Spatio-temporal modeling and forecasting with Fourier neural operators

Pratik Nag[1], Andrew Zammit-Mangion[1], Sumeetpal Singh[1], and Noel Cressie[1]

[1]School of Mathematics and Applied Statistics, University of Wollongong, Australia

Abstract

Spatio-temporal process models are often used for modeling dynamic physical and biological phenomena that evolve across space and time. These phenomena may exhibit environmental heterogeneity and complex interactions that are difficult to capture using traditional statistical process models such as Gaussian processes. This work proposes the use of Fourier neural operators (FNOs) for constructing statistical dynamical spatio-temporal models for forecasting. An FNO is a flexible mapping of functions that approximates the solution operator of possibly unknown linear or non-linear partial differential equations (PDEs) in a computationally efficient manner. It does so using samples of inputs and their respective outputs, and hence explicit knowledge of the underlying PDE is not required. Through simulations from a nonlinear PDE with known solution, we compare FNO forecasts to those from state-of-the-art statistical spatio-temporal-forecasting methods. Further, using sea surface temperature data over the Atlantic Ocean and precipitation data across Europe, we demonstrate the ability of FNO-based dynamic spatio-temporal (DST) statistical modeling to capture complex real-world spatio-temporal dependencies. Using collections of testing instances, we show that the FNO-DST forecasts are accurate with valid uncertainty quantification.

Keywords: Burgers' equation, Green's function, integro-difference equations, nonstationarity, partial differential equations.

1 Introduction

Spatio-temporal (ST) modeling and data analysis is often a component of physical, environmental, and biological studies. Such ST processes can exhibit complex dependencies across both space and time and hence can be difficult to model. ST statistical models fall into two main categories (Wikle et al., 2019). The first category consists of *descriptive* ST models, such as Gaussian processes (GPs; e.g., Cressie and Huang, 1999; Gneiting and Guttorp, 2007; Cressie and Wikle, 2011; Banerjee et al., 2014), where the time dimension is simply considered to be an extra dimension and treated in the same way as the spatial dimensions. This approach often faces challenges in computational scalability and its handling of complex dynamics. Moreover, these descriptive ST models typically rely on predefined classes of spatio-temporal covariance functions, which limit their flexibility.

The second category consists of *dynamical* ST (DST) models, which are constructed by specifying conditional-dependence relationships between and within spatial fields at successive time steps (e.g., Cressie and Wikle, 2011, Ch.7). These models provide an elegant framework for capturing the temporal evolution of a spatial process, making them particularly effective in applications where the underlying (physical, biological, etc.) system dynamics are complex. DST models offer a distinct advantage over descriptive ST models since they are able to encode the scientific process being modeled, which makes them well suited for forecasting tasks.

A class of DST models that has been extensively used for modeling spatio-temporal dynamical systems is the class of integro-difference equation (IDE) models (Wikle and Cressie, 1999; Wikle, 2002). These models are designed to characterize the conditional dependence between a spatial field at a future time step and its current state, through an integral operator. They have been successfully applied in various domains, such as ecology (e.g., Neubert et al., 1995), meteorology (e.g., Xu et al., 2005; Calder et al., 2011), and epidemiology (e.g., Kot and Schaffer, 1986).

Typically, DST models assume a linear operator, which simplifies their mathematical formulation. However, the linearity assumption may not be appropriate over long time horizons, where environmental heterogeneity and dynamic interactions can induce strong nonlinear dependencies. One way to address this is through quadratic nonlinear IDE-based DST models (Wikle and Hooten, 2010; Wikle and Holan, 2011). While these models are better at capturing complex nonstationary behaviors, they are less computationally efficient for making statistical inferences.

To mitigate the computational cost of fitting flexible DST models, de Bézenac et al. (2019) proposed the use of convolutional neural networks (CNNs) for learning dynamical parameters from data. Their approach was extended to stochastic IDE models by Zammit-Mangion and Wikle (2020). In both of these works, the CNN is trained offline but, once trained, parameter estimation, followed by forecasting, can be done with very little computational cost. The framework relies on a predefined governing linear partial differential equation (PDE), the solution to which is used to construct the IDE. However, many real-world applications involve heterogeneous or highly nonlinear dynamics that are not well represented through linear PDEs.

Recent advances in deep learning have revolutionized researchers' ability to model complex systems, particularly those governed by PDEs, linear or nonlinear. One such advance is the Fourier neural operator (FNO; Li et al., 2020), which are building blocks for modeling intricate physical dynamics of spatio-temporal processes. Unlike traditional solvers that explicitly solve PDEs, FNOs use training data to learn mappings between functional spaces based on samples of inputs and their respective outputs. This makes them particularly suited for spatio-temporal-forecasting applications where the underlying dynamics may not be known. Notably, they are more computationally efficient than standard PDE solvers, which makes them an attractive alternative to stochastic physical models.

In this study, we introduce a general class of stochastic DST models that uses an FNO framework to capture the temporal evolution of spatial processes; we refer to these as *FNO-DST models*. Their forecasting performance is evaluated here through simulation studies involving Burgers' equation, a nonlinear (non-stochastic) PDE whose forecasts can be obtained numerically through finite-difference approximations. Then we consider forecasting of geophysical variables. We use FNO-DSTs to forecast sea surface temperature (SST) over the North Atlantic Ocean, where it has been shown that the dynamics can be well represented through an advection-diffusion PDE (de Bézenac et al., 2019). In a second application, we use FNO-DSTs to forecast precipitation in a region over Europe, where the physical dynamics are complex and unknown. In both applications, we find that the FNO-DST model captures complex spatio-temporal interactions and that this leads to accurate forecasts.

The remainder of the paper is structured as follows: Section 2 gives the modeling and methodological details of an FNO, and it describes how we incorporate it into an FNO-DST model. Section 3 details the setup of the simulation experiment that compares the performance of the FNO-DST forecasts to other spatio-temporal forecasts, where the spatio-temporal data are generated using Burgers' equation that recall is nonlinear. Section 4 applies the FNO-DST approach to two geophysical variables, and we again compare its forecasting performance to that of other approaches. Section 5 discusses key findings and potential directions for future research. This article is accompanied by on-line Supplementary Material.

2 The FNO-DST model

In Section 2.1, we describe IDE models and a nonlinear generalization that employs neural operators. In Section 2.2, we introduce the FNO-DST model as a computationally tractable specification of the neural-operator-based model discussed in Section 2.1. Parameter estimation, specifically maximum likelihood estimation of the FNO-DST model's parameters, is presented in Section 2.3.

2.1 Classical IDE-based dynamical spatio-temporal models

Let $\mathcal{T} = [0, K]$ denote a continuous-time domain, and let $\mathcal{D} \subset \mathbb{R}^d$ be a spatial domain; in our examples, d is either 1 or 2. Although many models for environmental processes of interest assume continuous evolution on $\mathcal{D} \times \mathcal{T}$, for computational expediency their solutions are often discretized, resulting in a correlated time series of spatial processes. We adopt the same approach here: Let $\{t_1, t_2, \ldots, t_T\}$ be a finite set of time points in $\mathcal{T}$ with fixed increments $t_{k+1} - t_k = \delta > 0$, where $\delta \ll K$, and where $0 \leqslant t_1 \leqslant t_T \leqslant K$. For spatio-temporal processes governed by PDEs, a discrete-time representation can sometimes be constructed from integral solutions at these successive time points. As an example, we choose a simple linear advection-diffusion PDE, here a real-valued continuous-time spatial process $\mathcal{C}(\mathbf{s}, t)$ on $\mathcal{D} \times [t_k, t_{k+h}]$, that evolves according to,

$$\frac{\partial \mathcal{C}(\mathbf{s}, t)}{\partial t} + \gamma_1 \sum_{i=1}^{d} \frac{\partial \mathcal{C}(\mathbf{s}, t)}{\partial s_i} - \gamma_2 \sum_{i=1}^{d} \frac{\partial^2 \mathcal{C}(\mathbf{s}, t)}{\partial s_i^2} = 0; \quad \mathbf{s} \in \mathcal{D}, \quad t \in [t_k, t_{k+h}], \tag{1}$$

where the vector of parameters $\boldsymbol{\gamma} \equiv (\gamma_1, \gamma_2)^\top \in \mathbb{R} \times \mathbb{R}^+ \equiv \Gamma$, and where h is a fixed positive integer such that the end-point $t_{k+h} \in \mathcal{T}$. For fixed and known $\boldsymbol{\gamma}$, $\mathcal{D}$ bounded, and with suitable regularity conditions (Brezis and Brézis, 2011; Evans, 2022) on $\mathcal{C}(\mathbf{s}, t_k)$, the solution to (1) at the end-point t_{k+h} involves a Green's function $g(\cdot, \cdot)$, and is (Brauer, 1967; Pan, 2009):

$$\mathcal{C}(\mathbf{s}, t_{k+h}) = \mathcal{G}_{\boldsymbol{\gamma}}[\mathcal{C}(\cdot, t_k)] \equiv \int_{\mathcal{D}} g(\mathbf{s} - \mathbf{u}, h\delta; \boldsymbol{\gamma}) \mathcal{C}(\mathbf{u}, t_k) \mathrm{d}\mathbf{u}; \quad \mathbf{s} \in \mathcal{D}, k = 1, 2, \ldots, T - h. \tag{2}$$

In (2), we have used $\mathcal{G}_{\boldsymbol{\gamma}}$ to denote the integral operator that maps the input function $\mathcal{C}(\cdot, t_k) \in L^2(\mathcal{D}; \mathbb{R})$ to a corresponding output function in the same space $L^2(\mathcal{D}; \mathbb{R})$, where $L^2(\mathcal{D}; \mathbb{R})$ is the space of squared integrable functions that map from $\mathcal{D}$ to $\mathbb{R}$. In (2), the Green's function $g(\mathbf{s} - \mathbf{u}, h\delta; \boldsymbol{\gamma}) = (4\pi\gamma_2 h\delta)^{-d/2} \exp\big[-\|\mathbf{s} - \mathbf{u} - \gamma_1 h\delta\, \mathbf{1}\|^2/(4\gamma_2 h\delta)\big]$, which is derived in the Supplementary Material, Section S.1. The integral solution (2) is a deterministic linear autoregression of the spatial process with time step $h\delta$ and starting spatial field $\mathcal{C}(\mathbf{s}, t_1)$.

We now introduce two sources of uncertainty that change (2) into a stochastic process. First, we add a spatially correlated random term $\eta(\mathbf{s}, t_{k+h}), \mathbf{s} \in \mathcal{D}$, to (2), which captures variability of the underlying phenomenon being modeled and not accounted for by the integral solution. In this work, the spatial processes $\eta(\cdot, t_{k+h})$ and $\eta(\cdot, t_{k+h+k'})$ are assumed to be uncorrelated when conditioned on $Y_k(\cdot, t_k)$, for $k > 0$ and $k' < 0$ (their precise probabilistic dependence across time is made clear subsequently in (25)). Further, the processes $\{\eta(\mathbf{s}, t_k)\}_k$ are also each assumed to be at least second-order differentiable with respect to the spatial index $\mathbf{s}$, as is the solution to (1). Second, we model the parameters $\boldsymbol{\gamma}$ in (2) as random and independent of the process $\eta(\cdot, \cdot)$ entirely, across both space and time. These modeling assumptions lead to the following integro-difference equation model,

$$Y_{k+h}(\mathbf{s}) = \int_{\mathcal{D}} g(\mathbf{s} - \mathbf{u}, h\delta; \boldsymbol{\gamma}) Y_k(\mathbf{u}) \mathrm{d}\mathbf{u} + \eta(\mathbf{s}, t_{k+h}); \quad \mathbf{s} \in \mathcal{D}, \; k = 1, 2, \ldots, T - h, \tag{3}$$

where $g(\cdot, \cdot)$ is the Green's function, $Y_k(\cdot) \equiv \mathcal{C}(\cdot, t_k)$ for $k = 1, \ldots, h$, and $Y_k(\cdot) \equiv Y(\cdot, t_k)$ is the stochastic process at time point t_k. Note that, conditional on $\boldsymbol{\gamma}$, the h-step processes defined in (3) are Markov in time. For example when $h = 1$, conditional on $\boldsymbol{\gamma}$, the process $\{Y_k(\cdot)\}_k$ is Markov in time. However, we shall see that, marginally, the past and the future of the process are not independent conditional on the present.

Typically, a finite-dimensional representation of $Y_{k+h}(\cdot)$ is used. Here, we discretize $\mathcal{D}$ into basic areal units (BAUs) of common area Δ. Let $\mathcal{D}^G \subset \mathcal{D}$ denote the BAU centroids, and define the random column vectors $\mathbf{Y}_k \equiv (Y_k(\mathbf{s}) : \mathbf{s} \in \mathcal{D}^G)^\top$ and $\boldsymbol{\eta}_{k+h} \equiv (\eta(\mathbf{s}, t_{k+h}) : \mathbf{s} \in \mathcal{D}^G)^\top$. Approximating the integral in (3) with a sum, we see that (3) can be represented as the $|\mathcal{D}^G|$-dimensional vector autoregressive (VAR) process,

$$\mathbf{Y}_{k+h} = \mathbf{G}_{\boldsymbol{\gamma}} \mathbf{Y}_k + \boldsymbol{\eta}_{k+h}; \quad k = 1, 2, \ldots, T - h, \tag{4}$$

where $\mathbf{G}_{\boldsymbol{\gamma}} \equiv (g(\mathbf{s}-\mathbf{u}, h\delta; \boldsymbol{\gamma})\Delta : \mathbf{s}, \mathbf{u} \in \mathcal{D}^G)$ is a $|\mathcal{D}^G| \times |\mathcal{D}^G|$ matrix, and recall that $\boldsymbol{\gamma}$ and hence $\mathbf{G}_{\boldsymbol{\gamma}}$ are random. Now suppose that $\boldsymbol{\eta}_{k+h}$ is a mean-zero $|\mathcal{D}^G|$-dimensional Gaussian vector such that $\mathrm{var}(\boldsymbol{\eta}_{k+h} \mid \mathbf{Y}_k) = \boldsymbol{\Sigma}_{k+h}(\mathbf{Y}_k; \boldsymbol{\alpha})$ with parameters $\boldsymbol{\alpha}$. Furthermore, assume that $\mathrm{cov}(\boldsymbol{\eta}_k, \boldsymbol{\eta}_{k+h} \mid \mathbf{Y}_k) = \mathbf{0}$ for $k = (h+1), \dots, (T-h)$. Then,

$$\mathbf{Y}_{k+h} \mid \mathbf{Y}_k, \boldsymbol{\gamma} \sim \mathrm{Gau}\left(\mathbf{G}_{\boldsymbol{\gamma}}\mathbf{Y}_k, \boldsymbol{\Sigma}_{k+h}(\mathbf{Y}_k; \boldsymbol{\alpha})\right); \quad k = 1, 2, \dots, T-h. \tag{5}$$

In this work, we are mainly interested in forecasting the process $\mathbf{Y}_{k+h}$ at t_{k+h} from information up to and including t_k. Now, the forecasting distribution given only information $\mathbf{Y}_k$ at t_k is

$$p(\mathbf{Y}_{k+h} \mid \mathbf{Y}_k) = \int_{\Gamma} p(\mathbf{Y}_{k+h} \mid \mathbf{Y}_k, \boldsymbol{\gamma}) p(\boldsymbol{\gamma} \mid \mathbf{Y}_k) \mathrm{d}\boldsymbol{\gamma}. \tag{6}$$

Forecasting based on the predictive distribution given by (6) is likely to be of little value since there is scarce information on the dynamical parameters $\boldsymbol{\gamma}$ in just $\mathbf{Y}_k$ at time point k. That is, $p(\boldsymbol{\gamma} \mid \mathbf{Y}_k) \approx p(\boldsymbol{\gamma})$ in (6).

Intuitively, the dynamical parameters could be learned by incorporating information from recent history up to the time lag τ:

$$p(\mathbf{Y}_{k+h} \mid \mathbf{Y}_k, \dots, \mathbf{Y}_{k-\tau}) = \int_{\Gamma} p(\mathbf{Y}_{k+h} \mid \mathbf{Y}_k, \boldsymbol{\gamma}) p(\boldsymbol{\gamma} \mid \mathbf{Y}_k, \dots, \mathbf{Y}_{k-\tau}) \mathrm{d}\boldsymbol{\gamma}, \tag{7}$$

where the conditional distribution of $\boldsymbol{\gamma}$ is now used to sharpen the forecasting distribution. Now, as τ increases, the conditional distribution of $\boldsymbol{\gamma}$ can be expected to concentrate its probability around the true parameter, which we denote as $\boldsymbol{\gamma}^o$; that is, from (7), $p(\mathbf{Y}_{k+h} \mid \mathbf{Y}_k, \dots, \mathbf{Y}_{k-\tau}) \approx p(\mathbf{Y}_{k+h} \mid \mathbf{Y}_k, \boldsymbol{\gamma}^o)$ for large τ. This brief discussion reveals that when developing spatio-temporal forecasting models in the absence of knowledge on the governing dynamical parameters, using a value of $\tau > 0$ that includes the past, is essential, even when the latent spatio-temporal model is conditionally Markov across time.

There are several ways to incorporate this history dependence into a temporally dynamic spatio-temporal model with hidden dynamical parameters. One approach is to replace the spatial convolution in (3) with a sum (over time) of $(\tau + 1)$ spatial convolutions, resulting in,

$$Y_{k+h}(\mathbf{s}) = \sum_{j=0}^{\tau} \int_{\mathcal{D}} g_j(\mathbf{s}-\mathbf{u}, h\delta; \boldsymbol{\theta}') Y_{k-j}(\mathbf{u}) \mathrm{d}\mathbf{u} + \eta(\mathbf{s}, t_{k+h}); \quad \mathbf{s} \in \mathcal{D}, \quad k = \tau+1, \ldots, T-h, \tag{8}$$

where $\{g_j(\cdot, h\delta; \boldsymbol{\theta}')\}_{j=0}^{\tau}$ are kernels parameterized by $\boldsymbol{\theta}'$. Equation (8) reduces to (3) when $\tau = 0$ and $g_0(\cdot, \cdot) = g(\cdot, \cdot)$. As we did to obtain (4), we can discretize $\mathcal{D}$ into $\mathcal{D}^G$ to obtain a VAR process,

$$\mathbf{Y}_{k+h} = \sum_{j=0}^{\tau} \mathbf{G}_{j,\boldsymbol{\theta}'} \mathbf{Y}_{k-j} + \boldsymbol{\eta}_{k+h}; \quad k = \tau + 1, \ldots, T-h, \tag{9}$$

where $\mathbf{G}_{j,\boldsymbol{\theta}'} \equiv (g_j(\mathbf{s}-\mathbf{u}, h\delta; \boldsymbol{\theta}')\Delta : \mathbf{s}, \mathbf{u} \in \mathcal{D}^G)$ is a $|\mathcal{D}^G| \times |\mathcal{D}^G|$ matrix.

Another approach to incorporating history dependence, which relaxes the linearity structure of (8), is to make the kernel in (3) state-dependent; this leads to a state-dependent VAR process (Cressie and Wikle, 2011, Sec. 7.3.2, Zammit-Mangion and Wikle, 2020) of the form,

$$\mathbf{Y}_{k+h} = \mathbf{G}_{\boldsymbol{\theta}''}(\mathbf{Y}_k, \ldots, \mathbf{Y}_{k-\tau})\mathbf{Y}_k + \boldsymbol{\eta}_{k+h}; \quad k = \tau + 1, \ldots, T-h. \tag{10}$$

While both of these approaches lead to flexible classes of spatio-temporal models, practical implementations involve structure and restrictions on the matrices in (9) and (10), and constant regular gridding. Fitting these models to data is also computationally expensive. Fourier neural operator (FNO)-based models, discussed next, relieve the modeler of several of these drawbacks. Specifically, they possess the universal approximation property for continuous operators (Kovachki et al., 2023); they can be easily adapted to different regular griddings of $\mathcal{D}$; and, they are extremely fast to fit and forecast with, by virtue of the Fast Fourier Transform.

2.2 FNO-DST models for efficient computation

Motivated by the discussion in the previous section about hidden dynamics, spatio-temporal models for which the underlying dynamics are not prescribed should, in general, include classes that are Markov of order greater than 1. Here we consider a flexible operator that maps sample paths of the spatio-temporal process $Y(\cdot,\cdot)$ from $L^2(\mathcal{D}\times[t_{k-\tau},t_k];\mathbb{R})$ to $L^2(\mathcal{D}\times\{t_{k+h}\};\mathbb{R})$ for any $k\in\{\tau+1,\dots,T-h\}$. Note that the functional space of the input allows us to incorporate history-dependence into the operator, here by mixing convolutions over time. Specifically, at the core of our FNO defined below, is a space-time convolution of the form

$$v(\mathbf{s},t_{k+h}) = \int_{t_{k-\tau}}^{t_k}\int_{\mathcal{D}} g(\mathbf{s}-\mathbf{u},t_{k+h}-r;\boldsymbol{\gamma})v(\mathbf{u},r)\,\mathrm{d}\mathbf{u}\,\mathrm{d}r, \tag{11}$$

where $v\in L^2(\mathcal{D}\times\mathcal{T};\mathbb{R})$ and $g(\cdot,\cdot;\boldsymbol{\gamma})$ is a Green's function parameterized by $\boldsymbol{\gamma}$. Note from (11) that the operator is a convolution and hence time-invariant. Therefore, the time interval of the integral in (11) can be chosen to be $[0,\tau\delta]$, by shifting the time coordinate of $v(\cdot,\cdot)$:

$$v(\mathbf{s},t_{k+h}) = \int_0^{\tau\delta}\int_{\mathcal{D}} g(\mathbf{s}-\mathbf{u},(h+\tau)\delta-q;\boldsymbol{\gamma})v_0^{(k)}(\mathbf{u},q)\,\mathrm{d}\mathbf{u}\,\mathrm{d}q, \tag{12}$$

where the time-shifted spatio-temporal process $v_0^{(k)}(\mathbf{u},t)$ is defined as

$$v_0^{(k)}(\mathbf{u},t)\equiv v(\mathbf{u},t+t_{k-\tau}).$$

Then $v(\mathbf{s},t_{k+h})\equiv\mathcal{G}_0[v_0^{(k)}]$, where $\mathcal{G}_0[\,\cdot\,]$ is the k-invariant operator defined from (12). Without loss of generality we can therefore express the time-varying operator in (11) as a time-invariant operator on the shifted version of our spatio-temporal process, which corresponds to an implicit assumption that the underlying physical laws are temporally-invariant.

Now define $\mathcal{G}_0[\,\cdot\,]\equiv\mathcal{G}_{\boldsymbol{\theta}}[\,\cdot\,]$, where $\boldsymbol{\theta}$ are parameters that parameterize the FNO. The FNO uses a general (nonlinear) operator model for $\mathcal{G}_{\boldsymbol{\theta}}$, with universal approximation properties as described in Kovachki et al. (2023):

$$\mathcal{G}_{\boldsymbol{\theta}}[Y_0^{(k)}] = \mathcal{Q} \circ \sigma_{\mathrm{a}}(\mathcal{W}_L + \mathcal{K}_L) \circ \cdots \circ \sigma_{\mathrm{a}}(\mathcal{W}_1 + \mathcal{K}_1) \circ \mathcal{P}[Y_0^{(k)}]; \quad Y_0^{(k)} \in L^2(\mathcal{D} \times [0, \tau\delta]; \mathbb{R}), \tag{13}$$

where $Y_0^{(k)}(\cdot, t) \equiv Y(\cdot, t + t_{k-\tau})$. Equation (13) consists of three main components, which we now describe in detail.

- **Lifting operator $\mathcal{P}$:** The operator $\mathcal{P}[Y_0^{(k)}]$ is a pointwise mapping of $Y_0^{(k)}$'s graph into the higher-dimensional space $\mathcal{D}_v \subseteq \mathbb{R}^{d_v}$, where $d_v > d + 2$. Specifically, for $\mathbf{x} \equiv (\mathbf{s}^\top, t)^\top \in \mathcal{D} \times [0, \tau\delta]$, let $\mathbf{x}_Y \equiv (\mathbf{x}^\top, Y_0^{(k)}(\mathbf{s}, t))^\top$, which is a $(d+2)$-dimensional column vector. In our implementation,

$$\mathbf{v}_0(\mathbf{x}) \equiv \mathcal{P}[Y_0^{(k)}](\mathbf{x}) = \mathbf{H}_{\mathcal{P}} \mathbf{x}_Y + \mathbf{b}_{\mathcal{P}}; \quad \mathbf{x} \in \mathcal{D} \times [0, \tau\delta], \tag{14}$$

  where $\mathbf{H}_{\mathcal{P}}$ is a $d_v \times (d+2)$ matrix and $\mathbf{b}_{\mathcal{P}}$ is a d_v-dimensional vector. We collect the parameters in the operator $\mathcal{P}[\,\cdot\,]$ into the vector $\boldsymbol{\phi}_{\mathcal{P}} \equiv \left(\mathrm{vec}(\mathbf{H}_{\mathcal{P}})^\top, \mathbf{b}_{\mathcal{P}}^\top\right)^\top$.

- **Kernel integral operator $\mathcal{K}_l$ and local linear operator $\mathcal{W}_l$:** For layers $l = 1, \ldots, L$, the operators $\mathcal{K}_l[\,\cdot\,]$ and $\mathcal{W}_l[\,\cdot\,]$ are the core components of the neural operator. Specifically, $\mathcal{W}_l[\,\cdot\,]$ is the following linear transformation of the d_v-dimensional function $\mathbf{v}_{l-1}(\cdot)$,

$$\mathcal{W}_l[\mathbf{v}_{l-1}](\mathbf{x}) = \mathbf{A}_{\mathcal{W},l}\, \mathbf{v}_{l-1}(\mathbf{x}) + \mathbf{b}_{\mathcal{W},l}; \quad \mathbf{x} \in \mathcal{D} \times [0, \tau\delta], \tag{15}$$

  where $\mathbf{A}_{\mathcal{W},l}$ is a $d_v \times d_v$ matrix and $\mathbf{b}_{\mathcal{W},l}$ is a d_v-dimensional vector. We collect the parameters of $\mathcal{W}_l[\,\cdot\,]$ into the vector $\boldsymbol{\phi}_{\mathcal{W},l} \equiv \left(\mathrm{vec}(\mathbf{A}_{\mathcal{W},l})^\top, \mathbf{b}_{\mathcal{W},l}^\top\right)^\top$. The integral operator $\mathcal{K}_l[\,\cdot\,]$ is a multivariate version of a special case of (12) given by

$$\mathcal{K}_l[\mathbf{v}_{l-1}](\mathbf{x}) = \int_{\mathcal{D} \times [0, \tau\delta]} \mathbf{G}_l(\mathbf{x} - \mathbf{x}'; \boldsymbol{\phi}_{\mathcal{K},l})\, \mathbf{v}_{l-1}(\mathbf{x}')\, \mathrm{d}\mathbf{x}'; \quad \mathbf{x} \in \mathcal{D} \times [0, \tau\delta], \tag{16}$$

  where $\mathbf{G}_l(\cdot; \boldsymbol{\phi}_{\mathcal{K},l})$ is a $d_v \times d_v$ matrix of kernels with parameters collected into the vector

$\boldsymbol{\phi}_{\mathcal{K},l}$. Finally, the output of layer l, denoted $\mathbf{v}_l$, is the d_v-dimensional function,

$$\mathbf{v}_l(\mathbf{x}) \equiv \sigma_{\mathrm{a}}\left(\mathcal{W}_l[\mathbf{v}_{l-1}](\mathbf{x}) + \mathcal{K}_l[\mathbf{v}_{l-1}](\mathbf{x})\right); \quad l = 1, \ldots, L, \quad \mathbf{x} \in \mathcal{D} \times [0, \tau\delta], \tag{17}$$

where recall that $\mathbf{v}_0(\mathbf{x}) \equiv \mathcal{P}[Y_0^{(k)}](\mathbf{x})$ for some fixed k, and the map $\sigma_{\mathrm{a}}(\cdot)$ is a pointwise activation function that renders the operator nonlinear. In our case, we use a rectified linear unit (ReLU) for $\sigma_{\mathrm{a}}(\cdot)$, which returns its input vector with the negative components set to zero and with the positive components left unchanged.

- **Projection operator $\mathcal{Q}$:** This operator projects the d_v-dimensional output of the final layer back to the target output space. We implement this operator through

$$\mathcal{Q}[\mathbf{v}_L](\mathbf{x}) = \mathbf{h}_{\mathcal{Q}}^{\top}\mathbf{v}_L(\mathbf{x}) + b_{\mathcal{Q}}; \quad \mathbf{x} \in \mathcal{D} \times [0, \tau\delta], \tag{18}$$

  where $\mathbf{h}_{\mathcal{Q}}$ is a d_v-dimensional vector, and $b_{\mathcal{Q}}$ is a scalar. We collect the parameters in the operator $\mathcal{Q}[\,\cdot\,]$ into the vector $\boldsymbol{\phi}_{\mathcal{Q}} \equiv (\mathbf{h}_{\mathcal{Q}}^{\top}, b_{\mathcal{Q}})^{\top}$.

Recall that the neural operator given by (13) has universal approximation properties under suitable regulatory conditions. So too does the Fourier neural operator (FNO), where the operator $\mathcal{K}_l[\,\cdot\,]$ is expressed as a Fourier-inverse-Fourier pair. The Fourier transform of a convolution is a product of Fourier transforms, and hence,

$$\mathcal{K}_l\left[\mathbf{v}_{l-1}\right](\mathbf{x}) = \mathcal{F}^{-1}\big[\mathcal{F}[\mathbf{G}_l(\cdot; \boldsymbol{\phi}_{\mathcal{K},l})]\mathcal{F}[\mathbf{v}_{l-1}]\big](\mathbf{x}), \tag{19}$$

where the Fourier operator and the inverse Fourier operator are

$$U(\boldsymbol{\omega}) \equiv \mathcal{F}[u](\boldsymbol{\omega}) = \int_{\mathcal{D}\times[0,\tau\delta]} u(\mathbf{x})\exp(-\iota\boldsymbol{\omega}^{\top}\mathbf{x})\mathrm{d}\mathbf{x}; \quad \boldsymbol{\omega} \in \mathbb{R}^{d+1}, \tag{20}$$

$$\mathcal{F}^{-1}[U](\mathbf{x}) = \tfrac{1}{(2\pi)^{d+1}}\int_{\mathbb{R}^{d+1}} U(\boldsymbol{\omega})\exp(\iota\boldsymbol{\omega}^{\top}\mathbf{x})\mathrm{d}\boldsymbol{\omega}; \quad \mathbf{x} \in \mathcal{D} \times [0, \tau\delta], \tag{21}$$

with $\boldsymbol{\omega}$ denoting spatio-temporal angular frequency and $u(\cdot)$ a function in $L^2(\mathcal{D} \times [0, \tau\delta]; \mathbb{R})$.

Definition: Equation (13), together with the expressing of $\mathcal{K}_l[\,\cdot\,]$ as a Fourier–inverse-Fourier transformation pair in (19), defines the *Fourier neural operator* (FNO) that maps functions in $L^2(\mathcal{D} \times [0, \tau\delta]; \mathbb{R})$ to functions in $L^2(\mathcal{D} \times \{(h+\tau)\delta\}; \mathbb{R})$.

Fast computation is key in any neural operator, and in the FNO we obtain it by using the Cooley-Tukey Fast Fourier Transform (FFT) (Cooley and Tukey, 1965). Specifically, instead of using (20) and (21) we start with the discrete Fourier transform (DFT):

$$U(\boldsymbol{\kappa}) = \sum_{\mathbf{x} \in \mathcal{D}^G \times \mathcal{T}^G} u(\mathbf{x}) \exp\left(-2\pi\iota \sum_{j=1}^{d+1} \tfrac{\kappa_j x_j}{m_j}\right); \quad \boldsymbol{\kappa} \in \Omega, \tag{22}$$

$$u(\mathbf{x}) = \tfrac{1}{m_1 \cdots m_{d+1}} \sum_{\boldsymbol{\kappa} \in \Omega} U(\boldsymbol{\kappa}) \exp\left(2\pi\iota \sum_{j=1}^{d+1} \tfrac{\kappa_j x_j}{m_j}\right); \quad \mathbf{x} \in \mathcal{D}^G \times \mathcal{T}^G, \tag{23}$$

where $\mathcal{T}^G \equiv \{0, \delta, \ldots, \tau\delta\}$; m_j is the number of grid spacings in the the j-th spatio-temporal coordinate, so $m_{d+1} = \tau$; and $\Omega \equiv \times_{j=1}^{d+1}(0, \ldots, m_j - 1)$. Then, for computational efficiency, we shrink Ω to $\tilde{\Omega} \equiv \times_{j=1}^{d+1}(0, \ldots, \tilde{m}_j - 1)$, where $\tilde{m}_j \ll m_j,\ j = 1, \ldots d+1$. Hence, the product of Fourier transforms reduces to $\mathcal{O}(\prod_{j=1}^{d+1} \tilde{m}_j)$ matrix-vector multiplications of dimension d_v that can be computed very quickly; see the Supplementary Material, Section S.1.3, where the FFT algorithm is briefly described.

In our implementation of the FNO, we do not declare a parametric form for $\mathbf{G}_l(\,\cdot\,; \boldsymbol{\phi}_{\mathcal{K},l})$ in Equation (19) before finding its DFT. Instead, we define $\mathcal{F}[\mathbf{G}_l]$ in (19) as a spectral-weights tensor of complex numbers that is conjugately symmetric (since signals are taken to be real-valued), directly in the Fourier domain. We denote them by $\mathbf{F}_{\mathcal{K},l} \in \mathbb{C}^{d_v \times d_v \times \tilde{m}_1 \times \cdots \times \tilde{m}_{d+1}}$, in the high-dimensional space of complex numbers, which are trainable parameters that we collect into the vector $\boldsymbol{\phi}_{\mathcal{K},l} \equiv \mathrm{vec}(\mathbf{F}_{\mathcal{K},l})$. They give the FNO the flexibility it needs to represent complex spatio-temporal dynamics.

Let $\boldsymbol{\theta} = \left(\boldsymbol{\phi}_{\mathcal{P}}^\top, \boldsymbol{\phi}_{\mathcal{W},1}^\top, \boldsymbol{\phi}_{\mathcal{K},1}^\top, \ldots, \boldsymbol{\phi}_{\mathcal{W},L}^\top, \boldsymbol{\phi}_{\mathcal{K},L}^\top, \boldsymbol{\phi}_{\mathcal{Q}}^\top\right)^\top$ be the set of trainable parameters associated with the neural operator in (13). Let $\mathbf{G}(\,\cdot\,; \boldsymbol{\theta})$ denote the neural operator implemented with the DFT (22), which conveniently also admits a finite-dimensional parameterization. Now, for $k = \tau + 1, \dots, T - h$, define the vector-valued stochastic time series $\mathbf{Y}_{(k-\tau):k} \equiv \{\mathbf{Y}_{k-\tau}, \dots, \mathbf{Y}_k\}$ where the $|\mathcal{D}^G|$-dimensional vector $\mathbf{Y}_k$ is $Y_k(\cdot)$ evaluated at $\mathcal{D}^G$. Then we can rewrite the output of the operator defined in (13) on the discretized domain as $\mathbf{G}(\mathbf{Y}_{(k-\tau):k}; \boldsymbol{\theta})$. More details on these discretized tensors are discussed in the Supplementary Material, Section S.1.2. The discretized evaluation, $\mathbf{G}(\,\cdot\,; \boldsymbol{\theta})$, of the operator $\mathcal{G}_{\boldsymbol{\theta}}[\,\cdot\,]$ in (13) captures the spatio-temporal dynamics of a potentially complex physical process as follows:

$$\mathbf{Y}_{k+h} = \mathbf{G}(\mathbf{Y}_{(k-\tau):k}; \boldsymbol{\theta}) + \boldsymbol{\eta}_{k+h}; \quad k = \tau + 1, \dots, T - h. \tag{24}$$

This should be compared to Equation (10), where the form of the matrix $\mathbf{G}_{\boldsymbol{\theta}}(\cdot)$ is relatively constrained. Since the process model defined in (24) is a dynamic spatio-temporal (DST) model, we refer to this as the *FNO-DST* model.

We now turn to the additive terms $\{\boldsymbol{\eta}_{\tau+1+h}, \dots, \boldsymbol{\eta}_T\}$. We model the covariance matrix $\boldsymbol{\Sigma}_{k+h}(\mathbf{Y}_k; \boldsymbol{\alpha})$ of $\boldsymbol{\eta}_{k+h}$ using a squared exponential covariance function with spatially and temporally varying standard deviation, given by

$$\mathrm{cov}\left(\eta(\mathbf{s}, t_{k+h}), \eta(\mathbf{s}', t_{k+h}) \mid \mathbf{Y}_k\right) = \sigma_{k+h}(\mathbf{s}; \mathbf{Y}_k)\, \sigma_{k+h}(\mathbf{s}'; \mathbf{Y}_k)\, \exp\left(-\frac{\|\mathbf{s} - \mathbf{s}'\|^2}{2\alpha_{\mathrm{r}}^2}\right), \tag{25}$$

for $\mathbf{s}, \mathbf{s}' \in \mathcal{D}^G$. We cater for heteroscedasticity by modeling the standard deviation parameter at time t_{k+h} as a function of the process $\mathbf{Y}_k$. Specifically, we let

$$\left(\sigma_{k+h}(\mathbf{s}_1; \mathbf{Y}_k), \dots, \sigma_{k+h}(\mathbf{s}_N; \mathbf{Y}_k)\right)^\top = f_{\mathrm{NN}}(\mathbf{Y}_k; \boldsymbol{\alpha}_{\mathrm{NN}}); \quad \mathbf{s}_1, \dots, \mathbf{s}_N \in \mathcal{D}^G, \tag{26}$$

where $f_{\mathrm{NN}}(\cdot)$ is a shallow feed-forward neural network that takes $\mathbf{Y}_k$ as inputs and produces the vector of standard deviations across the spatial grid. This specifies $\boldsymbol{\Sigma}_{k+h}(\mathbf{Y}_k; \boldsymbol{\alpha})$, where $\boldsymbol{\alpha} \equiv (\mathrm{vec}(\boldsymbol{\alpha}_{\mathrm{NN}})^\top, \alpha_{\mathrm{r}})^\top$ are further trainable parameters in the FNO-DST model.

2.3 Likelihood computation, training, and forecasting

Consider N independent realisations from a spatio-temporal process model on $\mathcal{D} \times [0, K]$ generated using, for example, random initial conditions or dynamical parameters. Assume that, for $i = 1, \ldots, N$, we observe each process at time points $\{t_1, \ldots, t_T\}$ with temporal step size δ, to yield the discretized time series $\{\mathbf{Y}_1^{(i)}, \ldots, \mathbf{Y}_T^{(i)}\}$. Under the assumption of conditional independence given the history of τ observations, the likelihood function for the unknown parameters $(\boldsymbol{\theta}^\top, \boldsymbol{\alpha}^\top)^\top$ can be written as

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\alpha}) = \prod_{i=1}^{N} \prod_{k=\tau+1}^{T-h} p\Big(\mathbf{Y}_{k+h}^{(i)} \mid \mathbf{Y}_{(k-\tau):k}^{(i)}, \boldsymbol{\theta}, \boldsymbol{\alpha}\Big) = \prod_{i=1}^{N} \prod_{k=\tau+1}^{T-h} \mathcal{L}_k^{(i)}(\boldsymbol{\theta}, \boldsymbol{\alpha}), \tag{27}$$

where $\mathbf{Y}_{(k-\tau):k}^{(i)} \equiv \{\mathbf{Y}_{k-\tau}^{(i)}, \ldots, \mathbf{Y}_k^{(i)}\}$. From the Gaussian VAR model given by (24),

$$\mathbf{Y}_{k+h}^{(i)} \mid \mathbf{Y}_{(k-\tau):k}^{(i)}, \boldsymbol{\theta}, \boldsymbol{\alpha} \sim \mathrm{Gau}\Big(\mathbf{G}(\mathbf{Y}_{(k-\tau):k}^{(i)}; \boldsymbol{\theta}), \boldsymbol{\Sigma}_{k+h}^{(i)}(\mathbf{Y}_k; \boldsymbol{\alpha})\Big); i = 1, \ldots, N, k = \tau + 1, \ldots, T - h, \tag{28}$$

where $\boldsymbol{\Sigma}_{k+h}^{(i)}(\mathbf{Y}_k; \boldsymbol{\alpha})$ is the covariance matrix of $\boldsymbol{\eta}_{k+h}^{(i)}$ parameterized by $\boldsymbol{\alpha}$. We maximize $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\alpha})$ using a standard optimisation algorithm with an Adam learning schedule (Kingma, 2014). Let $(\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{\alpha}})$ denote the estimated parameters that maximize $\mathcal{L}$. Substituting these estimates into Equation (28) yields the forecasting distribution for replicate i:

$$\mathbf{Y}_{k+h}^{(i)} \mid \mathbf{Y}_{(k-\tau):k}^{(i)}, \widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{\alpha}} \sim \mathrm{Gau}(\mathbf{G}(\mathbf{Y}_{(k-\tau):k}^{(i)}; \widehat{\boldsymbol{\theta}}), \boldsymbol{\Sigma}_{k+h}^{(i)}(\mathbf{Y}_k; \widehat{\boldsymbol{\alpha}})); i = 1, \ldots, N, k = \tau + 1, \ldots, T - h, \tag{29}$$

where $\mathbf{G}(\mathbf{Y}_{(k-\tau):k}^{(i)}; \widehat{\boldsymbol{\theta}})$ is the estimated predictive mean and $\boldsymbol{\Sigma}_{k+h}^{(i)}(\mathbf{Y}_k; \widehat{\boldsymbol{\alpha}})$ is the corresponding estimated predictive covariance matrix.

3 Simulation experiments

In this section, we conduct a comparative analysis of the forecasting performance of our proposed FNO-DST approach against current state-of-the-art approaches using two-dimensional (1-D space $\times$ time) simulations from Burgers' equation, a nonlinear PDE whose solution can be obtained numerically. The comparison uses performance metrics such as mean squared prediction error (MSPE), prediction interval coverage probability (PICP), and mean prediction interval width (MPIW); for more details on PICP and MPIW, see Lakshminarayanan et al. (2017) and the Supplementary Material, Section S.2.2. In our simulation experiments, we compare forecasts from the FNO-DST model to those of the following:

- *Convolutional Long Short-Term Memory* (ConvLSTM) neural network model (Shi et al., 2015), which employs convolutional transitions within the inner LSTM module of the neural network. To train this model we replace $\mathbf{G}(\,\cdot\,;\boldsymbol{\theta})$ in (24) with ConvLSTM blocks.

- *Space-Time DeepKriging* (STDK) model (Nag et al., 2023), which performs spatio-temporal prediction by learning the relationship between embedded spatio-temporal locations $(\mathbf{s}^\top, t)^\top$, for $\mathbf{s} \in \mathcal{D}^G$ and $t \in \{t_1, \ldots, t_T\}$, and the underlying process $Y(\mathbf{s}, t)$, using neural networks. Nag et al. (2023) have shown that this approach outperforms another approach based on Gaussian regression with Vecchia's approximation (Katzfuss and Guinness, 2021) for prediction of complex spatio-temporal processes.

- *Persistence* model, which is a simple model where spatial observations from the last time point of the current sequence are used as forecasts for future time points at the corresponding locations. This approach can be thought of as providing a low-bar benchmark that other models' forecasts should improve upon.

Reproducible code is available from `https://github.com/pratiknag/FNO-DSTM-Code`.

3.1 Two-dimensional implementation

We consider the two-dimensional (1-D space $\times$ time) non-steady-state Burgers' equation (Hopf, 1950), which is a nonlinear PDE that has been used in various applications such as modeling the one-dimensional flow of a viscous fluid. A solution to Burgers' equation can be obtained numerically, and can serve as a gold standard against which the forecasts from the four models given above can be compared. Burgers' equation for the evolution of a spatio-temporal field $\{u(s,t) : s \in [0,1], t \in [0,1]\}$ (i.e., $d = 1$), is written here as:

$$\frac{\partial u(s,t)}{\partial t} + \frac{1}{2}\frac{\partial u^2(s,t)}{\partial s} = \gamma \frac{\partial^2 u(s,t)}{\partial s^2}; \quad (s,t) \in (0,1) \times (0,1), \tag{30}$$

with initial condition $u(s,0) = a_0(s)$, for $s \in [0,1]$, and periodic boundary conditions $u(0,t) = u(1,t)$, for $t \in [0,1]$. Here $\gamma \in [0,1]$ is the viscosity parameter. In order to solve (30) for a given γ and $a_0(\cdot)$, we rely on the Euler numerical method (Biswas et al., 2013), where the PDE is discretized onto a fine grid as follows: We replace temporal derivatives with a forward difference of step size $\Delta t = 1/10000$, we replace spatial first order derivatives with a backward difference of step size $\Delta s = 1/(2^{11} - 1)$, and we replace spatial second order derivatives with a symmetric second difference of the same step size. That is, the discretization of (30) results in,

$$\frac{u(s_i,\tilde{t}_{j+1}) - u(s_i,\tilde{t}_j)}{\Delta t} + u(s_i,\tilde{t}_j)\frac{u(s_i,\tilde{t}_j) - u(s_{i-1},\tilde{t}_j)}{\Delta s} = \gamma \frac{u(s_{i+1},\tilde{t}_j) - 2u(s_i,\tilde{t}_j) + u(s_{i-1},\tilde{t}_j)}{(\Delta s)^2}, \tag{31}$$

with $u(s_1,\tilde{t}_j) = u(s_{2^{11}},\tilde{t}_j)$, for each $i = 1,\ldots,2^{11}$, $j = 1,\ldots,10001$. Hence Burgers' equation, discretized, can be rewritten as a one-step-ahead forecast by rearranging terms in (31) as follows:

$$\begin{aligned} u(s_i,\tilde{t}_{j+1}) = u(s_i,\tilde{t}_j) - \tfrac{\Delta t}{\Delta s} u(s_i,\tilde{t}_j)\left(u(s_i,\tilde{t}_j) - u(s_{i-1},\tilde{t}_j)\right) + \\ \gamma \tfrac{\Delta t}{(\Delta s)^2}\left(u(s_{i+1},\tilde{t}_j) - 2u(s_i,\tilde{t}_j) + u(s_{i-1},\tilde{t}_j)\right), \end{aligned} \tag{32}$$

where the forecast at $\tilde{t}_{j+1}$ is made from local spatial knowledge at the current time $\tilde{t}_j$. (Note, we use $\tilde{t}_j$ for the simulator time-step to distinguish it from the FNO-DST time-step t_j.) Since we have periodic boundary conditions, we simulate $a_0(\cdot)$ on a unit circle once, from a wrapped Gaussian process with mean zero and covariance obtained from a chordal Matérn covariance function (Guinness and Fuentes, 2016) with variance 1.0, length scale 1.0, and smoothness 2.0. We consider two simulation settings: one has random $\gamma \equiv \gamma^{(1)} \sim \text{Uniform}(0.05, 0.7)$, and the other has fixed $\gamma \equiv \gamma^{(2)} = 0.4$.

In this simulation study, we consider two different neural operators of the form

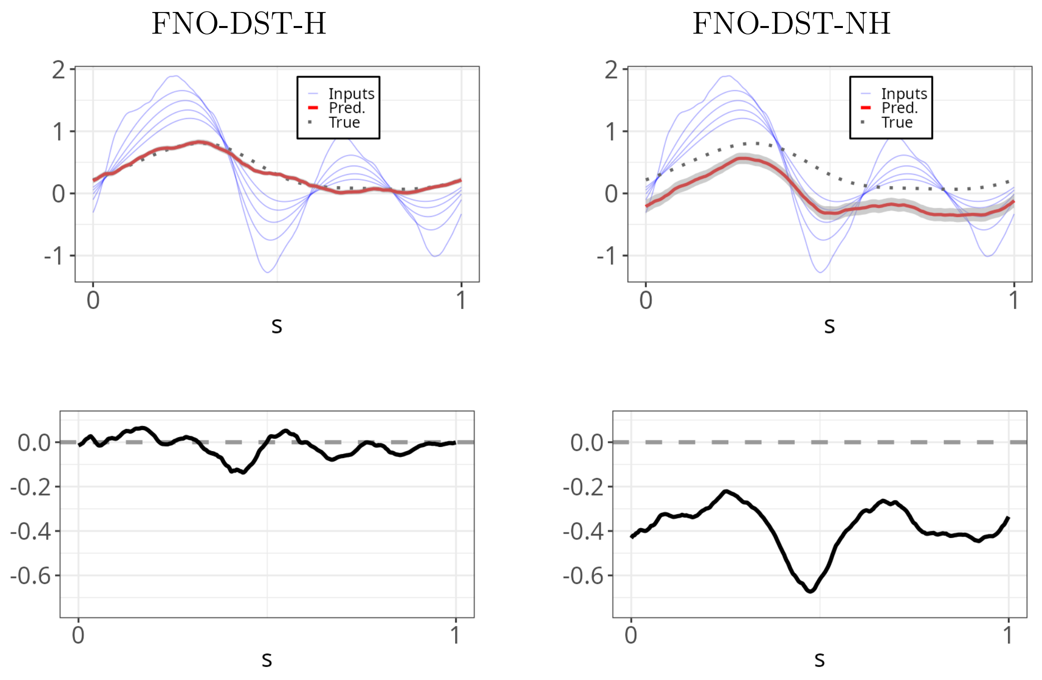
$$\mathcal{G}_{\boldsymbol{\theta}} : L^2([0,1]^2 \times [0, \tau\delta]; \mathbb{R}) \mapsto L^2([0,1]^2 \times \{(\tau + h)\delta\}; \mathbb{R}).$$

We define the first neural operator, which we label FNO-DST-H (FNO-DST with 'history'), to have $\tau = 4$. We define the second neural operator, which we label FNO-DST-NH (FNO-DST with 'no history'), to have $\tau = 0$. For both operators we let $\delta = 0.1$, and $h = 5$ (5-step ahead forecast). We consider these two neural operators to give evidence to the claim in Section 2.1 that one needs to condition on past trajectories to sharpen the forecasts when the dynamics are variable (in the case of $\gamma^{(1)}$), even though (31) suggests a Markov model.

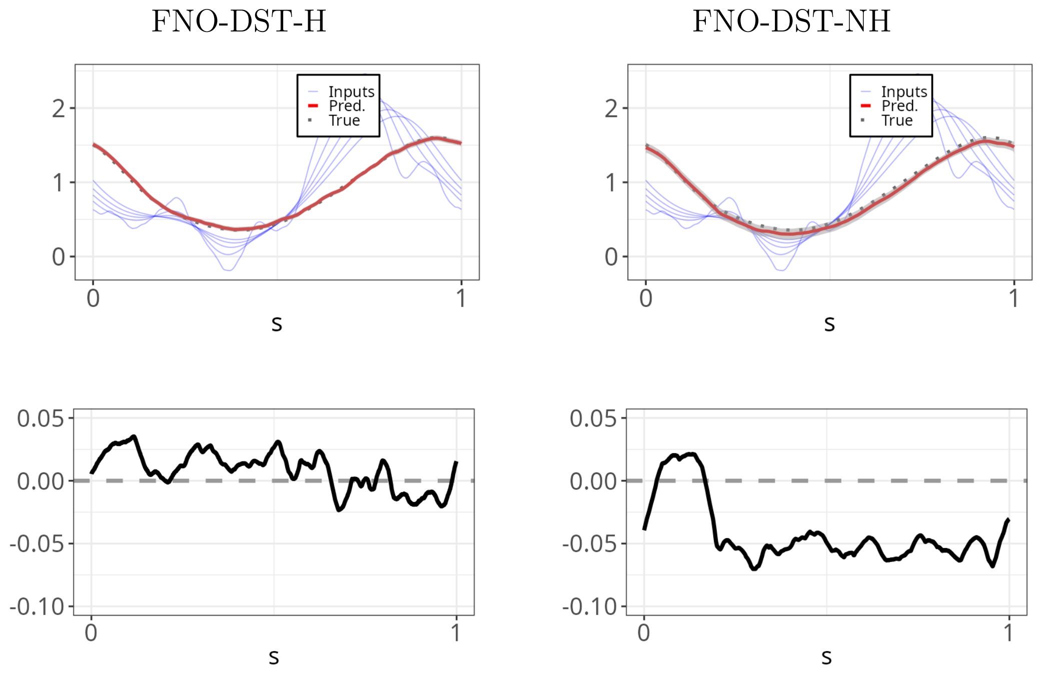
For each neural operator and simulation setting, we simulate 1000 independent instances on the domain $\mathcal{D}^G = \{(i-1)/(2^{11}-1) : i = 1, \ldots, 2^{11}\}$ with time step $\Delta_t = 0.0001$ using (32). The neural operators use a coarser time step of $\delta = 0.1$. Here, we let $\{t_1, \ldots, t_T\} = \{0.1, 0.2, \ldots, 1.0\}$, and, we evaluate the forecast performance at $t_{10} = 1$ from information up to $t_5 = 0.5$. Note that $\mathcal{D}^G$ defines a uniform discretization of the unit interval with 2^{11} equispaced points, chosen to facilitate efficient FFT computations as discussed in the Supplementary Material, Section S.1.2. Our temporal and spatial discretization are used to construct the vectors $\{\mathbf{Y}_k : k = 1, \ldots, T\}$, which are needed for optimizing the log-likelihood defined in (27). For each neural operator and simulation setting we use $N = 950$ instances for training; the remaining $N_0 = 50$ instances are used for testing.

3.2 Simulation results

Figure 1 presents a comparison between FNO-DST-H and FNO-DST-NH for the first testing instance with random $\gamma = \gamma^{(1)}$ and the first testing instance with fixed $\gamma = \gamma^{(2)}$. In the



(a) Predicted fields (top row) and errors (bottom row) for random $\gamma = \gamma^{(1)}$.



(b) Predicted fields (top row) and errors (bottom row) for fixed $\gamma = \gamma^{(2)}$.

Figure 1: First testing instance out of $N_0 = 50$ testing instances for, respectively, forecasts from FNO-DST-H and FNO-DST-NH models at $T = 10$. (a): $\gamma = \gamma^{(1)}$, (b): $\gamma = \gamma^{(2)}$. In the top panels of (a) and (b), the current and past four steps (blue lines), the true spatial field at $T = 10$ (black dotted line), and the forecasts (red line) with 95% prediction intervals (gray shaded area), are shown.

Table 1: Performance metrics for comparing FNO-DST-H and FNO-DST-NH for random $\gamma^{(1)}$ and fixed $\gamma^{(2)}$ in the simulation experiment of Section 3.1.

	Model	MSPE	PICP	MPIW
$\gamma = \gamma^{(1)}$	FNO-DST-NH	0.381	0.122	0.202
	FNO-DST-H	0.001	0.97	0.088
$\gamma = \gamma^{(2)}$	FNO-DST-NH	0.001	0.97	0.122
	FNO-DST-H	0.0007	0.98	0.065

Supplementary Material, Section S.2.2, the same types of comparisons are provided for two additional testing instances, which show patterns consistent with this first testing instance.

Table 1 provides a numerical comparison between FNO-DST-H and FNO-DST-NH using forecasting performance metrics MSPE, PICP, and MPIW. These metrics are obtained by averaging over replications, space, and forecast times (See the Supplementary Material, Section S.2.1) at $t = 1$. From the comparative table and plots, it can be observed that with γ given by random $\gamma^{(1)}$, the inputs from the current and past $\tau = 4$ steps during training enables FNO-DST-H to learn more effectively the underlying governing dynamics and produce a superior 5-step-ahead forecast of the spatial field $u(\cdot, 1)$ when compared to FNO-DST-NH. For γ given by fixed $\gamma^{(2)}$, training on a single current value gives comparable results to training on the current and past four values. This supports our argument in Section 2.1, advocating for the inclusion of temporal history rather than relying on a single snapshot for forecasting a spatial field when the underlying PDE parameters are variable.

To conclude this section, we compare the FNO-DST-H model with the three competing models given in the introduction to this section for simulation setting $\gamma = \gamma^{(1)}$. The comparison is shown visually in Figure 2 and using the forecasting performance metrics MSPE, PICP, and MPIW in Table 2. From the results, it can be seen that the FNO-DST-H produces valid forecasts (as do ConvLSTM and STDK), but it is more accurate, and it yields an MPIW that is smaller by an order of magnitude.

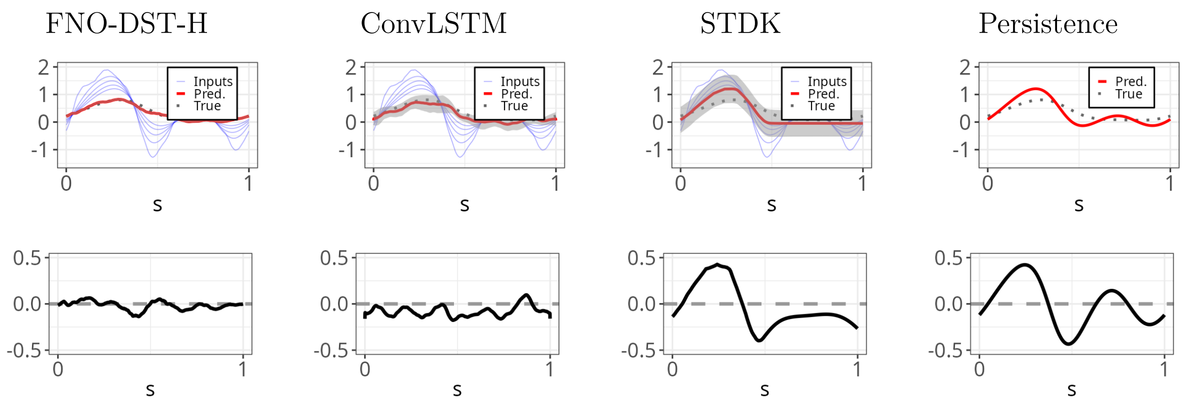


Figure 2: Forecast fields (top row by model) and corresponding prediction errors (bottom row by model) for the first testing instance out of $N_0 = 50$ testing instances. This visualization illustrates both the forecast quality at $T = 10$ and error distribution across space for the simulation setting with random $\gamma = \gamma^{(1)}$. In the top panels, the current and past four steps (blue lines), the true spatial field at $T = 10$ (black dotted line), and the forecasts (red line) with 95% prediction intervals (gray shaded area), are shown.

Table 2: Performance metrics for competing models for the simulation setting $\gamma = \gamma^{(1)}$ and FNO-DST-H, described in Section 3.1.

Model	MSPE	PICP	MPIW
FNO-DST-H	0.001	0.97	0.088
ConvLSTM	0.054	0.98	0.574
STDK	0.075	0.99	1.013
Persistence	0.089	-	-

4 FNO-DST forecasting in geophysical applications

In this section, we conduct a comparative analysis of FNO-DST forecasting applied to two geophysical data sets. We also compare these forecasts with those obtained from another dynamical forecasting approach based on the physically motivated integro-difference equation discussed in Section 2.1. A neural-network version of the linear integro-difference equation (CNN-IDE; Zammit-Mangion and Wikle, 2020) is competitive when those dynamics are expected (as in SST forecasting), but it performs less well when little is known about the underlying dynamics.

4.1 Application to SST data

In this study, we use sea surface temperature (SST) data generated by the NEMO (Nucleus for European Modeling of the Ocean) simulation framework (Madec and the NEMO System Team, 2024). NEMO is an advanced modeling system designed for oceanic simulations, employing physically motivated equations to model both regional and global ocean circulation. As in climate reanalysis, it also employs historical observations to produce an assimilated representation of oceanic states, which ensures that its data closely align with actual SST measurements.

We utilize NEMO SST data obtained from the `GLOBAL_ANALYSIS_FORECAST_PHY_001_024` product, which is provided by the Copernicus Marine Environment Monitoring Service (CMEMS). From this source, we aggregated the data onto a 0.5-degree longitude-latitude grid. Following Zammit-Mangion and Wikle (2020), we concentrated our analysis and forecasting of SST on the North Atlantic Ocean. The region is divided into 19 distinct zones, each spanning a standardized 64×64 grid $\mathcal{D}^G \equiv \left\{\left(\frac{i-1}{63}, \frac{j-1}{63}\right) : i, j \in \{1, \ldots, 64\}\right\}$.

In this study, we focus on leveraging the FNO-DST framework for modeling the operator mapping defined in Section 2.2 as

$$\mathcal{G}_{\boldsymbol{\theta}} : L^2([0,1]^2 \times [0, \tau\delta]; \mathbb{R}) \mapsto L^2([0,1]^2 \times \{(\tau + h)\delta\}; \mathbb{R}).$$

To train our models, we use the initial 4000 days of data available in this data product, covering the period from 27 December 2006 to 16 December 2017, with the 19 zones treated as replicates. The final 10 days of data are reserved for testing. Consequently there are $4000 \times 19 = 76000$ images that are used for training purposes. Figure 3 (motivated from figures in Zammit-Mangion and Wikle, 2020) presents the delineation of these 19 zones,

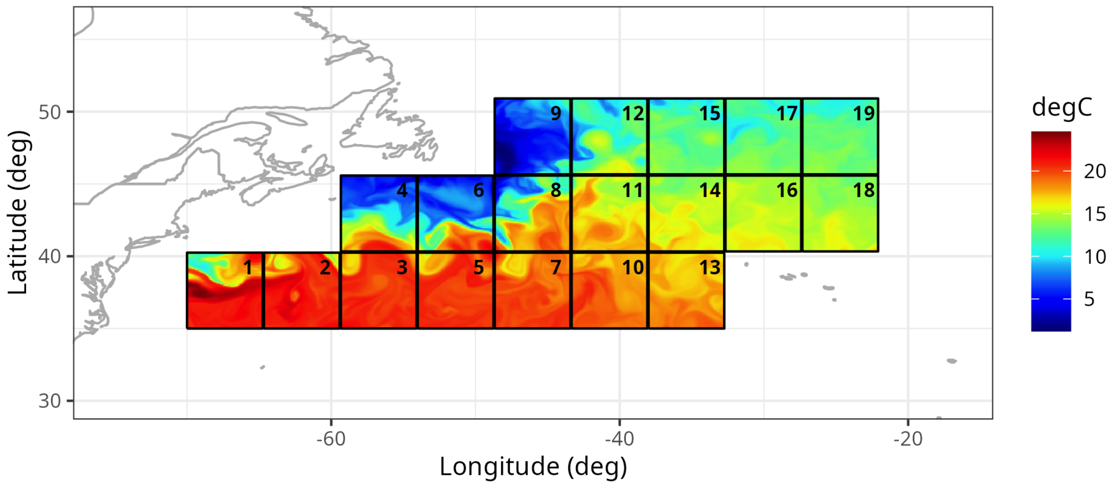


Figure 3: North Atlantic Ocean sea surface temperature (SST) NEMO data on December 27, 2006, with 19 zones numbered and delineated by the black boxes.

illustrated with the SST data on December 27, 2006 (the first day in the data product). This subdivision into the 19 zones is primarily for computational efficiency; however, it does not significantly affect the approach, as the dominant dynamics within each zone can still be used to forecast the future from the recent past. We take $\tau = 2$ (recent past), $h = 3$ (3-step-ahead forecast) and discretize the temporal dimension to the daily resolution. Hence we forecast 3 days ahead from the current day and two preceding days.

Figure 4 gives a visual comparison of the 3-day-ahead forecasts of the FNO-DST model and the CNN-IDE model. It is well established that SST data sets exhibit advection-diffusion properties, which the CNN-IDE model effectively incorporates through prior physical knowledge. Table 3 uses the same metrics used in Tables 1 and 2 (Section 3), showing an expanded comparison that includes the other three forecasts (ConvLSTM, STDK, and Persistence) that were compared in the simulations. While CNN-IDE forecasts of SST were best, the FNO-DST forecasts still performed well, even though no explicit geophysical knowledge was

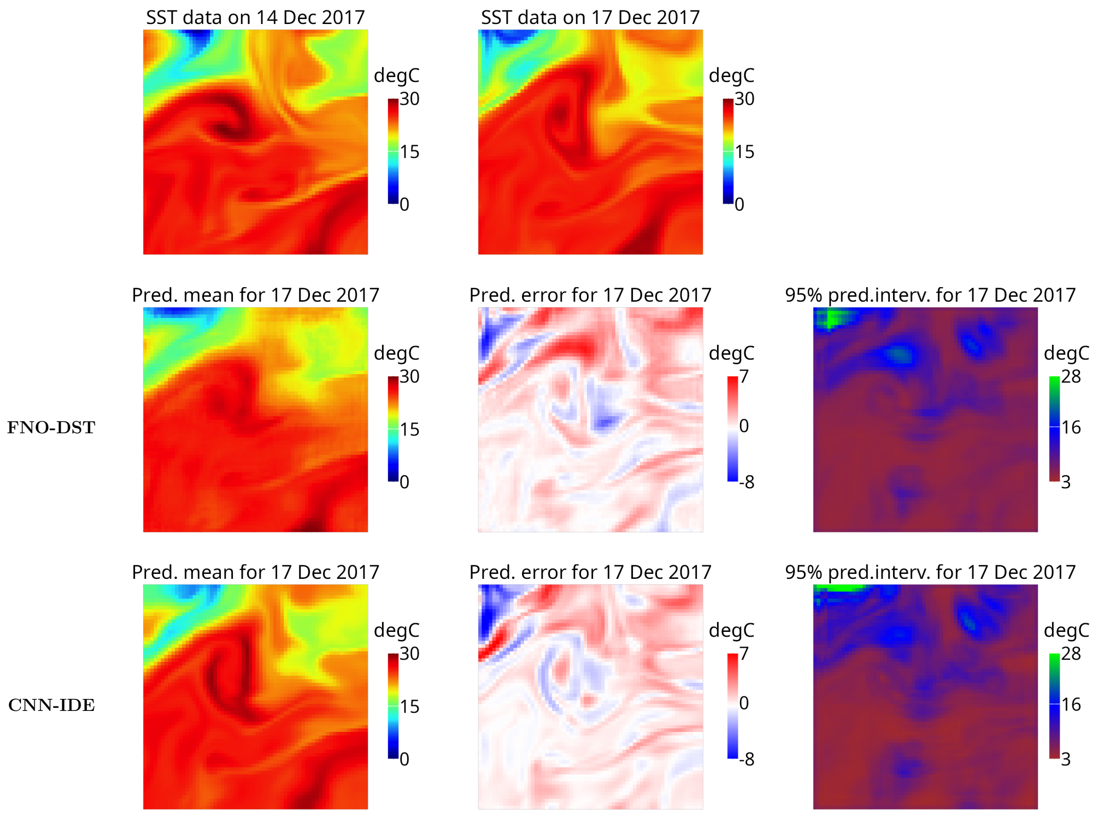


Figure 4: Visual comparison of FNO-DST and CNN-IDE forecasts for zone 3 in the North Atlantic Ocean. Forecasts of SST on 17 Dec 2017 were made from data on 12 Dec, 13 Dec, 14 Dec 2017. The top row shows 14 Dec 2017 data and its evolution to 17 Dec 2017, 3 days later. The middle row (FNO-DST) and bottom row (CNN-IDE) show the forecast results for 17 Dec 2017 (Pred. mean, Pred. error, and pixelwise 95% Pred. interval width).

incorporated. This demonstrates its forecasting potential for applications where the dynamics are scarcely known, as is the case for the daily precipitation data in the next subsection.

Table 3: Forecasting performance metrics of SST forecasts for competing models described in Sections 2.1 and 3, averaged over 19 zones and over all 3-day-ahead forecasts from 17 December 2017 to 26 December 2017. (The FNO-DST model includes 'history' from the current time back to the two previous times.)

Model	MSPE	PICP	MPIW
FNO-DST	1.56	0.95	5.77
CNN-IDE	1.34	0.95	5.65
ConvLSTM	1.66	0.96	5.85
STDK	2.27	0.93	3.98
Persistence	2.45	-	-

4.2 Application to precipitation data

The second application uses daily precipitation data collected from weather stations distributed across Western and Central Europe (Toreti, 2014). The data set includes a comprehensive observational record of daily precipitation measurements spanning the period from 01 January 2014, to 31 December 2024. Figure 5 illustrates the spatial distribution of the weather stations, where the subregion highlighted in red was selected for our forecasting study. To preprocess the data, a 10-day moving average was applied to a weather station's daily data to mitigate short-term fluctuations and fill in small gaps in the data. Each time series was then standardized to ensure consistent scaling across spatial locations and time points, with the sample mean and sample standard deviation saved in order to produce forecasts back on the original scale. Finally, spatio-temporal interpolation using the Deep-Kriging approach of Nag et al. (2023) was employed to generate precipitation fields on a 64×64 standardized grid $\mathcal{D}^G \equiv \left\{\left(\frac{i-1}{63}, \frac{j-1}{63}\right) : i, j \in \{1, \ldots, 64\}\right\}$ defined over the highlighted subregion in Figure 5, namely $[15°\mathrm{E}, 25°\mathrm{E}] \times [45°\mathrm{N}, 55°\mathrm{N}]$, for all days between 01 January 2014 and 31 December 2024 inclusive. Further details on the dataset, its preprocessing, and

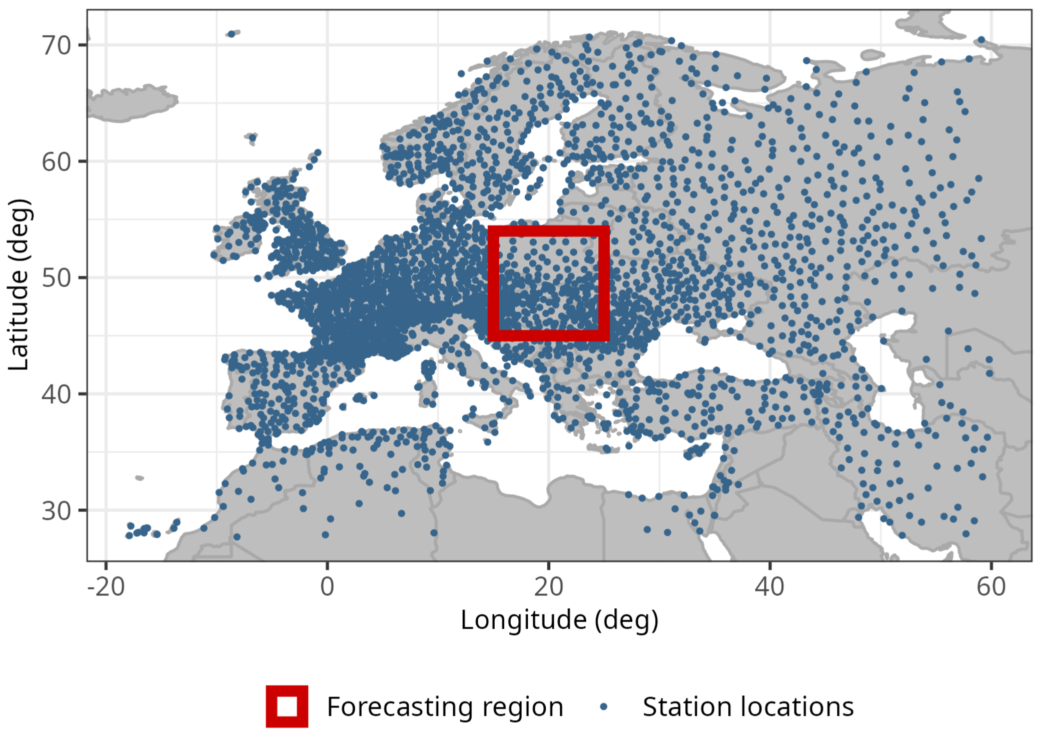


Figure 5: Distribution of weather-station locations across western and central Europe. The region highlighted in red has been specifically chosen for forecasting purposes.

the interpolation framework are provided in Nag (2025). To train the models, we use the first 3200 days of the gridded spatial dataset, spanning the period from 1 January 2014 to 5 November 2023. The remaining 421 days, from 6 November 2023 to 31 December 2024, are reserved for testing. As in Section 4.1, we take $\tau = 2$ (recent past), $h = 3$ (3-step-ahead forecast). Note that, unlike the SST application that had 19 replicates, here there is only one replicate, and so training is on 3200 images.

Figure 6 gives a visual comparison of the 3-day-ahead forecasts from the FNO-DST model and the CNN-IDE model. Table 4 uses the same metrics and forecasting approaches as in Table 3. The results clearly demonstrate that the FNO-DST forecasting of precipitation outperforms the other models' forecasts across all evaluation metrics. For the precipitation application, the dynamics are relatively poorly understood, and CNN-IDE comes third in performance. As shown in Figure 6, the FNO-DST model is able to forecast an East–West

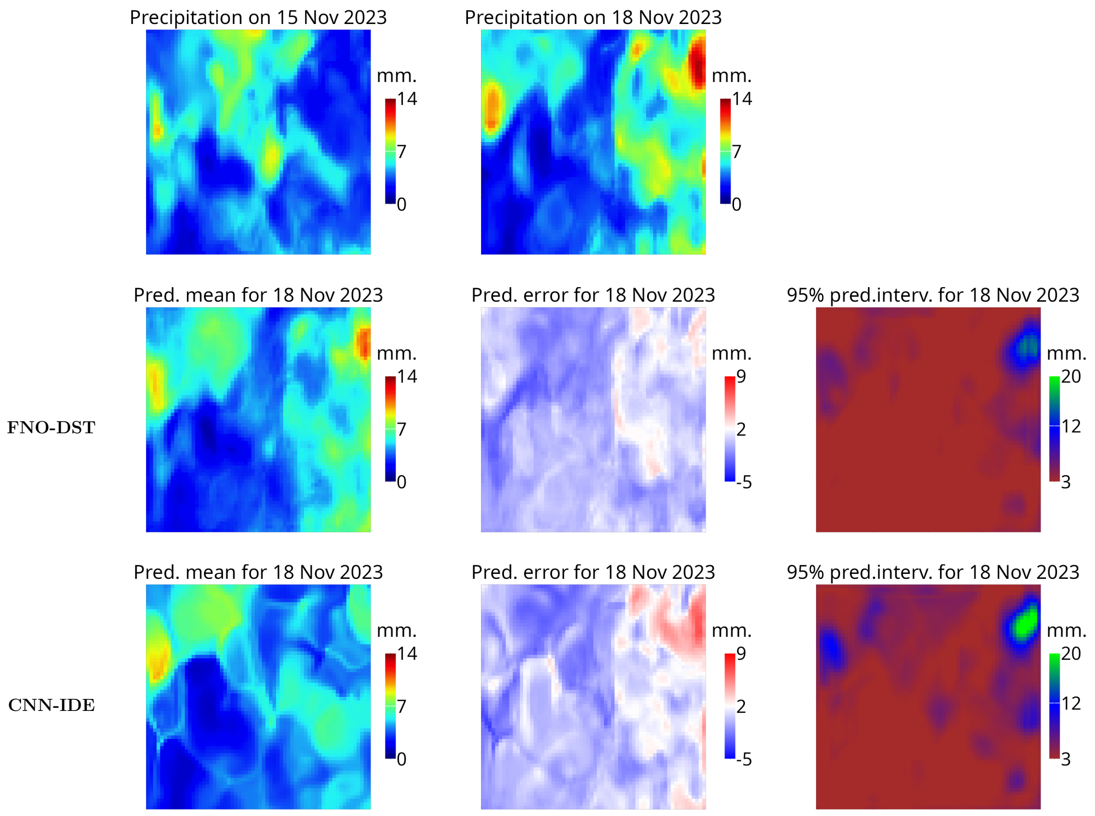

Precipitation on 15 Nov 2023
Precipitation on 18 Nov 2023
Pred. mean for 18 Nov 2023
Pred. error for 18 Nov 2023
95% pred.interv. for 18 Nov 2023
FNO-DST
CNN-IDE
mm.
14
7
0
9
2
-5
20
12
3

Figure 6: Visual comparison of FNO-DST and CNN-IDE forecasts for the precipitation data over the highlighted region shown in Figure 5. Forecasts on 18 Nov 2023 were made from data on 13 Nov, 14 Nov, 15 Nov 2023. The top row shows 15 Nov 2023 data and its evolution to 18 Nov 2023, 3 days later. The middle row (FNO-DST) and bottom row (CNN-IDE) show the forecast results (Pred. mean, Pred. error, and pixelwise 95% Pred. interval width).

precipitation front that is similar to what actually occurs on 18 November 2023. The CNN-IDE model, on the other hand, does not reproduce this feature. The simulation results in Section 3 and the applications in this section highlight the inferential and computational efficiency of FNO-DST forecasting in diverse settings.

Table 4: Performance metrics of different competing models described in Sections 3 and 4, averaged over all 3-step-ahead forecasts at all spatial locations and all days from 06 Nov 2023 to 31 Dec 2024.

Model	MSPE	PICP	MPIW
FNO-DST	0.32	0.96	1.95
CNN-IDE	0.46	0.96	2.43
ConvLSTM	0.38	0.94	2.23
STDK	0.82	0.92	1.97
Persistence	0.84	-	-

5 Conclusions

In this work, we develop a framework for dynamic spatio-temporal (DST) modeling where Fourier Neural Operators (FNOs) are used to forecast entire spatial fields in a computationally efficient manner. Discretized partial- and integro-differential-equation models often struggle to accommodate nonlinear interactions and complex dynamics inherent in many environmental and physical systems, and this affects their forecasting performance. Our approach leverages the FNO's ability to learn and then to approximate mappings between spatial fields across time using fast, parallelizable computations and without requiring explicit knowledge of the underlying dynamical mechanisms.

We evaluated the forecasting performance of the proposed FNO-DST model defined in Section 2.2 through a combination of simulation studies and real-world applications. In the simulation experiments based on Burgers' equation, which is a nonlinear PDE, the FNO-DST model demonstrated strong forecasting skill in capturing the nonlinear dynamics governing the system. In the real-data applications, we applied the model to two distinct spatio-temporal geophysical settings: forecasting sea surface temperature (SST) over the North

Atlantic Ocean, and forecasting precipitation over a region in Europe. In the SST-forecasting application, where physical PDE-based models have been previously developed, the FNO-DST model performed almost as well as a PDE-based model. In the precipitation-forecasting application, where the underlying dynamics are considerably more complex and less-well characterized, the FNO-DST model was able to learn the dynamics and provide superior forecasts according to several performance metrics. Our results demonstrate the ability of the FNO-based framework to adapt to a wide range of spatio-temporal systems for which the dynamics are unknown or little known.

Beyond its forecasting ability, a key strength of the FNO-based framework is its computational efficiency and scalability, both during training and post-training. This offers a substantial computational advantage for real-time forecasting applications, especially in high-dimensional domains, where traditional statistical and mechanistic models can become computationally prohibitive, or perhaps the traditional models are over-simplified for computational efficiency. Just as importantly, by embedding the FNO within a statistical model, the framework naturally accommodates uncertainty quantification, which is critical for decision making.

There remain several promising avenues for future research. First, extensions to incorporate physical constraints or conservation laws directly into the FNO framework will further improve forecasting skill while enhancing model interpretability. Second, developing a Bayesian hierarchical statistical formulation of the FNO-DST model will improve uncertainty quantification as data and parameter uncertainties are included. Finally, while the current work focuses on spatial lattices, adapting the methodology for irregular spatial domains, non-Euclidean spaces, or multivariate spatio-temporal processes represent important directions for broadening the applicability of the FNO-DST approach to forecasting.

Acknowledgement

This material is based upon work supported by the Air Force Office of Scientific Research under award number FA2386-23-1-4100. Nag and Zammit-Mangion also acknowledge travel support from the JRC to visit their research centre, which motivated the application case study presented in Section 4.2.

A Notation

$\mathcal{D}$**:** Continuous spatial domain.

$\mathcal{T}$**:** Continuous temporal domain.

$\mathbf{s}$**:** The spatial coordinate, $\mathbf{s} \in \mathcal{D}$.

t**:** The temporal coordinate, $t \in \mathcal{T}$.

$\mathcal{C}(\cdot, \cdot)$**:** The deterministic real-valued continuous-time PDE-based spatial process.

$\boldsymbol{\gamma}$**:** Parameters associated with the underlying PDE.

$g(\cdot, \cdot), g_j(\cdot, \cdot)$**:** Green's functions in different scenarios.

$\eta(\cdot, \cdot)$**:** Spatially correlated, temporally uncorrelated, random innovation term.

$Y(\cdot, \cdot)$**:** The spatio-temporal stochastic process being modeled.

$Y_k(\cdot) \equiv Y(\cdot, t_k)$**:** The stochastic (spatial) process evaluated at time point t_k.

$\{Y_k(\cdot)\}_k \equiv \{Y(\mathbf{s}, t_k) : \mathbf{s} \in \mathcal{D}\}_{k=1}^{T}$**:** The time series of spatial processes defined on the domain $\mathcal{D}$.

$\mathcal{D}^G$**:** BAU-discretization of $\mathcal{D}$.

$\boldsymbol{\eta}_{k+h}$**:** $\eta(\cdot, t_{k+h})$ evaluated on $\mathcal{D}^G$.

$\boldsymbol{\Sigma}_{k+h}(\mathbf{Y}_k; \boldsymbol{\alpha})$**:** The covariance matrix associated with $\boldsymbol{\eta}_{k+h}$, which is a function of $\mathbf{Y}_k$ and parameterized by $\boldsymbol{\alpha}$.

$\mathbf{Y}_k$**:** $Y_k(\cdot)$ evaluated on $\mathcal{D}^G$.

$\mathbf{G}_{\boldsymbol{\gamma}}, \mathbf{G}_{j,\boldsymbol{\theta}'}, \mathbf{G}_{\boldsymbol{\theta}''}$**:** BAU-discretized Green's functions of the operators in (3), (8), and (10), with parameters $\boldsymbol{\gamma}$, $\boldsymbol{\theta}'$, and $\boldsymbol{\theta}''$, respectively.

τ**:** Lag length used to form the conditioning history.

h**:** Forecast horizon.

$\mathbf{x} \equiv (\mathbf{s}^\top, t)^\top$**:** The spatio-temporal coordinate, $\mathbf{x} \in \mathcal{D} \times [0, \tau\delta]$.

$\mathcal{G}_{\boldsymbol{\theta}}$**:** The FNO operator parameterized by $\boldsymbol{\theta}$.

$\mathcal{P}, \mathcal{K}, \mathcal{W}, \mathcal{Q}$**:** The lifting operator, kernel integral operator, local linear operator, and projection operator, as defined in (14), (16), (15), and (18), respectively.

$\mathcal{F}, \mathcal{F}^{-1}$**:** The Fourier and inverse Fourier operators, as defined in (20) and (21), respectively.

$\mathbf{G}(\cdot; \boldsymbol{\theta})$**:** The BAU-discretized output of the operator $\mathcal{G}_{\boldsymbol{\theta}}$.

$\mathbf{Y}_{(k-\tau):k}$**:** Vector-valued stochastic time series $\{\mathbf{Y}_{k-\tau}, \dots, \mathbf{Y}_k\}$.

$\{\mathbf{Y}_k^{(i)}\}_k$**:** BAU-discretized time series for the i-th independent replicate.

$\boldsymbol{\Sigma}_{k+h}^{(i)}(\mathbf{Y}_k; \boldsymbol{\alpha})$**:** The covariance matrix associated with $\mathbf{Y}_{k+h}^{(i)}$.

$\mathcal{T}^G \equiv \{0, \delta, \dots, \tau\delta\}$**:** The temporal discretization of the interval $[0, \tau\delta]$ into $\tau + 1$ equally spaced points with step size δ.

$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\alpha})$: The likelihood function of the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\alpha}$, where $\boldsymbol{\alpha}$ is the vector of parameters that describe the spatial dependence of the innovation spatial field.

References

Banerjee, S., Carlin, B. P., and Gelfand, A. E. (2014). *Hierarchical Modeling and Analysis for Spatial Data.* Chapman and Hall, CRC Press, Boca Raton, FL.

Biswas, B. N., Chatterjee, S., Mukherjee, S., and Pal, S. (2013). A discussion on euler method: A review. *Electronic Journal of Mathematical Analysis and Applications*, 1(2):2090–2792.

Brauer, F. (1967). Green's functions for singular ordinary differential operators. *Canadian Journal of Mathematics*, 19:571–582.

Brezis, H. and Brézis, H. (2011). *Functional Analysis, Sobolev Spaces and Partial Differential Equations.* Vol. 2, Springer, New York, NY. See Chapter 8.

Calder, C. A., Berrett, C., Shi, T., Xiao, N., and Munroe, D. K. (2011). Modeling space-time dynamics of aerosols using satellite data and atmospheric transport model output. *Journal of Agricultural, Biological, and Environmental Statistics*, 16:495–512.

Cooley, J. and Tukey, J. (1965). An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, 19(90):297–301.

Cressie, N. and Huang, H.-C. (1999). Classes of nonseparable, spatio-temporal stationary covariance functions. *Journal of the American Statistical Association*, 94(448):1330–1340.

Cressie, N. and Wikle, C. K. (2011). *Statistics for Spatio-Temporal Data.* Wiley, Hoboken, NJ.

de Bézenac, E., Pajot, A., and Gallinari, P. (2019). Deep learning for physical processes: Incorporating prior scientific knowledge. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124009.

Evans, L. C. (2022). *Partial Differential Equations, Ch.5.* Vol. 19, American Mathematical Society, Providence, RI.

Gneiting, T. and Guttorp, P. (2007). Continuous parameter space-time models. *Environmetrics*, 18:412–429.

Guinness, J. and Fuentes, M. (2016). Isotropic covariance functions on spheres: Some properties and modeling considerations. *Journal of Multivariate Analysis*, 143:143–152.

Hopf, E. (1950). The partial differential equation $u_t + uu_x = \mu u_{xx}$. *Communications on Pure and Applied Mathematics*, 3(3):201–230.

Katzfuss, M. and Guinness, J. (2021). A general framework for Vecchia approximations of Gaussian processes. *Statistical Science*, 36(1):124–141.

Kingma, D. P. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980.*

Kot, M. and Schaffer, W. M. (1986). Discrete-time growth-dispersal models. *Mathematical Biosciences*, 80:109–136.

Kovachki, N., Li, Z., Liu, B., Azizzadenesheli, K., Bhattacharya, K., Stuart, A., and Anandkumar, A. (2023). Neural operator: Learning maps between function spaces with applications to PDEs. *Journal of Machine Learning Research*, 24(89):1–97.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in Neural Information Processing Systems*, 30:1–12.

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Stuart, A., Bhattacharya, K., and Anandkumar, A. (2020). Multipole graph neural operator for parametric partial differential equations. *Advances in Neural Information Processing Systems*, 33:6755–6766.

Madec, G. and the NEMO System Team (2024). NEMO ocean engine reference manual. https://doi.org/10.5281/zenodo.1464816.

Nag, P. (2025). Spatio-temporal deepkriging in pytorch: A supplementary application to precipitation data for interpolation and probabilistic forecasting. *arXiv preprint arXiv:2509.12708*.

Nag, P., Sun, Y., and Reich, B. J. (2023). Spatio-temporal deepkriging for interpolation and probabilistic forecasting. *Spatial Statistics*, 57:100773.

Neubert, M. G., Kot, M., and Lewis, M. A. (1995). Dispersal and pattern formation in a discrete-time predator-prey model. *Theoretical Population Biology*, 48(1):7–43.

Pan, E. (2009). *Green's Functions*. pp. 13-46, Springer, New York, NY.

Shi, X., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-K., and Woo, W.-C. (2015). Convolutional LSTM network: A machine learning approach for precipitation nowcasting. *Advances in Neural Information Processing Systems*, 28:1–9.

Toreti, A. (2014). Gridded Agro-meteorological data in Europe. http://data.europa.eu/89h/jrc-marsop4-7-weather_obs_grid_2019. Downloaded on 24 January 2025.

Wikle, C. K. (2002). A kernel-based spectral model for non-Gaussian spatio-temporal processes. *Statistical Modelling*, 2(4):299–314.

Wikle, C. K. and Cressie, N. (1999). A dimension-reduced approach to space-time Kalman filtering. *Biometrika*, 86(4):815–829.

Wikle, C. K. and Holan, S. H. (2011). Polynomial nonlinear spatio-temporal integro-difference equation models. *Journal of Time Series Analysis*, 32(4):339–350.

Wikle, C. K. and Hooten, M. B. (2010). A general science-based framework for dynamical spatio-temporal models. *Test*, 19:417–451.

Wikle, C. K., Zammit-Mangion, A., and Cressie, N. (2019). *Spatio-Temporal Statistics with R*. CRC Press, Boca Raton, FL.

Xu, K., Wikle, C. K., and Fox, N. I. (2005). A kernel-based spatio-temporal dynamical model for nowcasting weather radar reflectivities. *Journal of the American Statistical Association*, 100(472):1133–1144.

Zammit-Mangion, A. and Wikle, C. K. (2020). Deep integro-difference equation models for spatio-temporal forecasting. *Spatial Statistics*, 37:100408.

Supplementary Material: Spatio-temporal modeling and forecasting with Fourier neural operators

Pratik Nag[1], Andrew Zammit-Mangion[1], Sumeetpal Singh[1], and Noel Cressie[1]

[1]School of Mathematics and Applied Statistics, University of Wollongong, Australia

Here we provide additional details about the methodology (Green's function), the discretization, the computation (Fast Fourier Transform), and the simulation experiments (performance metrics, extra figures).

S.1 Further technical details

S.1.1 Derivation of the Green's function in Equation (2) in the main text

Let the Fourier transform of $\mathcal{C}(\mathbf{s}, t)$ be $\mathcal{F}[\mathcal{C}]$, where

$$\mathcal{F}[\mathcal{C}](\boldsymbol{\kappa}, t) \equiv \widehat{\mathcal{C}}(\boldsymbol{\kappa}, t) = \int_{\mathbb{R}^d} e^{-\iota \boldsymbol{\kappa}^\top \mathbf{s}}\, \mathcal{C}(\mathbf{s}, t)\, \mathrm{d}\mathbf{s},$$

with inverse

$$\mathcal{F}^{-1}[\widehat{\mathcal{C}}](\mathbf{s}, t) \equiv \mathcal{C}(\mathbf{s}, t) = \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} e^{\iota \boldsymbol{\kappa}^\top \mathbf{s}}\, \widehat{\mathcal{C}}(\boldsymbol{\kappa}, t)\, \mathrm{d}\boldsymbol{\kappa},$$

where ι is the imaginary number in the complex plane, and $\boldsymbol{\kappa} = (\kappa_1, \ldots, \kappa_d)^\top$. Taking the Fourier transform of Equation (1) in the main text and, using linearity,

$$\frac{\partial}{\partial t}\mathcal{F}[\mathcal{C}](\boldsymbol{\kappa}, t) + \gamma_1 \sum_{i=1}^{d} \mathcal{F}\Big[\frac{\partial \mathcal{C}}{\partial s_i}\Big] - \gamma_2 \sum_{i=1}^{d} \mathcal{F}\Big[\frac{\partial^2 \mathcal{C}}{\partial s_i^2}\Big] = 0, \quad \text{for } (\boldsymbol{\kappa}, t) \in \mathbb{R}^d \times [t_k, t_k + \tau]. \qquad \text{(S.1.1)}$$

For each $i = 1, \ldots, d$,

$$\mathcal{F}\Big[\frac{\partial \mathcal{C}}{\partial s_i}\Big](\boldsymbol{\kappa}, t) = \int_{\mathbb{R}^d} e^{-\iota \boldsymbol{\kappa}^\top \mathbf{s}}\, \frac{\partial \mathcal{C}}{\partial s_i}(\mathbf{s}, t)\, \mathrm{d}\mathbf{s}.$$

Integrating by parts with respect to s_i we get,

$$\mathcal{F}\Big[\frac{\partial \mathcal{C}}{\partial s_i}\Big](\boldsymbol{\kappa}, t) = \iota \kappa_i \widehat{\mathcal{C}}(\boldsymbol{\kappa}, t), \qquad \mathcal{F}\Big[\frac{\partial^2 \mathcal{C}}{\partial s_i^2}\Big](\boldsymbol{\kappa}, t) = -\kappa_i^2 \widehat{\mathcal{C}}(\boldsymbol{\kappa}, t).$$

Note that the integration-by-parts step uses the fact that a boundary term vanishes as $|s_i| \to \infty$. This holds under the mild assumption that, for each fixed t, $\mathcal{C}(\cdot, t)$ and $(\partial \mathcal{C}/\partial s_i)(\cdot, t)$ belong to $L^1(\mathbb{R}^d; \mathbb{R})$, which in particular implies $\mathcal{C}(\mathbf{s}, t) \to 0$ as $\|\mathbf{s}\| \to \infty$ (Brezis and Brézis, 2011; Evans, 2022).
Substituting these expressions into Equation (S.1.1), we obtain

$$\frac{\partial}{\partial t}\widehat{\mathcal{C}}(\boldsymbol{\kappa}, t) + \gamma_1 \sum_{i=1}^{d} \iota \kappa_i \widehat{\mathcal{C}}(\boldsymbol{\kappa}, t) + \gamma_2 \sum_{i=1}^{d} \kappa_i^2 \widehat{\mathcal{C}}(\boldsymbol{\kappa}, t) = 0,$$

or equivalently,

$$\frac{\partial \widehat{\mathcal{C}}}{\partial t}(\boldsymbol{\kappa}, t) = -\Big(\iota\gamma_1 \mathbf{1}^\top \boldsymbol{\kappa} + \gamma_2 \|\boldsymbol{\kappa}\|^2\Big)\, \widehat{\mathcal{C}}(\boldsymbol{\kappa}, t),$$

where $\mathbf{1}$ is the d-dimensional vector of 1s. We denote the equation above simply as $du/dt = -au$, where $u(t) \equiv \widehat{\mathcal{C}}(\boldsymbol{\kappa}, t)$, and $a \equiv \iota\gamma_1 \mathbf{1}^\top \boldsymbol{\kappa} + \gamma_2 \|\boldsymbol{\kappa}\|^2$, for a fixed $\boldsymbol{\kappa}$. Then,

$$u(t) \propto e^{-at}.$$

At the starting value $t = t_k$ in the interval $[t_k, t_k + \tau]$, we write $\widehat{\mathcal{C}}(\boldsymbol{\kappa}, t_k) \equiv \widehat{\mathcal{C}}_0(\boldsymbol{\kappa})$. Hence

$$u(t_k + \tau) = e^{-a\tau}\, \widehat{\mathcal{C}}_0(\boldsymbol{\kappa}),$$

or in terms of $\widehat{\mathcal{C}}(\boldsymbol{\kappa}, \cdot)$,

$$\widehat{\mathcal{C}}(\boldsymbol{\kappa}, t_k + \tau) = \exp\Big\{ -(\iota\gamma_1 \mathbf{1}^\top \boldsymbol{\kappa} + \gamma_2 \|\boldsymbol{\kappa}\|^2)\tau \Big\}\, \widehat{\mathcal{C}}_0(\boldsymbol{\kappa}).$$

Its inverse Fourier transform is,

$$\mathcal{C}(\mathbf{s}, t_k + \tau) = \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} e^{\iota \boldsymbol{\kappa}^\top \mathbf{s}}\, \widehat{\mathcal{C}}_0(\boldsymbol{\kappa})\, \exp\Big\{ -(\iota\gamma_1 \mathbf{1}^\top \boldsymbol{\kappa} + \gamma_2 \|\boldsymbol{\kappa}\|^2)\tau \Big\}\, \mathrm{d}\boldsymbol{\kappa}.$$

Using $\widehat{\mathcal{C}}_0(\boldsymbol{\kappa}) = \int_{\mathbb{R}^d} e^{-\iota \boldsymbol{\kappa}^\top \mathbf{u}}\, \mathcal{C}(\mathbf{u}, t_k)\, \mathrm{d}\mathbf{u}$, and applying Fubini's theorem, we have

$$\mathcal{C}(\mathbf{s}, t_k + \tau) = \int_{\mathbb{R}^d} \left[\frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} \exp\Big\{ \iota \boldsymbol{\kappa}^\top (\mathbf{s} - \mathbf{u}) - \iota\gamma_1 \tau \mathbf{1}^\top \boldsymbol{\kappa} - \gamma_2 \tau \|\boldsymbol{\kappa}\|^2 \Big\}\, \mathrm{d}\boldsymbol{\kappa} \right] \mathcal{C}(\mathbf{u}, t_k)\, \mathrm{d}\mathbf{u}.$$

Hence, for $\boldsymbol{\gamma} = (\gamma_1, \gamma_2)^\top$, where $\gamma_2 > 0$, the Green's function is

$$g(\mathbf{r}, \tau; \boldsymbol{\gamma}) = \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} \exp\Big\{ \iota \boldsymbol{\kappa}^\top \mathbf{r} - \iota\gamma_1 \tau \mathbf{1}^\top \boldsymbol{\kappa} - \gamma_2 \tau \|\boldsymbol{\kappa}\|^2 \Big\} \mathrm{d}\boldsymbol{\kappa},$$

which is real-valued and can be evaluated through a standard Gaussian integral to be:

$$g(\mathbf{r}, \tau; \boldsymbol{\gamma}) = \frac{1}{(4\pi\gamma_2\tau)^{d/2}} \exp\Big(-\frac{\|\mathbf{r} - \tau\gamma_1 \mathbf{1}\|^2}{4\gamma_2\tau} \Big), \quad \tau > 0.$$

Finally, substituting $\tau = h\delta$ yields the desired result in Equation (2) of the main text.

S.1.2 Further details on the discretization of the FNO input

In the notation of Section 2.2 from the main text, the input field $Y_0^{(k)}(\cdot)$, for $k \in \{\tau + 1, \dots, T - h\}$, evaluated on $\mathcal{D}^G \times \mathcal{T}^G$, where $\mathcal{T}^G \equiv \{0, \delta, \dots, \tau\delta\}$, is a tensor of size $m_1 \times \cdots \times m_{d+1}$. Similarly, the output of layer l, $\mathbf{v}_l(\cdot)$, evaluated on $\mathcal{D}^G \times \mathcal{T}^G$, is a tensor of size $d_v \times m_1 \times \cdots \times m_{d+1}$, for $l = 1, \dots, L$. This tensor formulation lends itself to efficient computation on graphics and tensor processing units. In particular, the local operations $\mathcal{P}[\,\cdot\,]$, $\mathcal{Q}[\,\cdot\,]$, and $\mathcal{W}_l[\,\cdot\,]$, for $l = 1, \dots, L$, simply reduce to multiple batch multiplications and additions that can be done in an embarrassingly parallel fashion for each point in $\mathcal{D}^G \times \mathcal{T}^G$.

S.1.3 Computation of discrete Fourier transforms

The tensor formulation of Section 2.2 from the main text allows us to take advantage of the discrete Fourier transform (DFT). Computation of Equations (21) and (22) have complexity $\mathcal{O}((\prod_{j=1}^{d+1} \tilde{m}_j)^2)$, where $\tilde{m}_j$ are defined below Equation (22) in the main text. To speed up computation, in practice we use the Cooley–Tukey Fast Fourier Transform (FFT) algorithm (Cooley and Tukey, 1965), where we let $\tilde{m}_j = 2^{q_j}, j = 1, \dots, d + 1$, and we compute the Fourier transforms at even and odd indices recursively. The computational complexity of the FFT is $\mathcal{O}((\prod_{j=1}^{d+1} \tilde{m}_j) \log(\prod_{j=1}^{d+1} \tilde{m}_j))$, which is exponentially faster than the naive DFT. Similarly, the Cooley-Tukey FFT algorithm can be used to compute the inverse DFT (IDFT), as defined in Equation (20), exponentially faster than the naive IDFT. We use the `pytorch` software package to implement the multi-dimensional FFT for our applications.

S.2 Further simulation-experiment details

S.2.1 Performance metrics

We assess the forecasting performance of each model using metrics evaluated at N_0 test instances, namely the time series of spatial data excluded from training. We compute these metrics by comparing the forecasts from all the models with the true process values at the corresponding spatial locations. The diagnostics we consider are the mean squared prediction error (MSPE), the prediction interval coverage probability (PICP) of a nominal 95% prediction interval, and the mean prediction interval width (MPIW). These are defined as

$$\text{MSPE} = \frac{1}{N_0}\frac{1}{|\mathcal{D}^G|}\frac{1}{T-h-\tau}\sum_{j=1}^{N_0}\sum_{\mathbf{s}\in\mathcal{D}^G}\sum_{k=\tau+1}^{T-h}(Y^{(j)}(\mathbf{s}, t_{k+h}) - \widehat{Y}^{(j)}(\mathbf{s}, t_{k+h}))^2,$$

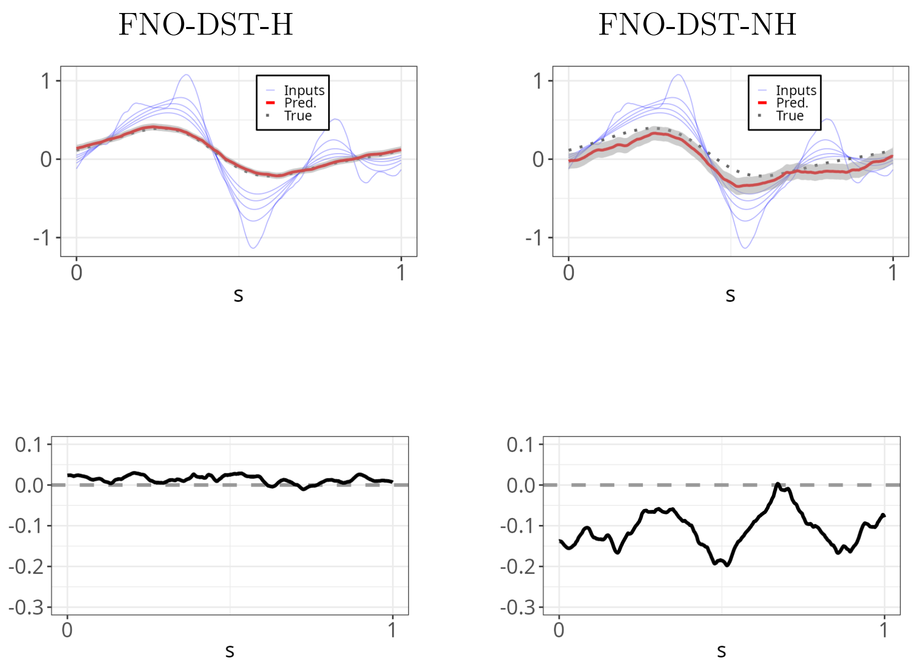
$$\text{PICP} = \frac{1}{N_0}\frac{1}{|\mathcal{D}^G|}\frac{1}{T-h-\tau}\sum_{j=1}^{N_0}\sum_{\mathbf{s}\in\mathcal{D}^G}\sum_{k=\tau+1}^{T-h}\mathbb{1}\{Y^{(j)}(\mathbf{s}, t_{k+h}) \in [L^{(j)}(\mathbf{s}, t_{k+h}), U^{(j)}(\mathbf{s}, t_{k+h})]\},$$

$$\text{MPIW} = \frac{1}{N_0}\frac{1}{|\mathcal{D}^G|}\frac{1}{T-h-\tau}\sum_{j=1}^{N_0}\sum_{\mathbf{s}\in\mathcal{D}^G}\sum_{k=\tau+1}^{T-h}[U^{(j)}(\mathbf{s}, t_{k+h}) - L^{(j)}(\mathbf{s}, t_{k+h})],$$

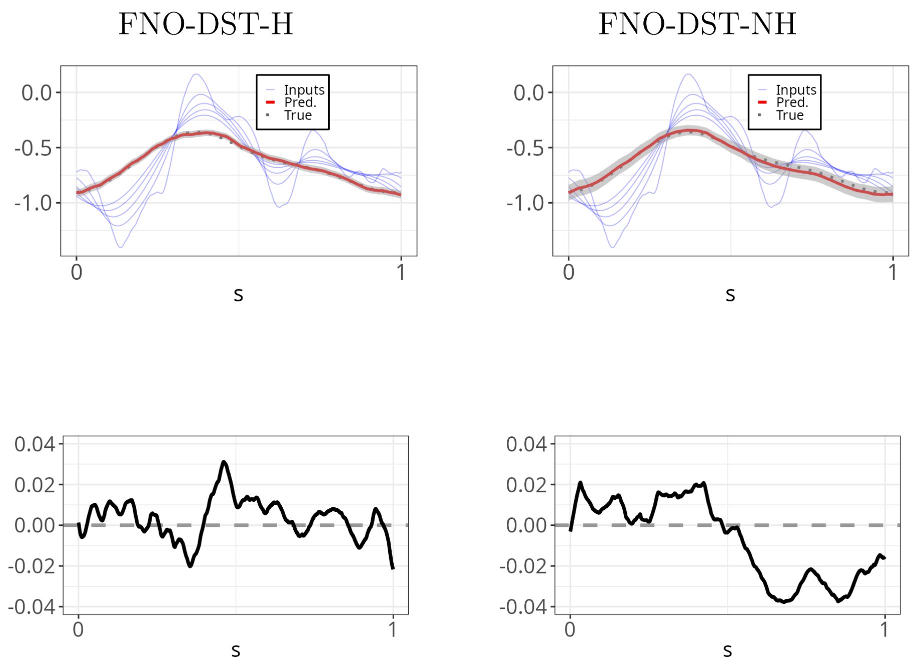
where $L^{(j)}(\mathbf{s}, t_{k+h})$ and $U^{(j)}(\mathbf{s}, t_{k+h})$ are the lower and upper forecast bounds of the 95% prediction interval of $Y^{(j)}(\mathbf{s}, t + h)$.

S.2.2 Additional testing instances for Section 3

This section presents supplementary figures corresponding to Section 3 of the main text, illustrating the results for two additional testing instances.



(a) Predicted fields (top row) and errors (bottom row) for random $\gamma = \gamma^{(1)}$.



(b) Predicted fields (top row) and errors (bottom row) for fixed $\gamma = \gamma^{(2)}$.

Figure S.1: Second testing instance out of $N_0 = 50$ testing instances for, respectively, forecasts from FNO-DST-H and FNO-DST-NH models at $T = 10$. (a): $\gamma = \gamma^{(1)}$, (b): $\gamma = \gamma^{(2)}$. In the top panels of (a) and (b), the current and past four steps (blue lines), the true spatial field at $T = 10$ (black dotted line), and the forecasts (red line) with 95% prediction intervals (gray shaded area), are shown.

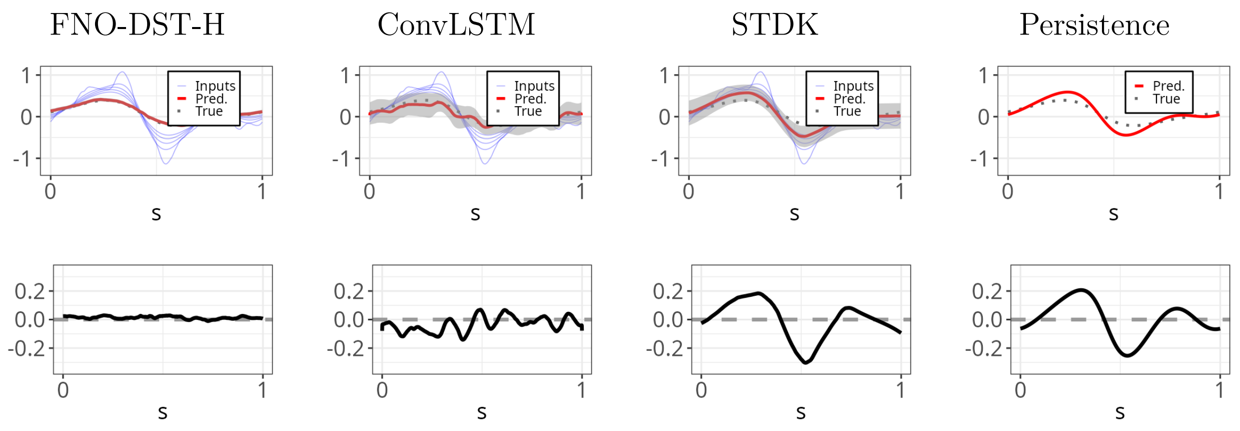
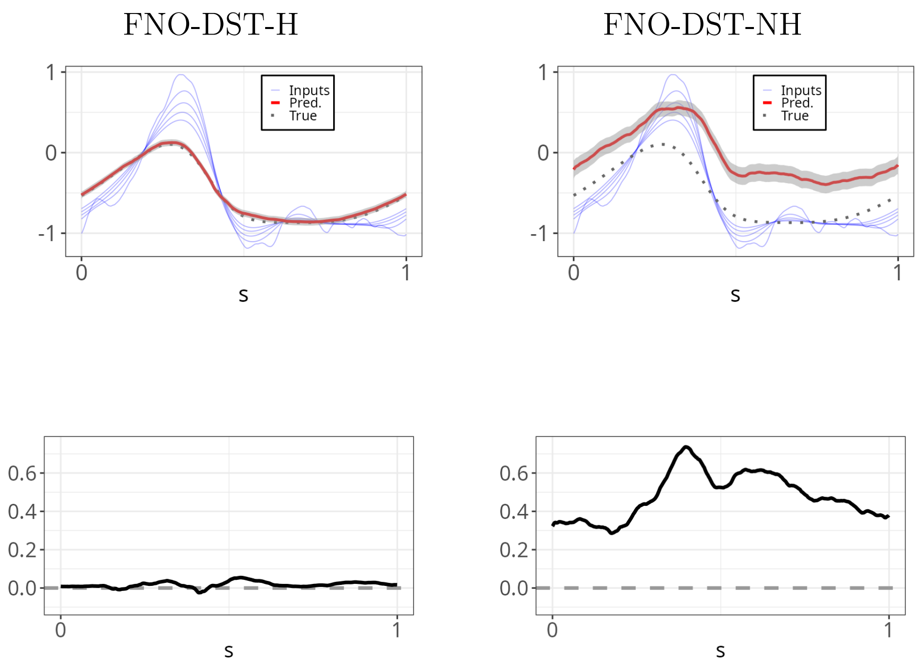
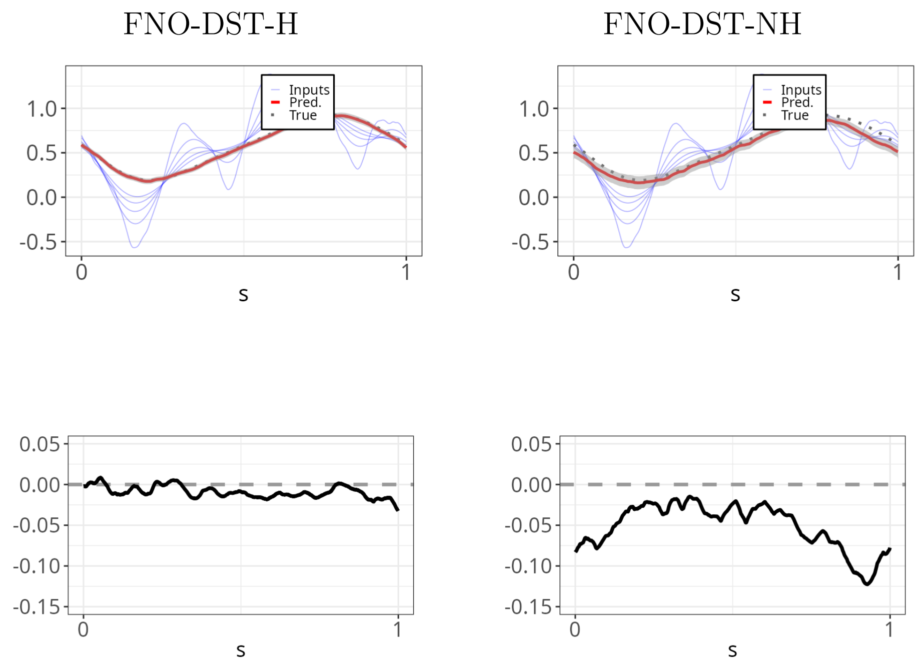


Figure S.2: Forecast fields (top row by model) and corresponding prediction errors (bottom row by model) for the second testing instance out of $N_0 = 50$ testing instances. This visualization illustrates both the forecast quality at $T = 10$ and error distribution across space for the simulation setting with random $\gamma = \gamma^{(1)}$. In the top panels, the current and past four steps (blue lines), the true spatial field at $T = 10$ (black dotted line), and the forecasts (red line) with 95% prediction intervals (gray shaded area), are shown.



(a) Predicted fields (top row) and errors (bottom row) for random $\gamma = \gamma^{(1)}$.



(b) Predicted fields (top row) and errors (bottom row) for fixed $\gamma = \gamma^{(2)}$.

Figure S.3: Third testing instance out of $N_0 = 50$ testing instances for, respectively, forecasts from FNO-DST-H and FNO-DST-NH models at $T = 10$. (a): $\gamma = \gamma^{(1)}$, (b): $\gamma = \gamma^{(2)}$. In the top panels of (a) and (b), the current and past four steps (blue lines), the true spatial field at $T = 10$ (black dotted line), and the forecasts (red line) with 95% prediction intervals (gray shaded area), are shown.

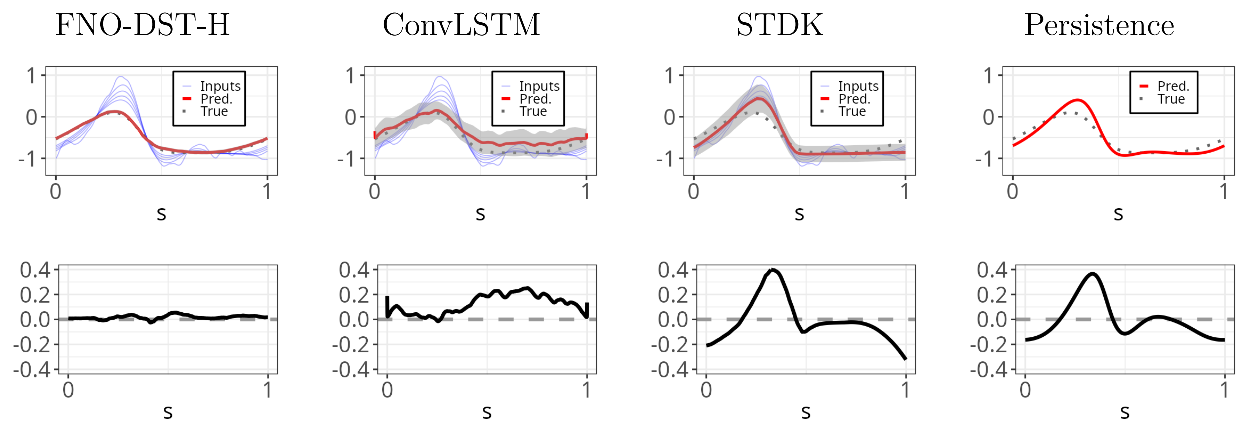


Figure S.4: Forecast fields (top row by model) and corresponding prediction errors (bottom row by model) for the third testing instance out of $N_0 = 50$ testing instances. This visualization illustrates both the forecast quality at $T = 10$ and error distribution across space for the simulation setting with random $\gamma = \gamma^{(1)}$. In the top panels, the current and past four steps (blue lines), the true spatial field at $T = 10$ (black dotted line), and the forecasts (red line) with 95% prediction intervals (gray shaded area), are shown.

References

Brezis, H. and Brézis, H. (2011). *Functional Analysis, Sobolev Spaces and Partial Differential Equations*. Vol. 2, Springer, New York, NY. See Chapter 8.

Cooley, J. and Tukey, J. (1965). An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, 19(90):297–301.

Evans, L. C. (2022). *Partial Differential Equations, Ch.5.* Vol. 19, American Mathematical Society, Providence, RI.