Y}}$$

for all $O \in \mathcal{O}, f \in \mathcal{X}, u \in \mathcal{Y}$.

We now state the Representer Theorem for operator RKHS's, as stated in Theorem 11 of Stepaniants (2023). Assume that $\mathcal{O}$ can be decomposed orthogonally into $\mathcal{O} = \mathcal{O}_0 \oplus \mathcal{O}_1$ where $\mathcal{O}_0$ is a finite-dimensional Hilbert space spanned by the operators $\{E_k\}_{k=1}^r$ and $\mathcal{O}_1$ is its orthogonal complement under the inner product $\langle \cdot, \cdot \rangle_{\mathcal{O}}$. We denote the inner product $\langle \cdot, \cdot \rangle_{\mathcal{O}}$ restricted to $\mathcal{O}_0, \mathcal{O}_1$ as $\langle \cdot, \cdot \rangle_{\mathcal{O}_0}, \langle \cdot, \cdot \rangle_{\mathcal{O}_0}$ respectively.

Theorem B.3. *Let $\psi : \mathbb{R} \to \mathbb{R}$ be a strictly increasing real-valued function and let $\mathcal{L}(\mathcal{X} \times \mathcal{Y} \times \mathcal{Y}) \to \mathbb{R}$ be an arbitrary loss function. Then*

$$\hat{O} = \underset{O \in \mathcal{O}}{\arg\min}\, \mathcal{L}\left(\{f^{(i)}, u^{(i)}, Of^{(i)}\}_{i=1}^{n}\right) + \psi(\|\mathrm{proj}_{\mathcal{O}_1}\, O\|_{\mathcal{O}_1})$$

has the form

$$\hat{O}(x, y) = \sum_{k=1}^{r} d_k E_k + \sum_{i=1}^{n} \alpha_i \kappa(f^{(i)}, \cdot)$$

for some $d \in \mathbb{R}^r$ and $\alpha_i \in \mathcal{Y}$.

We use this theorem to simplify the expression for gradient descent in operator space.

Lemma B.4. *Given any $O \in \mathcal{O}$, let G^* denote the steepest descent direction of the weighted least-squares loss with respect to $\|\cdot\|_{\mathcal{O}}$ as given in equation 18. Suppose $\mathcal{O}$ is an RKHS with kernel κ. Then*

$$G^*(\cdot) = c \sum_{i=1}^{n} \kappa(f^{(i)}, \cdot)(u^{(i)} - Of^{(i)})$$

for some scalar $c \in \mathbb{R}^+$.

Proof. We apply theorem B.3 to equation Equation (18) with $\mathcal{O}_0$ the trivial subspace and $\psi(s) = \frac{\lambda}{2} s^2$. Then our solution has the form

$$G^*(\cdot) = \sum_{i=1}^{n} \kappa(f^{(i)}, \cdot)\alpha_i. \tag{20}$$

We also know that

$$\|G^*\|_{\mathcal{O}}^2 = \sum_{i,j=1}^{n} \langle \kappa(f^{(i)}, \cdot)\alpha_i, \kappa(f^{(j)}, \cdot)\alpha_j \rangle_{\mathcal{O}} = \sum_{i,j=1}^{n} \langle \kappa(f^{(i)}, f^{(j)})\alpha_i, \alpha_j \rangle_{\mathcal{Y}}$$

where the last equality follows from the RKHS property. We observe that

$$\sum_{i,j=1}^{n} \langle \kappa(f^{(i)}, \cdot)\alpha_i, \kappa(f^{(j)}, \cdot)\alpha_j \rangle_{\mathcal{O}} = \sum_{i,j=1}^{n} \langle \alpha_i, \kappa(f^{(i)}, f^{(j)})\alpha_j \rangle_{\mathcal{Y}}$$

by the same RKHS property. Note that $\kappa(f^{(i)}, f^{(j)}) \in \mathcal{L}(\mathcal{Y})$, that is, is a linear operator from $\mathcal{Y}$ to $\mathcal{Y}$. Let $U \in \mathcal{X}^n, F \in \mathcal{Y}^n$ be such that $U_i = u^{(i)}$ and $F_i = Of^{(i)}$. Then

$$\begin{aligned}
\alpha^* &= \underset{\alpha \in \mathcal{Y}^n}{\arg\min} \sum_{i,j=1}^{n} 2\langle u^{(i)} - Of^{(i)}, \kappa(f^{(i)}, f^{(j)})\alpha_j \rangle_{\mathcal{Y}} + \lambda \langle \alpha_i, \kappa(f^{(i)}, f^{(j)})\alpha_j \rangle_{\mathcal{Y}} \\
&= \underset{\alpha \in \mathcal{Y}^n}{\arg\min} \sum_{i,j=1}^{n} \langle 2(u^{(i)} - Of^{(i)} + \lambda\alpha_i), \kappa(f^{(i)}, f^{(j)})\alpha_j \rangle_{\mathcal{Y}}
\end{aligned}$$

Taking the gradient of α as zero, that is, $\nabla_\alpha = 0$ gives us $\alpha \propto U - OF$ (here we are looking at α as an element of $\mathcal{Y}^n$). We also note that since $\|G^*\|_{\mathcal{O}} = 1$,

$$\sum_{i,j=1}^{n} \langle \alpha_i, \kappa(f^{(i)}, f^{(j)})\alpha_j \rangle_{\mathcal{Y}} = 1.$$

It follows that

$$\alpha^* = \frac{1}{\sum_{i,j=1}^{n} \langle \alpha_i, \kappa(f^{(i)}, f^{(j)})\alpha_j \rangle_{\mathcal{Y}}} (U - OF).$$

Therefore

$$G^*(\cdot) = \frac{1}{\sum_{i,j=1}^n \langle \alpha_i, \kappa(f^{(i)}, f^{(j)})\alpha_j \rangle_{\mathcal{Y}}} \sum_{i=1}^n \kappa(f^{(i)}, \cdot)(u^{(i)} - Of^{(i)}).$$

This gives us an exact form of c as stated in equation Equation (20). We can re-write c as

$$c = \sum_{i,j=1}^n \langle \alpha_i, \kappa(f^{(i)}, f^{(j)})\alpha_j \rangle_{\mathcal{Y}} = \mathbf{1}^T M \mathbf{1}$$

where $(M)_{ij} = \langle \alpha_i, \kappa(f^{(i)}, f^{(j)})\alpha_j \rangle_{\mathcal{Y}}$ and $\mathbf{1}$ is the vector of 1's. Then $c > 0$ by the positive definite property of the kernel κ. □

C TRANSFORMER OPERATOR IN-CONTEXT LEARNING PROOF

We first recall some notation from section 3.1. We are given n demonstrations $(f^{(i)}, u^{(i)}) \in \mathcal{X} \times \mathcal{Y}$ for all $i \in [n]$. The goal is to predict the output function for $f^{(n+1)}$. We stack these in a matrix Z_0 that serves as the input to our transformer:

$$Z_0 = [z^{(1)}, \ldots, z^{(n)}, z^{(n+1)}] = \begin{pmatrix} f^{(1)} & f^{(2)} & \ldots & f^{(n)} & f^{(n+1)} \\ u^{(1)} & u^{(2)} & \ldots & u^{(n)} & 0 \end{pmatrix}.$$

Z_ℓ denotes the output of layer ℓ of the transformer as given in equation 8.

Theorem C.1. *Let $\kappa : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ be an arbitrary operator-valued kernel. Let $\mathcal{O}$ denote the operator RKHS induced by κ. We are given in-context examples of the form $(f^{(i)}, u^{(i)})$ for $i = 1, \ldots, n$. The empirical loss functional $L : \mathcal{O} \to \mathbb{R}$ is given by*

$$L(O) = \sum_{i=1}^n \|u^{(i)} - Of^{(i)}\|_{\mathcal{Y}}^2.$$

Let $O_0 = \mathbf{0}$ and let O_ℓ denote the operator obtained from the ℓ^{th} operator-valued gradient descent sequence of L with respect to $\|\cdot\|_{\mathcal{O}}$ as defined in Equation (16)*. Then there exist scalar step sizes $r'_0, \ldots, r'_k$ such that the following holds-*

Let $[\widetilde{H}(U, W)]_{i,j} = \kappa(u^{(i)}, w^{(j)})$. Let $\mathcal{W}_{v,\ell} = \begin{pmatrix} 0 & 0 \\ 0 & -r'_\ell I \end{pmatrix}$, $\mathcal{W}_{q,\ell} = I$, $\mathcal{W}_{k,\ell} = I$. Then for any $f^{(n+1)}$, the transformer operator's prediction for $u^{(n+1)}$ at each layer ℓ matches the prediction of the operator-valued gradient descent sequence at step ℓ, that is

$$[\mathcal{T}_\ell(f; (\mathcal{W}_v, \mathcal{W}_q, \mathcal{W}_k)|z^{(1)}, \ldots, z^{(n)})]_{2,n+1} = -O_\ell f$$

for all $\ell = 0, 1, \ldots, k$.

Proof. From calculations in subsection Appendix B.1, we know that the ℓ-th step of gradient descent has the form

$$O_{\ell+1} = O_\ell + r'_\ell \sum_{i=1}^n \kappa(f^{(i)}, \cdot)(u^{(i)} - O_\ell f^{(i)}).$$

We now prove several basic facts. Firstly, note that $X_\ell \equiv X_0$ for all l. This can be proved by induction. For the base case, $X_0 = X_0$ trivially. Now suppose the claim holds for some layer ℓ. Then at the l^{th}, layer:

$$Z_{\ell+1} = Z_\ell + \mathcal{T}_\ell(f; (\mathcal{W}_v, \mathcal{W}_q, \mathcal{W}_k)|z^{(1)}, \ldots, z^{(n)}).$$

and the first row of $Z_\ell = X_0$ by the induction assumption. Hence it is enough to show that the first row of $\mathcal{T}_0(f)$ is all zeros.

$$\begin{aligned}
\mathcal{T}_{\ell+1}(f) &= \left(\widetilde{H}(X_\ell, X_\ell) \left(\begin{bmatrix} 0 & 0 \\ 0 & -r'_\ell I \end{bmatrix} \begin{bmatrix} f^{(1)} & \dots & f^{(n)} & f \\ u_\ell^{(1)} & \dots & u_\ell^{(n)} & u_\ell^{(n)} \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} \right)^T \right)^T \\
&= \left(\widetilde{H}(X_0, X_0) \begin{bmatrix} 0 & \dots & 0 & 0 \\ -r'_\ell u_\ell^{(1)} & \dots & -r'_\ell u_\ell^{(n)} & -r'_\ell u_\ell^{(n+1)} \end{bmatrix}^T \right)^T \\
&= \left(\begin{bmatrix} \kappa(f^{(1)}, f^{(1)}) & \kappa(f^{(1)}, f^{(2)}) & \dots & \kappa(f^{(1)}, f^{(n)}) & \kappa(f^{(1)}, f) \\ \kappa(f^{(2)}, f^{(1)}) & & \dots & \kappa(f^{(2)}, f^{(n)}) & \kappa(f^{(2)}, f) \\ \vdots & & \vdots & & \\ \kappa(f, f^{(1)}) & & \dots & \kappa(f, f^{(n)}) & \kappa(f, f) \end{bmatrix} \begin{bmatrix} -r'_\ell u_\ell^{(1)} & 0 \\ \vdots & \vdots \\ -r'_\ell u_\ell^{(n)} & 0 \\ -r'_\ell u_\ell^{(n+1)} & 0 \end{bmatrix} \right)^T \\
&= -r'_\ell \begin{bmatrix} 0 & \dots & 0 & 0 \\ \sum_{i=1}^n \kappa(f^{(1)}, f^{(i)}) u_\ell^{(i)} & \dots & \sum_{i=1}^n \kappa(f^{(n)}, f^{(i)}) u_\ell^{(i)} & \sum_{i=1}^n \kappa(f, f^{(i)}) u_\ell^{(i)} \end{bmatrix}.
\end{aligned} \tag{21}$$

This proves that $X_\ell \equiv X_0$ for all $l = 0, 1, \dots, k$.

From the bottom row of the matrix in Equation (21), we see that

$$u_\ell^{(i)} = u^{(i)} + (-r'_\ell) \sum_{j=1}^n \kappa(f^{(i)}, f^{(j)}) u_\ell^{(j)}.$$

This is equivalent to the fact:

$$u_\ell^{(i)} = u^{(i)} + \mathcal{T}_\ell(f^{(i)}; (\mathcal{W}_q, \mathcal{W}_v, \mathcal{W}_k) | z^{(1)}, \dots, z^{(n)}) \tag{22}$$

for all $i = 0, \dots, n$. In other words, "$u^{(i)} - u_\ell^{(i)}$ is equal to the predicted label for f, if $f^{(i)} = f$". Similar to equation Equation (21), the update at the ℓ^{th} layer looks like:

$$\mathcal{T}_\ell(f) = -r'_\ell \begin{bmatrix} 0 & \dots & 0 & 0 \\ \sum_{i=1}^n \kappa(f^{(1)}, f^{(i)}) u_\ell^{(i)} & \dots & \sum_{i=1}^n \kappa(f^{(n)}, f^{(i)}) u_\ell^{(i)} & \sum_{i=1}^n \kappa(f, f^{(i)}) u_\ell^{(i)} \end{bmatrix}.$$

We now proceed to the proof of the theorem using induction. At step 0, $Z_0 := 0 = O_0$. Now assume $\mathcal{T}_\ell(f; (\mathcal{W}_q, \mathcal{W}_v, \mathcal{W}_k) | z^{(1)}, \dots, z^{(n)}) = -O_\ell f$ holds up to some layer ℓ. For the next layer $\ell + 1$,

$$\begin{aligned}
\mathcal{T}_{\ell+1}(f; (\mathcal{W}_q, \mathcal{W}_v, \mathcal{W}_k)) &= \mathcal{T}_\ell(f; (\mathcal{W}_q, \mathcal{W}_v, \mathcal{W}_k) | z^{(1)}, \dots, z^{(n)}) - r'_\ell \sum_{i=1}^n [\widetilde{H}(X_0, X_0)]_{n+1,i} u_\ell^{(i)} \\
&= \mathcal{T}_\ell(f; (\mathcal{W}_q, \mathcal{W}_v, \mathcal{W}_k) | z^{(1)}, \dots, z^{(n)}) - r'_\ell \sum_{i=1}^n [\widetilde{H}(X_0, X_0)]_{n+1,i} (u^{(i)} - O_\ell f) \\
&= -O_\ell f - r'_\ell \sum_{i=1}^n \kappa(f, f^{(i)}) (u^{(i)} - O_\ell f) \\
&= -O_{\ell+1} f.
\end{aligned}$$

Here, the first line follows from plugging in $\mathcal{W}_q, \mathcal{W}_v, \mathcal{W}_k$ in Equation (21). The second line follows from Equation (22). □