Thompson Sampling in Function Spaces via Neural Operators

Rafael Oliveira [1] **Xuesong Wang** [1] **Kian Ming A. Chai** [2] **Edwin V. Bonilla** [1]

Abstract

We propose an extension of Thompson sampling to optimization problems over function spaces where the objective is a known functional of an unknown operator's output. We assume that functional evaluations are inexpensive, while queries to the operator (such as running a high-fidelity simulator) are costly. Our algorithm employs a sample-then-optimize approach using neural operator surrogates. This strategy avoids explicit uncertainty quantification by treating trained neural operators as approximate samples from a Gaussian process. We provide novel theoretical convergence guarantees based on Gaussian processes in the infinite-dimensional setting, under minimal assumptions. We benchmark our method against existing baselines on functional optimization tasks involving partial differential equations and other nonlinear operator-driven phenomena, demonstrating improved sample efficiency and competitive performance.

1. Introduction

Neural operators have been established as versatile models capable of learning complex, nonlinear mappings between function spaces (Kovachki et al., 2023), with demonstrated success across diverse fields, including climate science (Kurth et al., 2023), materials engineering (Oommen et al., 2024), and computational fluid dynamics (Li et al., 2023). Although their applications in supervised learning and physical system emulation are well-studied, their potential for online learning and optimization within infinite-dimensional function spaces remains relatively untapped.

In many scientific contexts, learning operators that map between entire function spaces naturally arises, such as the task of approximating solution operators for a partial differential equation (PDE) (Kovachki et al., 2023). However, adaptive methods that efficiently query these operators to optimize specific functionals of their outputs (particularly in an active learning setting) are still underdeveloped. Examples of such functionals are fluid pressure based on pore structure design using Darcy flows (Wiker et al., 2007), and the inverse problem of atmospheric dynamics using shallow-water equations (Bonev et al., 2023).

We propose a framework that integrates neural operator surrogates with Thompson sampling strategies. Namely, let $\mathcal{A}$ and $\mathcal{U}$ denote spaces of input and output functions, respectively. The unknown target operator[1] is $G_* : \mathcal{A} \to \mathcal{U}$. We consider $f : \mathcal{U} \to \mathbb{R}$ as a known and cheap-to-evaluate objective functional. Given a compact search space $\mathcal{S} \subset \mathcal{A}$, we aim to solve:

$$a^* \in \operatorname*{argmax}_{a \in \mathcal{S}} f(G_*(a)), \tag{1}$$

while G_* is only accessible via expensive oracle queries: for a chosen a, we observe $y = G_*(a) + \xi$, where ξ is i.i.d. $\mathcal{U}$-valued noise. The algorithm is allowed to query the oracle with any function in the input function space for up to a budget of N queries.

We follow the steps of Bayesian optimization frameworks for composite functions (Astudillo & Frazier, 2019; Guilhoto & Perdikaris, 2024), which leverage knowledge of the composite structure to speed-up optimization, extending the framework to functional domains. Applying the theoretical results for the infinite-width limit of neural networks (Jacot et al., 2018; Lee et al., 2019), we show that a trained neural operator approximates a posterior sample from a vector-valued Gaussian process (Rasmussen & Williams, 2006; Alvarez et al., 2012; Jorgensen & Tian, 2024) in a sample-then-optimize approach (Dai et al., 2022), allowing us to implement Thompson sampling in a simple and effective way, without the need for explicit uncertainty quantification. Experiments on PDE benchmarks validate our approach against Bayesian optimization (Shahriari et al., 2016) baselines.

[1]CSIRO's Data61, Sydney, Australia [2]DSO National Laboratories, Singapore. Correspondence to: Rafael Oliveira <rafael.dossantosdeoliveira@data61.csiro.au>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

[1]Here, we use the term *unknown* loosely, in the sense that it is not fully implementable within the computational resources or paradigms accessible to us. For example, the target operator can be a simulator in a high-performance computing facility which we have limited access to.

2. Related Work

Bayesian optimization (BO) efficiently solves costly black-box optimization problems (Shahriari et al., 2016). Prior work includes Bayesian functional optimization (BFO) with Gaussian process (GP) surrogates over function spaces (Vien & Toussaint, 2018; Vellanki et al., 2019) and composite BO for vector-to-function mappings (Astudillo & Frazier, 2019; Guilhoto & Perdikaris, 2024). In contrast, we optimize function-to-function operators, a setting unexplored by existing BO or bandit methods despite advances in GP models for such mappings (Mora et al., 2025). Lastly, the functional bandits literature (Tran-Thanh & Yu, 2014) addresses known functionals of reward *distributions*.

Neural Thompson Sampling (NTS) (Zhang et al., 2021) leverages neural networks and neural tangent kernel theory (Jacot et al., 2018) to approximate GPs. The STO-BNTS variant (Dai et al., 2022) improves this by using linearized network training to emulate GP posterior samples, but these methods have not been extended to function-valued inputs.

In Bayesian experimental design, deep operator networks (DeepONets) and Fourier neural operators (FNOs) have been used for uncertainty reduction with ensemble-based mixture models (Pickering et al., 2022; Li et al., 2024a). Similarly, Musekamp et al. (2025) applied variance-based active learning for forecasting. Unlike these, our work targets optimization objectives beyond information gain.

3. Neural Operator Thompson Sampling

We propose a Thompson sampling algorithm for optimizing functionals of unknown operators, leveraging neural operator surrogates. The method efficiently explores the input space while balancing exploration and exploitation.

3.1. Approximate Posterior Sampling

Given data $\mathcal{D}_t = \{(a_i, y_i)\}_{i=1}^t$, we train a neural operator $G_{\boldsymbol{\theta}}$ with parameters $\boldsymbol{\theta}$ to minimize the empirical loss:

$$\boldsymbol{\theta}_t := \underset{\boldsymbol{\theta}\in\mathcal{W}}{\operatorname{argmin}} \sum_{j=1}^{t} \|y_j - G_{\boldsymbol{\theta}}(a_j)\|_{\mathcal{U}}^2 + \lambda\|\boldsymbol{\theta}\|^2, \qquad (2)$$

where $\|\cdot\|_{\mathcal{U}}$ represents the norm in the operator's output function space $\mathcal{U}$ and $\lambda > 0$ is a regularization factor which relates to the noise process ξ (see Proposition 1 and Ordoñez et al., 2025). The argmin operator is implemented via gradient descent starting from $\boldsymbol{\theta}_{t,0} \sim \mathcal{N}(0, \boldsymbol{\Sigma}_0)$, where $\boldsymbol{\Sigma}_0$ is a diagonal matrix following Kaiming (He et al., 2015) or LeCun initialization (LeCun et al., 1998), which scale the weights intialization variance by the width of the previous layer. By standard results on the infinite-width limit of neural networks, we can show that the trained neural operator converges to a posterior sample from a vector-valued GP when, e.g., we train only the last linear layer (Lee et al., 2019, App. D), which in turn guarantees regret bounds (Section 4). In practice, we also note that inputs a and observations y are discretized either over a finite grid or other finite-dimensional representations (Kovachki et al., 2023).

Algorithm 1 Neural Operator Thompson Sampling (NOTS)

Input: Search space $\mathcal{S}$, Initial dataset $\mathcal{D}_0$ (optional)
for $t = 1$ **to** T **do**
  Randomly initialize $\boldsymbol{\theta}_{t,0} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_0)$
  $\boldsymbol{\theta}_t \leftarrow \operatorname{argmin}_{\boldsymbol{\theta}} \sum_{j=1}^{t-1} \|y_j - G_{\boldsymbol{\theta}}(a_j)\|_{\mathcal{U}}^2 + \lambda\|\boldsymbol{\theta}\|^2$
  Define $G_t := G_{\boldsymbol{\theta}_t}$
  $a_t \leftarrow \operatorname{argmax}_{a\in\mathcal{S}} f(G_t(a))$
  Query $y_t \leftarrow G_*(a_t) + \xi_t$
  $\mathcal{D}_t \leftarrow \mathcal{D}_{t-1} \cup \{(a_t, y_t)\}$
end for

3.2. Thompson Sampling Algorithm

In Algorithm 1, we present the Neural Operator Thompson Sampling (NOTS) algorithm, designed to efficiently optimize a scalar functional of an unknown operator's output. The algorithm operates over T iterations. Each iteration begins with the random initialization of the parameters of a neural operator that serves as a surrogate model for the true unknown operator. This surrogate is trained by minimizing a regularized least-squares loss based on previously observed data, ensuring that it does not overfit to potentially noisy data. The next step involves selecting the input for querying the oracle by maximizing the value of the objective functional over the neural operator's predictions. Finally, the algorithm queries the oracle at the selected input and updates the dataset with the new observation. This process repeats for up to a given budget of T iterations.

4. Theoretical Results

We establish the theoretical foundation of our proposed method. We show how the trained neural operator converges to a Gaussian process in the infinite-width limit through the use of the conjugate kernel, also known as NNGP kernel (Neal, 1996; Daniely, 2017; Lee et al., 2018). This allows us to extend existing guarantees for Gaussian process Thompson sampling (GP-TS) to our setting (Takeno et al., 2024).

We analyze the performance of a sequential decision-making algorithm via its Bayesian cumulative regret. An algorithm's regret for querying $x_t \in \mathcal{X}$ at iteration $t \geq 1$ is given by:

$$r_t := f(G_*(a^*)) - f(G_*(a_t)) \qquad (3)$$

where a^* is defined as in Equation 1. The Bayesian cumula-

tive regret after T iterations is then defined as:

$$R_T := \mathbb{E}\left[\sum_{t=1}^{T} r_t\right], \tag{4}$$

where the expectation is over the function space prior for f and all other random variables involved in the decision-making process and the function space prior. If the algorithm achieves sub-linear cumulative regret, its regret vanishes, as $\lim_{T\to\infty} \mathbb{E}\left[\min_{t\in\{1,\dots,T\}} r_t\right] \leq \lim_{T\to\infty} \frac{1}{T} R_T$.

Regularity assumptions. For our analysis, we assume that $\mathcal{U} \subset \mathcal{L}_2(\nu)$, where $\mathcal{L}_2(\nu)$ denotes the Hilbert space of square-integrable ν-measurable functions, for a given finite Borel measure ν on a compact domain $\mathcal{Z}$. The search space $\mathcal{S} \subset \mathcal{A}$ is a compact metric space. The true operator $G_* : \mathcal{A} \to \mathcal{U}$ will be assumed to be a sample from a vector-valued Gaussian process $G_* \sim \mathcal{GP}(0, K)$, where the operator-valued kernel $K : \mathcal{A} \times \mathcal{A} \to \mathcal{L}(\mathcal{U})$ is given by a neural operator's infinite-width limit. Observations $y = G_*(a) + \xi$ are assumed to be corrupted by zero-mean Gaussian (process) noise, $\xi \sim \mathcal{GP}(0, k_\xi)$, where $k_\xi : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$ is a positive-semidefinite kernel on the same domain $\mathcal{Z}$ as the operator's output functions.

We adapt state-of-the-art regret bounds for Gaussian process Thompson sampling (Takeno et al., 2024) to our setting. To do so, we show that randomly initialized neural operators behave as Gaussian processes in the infinite-width limit (see Proposition C.1). Their composition with fixed functionals then yields a scalar-valued GP, allowing us to apply GP-TS regret bounds.

Proposition 1. *Let $f : \mathcal{U} \to \mathbb{R}$ be a bounded linear functional. Assume that the search space $\mathcal{S} \subset \mathcal{A}$ is finite, i.e., $|\mathcal{S}| < \infty$, and that observations are corrupted by noise $\xi \sim \mathcal{GP}(0, k_\xi)$ such that $k_\xi : (z, z') \mapsto \sigma_\xi^2 \mathbb{I}[z = z']$, for a given $\sigma_\xi > 0$. Let NOTS train only the last linear layer of the neural operator surrogate. Then, in the infinite-width limit, NOTS equipped with a single-hidden-layer neural operator surrogate and $\lambda := \sigma_\xi^2$ achieves:*

$$R_T \in \mathcal{O}(\sqrt{T\gamma_{f,T}}), \tag{5}$$

where $\gamma_{f,T}$ corresponds to the maximum information gain after T iterations for a GP prior with kernel $k_f := f^\mathsf{T} K f$, i.e., $\gamma_{f,T} := \max_{\mathcal{S}_T \subset \mathcal{S} : |\mathcal{S}_T| \leq T} \frac{1}{2} \log|\mathbf{I} + \lambda^{-1}\mathbf{K}_{f,T}|$, with $\mathbf{K}_{f,T} := [k_f(a, a')]_{a,a' \in \mathcal{S}_T}$.

The result above connects existing GP-TS guarantees to NOTS, and it differs from existing guarantees for other neural network based Thompson sampling algorithms (Zhang et al., 2021; Dai et al., 2022), which explored a frequentist setting (i.e., the objective function being a fixed element of the reproducing kernel Hilbert space associated with the network's neural tangent kernel). In the Bayesian setting, there is also no need for a time-dependent regularization parameter, allowing for a simpler implementation. The last-layer-only assumption on training ensures that the trained network follows an exact GP posterior in the infinite-width limit (Lee et al., 2019, App. D), while explicit regularization accounts for observation noise (Ordoñez et al., 2025). Full proofs and further discussions can be found in the appendix.

5. Experiments

We evaluate NOTS on PDE benchmarks (Darcy flow and shallow water), comparing to GP-based Bayesian optimization and neural network baselines that model the mapping from function-valued inputs $a \in \mathcal{A}$ (discretized over a grid) to scalar-valued functional evaluations $f(G_*(a))$, alongside a random search (RS) baseline. NOTS utilizes standard and spherical FNOs, adhering to recommended dataset settings (Kossaifi et al., 2024). We first implement BO with a 3-layer infinite-width ReLU BNN model, represented as a GP with its equivalent kernel, based on Li et al. (2024b)'s findings of optimal performance in high dimensions. Our results include two versions of BO: one with log-expected improvement (Ament et al., 2023), noted as "BO" in our plots, and one using Thompson sampling (GP-TS) (Russo & Van Roy, 2016). We also assess Bayesian functional optimization (BFO), encoding input functions in an RKHS via their minimum-norm interpolant and employing a squared-exponential kernel, as in (Vien & Toussaint, 2018). Finally, we evaluate sample-then-optimize neural Thompson sampling (STO-NTS), training a 2-layer 256-width fully connected neural network with a regularized least-squares loss (Dai et al., 2022).

5.1. Optimization Functionals

We define several problem-dependent optimization functionals, clarifying their physical interpretations for the benchmark flows in our maximization context, where negative signs indicate quantities to be minimized.

Negative total flow rates (Katz, 1979): $f(u, a) = -\int_{\partial\mathcal{Z}} a(z)(\nabla u(z) \cdot \mathbf{n})ds$ integrates the volumetric flux $-a(z)\nabla u(z)$ across the boundary $\partial\mathcal{Z}$ (with outward normal $\mathbf{n}$), reflecting the fluid's total flow rate, useful for leakage reduction and contaminant control.

Negative total pressure (Jeong & Lee, 2025): $f(u) = -\frac{1}{2}\int_{\mathcal{Z}} \|u(z)\|_2 dz$ computes the average L^2-norm of the output function over the domain.

Negative total potential power (Wiker et al., 2007): $f(u, g) = -\frac{1}{2}\alpha(u, u) - \frac{1}{2}\beta(u, u) + \langle g, u\rangle$, where $\alpha(u, v) = \int_{\mathcal{Z}} \frac{1}{\int_{\mathcal{Z}} a(z)dz} u(z) \cdot v(z) dz$ normalizes total power, $\beta(u, v) = 2\int_{\mathcal{Z}} D(u(z)) \cdot D(v(z)) dz$ measures symmetrical power

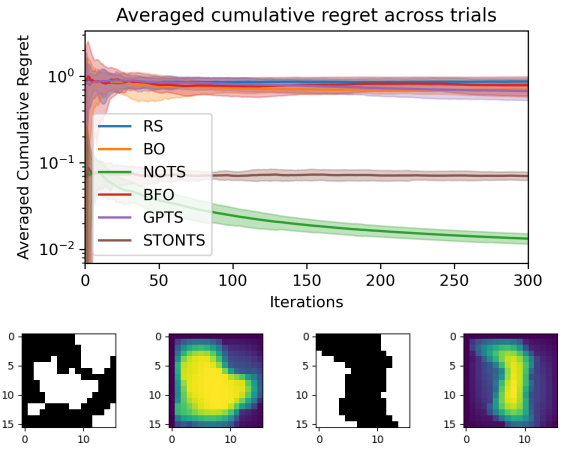

Figure 1. Darcy flow rate optimization. Average cumulative regret across trials for the negative total flow rates case in the Darcy flow problem. The shaded areas correspond to one standard deviation across 10 trials. The corresponding input-output functions that achieved the best (left) and worst (right) flow rates are presented (bottom). White regions $a(x) = 1$ means fully open permeability and black regions $a(x) = 0$ represents impermeable pore material. Output functions are pressure fields, where brighter color indicates higher pressure.

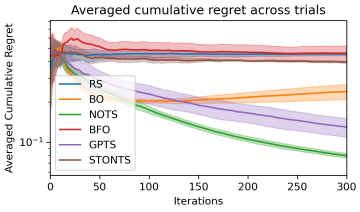

Figure 2. Darcy flow potential power. Average cumulative regret across trials for the negative total potential power functional in the Darcy flow problem. The shaded areas correspond to one standard deviation across 10 trials.

with $D(u(z)) = \frac{1}{2}(\nabla u(z) + (\nabla u(z))^T)$, and $\langle g, u \rangle = \int_{\mathcal{Z}} g(z) \cdot u(z) dz$ captures power induced by the forcing function $g(z)$ (set to 1 in Darcy flow).

Inverse problem: $f(u) = -\frac{1}{2}\|u - u_t\|^2$ targets the initial condition a that produces the ground truth solution u_t, specifically in shallow water modeling. This mimics the assimilation objective used in weather forecasting (Rabier et al., 1998; Xiao et al., 2023).

5.2. Results

The results presented in Fig. 1 to 4 highlight the cumulative regret of NOTS compared to baselines across various PDE problem settings. In the Darcy flow benchmark (Fig. 1), GP-based BO methods underperform in high-dimensional

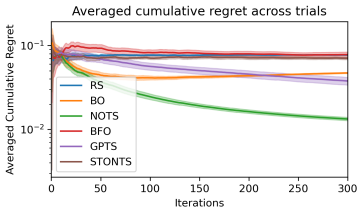

Figure 3. Darcy flow pressure. Overlay of cumulative regret (left) and its average (right) metrics across trials for the negative total pressure functional in the Darcy flow problem.

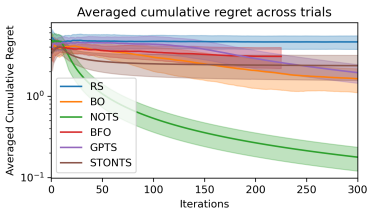

Figure 4. Shallow water inverse problem. Overlay of cumulative regret (left) and its average (right) metrics across trials for the inverse problem in the shallow water data.

scenarios, while NOTS and other neural network Thompson sampling methods excel. The "best candidate" design effectively blocks fluid outflow with impermeable regions, whereas the "worst candidate" features permeable zones that allow high flow rates. Additional results on optimizing power and pressure in Darcy flow (Figs. 2 and 3) further confirm NOTS's advantages. For the inverse problem in the shallow water benchmark (Fig. 4), NOTS successfully navigates the challenging 6144-dimensional input space, leveraging the problem's underlying physics for more efficient exploration.

6. Conclusion

We introduced Neural operator Thompson sampling (NOTS), demonstrating notable gains in capturing the compositional structure of black-box operator problems like complex physics simulators. NOTS is supported by theoretical guarantees linking Thompson sampling to neural operator frameworks. Practically, using neural operators as surrogates for Thompson sampling is effective and avoids costly uncertainty quantification by leveraging insights from infinitely wide deep networks and Gaussian processes. Neural operators enable scalable representation learning in very high-dimensional spaces where traditional bandits and Bayesian optimization struggle. Currently, our results address finite search spaces and well-specified models; future work will extend to continuous domains and batch settings for enhanced scalability.

Acknowledgements

This research was carried out solely using CSIRO's resources. Chai contributed while on sabbatical leave visiting the Machine Learning and Data Science Unit at Okinawa Institute of Science and Technology, and the Department of Statistics in the University of Oxford. This project was supported by resources and expertise provided by CSIRO IMT Scientific Computing.

References

Alvarez, M. A., Rosasco, L., and Lawrence, N. D. Kernels for vector-valued functions: a review. *Foundations and Trends in Machine Learning*, 4(3), 2012.

Ament, S., Daulton, S., Eriksson, D., Balandat, M., and Bakshy, E. Unexpected improvements to expected improvement for Bayesian optimization. In *37th Conference on Neural Information Processing Systems (NeurIPS 2023)*, New Orleans, LA, USA, 2023.

Astudillo, R. and Frazier, P. I. Bayesian optimization of composite functions. In *36th International Conference on Machine Learning, ICML 2019*, volume 2019-June, 2019.

Bonev, B., Kurth, T., Hundt, C., Pathak, J., Baust, M., Kashinath, K., and Anandkumar, A. Spherical Fourier neural operators: Learning stable dynamics on the sphere. In *International conference on machine learning (ICML)*, pp. 2806–2823. PMLR, 2023.

Chowdhury, S. R. and Gopalan, A. On kernelized multi-armed bandits. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, Sydney, Australia, 2017.

Dai, Z., Shu, Y., Low, B. K. H., and Jaillet, P. Sample-then-optimize batch neural Thompson sampling. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc.

Daniely, A. SGD learns the conjugate kernel class of the network. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Guilhoto, L. F. and Perdikaris, P. Composite Bayesian optimization in function spaces using NEON – Neural Epistemic Operator Networks. *Scientific Reports*, 14, 2024.

Hanin, B. Random neural networks in the infinite width limit as Gaussian processes. *The Annals of Applied Probability*, 33(6A):4798 – 4819, 2023. URL https://doi.org/10.1214/23-AAP1933.

He, K., Zhang, X., Ren, S., and Sun, J. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 1026–1034, 2015.

Hu, Z. and Huang, H. On the random conjugate kernel and neural tangent kernel. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 4359–4368. PMLR, 18–24 Jul 2021.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, Montreal, Canada, 2018.

Jeong, S. and Lee, S. Optimal control for Darcy's equation in a heterogeneous porous media. *Applied Numerical Mathematics*, 207:303–322, 2025.

Jorgensen, P. E. T. and Tian, J. Operator-valued Gaussian processes and their covariance kernels. *Infinite Dimensional Analysis, Quantum Probability and Related Topics*, 27(02), 2024.

Katz, V. J. The history of Stokes' theorem. *Mathematics Magazine*, 52(3):146–156, 1979.

Kossaifi, J., Kovachki, N., Li, Z., Pitt, D., Liu-Schiaffini, M., George, R. J., Bonev, B., Azizzadenesheli, K., Berner, J., and Anandkumar, A. A library for learning neural operators, 2024.

Kovachki, N., Lanthaler, S., and Mishra, S. On universal approximation and error bounds for Fourier neural operators. *Journal of Machine Learning Research*, 22(290):1–76, 2021. URL http://jmlr.org/papers/v22/21-0806.html.

Kovachki, N., Li, Z., Liu, B., Azizzadenesheli, K., Bhattacharya, K., Stuart, A., and Anandkumar, A. Neural operator: Learning maps between function spaces. *Journal of Machine Learning Research*, 24(89), 2023.

Kurth, T., Subramanian, S., Harrington, P., Pathak, J., Mardani, M., Hall, D., Miele, A., Kashinath, K., and Anandkumar, A. FourCastNet: Accelerating global high-resolution weather forecasting using adaptive Fourier neural operators. In *Proceedings of the Platform for Advanced Scientific Computing Conference*, PASC '23, New York, NY, USA, 2023. Association for Computing Machinery.

LeCun, Y., Bottou, L., Orr, G. B., and Müller, K. R. *Efficient BackProp*, pp. 9–50. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

Lee, J., Bahri, Y., Novak, R., Schoenholz, S. S., Pennington, J., and Sohl-dickstein, J. Deep neural networks as Gaussian processes. In *International Conference on Learning Representations (ICLR)*, 2018.

Lee, J., Xiao, L., Schoenholz, S. S., Bahri, Y., Novak, R., Sohl-Dickstein, J., and Pennington, J. Wide neural networks of any depth evolve as linear models under gradient descent. In *33rd Conference on Neural Information Processing Systems (NeurIPS)*, Vancouver, Canada, 2019.

Li, S., Yu, X., Xing, W., Kirby, R., Narayan, A., and Zhe, S. Multi-resolution active learning of Fourier neural operators. In Dasgupta, S., Mandt, S., and Li, Y. (eds.), *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pp. 2440–2448. PMLR, 2024a. URL https://proceedings.mlr.press/v238/li24k.html.

Li, Y. L., Rudner, T. G. J., and Wilson, A. G. A study of Bayesian neural network surrogates for Bayesian optimization. In *2024 International Conference on Learning Representations (ICLR)*, Vienna, Austria, 2024b. OpenReview.

Li, Z., Kovachki, N. B., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*. OpenReview, 2021.

Li, Z., Kovachki, N., Choy, C., Li, B., Kossaifi, J., Otta, S., Nabian, M. A., Stadler, M., Hundt, C., Azizzadenesheli, K., and Anandkumar, A. Geometry-informed neural operator for large-scale 3d PDEs. *Advances in Neural Information Processing Systems*, 36:35836–35854, 2023.

Liu, C., Zhu, L., and Belkin, M. On the linearity of large non-linear models: When and why the tangent kernel is constant. In *Advances in Neural Information Processing Systems*, 2020.

Mao, Z. and Meng, X. Physics-informed neural networks with residual/gradient-based adaptive sampling methods for solving partial differential equations with sharp solutions. *Applied Mathematics and Mechanics*, 44(7): 1069–1084, 2023.

Matthews, A. G. d. G., Hron, J., Rowland, M., Turner, R. E., and Ghahramani, Z. Gaussian Process Behaviour in Wide Deep Neural Networks. In *International Conference on Learning Representations*, Vancouver, Canada, 2018. OpenReview.net. URL https://openreview.net/forum?id=H1-nGgWC-.

Mora, C., Yousefpour, A., Hosseinmardi, S., Owhadi, H., and Bostanabad, R. Operator learning with Gaussian processes. *Computer Methods in Applied Mechanics and Engineering*, 434:117581, 2025.

Musekamp, D., Kalimuthu, M., Holzmüller, D., Takamoto, M., and Niepert, M. Active learning for neural PDE solvers. In *International Conference on Learning Representations (ICLR)*, Singapore, 2025. OpenReview.

Neal, R. M. *Priors for Infinite Networks*, chapter 2, pp. 29–53. Springer New York, New York, NY, 1996.

Nguyen, M. and Mücke, N. Optimal convergence rates for neural operators. *arXiv e-prints*, 2024. URL http://arxiv.org/abs/2412.17518.

Oommen, V., Shukla, K., Desai, S., Dingreville, R., and Karniadakis, G. E. Rethinking materials simulations: Blending direct numerical simulations with neural operators. *npj Computational Materials*, 10(1): 145, 2024. URL https://doi.org/10.1038/s41524-024-01319-1.

Ordoñez, S. C., Plenk, J., Bergna, R., Cartea, A., Hernández-Lobato, J. M., Palla, K., and Ciosek, K. Observation noise and initialization in wide neural networks. In *7th Symposium on Advances in Approximate Bayesian Inference – Workshop Track*, 2025. URL https://openreview.net/forum?id=9A9p2lkPDI.

Owhadi, H. and Scovel, C. *Gaussian Measures, Cylinder Measures, and Fields on $\mathcal{B}$*, pp. 347–359. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 2019.

Pickering, E., Guth, S., Karniadakis, G. E., and Sapsis, T. P. Discovering and forecasting extreme events via active learning in neural operators. *Nature Computational Science*, 2(12):823–833, 2022.

Pisier, G. Probabilistic methods in the geometry of Banach spaces. *Probability and analysis*, pp. 167–241, 1986.

Rabier, F., Thépaut, J.-N., and Courtier, P. Extended assimilation and forecast experiments with a four-dimensional variational assimilation system. *Quarterly Journal of the Royal Meteorological Society*, 124(550):1861–1887, 1998.

Rasmussen, C. E. and Williams, C. K. I. *Gaussian Processes for Machine Learning*. The MIT Press, Cambridge, MA, 2006.

Russo, D. and Van Roy, B. Learning to optimize via posterior sampling. *Mathematics of Operations Research*, 39 (4):1221–1243, 2014.

Russo, D. and Van Roy, B. An information-theoretic analysis of Thompson sampling. *Journal of Machine Learning Research (JMLR)*, 17:1–30, 2016.

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., and De Freitas, N. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104 (1):148–175, 2016.

Takeno, S., Inatsu, Y., Karasuyama, M., and Takeuchi, I. Posterior sampling-based Bayesian optimization with tighter Bayesian regret bounds. In *Proceedings of the 41 st International Conference on Machine Learning (ICML 2024)*, volume 235, Vienna, Austria, 2024. PMLR.

Tran-Thanh, L. and Yu, J. Y. Functional bandits. *arXiv e-prints*, 2014. URL http://arxiv.org/abs/1405.2432.

Vakili, S., Bromberg, M., Garcia, J., Shiu, D. S., and Bernacchia, A. Information gain and uniform generalization bounds for neural kernel models. In *IEEE International Symposium on Information Theory (ISIT)*, pp. 555–560. IEEE, 2023.

Vellanki, P., Rana, S., Gupta, S., de Celis Leal, D., Sutti, A., Height, M., and Venkatesh, S. Bayesian functional optimisation with shape prior. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):1617–1624, 2019.

Vien, N. A. and Toussaint, M. Bayesian functional optimization. In *AAAI Conference on Artificial Intelligence*, pp. 4171–4178, New Orleans, LA, USA, 2018.

Wiker, N., Klarbring, A., and Borrvall, T. Topology optimization of regions of Darcy and Stokes flow. *International journal for numerical methods in engineering*, 69 (7):1374–1404, 2007.

Xiao, Y., Bai, L., Xue, W., Chen, K., Han, T., and Ouyang, W. Fengwu-4dvar: Coupling the data-driven weather forecasting model with 4d variational assimilation. *arXiv preprint arXiv:2312.12455*, 2023.

Zhang, W., Zhou, D., Li, L., and Gu, Q. Neural Thompson sampling. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=tkAtoZkcUnm.

A. Background

Neural operators. A neural operator is a specialized neural network architecture modeling operators $G : \mathcal{A} \to \mathcal{U}$ between function spaces $\mathcal{A}$ and $\mathcal{U}$ (Kovachki et al., 2023). Assume $\mathcal{A} \subset \mathcal{C}(\mathcal{X}, \mathbb{R}^{d_a})$ and $\mathcal{U} \subset \mathcal{C}(\mathcal{Z}, \mathbb{R}^{d_u})$, where $\mathcal{C}(\mathcal{S}, \mathcal{S}')$ denotes the space of continuous functions between sets $\mathcal{S}$ and $\mathcal{S}'$. Given an input function $a \in \mathcal{A}$, a neural operator $G_{\boldsymbol{\theta}}$ performs a sequence of transformations $a =: u_0 \mapsto u_1 \mapsto \cdots \mapsto u_{L-1} \mapsto u_L$ through L layers of neural networks, where $u_l : \mathcal{X}_l \to \mathbb{R}^{d_l}$ is a continuous function for each layer $l \in \{1, \ldots, L\}$, and $\mathcal{X}_L := \mathcal{Z}$ is the domain of the output functions and $d_L := d_u$. In one of its general formulations, for a given $l \in \{0, 1, \ldots, L-1\}$, the result of the transform (or update) at any $x \in \mathcal{X}_{l+1}$ and $z \in \mathcal{Z}$ can be described as:

$$u_0 := a \tag{A.1}$$

$$u_{l+1}(x) := \alpha_{l+1}\left(\int_{\mathcal{X}_l} \mathbf{R}_l(x, x', u_l(\Pi_l(x)), u_l(x'))\, u_l(x')\, \mathrm{d}\nu_l(x') + \mathbf{W}_l\, u_l(\Pi_l(x)) + b_l(x)\right) \tag{A.2}$$

$$G_{\boldsymbol{\theta}}(a)(z) := u_L(z)\,, \tag{A.3}$$

where $\Pi_l : \mathcal{X}_{l+1} \to \mathcal{X}_l$ is a fixed mapping (often the identity), $\alpha_l : \mathbb{R} \to \mathbb{R}$ denotes an activation function applied elementwise, $\mathbf{R}_l : \mathcal{X}_{l+1} \times \mathcal{X}_l \times \mathbb{R}^{d_l} \times \mathbb{R}^{d_l} \to \mathbb{R}^{d_{l+1} \times d_l}$ defines a (possibly nonlinear or positive-semidefinite) kernel integral operator with respect to a measure ν_l on $\mathcal{X}_l$, $\mathbf{W}_l \in \mathbb{R}^{d_{l+1} \times d_l}$ is a weight matrix, and $b_l : \mathcal{X}_{l+1} \to \mathbb{R}^{d_{l+1}}$ is a bias function. We denote by $\boldsymbol{\theta}$ the collection of all learnable parameters of the neural operator: the weights matrices $\mathbf{W}_l$, the parameters of the bias functions b_l and the matrix-valued kernels $\mathbf{R}_l$, for all layers $l \in \{1, \ldots, L\}$. Variations to the formulation above correspond to various neural operator architectures based on low-rank kernel approximations, graph structures, Fourier transforms, etc. (Kovachki et al., 2023).

Vector-valued Gaussian processes. Vector-valued Gaussian processes extend scalar GPs (Rasmussen & Williams, 2006) to the case of vector-valued functions (Alvarez et al., 2012). Let $\mathcal{A}$ be an arbitrary domain, and let $\mathcal{U}$ be a Hilbert space representing a codomain. We consider the case where both the domain $\mathcal{A}$ and codomain $\mathcal{U}$ might be infinite-dimensional vector spaces, which leads to GPs whose realizations are operators $G_* : \mathcal{A} \to \mathcal{U}$ (Jorgensen & Tian, 2024). To simplify our exposition, we assume that $\mathcal{U}$ is a separable Hilbert space, though the theoretical framework is general enough to be extended to arbitrary Banach spaces (Owhadi & Scovel, 2019). A vector-valued Gaussian process $G_* \sim \mathcal{GP}(\widehat{G}, K)$ on $\mathcal{A}$ is fully specified by a mean operator $\widehat{G} : \mathcal{A} \to \mathcal{U}$ and a positive-semidefinite operator-valued covariance function $K : \mathcal{A} \times \mathcal{A} \to \mathcal{L}(\mathcal{U})$, where $\mathcal{L}(\mathcal{U})$ denotes the space of bounded linear operators on $\mathcal{U}$. Formally, given any $a, a' \in \mathcal{A}$ and any $u, u' \in \mathcal{U}$, it follows that:

$$\mathbb{E}[G_*(a)] = \widehat{G}(a), \tag{A.4}$$

$$\mathrm{Cov}(\langle G_*(a), u\rangle, \langle G_*(a'), u'\rangle) = \langle u, K(a, a')u'\rangle\,, \tag{A.5}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product and $\mathrm{Cov}(\cdot, \cdot)$ stands for the covariance between scalar variables. Assume we are given a set of observations $\mathcal{D}_t := \{(a_i, y_i)\}_{i=1}^t \subset \mathcal{A} \times \mathcal{U}$, where $y_i = G_*(a_i) + \xi_i$, and $\xi_i \sim \mathcal{N}(0, \Sigma)$ corresponds to Gaussian $\mathcal{U}$-valued noise with covariance operator $\Sigma \in \mathcal{L}(\mathcal{U})$. The posterior mean and covariance can then be defined by the following recursive relations:

$$\widehat{G}_t(a) = \widehat{G}_{t-1}(a) + K_{t-1}(a, a_t)(K_{t-1}(a_t, a_t) + \Sigma)^{-1} u_t \tag{A.6}$$

$$K_t(a, a') = K_{t-1}(a, a') - K_{t-1}(a, a_t)(K_{t-1}(a_t, a_t) + \Sigma)^{-1} K_{t-1}(a_t, a') \tag{A.7}$$

for any $a, a' \in \mathcal{A}$, and $t \in \mathbb{N}$, which are an extension of the same recursions from the scalar-valued case (Chowdhury & Gopalan, 2017, App. F) to the case of vector-valued processes. Such definition arises from sequentially conditioning the GP posterior on each observation, starting from the prior $\mathcal{GP}(\widehat{G}_0, K_0)$. It leads to the same matrix-based definitions of the usual GP posterior equations (Rasmussen & Williams, 2006), but in our case avoids complications with the resulting higher-order tensors that arise when kernels are operator-valued.

B. Conjugate Kernel vs. Neural Tangent Kernel

In this section, we discuss the main differences between the neural tangent kernel (Jacot et al., 2018) and the conjugate kernel, also known as the neural network Gaussian process (NNGP) kernel (Lee et al., 2019). Both kernels are used to approximate the behavior of neural networks, but they differ in how they use Gaussian processes to describe the network's behavior.

B.1. Conjugate Kernel (NNGP)

The conjugate kernel has long been studied in the neural networks literature, describing the correspondence neural networks with randomized parameters and their limiting distribution as the network width approaches infinity (Neal, 1996; Daniely, 2017; Lee et al., 2018; Matthews et al., 2018; Hu & Huang, 2021). Neal (1996) first showed the correspondence between an infinitely wide single-hidden-layer network and a Gaussian process by applying the central limit theorem. More recent works (Daniely, 2017; Lee et al., 2018; Matthews et al., 2018) later showed that the same reasoning can be extended to neural networks with multiple hidden layers. The NNGP kernel is particularly useful for Bayesian inference as it allows us to define GP priors for neural networks and analyze how they change when conditioned on data, providing us with closed-form expressions for an exact GP posterior in the infinite-width limit (Lee et al., 2018).

Define an L-layer neural network $h(\cdot, \boldsymbol{\theta}) : \mathcal{X} \to \mathbb{R}$ with $h(x; \boldsymbol{\theta}) := h_L(x; \boldsymbol{\theta})$ via the recursion:

$$\begin{aligned} h_0(x; \boldsymbol{\theta}) &:= x \\ h_l(x; \boldsymbol{\theta}) &:= \alpha_l(\mathbf{W}_l h_{l-1}(x; \boldsymbol{\theta}) + \mathbf{b}_l)\,, \quad l \in \{1, \dots, L\}, \end{aligned} \tag{B.1}$$

where $x \in \mathcal{X}$ represents an arbitrary input on a finite-dimensional domain $\mathcal{X}$, $\mathbf{W}_l \in \mathbb{R}^{M_l \times M_{l-1}}$ denotes a layer's weights matrix, M_l is the width of the lth layer, $\mathbf{b}_l \in \mathbb{R}^{M_l}$ is a bias vector, $\alpha_l : \mathbb{R} \to \mathbb{R}$ denotes the layer's activation function, which is applied elementwise on vector-valued inputs, and $\boldsymbol{\theta} := \mathrm{vec}(\{\mathbf{W}_l, b_l\}_{l=1}^L)$ collects all the network parameters into a vector. Assume $[\mathbf{W}_l]_{i,j} \sim \mathcal{N}\left(0, \frac{1}{M_{l-1}}\right)$ and $[\mathbf{b}_l]_i \sim \mathcal{N}(0, 1)$, for $i \in \{1, \dots, M_l\}$, $j \in \{1, \dots, M_{l-1}\}$ and $l \in \{1, \dots, L\}$, and let $M := \min\{M_1, \dots, M_L\}$. The NNGP kernel then corresponds to the infinite-width limit of the network outputs covariance function (Lee et al., 2018) as:

$$k_{\texttt{NNGP}}(x, x') := \lim_{M \to \infty} \mathbb{E}[h(x; \boldsymbol{\theta}) h(x'; \boldsymbol{\theta})], \quad x, x' \in \mathcal{X}\,, \tag{B.2}$$

where the expectation is taken under the parameters distribution. By an application of the central limit theorem, it can be shown (Neal, 1996; Lee et al., 2018) that the neural network converges in distribution to a Gaussian process with the kernel defined above, i.e.:

$$h_{\boldsymbol{\theta}} \xrightarrow{d} h \sim \mathcal{GP}(0, k_{\texttt{NNGP}})\,, \tag{B.3}$$

where $\xrightarrow{d}$ denotes convergence in distribution as $M \to \infty$. In other words, the randomly initialized network follows a GP prior in the infinite-width limit. Moreover, it follows that, when conditioned on data $\mathcal{D}_N := \{x_i, y_i\}_{i=1}^N$, assuming $y_i = h(x_i) + \epsilon_i$ and $\epsilon_i \sim \mathcal{N}(0, \sigma_\epsilon^2)$, a Bayesian neural network is distributed according to a GP posterior in the infinite-width limit as:

$$h | \mathcal{D}_N \sim \mathcal{GP}(\mu_N, k_N) \tag{B.4}$$

$$\mu_N(x) := \mathbb{E}[h(x) \mid \mathcal{D}_N] = \mathbf{k}_N(x)^\mathsf{T} (\mathbf{K}_N + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}_N \tag{B.5}$$

$$k_N(x, x') := \mathrm{Cov}[h(x), h(x') \mid \mathcal{D}_N] = k(x, x') - \mathbf{k}_N(x)^\mathsf{T} (\mathbf{K}_N + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}_N(x'), \tag{B.6}$$

for any $x, x' \in \mathcal{X}$, where $\mathbf{K}_N := [k(x_i, x_j)]_{i,j=1}^N \in \mathbb{R}^{N \times N}$, $\mathbf{k}_N(x) := [k(x_i, x)]_{i=1}^N \in \mathbb{R}^N$, $\mathbf{y}_N := [y_i]_{i=1}^N$, and we set $k := k_{\texttt{NNGP}}$ to avoid notation clutter. Hence, the NNGP kernel allows us to compute exact GP posteriors for neural network models. However, we emphasize that the conjugate kernel should not be confused with the neural tangent kernel (Jacot et al., 2018), which corresponds to the infinite-width limit of $\mathbb{E}[\nabla_{\boldsymbol{\theta}} h(x; \boldsymbol{\theta}) \cdot \nabla_{\boldsymbol{\theta}} h(x'; \boldsymbol{\theta})]$, instead.

B.2. Neural Tangent Kernel (NTK)

The NTK approximates the behavior of a neural network during training via gradient descent by considering the gradients of the network with respect to its parameters (Jacot et al., 2018). Consider an L-layer feedforward neural network $h_{\boldsymbol{\theta}} : \mathcal{X} \to \mathbb{R}$ as defined in Equation B.1. In its original formulation, Jacot et al. (2018) applied a scaling factor of $\frac{1}{\sqrt{M}}$ to the output of each layer to ensure asymptotic convergence in the limit $M \to \infty$ of the network trained via gradient descent. However, later works showed that standard network parameterizations (without explicit output scaling) also converge to the same limit as long as a LeCun or Kaiming/He type of initialization scheme is applied to the parameters with appropriate scaling of the learning rates (Lee et al., 2019; Liu et al., 2020), which ensure bounded variance in the infinite-width limit. The NTK describes the limit:

$$k_{\texttt{NTK}}(x, x') = \lim_{M \to \infty} \mathbb{E}[\nabla_{\boldsymbol{\theta}} h_{\boldsymbol{\theta}}(x) \cdot \nabla_{\boldsymbol{\theta}} h_{\boldsymbol{\theta}}(x')]\,, \tag{B.7}$$

for any $x, x' \in \mathcal{X}$, where the expectation is taken under the parameters initialization distribution. Under mild assumptions, the trained network's output distribution converges to a Gaussian process described by the NTK (Jacot et al., 2018; Lee et al., 2018). Although originally derived for the unregularized case, applying L2 regularization to the parameters norm yields a GP posterior with a term that can account for observation noise (Ordoñez et al., 2025). Namely, consider the following loss function:

$$\ell_N(\boldsymbol{\theta}) := \sum_{i=1}^{N} (y_i - h_{\boldsymbol{\theta}}(x_i))_2^2 + \lambda \|\boldsymbol{\theta} - \boldsymbol{\theta}_0\|_2^2 \,, \tag{B.8}$$

where $\boldsymbol{\theta}_0$ denotes the initial parameters. As the network width grows larger, the NTK tells us that the network behaves like a linear model (Jacot et al., 2018; Liu et al., 2020) as:

$$h(x; \boldsymbol{\theta}) \approx h(x; \boldsymbol{\theta}_0) + \nabla_{\boldsymbol{\theta}} h(x; \boldsymbol{\theta})\big|_{\boldsymbol{\theta} := \boldsymbol{\theta}_0} \cdot (\boldsymbol{\theta} - \boldsymbol{\theta}_0) \,, \quad x \in \mathcal{X} \,. \tag{B.9}$$

The approximation becomes exact in the infinite width limit within any bounded neighborhood $\mathcal{B}_R(\boldsymbol{\theta}_0) := \{\boldsymbol{\theta} \mid \|\boldsymbol{\theta} - \boldsymbol{\theta}_0\| \leq R\}$ of arbitrary radius $0 < R < \infty$ around $\boldsymbol{\theta}_0$, as the second-order error term vanishes (Liu et al., 2020). The latter also means that $\nabla_{\boldsymbol{\theta}} h(\cdot; \boldsymbol{\theta})$ converges to fixed feature map $\phi : \mathcal{X} \to \mathcal{H}_0$, where $\mathcal{H}_0$ is the Hilbert space spanned by the limiting gradient vectors. With this observation, our loss function can be rewritten as:

$$\ell_N(\boldsymbol{\theta}) \approx \sum_{i=1}^{N} \Big(y_i - h(x_i; \boldsymbol{\theta}_0) - \nabla_{\boldsymbol{\theta}} h(x_i; \boldsymbol{\theta})\big|_{\boldsymbol{\theta} := \boldsymbol{\theta}_0} \cdot (\boldsymbol{\theta} - \boldsymbol{\theta}_0) \Big)^2 + \lambda \|\boldsymbol{\theta} - \boldsymbol{\theta}_0\|_2^2 \,. \tag{B.10}$$

The minimizer of the approximate loss can be derived in closed form. Applying the NTK then yields the infinite-width model:

$$h_N(x) = h(x) + \mathbf{k}_N^{\texttt{NTK}}(x)^\intercal (\mathbf{K}_N^{\texttt{NTK}} + \lambda \mathbf{I})^{-1} (\mathbf{y}_N - \mathbf{h}_N) \,, \tag{B.11}$$

where $h \sim \mathcal{GP}(0, k_{\texttt{NNGP}})$ denotes the network at its random initialization, as defined above, $\mathbf{k}_N^{\texttt{NTK}}(x) := [k_{\texttt{NTK}}(x_i, x)]_{i=1}^N \in \mathbb{R}^N$, $\mathbf{K}_N^{\texttt{NTK}} := [k_{\texttt{NTK}}(x_i, x_j)]_{i,j=1}^N \in \mathbb{R}^{N \times N}$, and $\mathbf{h}_N := [h(x_i)]_{i=1}^N \in \mathbb{R}^N$. Now applying the GP limit to the randomly initialized network h (Lee et al., 2019; Ordoñez et al., 2025), we have that:

$$h_N \sim \mathcal{GP}(\hat{\mu}_N, \hat{k}_N) \tag{B.12}$$

$$\hat{\mu}_N(x) = \mathbf{k}_N^{\texttt{NTK}}(x)^\intercal (\mathbf{K}_N^{\texttt{NTK}} + \lambda \mathbf{I})^{-1} \mathbf{y}_N \tag{B.13}$$

$$\begin{aligned} \hat{k}_N(x, x') = k(x, x') &+ \mathbf{k}_N^{\texttt{NTK}}(x)^\intercal (\mathbf{K}_N^{\texttt{NTK}} + \lambda \mathbf{I})^{-1} \mathbf{K}_N (\mathbf{K}_N^{\texttt{NTK}} + \lambda \mathbf{I})^{-1} \mathbf{k}_N^{\texttt{NTK}}(x') \\ &- \mathbf{k}_N^{\texttt{NTK}}(x)^\intercal (\mathbf{K}_N^{\texttt{NTK}} + \lambda \mathbf{I})^{-1} \mathbf{k}_N(x') - \mathbf{k}_N(x)^\intercal (\mathbf{K}_N^{\texttt{NTK}} + \lambda \mathbf{I})^{-1} \mathbf{k}_N^{\texttt{NTK}}(x'), \end{aligned} \tag{B.14}$$

where we again set $k := k_{\texttt{NNGP}}$ to avoid clutter. However, note that such GP model does not generally correspond to a Bayesian posterior. An exception is where only the last linear layer is trained, while the rest are kept fixed at their random initialization; in which case case, the GP described by the NTK and the exact GP posterior according to the NNGP kernel match (Lee et al., 2019, App. D).

B.3. Application to Thompson Sampling

For our purpose, it is important to have a Bayesian posterior in order to apply Gaussian process Thompson sampling (GP-TS) (Takeno et al., 2024) for the regret bounds in Proposition 2. Therefore, we are constrained by existing theories connecting neural networks to Gaussian processes to assume training only the last layer of neural networks of infinite width, which gives a Bayesian posterior of the NNGP after training. In addition, we had to consider the case of a single hidden layer neural operator, as the usual recursive step applied to derive the infinite-width limit would require an intermediate (infinite-dimensional) function space in our case, making the extension to the multi-layer case not trivial. Nonetheless, the NOTS algorithm suggested by our theory has demonstrated competitive performance in our experiments even in more relaxed settings with a multi-layer model. Future theoretical developments in Bayesian analysis of neural networks may eventually permit the convergence analysis of the more relaxed settings in our experiments. In any case, we present an experiment with a wide single-hidden-layer model with training only on the last layer in Appendix F.

C. The Infinite-Width Limit of Neural Operators

Neural operator abstraction. A neural operator is a specialized neural network architecture modeling nonlinear operators $G : \mathcal{A} \to \mathcal{U}$ between possibly infinite-dimensional function spaces $\mathcal{A}$ and $\mathcal{U}$. Current results in NTK (Jacot et al., 2018)

and Gaussian process theory for neural networks (Lee et al., 2019) do not immediately apply to this setting, as they are formulated for finite-dimensional domains. However, we can leverage an abstraction for neural operator architectures to see their layers as operating over finite-dimensional inputs (Nguyen & Mücke, 2024), which result from truncations that make the modeling problem tractable.

Considering a *single* hidden layer neural operator, let $M \in \mathbb{N}$ represent the layer's width, $A_{\mathbf{R}} : \mathcal{A} \to \mathcal{C}(\mathcal{Z}, \mathbb{R}^{d_{\mathbf{R}}})$ denote a (fixed) continuous operator, and $b_0 : \mathcal{Z} \to \mathbb{R}^{d_b}$ denote a (fixed) continuous function. For simplicity, we will assume scalar outputs with $d_u = 1$. In general, a single hidden layer of the model described in Equation A.3 can be rewritten as:

$$G_{\boldsymbol{\theta}}(a)(z) = \mathbf{w}_o^{\mathsf{T}} \alpha \left(\mathbf{W}_{\mathbf{R}} A_{\mathbf{R}}(a)(z) + \mathbf{W}_u a(\Pi_0(z)) + \mathbf{W}_b b_0(z) \right) , \tag{C.1}$$

for $z \in \mathcal{Z}$, where $\boldsymbol{\theta} := \mathrm{vec}(\mathbf{w}_o, \mathbf{W}_{\mathbf{R}}, \mathbf{W}_u, \mathbf{W}_b) \in \mathbb{R}^{M(1+d_{\mathbf{R}}+d_a+d_b)} =: \mathcal{W}$ represents the model's flattened parameters. The finite weight matrix $\mathbf{W}_{\mathbf{R}}$ representing the kernel convolution integral arises as a result of truncations required in the practical implementation of neural operators (e.g., a finite number of Fourier modes or quadrature points). With this formulation, one can recover most popular neural operator architectures (Nguyen & Mücke, 2024). In the appendix, we discuss how Fourier neural operators (Li et al., 2021) fit under this formulation, though it is general enough to incorporate other cases. We also highlight that neural operators possess universal approximation properties (Kovachki et al., 2021), given sufficient data and computational resources, despite the inherent truncations in their architecture.

C.1. Application to Fourier Neural Operators

As an example, we show how the formulation above applies to the Fourier neural operator (FNO) architecture (Li et al., 2021). For simplicity, assume that $\mathcal{X}$ is the d-dimensional periodic torus, i.e., $\mathcal{X} = [0, 2\pi)^d$, and $\mathcal{Z} = \mathcal{X}$. Then any square-integrable function $a : \mathcal{X} \to \mathbb{C}^{d_a}$ can be expressed as a Fourier series:

$$a(x) = \sum_{s \in \mathbb{Z}^d} \hat{a}(s) e^{\iota \langle s, x \rangle}, \quad \forall x \in \mathcal{X} , \tag{C.2}$$

where $\iota := \sqrt{-1} \in \mathbb{C}$ denotes the imaginary unit, and $\hat{a}(s)$ are coefficients given by the function's Fourier transform $F : \mathcal{L}_2(\mathcal{X}, \mathbb{C}^{d_a}) \to \mathcal{L}_2(\mathbb{Z}^d, \mathbb{C}^{d_a})$ as:

$$\hat{a}(s) := (Fa)(s) = \frac{1}{(2\pi)^d} \int_{\mathcal{X}} a(x) e^{-\iota \langle s, x \rangle} \, \mathrm{d}x , \quad s \in \mathbb{Z}^d . \tag{C.3}$$

For a translation-invariant kernel $\mathbf{R}(x, x') = \mathbf{R}(x - x')$, applying the convolution theorem, the integral operator is:

$$\begin{aligned} \int_{\mathcal{X}} \mathbf{R}(\cdot, x) a(x) \, \mathrm{d}x &= \mathbf{R} * a \\ &= F^{-1}(F(\mathbf{R}) \cdot F(a)) \\ &= \sum_{s \in \mathbb{Z}^d} \widehat{\mathbf{R}}(s) \hat{a}(s) e^{\iota \langle s, \cdot \rangle} \end{aligned} \tag{C.4}$$

In practice, function observations are only available at a discrete set of points and the Fourier series is truncated at a maximum frequency $s_{\max} \in \mathbb{Z}^d$, which allows one to efficiently compute it via the fast Fourier transform (FFT). Considering these facts, FNOs approximate the integral as (Li et al., 2021):

$$\int_{\mathcal{X}} \mathbf{R}(x, x') a(x') \, \mathrm{d}x' \approx \sum_{n=1}^{N} \widehat{\mathbf{R}}(s_n) \hat{a}(s_n) e^{\iota \langle s_n, x \rangle}, \quad x \in \mathcal{Z} , \tag{C.5}$$

where the N values of s_n range from 0 to $s_{\max}$ in all d coordinates. Now we can finally see that, defining $A_{\mathbf{R}}$ as:

$$\begin{aligned} A_{\mathbf{R}} : \mathcal{C}(\mathcal{X}, \mathbb{C}^{d_a}) &\to \mathcal{C}(\mathcal{X}, \mathbb{C}^{N d_a}) \\ a &\mapsto \begin{bmatrix} (Fa)(s_1) e^{\iota \langle s_1, \cdot \rangle} \\ \vdots \\ (Fa)(s_N) e^{\iota \langle s_N, \cdot \rangle} \end{bmatrix} , \end{aligned} \tag{C.6}$$

and letting $\mathbf{W}_{\mathbf{R}} = [\widehat{\mathbf{R}}(s_1), \ldots, \widehat{\mathbf{R}}(s_N)]$, we recover Equation C.1 for FNOs in the complex-valued case.

For real-valued functions, to ensure that the result is again real-valued, a symmetry condition is imposed on $\widehat{\mathbf{R}}$, so that its values for negative frequencies are the conjugate transpose of the corresponding values for positive frequencies. However, we can still represent it via a single matrix of weights, which is simply conjugate transposed for the negative frequencies. Lastly, note that complex numbers can be represented as tuples of real numbers.

C.2. Infinitely Wide Neural Operator as Gaussian Process

With the construction in Equation C.1, we can simply see the result of a neural operator layer when evaluated at a fixed $z \in \mathcal{Z}$ equivalently as a M-width feedforward neural network:

$$G_{\boldsymbol{\theta}}(a)(x) = h_{\boldsymbol{\theta}}(\mathbf{v}_z(a)) := \langle \mathbf{w}_o, \alpha(\mathbf{W}\mathbf{v}_z(a)) \rangle \,, \tag{C.7}$$

where the input is given by $\mathbf{v}_z(a) := [A_{\mathbf{R}}(a)(z),\ a(\Pi_0(z)),\ b_0(z)] \in \mathcal{V}$, and $\mathcal{V} := \mathbb{R}^{d_{\mathbf{R}}+d_a+d_b}$.

Conjugate kernel. As seen above, a single-hidden-layer neural operator can be seen as a fully connected neural network $h_{\boldsymbol{\theta}} : \mathcal{V} \to \mathbb{R}$ when its output is evaluated at a fixed $z \in \mathcal{Z}$. We can now derive its infinite-width limits. The conjugate kernel describes the distribution of the untrained neural network under Gaussian weights initialization, whose infinite-width limit yields a Gaussian process (Neal, 1996; Lee et al., 2018). Formally, the conjugate kernel is defined as:

$$k_h(\mathbf{v}, \mathbf{v}') := \lim_{M\to\infty} \mathbb{E}_{\boldsymbol{\theta}_0 \sim \mathcal{N}(\mathbf{0},\mathbf{I})}[h_{\boldsymbol{\theta}_0}(\mathbf{v}) h_{\boldsymbol{\theta}_0}(\mathbf{v}')], \quad \mathbf{v}, \mathbf{v}' \in \mathcal{V} \,. \tag{C.8}$$

As the composition of the map $\mathcal{A} \times \mathcal{Z} \ni (a, z) \mapsto \mathbf{v}_z(a) \in \mathcal{V}$ with a kernel on $\mathcal{V}$ yields a kernel on $\mathcal{A} \times \mathcal{Z}$, the conjugate kernel of $G_{\boldsymbol{\theta}}$ is determined by:

$$k_G(a, z, a', z') := k_h(\mathbf{v}_z(a), \mathbf{v}_{z'}(a')), \quad a, a' \in \mathcal{A}, \quad z, z' \in \mathcal{Z} \,, \tag{C.9}$$

where k_h is the conjugate kernel of the neural network $h_{\boldsymbol{\theta}}$. Such a kernel defines a covariance function for a GP over the space of operators mapping $\mathcal{A}$ to $\mathcal{U}$. Assume $\mathcal{U} \subset \mathcal{L}_2(\nu)$ for some σ-finite Borel measure on $\mathcal{Z}$, and let $\mathcal{L}(\mathcal{U})$ denote the space of linear operators on $\mathcal{U}$. The following then defines a positive-semidefinite operator-valued kernel $K_G : \mathcal{A} \times \mathcal{A} \to \mathcal{L}(\mathcal{U})$:

$$(K_G(a, a')u)(z) = \int_{\mathcal{Z}} k_G(a, z, a', z') u(z') \, \mathrm{d}\nu(z'), \tag{C.10}$$

for any $u \in \mathcal{U}$, $a, a' \in \mathcal{A}$ and $z \in \mathcal{Z}$. Hence, we can state the following result.

Proposition C.1. *Let $G_{\boldsymbol{\theta}}$ be a neural operator with a single hidden layer, as defined as in Equation C.1. Assume $\mathbf{w}_o \sim \mathcal{N}(\mathbf{0}, \sigma_{\boldsymbol{\theta}}^2 \mathbf{I})$, for $\sigma_{\boldsymbol{\theta}}^2 > 0$ such that $\sigma_{\boldsymbol{\theta}}^2 \in \mathcal{O}(\frac{1}{M})$, and let the remaining parameters have their entries sampled from a standard normal distribution. Then, as $M \to \infty$, the neural operator converges in distribution to a zero-mean vector-valued Gaussian process with operator-valued covariance function given by:*

$$\lim_{M\to\infty} \mathbb{E}_{\boldsymbol{\theta} \sim \mathcal{N}(\mathbf{0}, \sigma_{\boldsymbol{\theta}}^2 \mathbf{I})}[G_{\boldsymbol{\theta}}(a) \otimes G_{\boldsymbol{\theta}}(a')] = K_G(a, a') \,, \quad a, a' \in \mathcal{A} \,, \tag{C.11}$$

where $K_G : \mathcal{A} \times \mathcal{A} \to \mathcal{L}(\mathcal{U})$ is defined in Equation C.10, and $\otimes$ denotes the outer product.

To prove the result above, we first need the following definitions and auxiliary results. We start with a basic definition for a randomly initialized neural network.

Definition C.1 (Multi-Layer Fully-Connected Neural Network)**.** *A multi-layer fully-connected neural network with L hidden layers, input dimension d_0, output dimension d_{L+1}, and hidden layer widths $d_1, \ldots, d_L$, is defined recursively as follows. For input $x \in \mathcal{X}$, the pre-activations and activations at layer $l = 1, \ldots, L+1$ are:*

$$\mathbf{v}^{(1)}(x) = \mathbf{W}^{(0)} x + \mathbf{b}^{(0)} \tag{C.12}$$

$$\mathbf{v}^{(l)}(x) = \mathbf{W}^{(l-1)} \alpha(\mathbf{v}^{(l-1)}(x)) + \mathbf{b}^{(l-1)}, \quad l = 2, \ldots, L, \tag{C.13}$$

$$\mathbf{v}^{(L+1)}(x) = \mathbf{W}^{(L)} \alpha(\mathbf{v}^{(L)}(x)), \tag{C.14}$$

where $\mathbf{W}^{(l)} \in \mathbb{R}^{d_{l+1} \times d_l}$ are weight matrices, $\mathbf{b}^{(l)} \in \mathbb{R}^{d_{l+1}}$ are bias vectors, $\alpha : \mathbb{R} \to \mathbb{R}$ is a coordinate-wise non-linearity, and the network output is $f(x) = \mathbf{v}^{(L+1)}(x)$. The weights are initialized as $W_{ij}^{(l)} = \left(\frac{c_W}{d_l}\right)^{1/2} \widehat{W}_{ij}^{(l)}$, where $\widehat{W}_{ij}^{(l)} \sim \mu$ with mean 0, variance 1, and finite higher moments, and biases as $b_i^{(l)} \sim \mathcal{N}(0, c_b)$, given fixed constants $c_W > 0$ and $c_b \geq 0$.

Lemma C.1 (Infinite-width limit (Hanin, 2023)). *Consider a feedforward fully connected neural network as in Definition C.1 with non-linearity $\alpha : \mathbb{R} \to \mathbb{R}$ that is absolutely continuous with polynomially bounded derivative. Fix the input dimension d_0, the output dimension d_{L+1}, the number of layers L, and a compact set $\mathcal{X} \subset \mathbb{R}^{d_0}$. As hidden layer widths $d_1, \ldots, d_L \to \infty$, the random field $x \mapsto f(x)$ converges weakly in $\mathcal{C}(\mathcal{X}, \mathbb{R}^{d_{L+1}})$ to a centered Gaussian process with covariance $\mathbf{K}^{(L+1)} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^{d_{l+1} \times d_{l+1}}$ defined recursively by:*

$$\mathbf{K}^{(l+1)}(x, x') = c_b \mathbf{I} + c_W \, \mathbb{E}_{(\mathbf{v}, \mathbf{v}')} \left[\alpha(\mathbf{v}) \otimes \alpha(\mathbf{v}') \right], \tag{C.15}$$

where $(\mathbf{v}, \mathbf{v}') \sim \mathcal{N}\left(0, \begin{bmatrix} \mathbf{K}^{(l)}(x, x) & \mathbf{K}^{(l)}(x, x') \\ \mathbf{K}^{(l)}(x, x') & \mathbf{K}^{(l)}(x', x') \end{bmatrix}\right)$ *for $l \geq 2$, with the initial condition for $l = 1$ determined by the first-layer weights and biases.*

Assumption 1. *The activation function $\alpha : \mathbb{R} \to \mathbb{R}$ is absolutely continuous with derivative bounded almost everywhere.*

Considering the assumption above, the main component of our proof of Proposition C.1 is the following lemma.

Lemma C.2 (Continuity of limiting GP). *Let $G_{\boldsymbol{\theta}} : \mathcal{A} \to \mathcal{C}(\mathcal{Z})$ be a neural operator with a single hidden layer, as defined as in Equation C.1. Assume $\mathbf{w}_o \sim \mathcal{N}(\mathbf{0}, \sigma_{\boldsymbol{\theta}}^2 \mathbf{I})$, for $\sigma_{\boldsymbol{\theta}}^2 > 0$ such that $\sigma_{\boldsymbol{\theta}}^2 \in \mathcal{O}(\frac{1}{M})$, and let the remaining parameters have their elements sampled from a standard normal distribution. Then, as $M \to \infty$, the neural operator converges in distribution to a zero-mean Gaussian process with continuous realizations $G : \mathcal{A} \to \mathcal{C}(\mathcal{Z})$.*

Proof. As shown above, when evaluated at a fixed point $z \in \mathcal{Z}$, a neural operator with a single hidden layer can be seen as:

$$G_{\boldsymbol{\theta}}(a)(z) = h_{\boldsymbol{\theta}}(\phi(a, z)), \quad a \in \mathcal{A}, \tag{C.16}$$

where $\phi(a, z) := \mathbf{v}_z(a)$ is a fixed map $\phi : \mathcal{A} \times \mathcal{Z} \to \mathcal{V}$, with $\mathcal{V} = \mathbb{R}^{d_{\mathbf{R}} + d_a + d_b}$, and $h_{\boldsymbol{\theta}}$ is a conventional feedforward neural network. By Assumption 1 and Lemma C.1, it follows that, as $M \to \infty$, $h_{\boldsymbol{\theta}}$ converges in distribution to a Gaussian process $h \sim \mathcal{GP}(0, k_h)$ with continuous sample paths, i.e., $h \in \mathcal{C}(\mathcal{V})$ almost surely. The continuity of $\phi : \mathcal{A} \times \mathcal{Z} \to \mathcal{V}$ then implies that $g := h \circ \phi$ is a zero-mean GP whose sample paths lie almost surely in $\mathcal{C}(\mathcal{A} \times \mathcal{Z})$. Therefore, for each $a \in \mathcal{A}$, we have $\mathbb{P}\left[g(a, \cdot) \in \mathcal{C}(\mathcal{Z})\right] = 1$, so that $G(a) := g(a, \cdot)$ defines an almost surely continuous operator $G : \mathcal{A} \to \mathcal{C}(\mathcal{Z})$. The verification that G is a vector-valued GP trivially follows. □

The proof of Proposition C.1 now proceeds as follows.

Proof of Proposition C.1. We start by noting that any continuous function $u \in \mathcal{C}(\mathcal{Z})$ is automatically included in $\mathcal{U} = \mathcal{L}_2(\nu)$, since $\|u\|_{\mathcal{U}}^2 = \int_{\mathcal{Z}} u(z)^2 \, \mathrm{d}\nu(z) \leq \nu(\mathcal{Z}) \|u\|_\infty^2 < \infty$. Hence, any operator mapping into $\mathcal{C}(\mathcal{Z})$ also maps into $\mathcal{U}$ by inclusion.

Applying Lemma C.2, it follows that $G_{\boldsymbol{\theta}} \xrightarrow{d} G$, where G is a zero-mean GP, as $M \to \infty$. Now, given any $u \in \mathcal{U}$, $a, a' \in \mathcal{A}$ and $z \in \mathcal{Z}$, we have that:

$$\begin{aligned}
(\mathbb{E}[G(a) \otimes G(a')]u)(z) &= \mathbb{E}[G(a)\langle G(a'), u\rangle] \\
&= \left(\mathbb{E}\left[g(a, \cdot) \int_{\mathcal{Z}} g(a', z') u(z') \, \mathrm{d}\nu(z') \right] \right)(z) \\
&= \mathbb{E}\left[\int_{\mathcal{Z}} g(a, z) g(a', z') u(z') \, \mathrm{d}\nu(z') \right] \\
&= \int_{\mathcal{Z}} \mathbb{E}[g(a, z) g(a', z')] u(z') \, \mathrm{d}\nu(z') \\
&= \int_{\mathcal{Z}} k_G(a, z, a', z') u(z') \, \mathrm{d}\nu(z'),
\end{aligned} \tag{C.17}$$

where we applied the linearity of expectations and the correspondence between $g : \mathcal{A} \times \mathcal{Z} \to \mathbb{R}$ and the limiting operator $G : \mathcal{A} \to \mathcal{U}$. As the choice of elements was arbitrary, it follows that the above defines an operator-valued kernel K_G. Linearity follows from the expectations. Given any $a \in \mathcal{A}$, the operator norm of $K_G(a, a)$ is bounded by its trace, which is such that:

$$\mathrm{Tr}(K_G(a, a)) = \mathbb{E}[\|G(a)\|_{\mathcal{U}}^2] = \mathbb{E}\left[\int_{\mathcal{Z}} g(a, z)^2 \, \mathrm{d}\nu(z) \right] < \nu(\mathcal{Z}) \mathbb{E}[\|g\|_\infty^2], \tag{C.18}$$

and the last expectation is finite, since g is almost surely continuous. Hence, $K_G(a, a) \in \mathcal{L}(\mathcal{U})$. □

D. Regret Bound

Lemma D.1 (Thm. 3.1 in Takeno et al. (2024)). *Let $f \sim \mathcal{GP}(0, k)$, where $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a positive-definite kernel on a finite $\mathcal{X}$. Then the Bayesian cumulative regret of GP Thompson sampling is such that:*

$$R_T \in \mathcal{O}(\sqrt{T\gamma_T}),$$

where γ_T denotes the maximum information gain after T iterations with the GP model.

Proof of Proposition 1. Following Proposition C.1, the infinite-width limit yields $G \sim \mathcal{GP}(0, K_G)$. By linearity, it follows that $f \circ G \sim \mathcal{GP}(0, f^\mathsf{T} K_G f)$ for any fixed bounded linear functional $f : \mathcal{U} \to \mathbb{R}$.

As in practice we only observe the operators output on a finite domain, i.e., $\mathcal{Z} := \{z_i\}_{i=1}^{n_z}$, the assumed noise model yields a finite-dimensional vector of observations $\mathbf{y} = G_*(a) + \boldsymbol{\xi}$, where $\boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \sigma_\xi^2 \mathbf{I})$, for any $a \in \mathcal{S}$. Training our model via regularized gradient descent with $\lambda := \sigma_\xi^2$ then yields a neural operator GP posterior $G_* | \mathcal{D}_t \sim \mathcal{GP}(\widehat{G}_t, K_t)$ with mean and covariance operators defined in (A.6) and (A.7), respectively, as previously discussed. Correspondingly, we have $f \circ G_* | \mathcal{D}_t \sim \mathcal{GP}(f \circ \widehat{G}_t, f^\mathsf{T} K_t f)$. The result then follows by a straightforward application of Lemma D.1. □

Remark D.1. *Despite the result above assuming that f is only a function of $G(a)$, there is a straightforward extension to functionals of the form $f : \mathcal{U} \times \mathcal{A} \to \mathbb{R}$, as considered in our experiments. We simply need to replace $G : \mathcal{A} \to \mathcal{U}$ with the operator $G' : a \mapsto (G(a), a)$ by a concatenation with an identity map $a \mapsto a$, which is deterministic. A similar result then follows after minor adjustments.*

If the cumulative regret bound above grows sublinearly over time, i.e., $\gamma_{f,T} \in o(\sqrt{T})$, the algorithm should eventually find the true optimum, since its simple regret would vanish. Showing that the maximum information gain satisfies these assumptions, however, requires specific knowledge of the chosen neural operator architecture. In the case of activation functions of the form $\alpha_s(y) = (\max(0, y))^s$, for $s \in \mathbb{N}$, Vakili et al. (2023) showed that $\gamma_T \in \tilde{\mathcal{O}}(T^{\frac{d-1}{d+2s}})$, where the $\tilde{\mathcal{O}}$ notation suppresses logarithmic factors. Hence, for a sufficiently smooth activation function (i.e., with a large enough smoothness parameter s), NOTS should be able to achieve sublinear cumulative regret.

E. Experiment Details

E.1. PDE Problems

In this section, we provide the details of the problems used in the experiments.

Darcy flow. Darcy flow describes the flow of a fluid through a porous medium with the following PDE form

$$-\nabla \cdot (a(x) \nabla u(x)) = g(x) \quad x \in \Omega = (0, 1)^2$$

$$u(x) = 0 \quad x \in \partial\Omega = \partial(0, 1)^2$$

where $u(x)$ is the flow pressure, $a(x)$ is the permeability coefficient and $g(x)$ is the forcing function. We fix $g(x) = 1$ and generate different solutions at random with zero Neumann on the Laplacian, following the setting in Li et al. (2021) via the neural operator library implementation (Kossaifi et al., 2024). In particular, for this problem, we generate a search space $\mathcal{S}$ with $|\mathcal{S}| = 1000$ data points. The divergence of f is $\nabla \cdot f = \frac{\partial f_x}{\partial x} + \frac{\partial f_y}{\partial y}$ where $f : \Omega \to \mathbb{R}^2$ is a vector field $f = (f_x, f_y)$. The gradient $\nabla u = (\frac{\partial u(x,y)}{\partial x}, \frac{\partial u(x,y)}{\partial y})$ where $u(x, y) : \Omega \to \mathbb{R}$ is a scalar field. Inspired by previous works (Wiker et al., 2007; Katz, 1979; Mao & Meng, 2023), we chose the following objective functions to evaluate the functions assuming that we aim to maximize the objective function $f(\cdot)$:

1. Negative total flow rates (Katz, 1979)

$$f(u, a) = \int_{\partial\Omega} a(x)(\nabla u(x) \cdot n) ds$$

where $s = \partial\Omega$ is the boundary of the domain and n is the outward pointing unit normal vector of the boundary. $q(x) = -a(x)\nabla u(x)$ is the volumetric flux which describes the rate of volume flow across a unit area. Therefore, the objective

function measures the boundary outflux. Since the boundary is defined on a grid, $n \in \{[-1,0],[1,0],[0,1],[0,-1]\}$ for the left, right, top and bottom boundaries. The boundary integral can be simplified as

$$\int_0^1 [-a(0,y)u_x(0,y) + a(1,y)u_x(1,y)]dy + \int_0^1 [-a(x,0)u_y(x,0) + a(x,1)u_y(x,1)]dx$$

where $u_x(x,y) = \frac{\partial u}{\partial x}, u_y(x,y) = \frac{\partial u}{\partial y}$

2. High gradient solutions (Algorithm 2 in (Mao & Meng, 2023))

$$f(u) = \frac{1}{K}\sum_k top_k(\|\nabla u(x)\|_2), \quad x \in \Omega$$

where the top-K highest gradients of the solution functions are averaged to approximate the high gradients of the overall solutions. In the experiments we set $K = 10$ for computation efficiency.

3. Negative total pressure (Eq 2.1 in (Jeong & Lee, 2025))

$$f(u,g) = -\frac{1}{2}\int_\Omega (\|u(x)\|_2 + \beta\|g(x)\|_2)dx$$

with $\beta > 0$ is a coefficient for the forcing term $g(x)$. With a constant $g(x)$, the objective is simplified as $-\frac{1}{2}\int_\Omega \|u(x)\|_2 dx$.

4. Negative total potential power (Eq (14) in (Wiker et al., 2007))

$$f(u,g) = -\frac{1}{2}\alpha(u,u) - \frac{1}{2}\beta(u,u) + \langle g,u\rangle$$

where $\alpha(u,v) = \int_\Omega \frac{1}{\int_\Omega a(x)dx} u(x)\cdot v(x)dx$ evaluates the normalized total power, $\beta(u,v = 2\int_\Omega D(u(x))\cdot D(v(x))dx$ measures the symmetrical power with $D(u(x)) = \frac{1}{2}(\nabla u(x) + (\nabla u(x))^T)$, and $\langle g,u\rangle = \int_\Omega g(x)\cdot u(x)dx$ is the power induced by the forcing function.

Shallow Water. The shallow water equation on the rotating sphere is often used to model ocean waters over the surface of the globe. This problem can be described by the following PDE (Bonev et al., 2023):

$$\frac{\partial \varphi}{\partial t} + \nabla\cdot(\varphi v) = 0 \quad \text{in} \quad S^2 \times \{0,+\infty\}$$

$$\frac{\partial(\varphi v)}{\partial t} + \nabla\cdot(\varphi v \otimes v) = g \quad \text{in} \quad S^2 \times \{0,+\infty\}$$

$$\varphi = \varphi_0, \quad v = v_0 \quad \text{on} \quad S^2 \times \{0\}$$

where the input function is defined as the initial condition of the state $a = (\varphi_0, \varphi_0 v_0)$ with the geopotential layer depth φ and the discharge (v is the velocity field), g is the Coriolis force term. The output function u predicts the state function at time t: $(\varphi_t, \varphi_t v_t)$. For this problem, we use a search space $\mathcal{S}$ with $|\mathcal{S}| = 200$ data points.

As the shallow water equation is usually chosen as a simulator of global atmospheric variables, we adopt the most common data assimilation objective (Rabier et al., 1998; Xiao et al., 2023) in the weather forecast literature defined as

$$f(u,a) = \frac{1}{2}\langle a - a_p, B^{-1}(a-a_p)\rangle + \frac{1}{2}\langle u - u_t, R^{-1}(u-u_t)\rangle$$

where a_p describes the prior estimate of the initial condition, u_t represents the ground truth function, the background kernel B and error kernel R can be computed with historical data. The objective can be defined as an inverse problem which corresponds to finding the initial condition a that generates the ground truth solution function u_t. Here we simplify the objective by not penalizing the initial condition (dropping the prior term) and assuming independence and unit variance on the solution functions using an identity kernel R), the simplified objective function $f(u) = \frac{1}{2}\langle u - u_t, u - u_t\rangle$ can be used to measure different initial conditions.

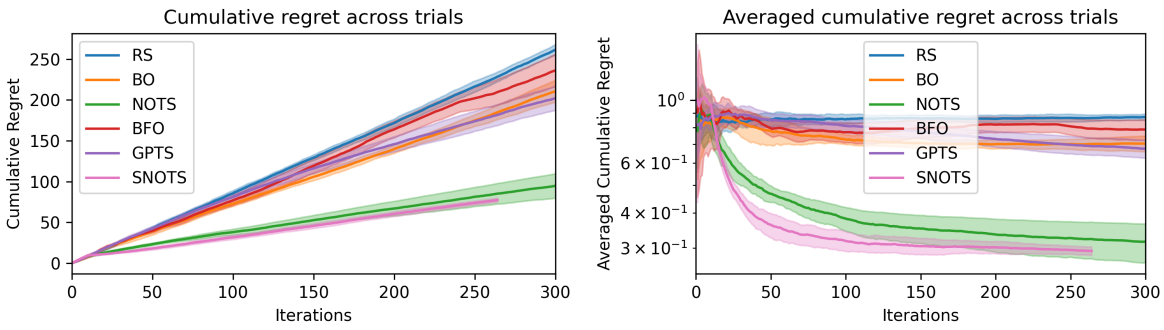

Figure 5. Cumulative regret across trials for the Darcy *flow rate* optimization problem with only the last linear layer of a single-hidden-layer FNO trained via full-batch gradient descent for NOTS (labelled as SNOTS). Results were averaged over 10 independent trials, and shaded areas represent ± 1 standard deviation.

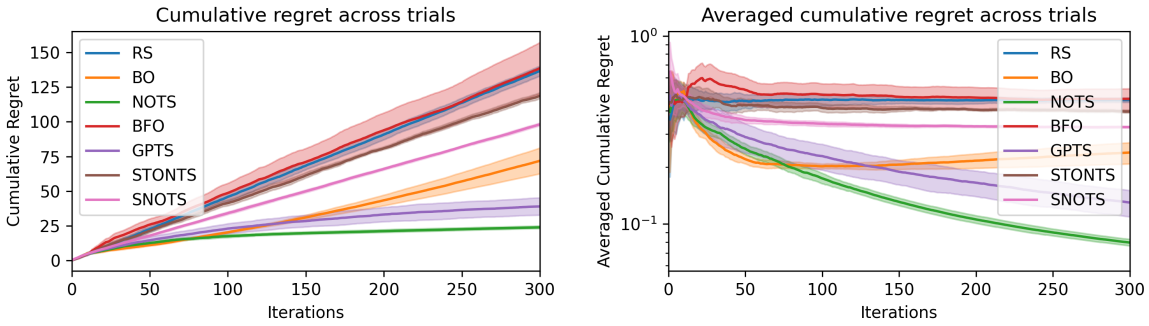

Figure 6. Cumulative regret across trials for the Darcy flow *potential power* optimization problem with only the last linear layer of a single-hidden-layer FNO trained via full-batch gradient descent for NOTS (labelled as SNOTS). Results were averaged over 10 independent trials, and shaded areas represent ± 1 standard deviation.

E.2. Algorithm Settings

NOTS was implemented using the *Neural Operator* library (Kossaifi et al., 2024). For each dataset, we selected the recommended settings for FNO models according to examples in the library. Parameters were randomly initialized using Kaiming (or He) initialization (He et al., 2015) for the network weights, sampling from a normal distribution with variance inversely proportional to the input dimensionality of each layer, while biases were initialized at zero. For all experiments, we trained the model for 10 epochs of mini-batch stochastic gradient descent with an initial learning rate of 10^{-3} and a cosine annealing scheduler. The regularization factor for the L2 penalty was set as $\lambda := 10^{-4}$. This same setting for the regularization factor was also applied to our implementation of STO-NTS.

F. Additional Results with Single-Hidden-Layer Model

More closely to the setting in our theoretical results, we tested a single-hidden-layer FNO on the Darcy flow PDE. Only the last hidden layer of the model was trained via full-batch gradient descent. The FNO was configured without any lifting layer, having only a single Fourier kernel convolution and a residual connection, as in the original formulation. The number of hidden channels was set to 2048 to approximate the infinite-width limit, and the model was trained for 2000 steps with a fixed learning rate of 10^{-3}.

The results in Figure 5 show that the algorithm with the simpler model (SNOTS) can perform well in this setting, even surpassing the performance of the original NOTS. However, in the more challenging scenario imposed by the potential power problem, we note that SNOTS struggles, only achieving mid-range performance when compared to other baselines, as shown in Figure 6. This performance drop suggests that the complexity of the potential power problem may require more accurate predictions to capture details in the output functions that might heavily influence the potential power. In general, a

quadratic objective will be more sensitive to small disturbances than a linear functional, requiring a more elaborate model.

G. Limitations and Extensions

Noise. We note that, although our result in Proposition 2 assumes a well specified noise model, it should be possible to show that the same holds for noise which is sub-Gaussian with respect to the regularization factor. The latter would allow for configuring the algorithm with any regularization factor which is at least as large as the assumed noise sub-Gaussian parameter (i.e., its variance if Gaussian distributed). However, this analysis can be quite involved and out of the immediate scope of this paper, so we leave it for further research.

Nonlinear functionals. We assumed a bounded linear functional in Proposition 2, which should cover a variety of objectives involve integrals and derivatives of the operator's output. However, this assumption may not hold for more interesting functionals, such as most objectives considered in our experiments. Similar to the case with noise, any Lipschitz continuous functional of the neural operator's output should follow a sub-Gaussian distribution (Pisier, 1986). Hence, the Gaussian approximation remains reasonable, though a more in-depth analysis would be needed to derive the exact rate of growth for the cumulative regret in these settings.

Mult-layer models. We assumed a single hidden layer neural network as the basis of our Thompson sampling algorithm. While this choice provides a simple and computationally efficient framework, it may not be optimal for all applications or datasets. For instance, in some cases, a deeper neural network with more layers might provide better performance due to increased capacity to capture complex patterns in the data. Extending our analysis to this setting involves extending the inductive proofs for the multi-layer NNGP (Lee et al., 2018; Matthews et al., 2018) to the case of neural operators. Such extension, however, may require transforming the operator layer's output back into a function in an infinite-dimensional space. In the single-hidden-layer case, the latter was not required by operating directly with the finite-dimensional input function embedding $A_{\mathbf{R}}(a)(z) \in \mathbb{R}^{d_{\mathbf{R}}}$. We conjecture that such extension should be possible, though we leave it as subject of future work.

Prior misspecification. We assumed that the true operator G_* follows the same prior as our model, which was also considered to be infinitely wide. While this assumption greatly simplifies our analysis, more practical results may be derived by considering finite-width neural operators and a true operator which might not exactly correspond to a realization of the chosen class of neural operator models. For the case of finite widths, one simple way to obtain a similar regret bound is to let the width of the network grow at each Thompson sampling iteration. The approximation error between the GP model and the finite width neural operator can potentially be bounded as $\mathcal{O}(M^{-1/2})$ (Liu et al., 2020). Hence if the sequence of network widths $\{M_t\}_{t=1}^{\infty}$ is such that $\sum_{t=1}^{\infty} \frac{1}{\sqrt{M_t}} < \infty$, a similar regret bound to the one in Proposition 2 should be possible. Furthermore, if other forms of prior misspecification need to be considered, analyzing the Bayesian cumulative regret (instead of the more usual frequentist regret), as we did, allows one to bound the resulting cumulative regret of the misspecified algorithm via the Radon-Nikodym derivative $\frac{\mathrm{d}P}{\mathrm{d}\hat{P}}$ of the true prior P with respect to the algorithm's prior probability measure $\hat{P}$. If its essential supremum $\left\| \frac{\mathrm{d}P}{\mathrm{d}\hat{P}} \right\|_{\infty}$ is bounded, then the resulting cumulative regret remains proportional to the same bound derived as if the algorithm's prior was the correct one (Russo & Van Roy, 2014).