Physics and Lie symmetry informed Gaussian processes

David Dalton [1] **Dirk Husmeier** [1] **Hao Gao** [1]

Abstract

Physics-informed machine learning (PIML) has established itself as a new scientific paradigm which enables the seamless integration of observational data with partial differential equation (PDE) based physics models. A powerful tool for the analysis, reduction and solution of PDEs is the Lie symmetry method. Nevertheless, only recently has the integration of such symmetries into PIML frameworks begun to be explored. The present work adds to this growing literature by introducing an approach for incorporating a Lie symmetry into a physics-informed Gaussian process (GP) model. The symmetry is introduced as a constraint on the GP; either in a soft manner via virtual observations of an induced PDE called the invariant surface condition, or explicitly through the design of the kernel. Experimental results demonstrate that the use of symmetry constraints improves the performance of the GP for both forward and inverse problems, and that our approach offers competitive performance with neural networks in the low-data environment.

1. Introduction

Partial differential equations (PDEs) have become ubiquitous across science and engineering for describing the dynamics of a wide variety of physical, chemical and biological processes. Simulation of PDE based models has traditionally been the domain of numerical solvers such as the finite difference method or finite element method (Li et al., 2017). Such methods can have drawbacks, however, including for instance excessive computational costs and difficulty in incorporating experimental data. For this reason, there has been an explosion of interest in the past half decade in an alternative paradigm for modelling PDE systems called *physics-informed machine learning* (PIML).

[1]School of Mathematics and Statistics, University of Glasgow, United Kingdom. Correspondence to: David Dalton <david.dalton@glasgow.ac.uk>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

PIML refers to a set of methods which embed physical laws and constraints into machine learning algorithms, through the use of inductive, learning or observational biases (Karniadakis et al., 2021). PIML approaches can alleviate some of the problems of traditional solvers, and have been deployed in the context of forward, inverse, design and optimisation problems (Hao et al., 2022).

Physics-informed neural networks (PINNs) (Raissi et al., 2019) constitute the most widely used class of PIML model, where the idea is to represent the unknown PDE solution as a neural network. In the original formulation, training is performed in a multitask manner against a loss function which includes both PDE and data fit terms, where automatic differentiation is used to apply the PDE operator to the neural network. The range of applications for which PINNs have been deployed is vast (Pateras et al., 2023), while numerous extensions of the original PINN framework have been proposed, for example (Nguyen-Thanh et al., 2020; Meng et al., 2020; Kharazmi et al., 2021).

PIML models for PDE and ordinary differential equation (ODE) systems based on Gaussian processes (GPs) have also been proposed, which leverage the closure of GPs under linear operators (Adler, 2010). For instance, the gradient matching technique fits a GP to (noisy) observational data, and then estimates the unknown parameters of the differential operator by minimising the mismatch between the ODEs and the derivative process induced by the fitted GP (Calderhead et al., 2008; Dondelinger et al., 2013). However, this approach does not define a proper probabilistic generative model (Barber & Wang, 2014), and performance is critically dependent on the way in which the tradeoff between the GP fit and ODE mismatch terms is managed (Macdonald et al., 2015). Latent force models are a related approach to inference in PDE systems (Alvarez et al., 2013). This approach does define a generative model, however it is limited to the case of linear PDEs for which the Green's function is available. For these reasons, in the present work we adopt the philosophy of physics-informed GPs for linear PDE modelling (Raissi et al., 2017). Here, the idea is to represent the unknown solution function with a Gaussian process (GP). This implies that the linear PDE itself also follows a GP, allowing for joint inference to be performed using data from both solution space and PDE space. PIGPs have been deployed in a range of application contexts (Tar-

takovsky et al., 2023; Pan et al., 2023; Nevin et al., 2021), and several extensions of the original approach have been proposed to handle nonlinear PDEs, for instance (Raissi et al., 2018; Chen et al., 2020; Long et al., 2022).

When modelling physical systems, it is common that certain *symmetries* of the underlying process may be known, which can in turn be encoded in the PIML model for improved performance. For example, there has recently been significant interest in the design of NN architectures which satisfy equivariance under symmetry groups relating to actions such as translation, rotation and scaling (Fuchs et al., 2020; Thomas et al., 2018; Satorras et al., 2021). In the context of PDEs, a symmetry maps a solution of the equation to another solution, i.e. it leaves the PDE *invariant*. PDE symmetry analysis constitutes a powerful tool for finding explicit solutions, conservation laws, and simplifications of PDE systems (Bluman & Anco, 2002). In this work, we will focus on the classical symmetry method due to Lie, the use of which remains an active research topic (Champala et al., 2023; Bakhshandeh-Chamazkoti & Alipour, 2022; Al-Nassar & Nadjafikhah, 2023), including extensions to symmetries admitted by initial and boundary conditions (Zhang & Chen, 2010; Goard, 2008; Cherniha & Kovalenko, 2012). Nevertheless, only very recently has this approach been made use of in the context of PIML. This includes the work of Wang et al. (2021), which developed equivariant NN architectures for the heat and Navier-Stokes equations, Bransetter et al. (2022), which leveraged Lie symmetries to improve efficiency through data augmentation, and Mialon et al. (2023), which performed self-supervised learning in PINNs. Finally, several methods have been proposed for embedding symmetry information into the loss function used for PINN training, by including an additional term to penalise deviation from the symmetry (Akhound-Sadegh et al., 2023; Zhang et al., 2023a;b;c).

In this work, we introduce the physics and Lie symmetry informed Gaussian process (PSGP), which incorporates a given Lie symmetry into the inference framework using an induced PDE called the invariant surface condition (ISC). In the general case, this requires a likelihood which assimilates data from the solution, PDE and ISC spaces respectively, and a prior which defines a consistent distribution over the process and its derivative values, before inference is performed by maximisation of a variational lower bound. Various simplifications are possible depending on the specific PDE and symmetry of interest, including exact marginalisation of the latent variables and explicit enforcement of the ISC. Through numerical experiments involving three PDEs, we show that the incorporation of symmetry information enables highly sample efficient performance in both forward and inverse problems.

2. Background

2.1. Gaussian Process Regression

Gaussian processes (GPs) allow for a distribution to be specified directly on a function space, and can be used to perform Bayesian non-parametric regression (Rasmussen & Williams, 2006). Specifically, consider the task of learning an unknown function $u : \mathbb{R}^{D+1} \to \mathbb{R}$, given a finite set of N_u (noisy) observations $\boldsymbol{y}_u = [y_u^{(1)}, y_u^{(2)}, \dots, y_u^{(N_u)}]^\top$ at input locations $\mathbf{X}_u = [\boldsymbol{x}_u^{(1)}, \boldsymbol{x}_u^{(2)}, \dots, \boldsymbol{x}_u^{(N_u)}]^\top$. We assume any observation noise is iid Gaussian with variance σ_u^2. Performing GP regression first involves assuming that u is drawn from a Gaussian process with mean function m_u and covariance or kernel function k_{uu}, which is denoted

$$u \sim \mathcal{GP}(m_u(\cdot), k_{uu}(\cdot, \cdot)). \tag{1}$$

Popular choices of kernel include the squared-exponential, Matefn and rational-quadratic, respectively (Duvenaud, 2014). By definition, the GP prior on u means the projection of the process at the N_u input locations is a multivariate Gaussian. Coupled with the Gaussian noise model, this implies $p(\boldsymbol{y}_u) = \mathcal{N}(\boldsymbol{m}_u, \mathbf{K}_{uu} + \sigma_u^2 \mathbf{I}_u)$, where $[\boldsymbol{m}_u]^{(i)} = m_u(\boldsymbol{x}_u^{(i)}) = \mathbb{E}[u(\boldsymbol{x}_u^{(i)})]$, $[\mathbf{K}_{uu}]^{(i,j)} = k_{uu}(\boldsymbol{x}_u^{(i)}, \boldsymbol{x}_u^{(j)}) = \mathrm{Cov}\left(u(\boldsymbol{x}_u^{(i)}), u(\boldsymbol{x}_u^{(j)})\right)$ and $\mathbf{I}_u$ is the identity matrix. When viewed as a function of any mean, kernel and noise parameters, $p(\boldsymbol{y}_u)$ is called the *marginal likelihood* of $\boldsymbol{y}_u$ and can be used to perform model inference.

Given any N_s input points of interest $\mathbf{X}_s$, the posterior distribution $p(\boldsymbol{u}_s \mid \boldsymbol{y}_u)$ over the unknown function values $\boldsymbol{u}_s = [u_s^{(1)}, u_s^{(2)}, \dots, u_s^{(N_s)}]^\top$ at these points can be shown to be a Gaussian $\mathcal{N}(\boldsymbol{\mu}_s, \boldsymbol{\Sigma}_s)$, where

$$\begin{aligned} \boldsymbol{\mu}_s &= \boldsymbol{m}_s + \mathbf{K}_{us}^\top \left(\mathbf{K}_{uu} + \sigma_u^2 \mathbf{I}_{N_u}\right)^{-1} (\boldsymbol{y}_u - \boldsymbol{m}_u), \\ \boldsymbol{\Sigma}_s &= \mathbf{K}_{ss} - \mathbf{K}_{us}^\top \left(\mathbf{K}_{uu} + \sigma_u^2 \mathbf{I}_{N_u}\right)^{-1} \mathbf{K}_{us}. \end{aligned} \tag{2}$$

The mean vector $\boldsymbol{m}_s$ and covariance matrix $\mathbf{K}_{ss}$ above are found by applying the mean and covariance functions respectively to $\mathbf{X}_s$, while $[\mathbf{K}_{us}]^{(i,j)} = k_{uu}(\boldsymbol{x}_u^{(i)}, \boldsymbol{x}_s^{(j)})$.

2.2. Partial Differential Equations

Partial differential equations (PDEs) are mathematical models used to describe the dynamics of systems which evolve over space and time. In this work we consider scalar PDEs of the form

$$\mathcal{F}_{\boldsymbol{x},t}^{\boldsymbol{\theta}}[u] = f, \; t \in [t_{\min}, t_{\max}] = T, \; \boldsymbol{x} \in \Omega \subset \mathbb{R}^D, \tag{3}$$

where $\mathcal{F}_{\boldsymbol{x},t}^{\boldsymbol{\theta}}$ is a differential operator, u is the solution function, t is time[1], $\boldsymbol{x}$ is the spatial coordinate and Ω is the

[1]For notational simplicity sometimes we do not make the temporal component explicit.

domain, while $\boldsymbol{\theta}$ denotes any parameters of the PDE. A PDE can equivalently be represented using an algebraic equation $F_{\boldsymbol{\theta}}$ which satisfies

$$F_{\boldsymbol{\theta}}\left([\boldsymbol{x}, t], u(\boldsymbol{x}, t), d_1(\boldsymbol{x}, t), \ldots, d_m(\boldsymbol{x}, t)\right) = f, \quad (4)$$

where we use the functions $d_1, \ldots, d_m$ to represent all m possible partial derivative operators up to the highest order of the PDE (Bluman & Anco, 2002, Eq. (4.1)). For a second order PDE over one spatial dimension, for instance, we have $m = 5$ and the individual operators are $d_1 = u_x, d_2 = u_t, d_3 = u_{xt}, d_4 = u_{tt}$ and $d_5 = u_{xx}$[2]. In this representation, the PDE defines a hypersurface on the *prolonged* input space, which is referred to as the *jet-space*. Typically, a PDE will only incorporate a small subset of these m operators - see the below for an example.

Example 2.1. The following equation of Fisher's type

$$\mathcal{F}^{\theta}_{x,t}[u] = u_t - u_{xx} - \theta u^2(1-u) = f = 0, \quad (5)$$

is an example of a non-linear PDE over one spatial dimension, where θ is a parameter. This equation can be alternatively represented in algebraic form as

$$F_\theta(u, d_2, d_5) = d_2 - d_5 - \theta u^2(1-u) = f = 0, \quad (6)$$

where $d_2 = u_t$ and $d_5 = u_{xx}$, as in the preceding paragraph.

Solutions to PDEs are not uniquely defined. To make a problem well posed, an initial boundary value problem (IBVP) can be specified, which imposes constraints on the behaviour of the solution function at the initial time t_{min} and domain boundary $\partial\Omega$ respectively. For instance, Dirichlet conditions specify the exact value of the solution,

$$u(\boldsymbol{x}, t_{\text{min}}) = \mathcal{I}(\boldsymbol{x}) \;\forall\; \boldsymbol{x} \in \Omega, \quad (7)$$
$$u(\boldsymbol{x}, t) = \mathcal{B}(\boldsymbol{x}, t) \;\forall\; \boldsymbol{x} \in \partial\Omega, t \in T, \quad (8)$$

where $\mathcal{I} : \Omega \to \mathbb{R}$ is the initial condition while $\mathcal{B} : \partial\Omega \times T \to \mathbb{R}$ is the boundary function.

2.3. Physics-Informed Gaussian Processes

A key property of GPs is that they are closed under linear operators (Pförtner et al., 2024). Specifically, if we have $f = \mathcal{L}^{\boldsymbol{\theta}}_{\boldsymbol{x}}[u]$ with $\mathcal{L}^{\boldsymbol{\theta}}_{\boldsymbol{x}}$ a *linear* PDE operator[3], then our GP prior assumption for u (see Eq. (1)) implies that $f \sim \mathcal{GP}\left(m_f(\cdot\,;\boldsymbol{\theta}), k_{ff}(\cdot,\cdot\,;\boldsymbol{\theta})\right)$, where

$$m_f(\boldsymbol{x};\boldsymbol{\theta}) = \mathcal{L}^{\boldsymbol{\theta}}_{\boldsymbol{x}} m_u(\boldsymbol{x}), \quad (9)$$
$$k_{ff}\left(\boldsymbol{x}, \boldsymbol{x}';\boldsymbol{\theta}\right) = \mathcal{L}^{\boldsymbol{\theta}}_{\boldsymbol{x}}\mathcal{L}^{\boldsymbol{\theta}}_{\boldsymbol{x}'} k_{uu}\left(\boldsymbol{x}, \boldsymbol{x}'\right). \quad (10)$$

[2]Here and throughout the text we use the shorthand notation $u_t = \frac{\partial u}{\partial t}$, $u_x = \frac{\partial u}{\partial x}$, $u_{tt} = \frac{\partial^2 u}{\partial t^2}$, $u_{xt} = \frac{\partial^2 u}{\partial x \partial t}$, etc.

[3]i.e. $\mathcal{L}^{\boldsymbol{\theta}}_{\boldsymbol{x}}[\alpha u_1 + \beta u_2] = \alpha\mathcal{L}^{\boldsymbol{\theta}}_{\boldsymbol{x}}[u_1] + \beta\mathcal{L}^{\boldsymbol{\theta}}_{\boldsymbol{x}}[u_2]$, with $\alpha, \beta \in \mathbb{R}$.

In addition, the cross covariance between observations in u and f space respectively can be found as

$$k_{uf}\left(\boldsymbol{x}, \boldsymbol{x}';\boldsymbol{\theta}\right) = \mathcal{L}^{\boldsymbol{\theta}}_{\boldsymbol{x}'} k_{uu}\left(\boldsymbol{x}, \boldsymbol{x}'\right), \quad (11)$$
$$k_{fu}\left(\boldsymbol{x}, \boldsymbol{x}';\boldsymbol{\theta}\right) = \mathcal{L}^{\boldsymbol{\theta}}_{\boldsymbol{x}} k_{uu}\left(\boldsymbol{x}, \boldsymbol{x}'\right). \quad (12)$$

This property of GPs was made use of by Raissi et al. (2017) with the introduction of physics-informed Gaussian processes (PIGPs). PIGPs provide an elegant framework for solving both forward and inverse problems for systems where linear PDE information is available. Because joint Gaussianity is maintained by linear PDEs, exact inference is possible, with the marginal likelihood and posterior predictive distribution taking the same multivariate Gaussian forms as in usual GP regression.

2.4. Lie Symmetry Method

Remark 2.2. This section gives a brief overview of the classical approach to Lie symmetry analysis, material which is covered comprehensively in (Bluman & Anco, 2002). For an illustrative example of this method in the particular case of rotational symmetry, see Appendix A.

Intuitively, a symmetry of a geometric object is a transformation which leaves its relevant properties unchanged. The collection of all symmetries of an object form a *group*, which is a set equipped with a binary operator that composes any two symmetry transformations, such that associativity and closure are satisfied; there exists an identity transformation; and there exists an inverse transformation for each symmetry. *Continuous* symmetries can be naturally described by a group which is also a differentiable manifold, which is called a *Lie group*, in which case the binary operator is smooth, as is the inverse transformation.

A Lie group of transformations associated to a PDE corresponds to a mapping of each solution of the PDE to another solution. The classical approach due to Lie is to find the one-parameter ($\varepsilon \in \mathbb{R}$) Lie group of point transformations

$$\begin{aligned} x^* &= x + \varepsilon\xi(x,t,u) + O\left(\varepsilon^2\right), \\ t^* &= t + \varepsilon\tau(x,t,u) + O\left(\varepsilon^2\right), \\ u^* &= u + \varepsilon\eta(x,t,u) + O\left(\varepsilon^2\right), \end{aligned} \quad (13)$$

which leaves the PDE invariant, where we consider only one spatial dimension to simplify notation. Here $\xi = \xi(x,t,u)$, $\tau = \tau(x,t,u)$ and $\eta = \eta(x,t,u)$ are called the *infinitesimals* of the Lie group.

Definition 2.3. The **infinitesimal generator** of a one-parameter Lie group of transformations is the operator

$$\mathcal{X} = \xi\partial_x + \tau\partial_t + \eta\partial_u. \quad (14)$$

A corollary of Lie's First Fundamental Theorem (Bluman & Anco, 2002, Thm. 2.3.1-1) is that the one-parameter Lie

group (Eq. (13) is determined by its infinitesimal generator (Bluman & Anco, 2002, page 43). For this reason, we refer to the Lie group and its infinitesimal generator together as a Lie symmetry. To determine if a Lie symmetry admits a given PDE, Lie's *infinitesimal criterion* (Bluman & Anco, 2002, Thm. 4.1.1-1) can be applied, however this is beyond the scope of the present work.

A function which is invariant under the Lie symmetry and which solves the underlying PDE is called an *invariant solution*. Such solutions must also satisfy an additional PDE called the *invariant surface condition* (ISC).

Definition 2.4. The **invariant surface condition** (ISC) is a PDE of the form

$$\eta - \tau u_t - \xi u_x = g = 0. \tag{15}$$

The ISC induced by a Lie symmetry can allow the underlying PDE itself to be analysed, reduced and in some cases solved analytically - see (Bluman & Anco, 2002, Section 4.2.3) for some examples.

3. Methods

3.1. Physics and Lie Symmetry Informed GPs

We introduce physics and Lie symmetry informed Gaussian processes (PSGPs), a framework for incorporating observational data together with PDE and symmetry information into a joint, GP-based inference framework. PIGPs are restricted to linear PDEs and are therefore not appropriate for general differential operators. Several methods have been proposed to overcome this restriction (Raissi et al., 2018; Chen et al., 2020; Long et al., 2022). - here we adapt the approach of (Long et al., 2022). As in Section 2.1, we assume data $(\mathbf{X}_u, \boldsymbol{y}_u)$ is available in u-space. The underlying PDE (Eq. (3)) is incorporated using a vector $\boldsymbol{y}_f$ of *virtual observations* of f at N_f collocation points $\mathbf{X}_f = [\boldsymbol{x}_f^{(1)}, \boldsymbol{x}_f^{(2)}, \ldots, \boldsymbol{x}_f^{(N_f)}]^\top$ in the spatio-temporal domain. Similarly, symmetry information is incorporated via the ISC (Eq. (15)) at N_g collocation points $\mathbf{X}_g$, using virtual observations $\mathbf{0}_g$ of $g = 0$.

Prior. As in general GP regression, we assign a GP prior to the function u (see Eq. (1)). We now introduce some notation. Let $\boldsymbol{u}$ be the vector corresponding to the values of u evaluated at input locations $\mathbf{X}_u$, i.e. $[\boldsymbol{u}]^{(i)} = u(\boldsymbol{x}_u^{(i)})$, for $i = 1, 2, \ldots, N_u$. Similarly, let $\boldsymbol{d}_0$ give the values of u at $\mathbf{X}_f$. For $j = 1, 2, \ldots, m$, we let $\boldsymbol{d}_j$ be the vector of evaluations of d_j (see Eq. (4)) at $\mathbf{X}_f$, i.e $[\boldsymbol{d}_j]^{(i)} = d_j(\boldsymbol{x}_f^{(i)})$, for $i = 1, 2, \ldots, N_f$. Finally, we denote the vector of evaluations of the ISC at each point in $\mathbf{X}_g$ as $\boldsymbol{g}$. For notational simplicity, we will assume here that the ISC is linear in u. Consider then the vector $\boldsymbol{h} = [\boldsymbol{u}; \boldsymbol{d}_0; \boldsymbol{d}_1; \ldots; \boldsymbol{d}_m; \boldsymbol{g}]$. Each element of $\boldsymbol{h}$ corresponds to an evaluation of either u, or a process which is found by applying a linear operator to u. Since u follows a GP, this implies

$$p(\boldsymbol{h}) = \mathcal{N}(\boldsymbol{h} \mid \boldsymbol{m}_h, \mathbf{K}_{hh}), \tag{16}$$

where the individual terms of $\boldsymbol{m}_h$ and $\mathbf{K}_{hh}$ are found by applying the rules given in Eqs. (9-12) for each linear operator used in the specification of $\boldsymbol{h}$.

Likelihood. The PSGP must incorporate data from three different function spaces - in each case, we assume an iid Gaussian noise model. For the data in solution (u) space, this assumption yields a likelihood

$$p(\boldsymbol{y}_u \mid \boldsymbol{u}) = \mathcal{N}(\boldsymbol{y}_u \mid \boldsymbol{u}, \sigma_u^2 \mathbf{I}_u). \tag{17}$$

For virtual data in PDE (f) space, a *virtual likelihood* can be defined in terms of the algebraic representation of the PDE

$$p(\boldsymbol{y}_f \mid \boldsymbol{d}_0, \ldots, \boldsymbol{d}_m) = \mathcal{N}(\boldsymbol{y}_f \mid F_{\boldsymbol{\theta}}(\mathbf{X}_f, \boldsymbol{d}_0, \ldots, \boldsymbol{d}_m), \sigma_f^2 \mathbf{I}_f) \tag{18}$$

where $F_{\boldsymbol{\theta}}$ (see Eq. (4)) is applied element-wise. Here, σ_f^2 is a nugget term which can be trained to control the degree to which the model conforms to the PDE. Finally, for virtual data in ISC (g) space, the corresponding virtual-likelihood takes the below form, given nugget term σ_g^2

$$p(\mathbf{0}_g \mid \boldsymbol{g}) = \mathcal{N}(\mathbf{0}_g \mid \boldsymbol{g}, \sigma_g^2 \mathbf{I}_g). \tag{19}$$

Joint distribution. The joint probability distribution over all random variables is then given by the product of the likelihood terms and the prior:

$$\begin{aligned} p(\boldsymbol{h}, \boldsymbol{y}_u, \boldsymbol{y}_f, \mathbf{0}_g) = &\mathcal{N}(\boldsymbol{y}_u \mid \boldsymbol{u}, \sigma_u^2 \mathbf{I}_u) \cdot \mathcal{N}(\mathbf{0}_g \mid \boldsymbol{g}, \sigma_g^2 \mathbf{I}_g) \\ &\cdot \mathcal{N}(\boldsymbol{y}_f \mid F_{\boldsymbol{\theta}}(\mathbf{X}_f, \boldsymbol{d}_0, \ldots, \boldsymbol{d}_n), \sigma_f^2 \mathbf{I}_f) \\ &\cdot \mathcal{N}(\boldsymbol{h} \mid \boldsymbol{m}_h, \mathbf{K}_{hh}) \end{aligned} \tag{20}$$

This model specifies a consistent prior over the function, its derivative values and the ISC, with likelihood terms that incorporate observations from the different function spaces. This allows for joint inference to be performed of the kernel hyperparameters, noise levels, and any PDE parameters $\boldsymbol{\theta}$ which are unknown.

Inference. For general nonlinear PDEs, exact inference of Eq. (20) is not possible and therefore the posterior $p(\boldsymbol{h} \mid \boldsymbol{y}_u, \boldsymbol{y}_f, \mathbf{0}_g)$ cannot be computed. As in Long et al. (2022), we proceed by using variational inference and introduce an approximate posterior $q(\boldsymbol{h}) = \mathcal{N}(\boldsymbol{h} \mid \boldsymbol{a}, \mathbf{A}\mathbf{A}^\top)$ with $\mathbf{A}$ a lower-triangular matrix, which ensures the variational posterior covariance is positive-definite. The variational parameters $\boldsymbol{a}$ and $\mathbf{A}$ together with any kernel/noise/PDE parameters are then jointly inferred by maximisation of the *evidence lower bound* (ELBO):

$$\begin{aligned} L = &- \mathrm{KL}(q(\boldsymbol{h}) \| p(\boldsymbol{h})) + \mathbb{E}_q[\log p(\boldsymbol{y}_f \mid \boldsymbol{d}_0, \ldots, \boldsymbol{d}_n))] \\ &+ \mathbb{E}_q[\log p(\mathbf{0}_g \mid \boldsymbol{g})] + \mathbb{E}_q[\log p(\boldsymbol{y}_u \mid \boldsymbol{u})], \end{aligned} \tag{21}$$

where $\mathrm{KL}(\cdot\|\cdot)$ indicates the Kullback-Leibler divergence. For exact details on computing the ELBO, see Appendix D.1. In brief, the second term involving the pseudo PDE observations $\boldsymbol{y}_f$ is evaluated using Monte-Carlo sampling, while the other three terms are available in closed form.

Prediction. Once inference has been performed, an approximate posterior $\hat{p}$ over the function values $\boldsymbol{u}_s$ at any input locations of interest $\mathbf{X}_s$ can be found as

$$\begin{aligned}\hat{p}(\boldsymbol{u}_s) &= \int p(\boldsymbol{u}_s \mid \boldsymbol{h}) q(\boldsymbol{h}) d\boldsymbol{h} \\ &= \mathcal{N}(\boldsymbol{u}_s \mid \hat{\boldsymbol{\mu}}_s, \hat{\boldsymbol{\Sigma}}_s). \end{aligned} \tag{22}$$

Appendix D.2 presents the derivation of the above posterior mean and covariance, yielding $\hat{\boldsymbol{\mu}}_s = \boldsymbol{m}_s + \mathbf{K}_{sh}\mathbf{K}_{hh}^{-1}(\boldsymbol{a} - \boldsymbol{m}_h)$ and $\hat{\boldsymbol{\Sigma}}_s = \mathbf{K}_{ss} - \mathbf{K}_{sh}\left[\mathbf{K}_{hh}^{-1} - \mathbf{K}_{hh}^{-1}\mathbf{A}\mathbf{A}^\top\mathbf{K}_{hh}^{-1}\right]\mathbf{K}_{hs}$, where $[\mathbf{K}_{sh}]^{(i,j)} = \mathrm{Cov}\left(u_s^{(i)}, h^{(j)}\right)$ is found by applying k_{uu} and the differential operators used to define the latent vector $\boldsymbol{h}$.

3.1.1. Simplifications of general approach

In the general case detailed above, the dimensionality of the latent vector $\boldsymbol{h}$ will be high, imposing a computational bottleneck in the evaluation of the ELBO and the approximate posterior. Depending on the specific PDE and Lie symmetry of interest, however, simplifications are typically possible which improve computational efficiency. Firstly, it is unlikely that a PDE will make use of all possible partial derivative operators from the full jet-space (see Eq. (4)), and any operators which are not used can be discarded from $\boldsymbol{h}$. In addition, it may be possible to identify in the PDE certain sublinear differential operators to incorporate into $\boldsymbol{h}$, rather than considering each derivative operator individually, allowing for the dimensionality of $\boldsymbol{h}$ to be reduced. In Eq. (6) for instance, the single linear operator $d_2 - d_5$ can be considered, rather than considering each operator separately.

If the PDE itself is linear, a further simplification arises as exact inference is possible in this case, which allows $\boldsymbol{h}$ to be marginalised out and obviating the need for the approximate posterior q (Raissi et al., 2017). This is the case for example in the heat equation considered in Section 4.3. Finally, it may be possible to define the kernel k_{uu} on a system of *canonical coordinates* such that the ISC and possibly the PDE itself are explicitly satisfied by the PSGP. In this case, one or both of the virtual likelihoods from the joint model (Eq. (20)) are not required. An example of this is provided in Section 4.4.

3.1.2. Enforcement of boundary conditions

There has recently been an interest in the development of methods for explicitly enforcing known initial/boundary conditions, both for GPs (Tan, 2016; Li & Tan, 2022) and neural networks (NNs) (Nguyen-Thanh et al., 2020; Sheng & Yang, 2021). Dirichlet conditions can be enforced by using a mean function which interpolates between the known boundary values, and a distance function which collapses the GP/NN to zero at the boundary [4]. With Thm. 3.2 below, we prove (see Appendix C.1) that the same technique can be used to enforce such constraints in the approximate PSGP posterior (Eq. (22)). Given this result, we use explicit enforcement in the numerical experiments for all known initial/boundary conditions in the PSGP model (and all benchmark PIML models considered), precluding the requirement for penalty enforcement via virtual observations on the boundary.

Definition 3.1. $\mathcal{L}$-smoothness. We say a continuous kernel k_{uu} is $\mathcal{L}$-smooth with respect to the differential operator $\mathcal{L}$ if $\mathcal{L}_{\boldsymbol{x}}[k_{uu}](\cdot,\cdot), \mathcal{L}_{\boldsymbol{x}'}[k_{uu}](\cdot,\cdot), \mathcal{L}_{\boldsymbol{x}}\mathcal{L}_{\boldsymbol{x}'}[k_{uu}](\cdot,\cdot)$ all exist and are continuous functions.

Theorem 3.2. *Let $u : \Omega \to \mathbb{R}$ be subject to Dirichlet boundary conditions, i.e. $u(\boldsymbol{x}) = \mathcal{B}(\boldsymbol{x}) \,\forall\, \boldsymbol{x} \in \partial\Omega$. Assign a prior $u(\cdot) \sim \mathcal{GP}\left(\tilde{m}_u(\cdot), \tilde{k}_{uu}(\cdot,\cdot)\right)$ with $\tilde{m}_u$ a continuous function which equals $\mathcal{B}$ on $\partial\Omega$, and $\tilde{k}_{uu}(\boldsymbol{x},\boldsymbol{x}') = \phi(\boldsymbol{x})\phi(\boldsymbol{x}')k_{uu}(\boldsymbol{x},\boldsymbol{x}')$ where k_{uu} is a kernel and ϕ is a continuous distance function which equals zero on $\partial\Omega$ and is otherwise positive. We assume data $\boldsymbol{y}_u$ and virtual observations $\boldsymbol{y}_f$ ($\mathbf{0}_g$) are available for a given PDE (ISC). Now, consider any sequence of points in the interior of the domain $\boldsymbol{x}_s^{(1)}, \boldsymbol{x}_s^{(2)}, \ldots$ such that $\lim_{l\to\infty} \boldsymbol{x}_s^{(l)} = \boldsymbol{x}_b \in \partial\Omega$. Then, assuming that k_{uu} is $\mathcal{L}$-smooth with respect to every differential operator used in the specification of $\boldsymbol{h}$, the PSGP posterior (Eq. (22)) satisfies $\hat{p}\left(u(\boldsymbol{x}_s^{(l)})\right) \xrightarrow{p} \mathcal{B}(\boldsymbol{x}_b)$ for arbitrary form of the variational posterior $q(\boldsymbol{h})$, where $\xrightarrow{p}$ indicates convergence in probability.*

4. Numerical Experiments

We evaluated the performance of **PSGPs** on three tasks commonly encountered in PIML; solving IBVPs (Section 4.2), the inverse problem of learning unknown PDE parameters (Section 4.3), and the forward problem of learning the PDE solution from noisy observations (Section 4.4). For benchmarking, we used three existing PIML models. Firstly, we compared with a GP which incorporates observational data and the underlying PDE, but ignores symmetry information. For notational simplicity, we will refer to this approach as a **PIGP** irrespective of whether the underlying PDE is linear or nonlinear (in which case the model is identical to

[4] For grid-like domains, the construction of these mean and distance functions is straightforward (see Appendix C.1.1 for an example). For Neumann/Cauchy/Robbins conditions and domains of more complex shape, an extension of this approach can be used (Liu et al., 2022).

that described in Section 3.1, without the virtual ISC observations). Secondly, we compared with a **PINN**. A PINN is trained on an objective function comprised of one loss term for observation data $\boldsymbol{y}_u$ and one loss term for virtual PDE data $\boldsymbol{y}_f$, but ignores symmetry information. Finally, we compared with a PINN which also accounts for a given Lie symmetry, which we call a physics and Lie symmetry informed neural network (**PSNN**) for notational consistency. The symmetry is incorporated either through the addition of a third term to the objective function to account for the loss against virtual ISC observations $\mathbf{0}_g$ (in which case the PSNN is equivalent to the symmetry enhanced PINN (Zhang et al., 2023c)), or through explicit enforcement via a coordinate transformation.

4.1. Implementation Details

For the GP models, k_{uu} was specified to be the rational quadratic kernel. We experimented with different neural network architectures (using tanh activation function), and found that four hidden layers each of width 20 yielded the best accuracy. Each model was trained using the Adam optimiser with exponentially decaying learning rate (Kingma & Ba, 2017). As suggested in (Long et al., 2022), we use the whitening trick (Murray & Adams, 2010) when evaluating the ELBO (Eq. (21)), to improve training efficiency. Experiments were performed in Python using JAX (Bradbury et al., 2018). All differential operators were implemented using the automatic differentiation system provided by JAX, meaning that no hand derivations of mean/kernel functions were required. Code and data are available at github.com/dodaltuin/jax-pigp/tree/main/examples/PSGPs.

4.2. Fisher-like Equation

We first considered the non-linear diffusion equation of Fisher's type from Eq. (5) with $\theta = 1$, which was analysed in (Verma et al., 2014). This PDE admits the following spatio-temporal translation symmetry

$$\mathcal{X}_{fish} = \frac{1}{\sqrt{2}}\partial_x + \partial_t, \tag{23}$$

which implies that the below ISC holds (see Eq. (15)).

$$u_t + \frac{1}{\sqrt{2}}u_x = g = 0. \tag{24}$$

In this instance, the ISC can be used to solve for u exactly (Verma et al., 2014, Eq. (35)), which is displayed in Figure 1 (a) for $t \in [0, 10]$ and $x \in [-5, 5]$.

We considered an adjustment of the IBVP from (Verma et al., 2014, Ex. 2), where the objective was to learn the PDE solution using N_f virtual observations $\boldsymbol{y}_f$ in f-space for the PIGP and PINN models, and additionally $N_g = N_f$ virtual observations $\mathbf{0}_g$ in g-space for the PSGP and PSNN approaches, given known initial and boundary conditions. To examine the impact of data set size on results, we performed the experiment for $N_f/N_g = 16, 32, 64$ and 128 virtual observations, where the input locations $\mathbf{X}_f = \mathbf{X}_g$ were chosen using a Sobol sequence. Once trained, the performance of each model was evaluated using mean absolute error (MAE) against the true function value on a grid of independent test points in the domain.

Table 1 displays the test set results. First comparing the PIGP and PSGP results, the inclusion of symmetry information clearly yields a gain in predictive performance, however the difference narrows as more collocation points are included. Comparing the GP and NN results, we see that the NN models are clearly outperformed for lower number of collocation points. Again, as more points are included, the performance of the PSNN begins to converge to that of the GP models, which aligns with the results reported in (Long et al., 2022). Figure 1 displays prediction errors for the PIGP, PSGP and PSNN models when 32 collocation locations are considered. The PSGP obtains lower errors than the other two models across the entire spatio-temporal domain.

Table 1. Test-set MAE ($\times 10^5$) for Fisher-like equation with different numbers $N_f = N_g$ virtual observations.

MODEL	16	32	64	128
PINN	37600	16000	6000	510
PSNN	25000	530	130	7.1
PIGP	1120	230	25.6	5.1
PSGP	**360**	**44.9**	**10.3**	**3.3**

4.3. Heat Equation

We next considered the heat equation over one spatial dimension, which takes the form

$$u_t - \theta u_{xx} = f = 0. \tag{25}$$

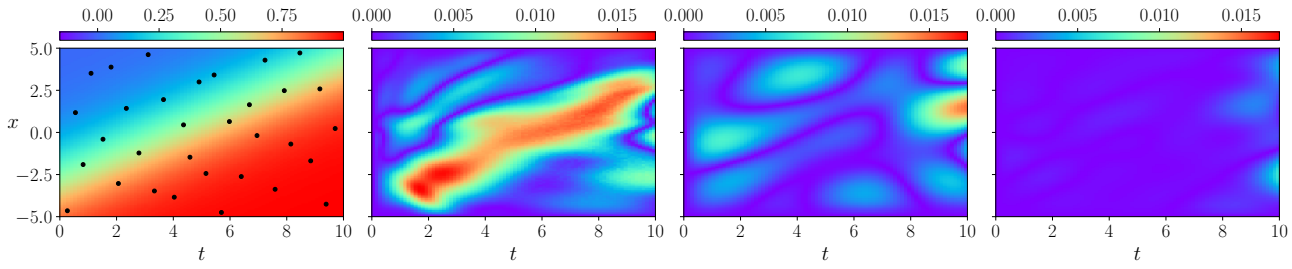
In this case, the equation models the process of thermal conduction in a one dimensional rod, with thermal conductivity level θ. Eq. (25) admits a Lie symmetry (Zhang et al., 2023c)

$$\mathcal{X}_{heat} = xt\partial_x + t^2\partial_t - \left(\frac{x^2}{4\theta} + \frac{t}{2}\right)u\partial_u, \tag{26}$$

with corresponding ISC (see Eq. (15)) of the form

$$\left(\frac{x^2}{4\theta} + \frac{t}{2}\right)u + xtu_x + t^2u_t = g = 0. \tag{27}$$

$\mathcal{X}_{heat}$ generates an exact solution of $u(x,t) = xt^{-3/2}\mathrm{e}^{-\frac{x^2}{4t}}$ with $\theta = 1$.



(a) True solution (b) PSNN - $|u - \hat{u}|$ (c) PIGP - $|u - \hat{u}|$ (d) PSGP - $|u - \hat{u}|$

Figure 1. Results for the **Fisher-like equation** with $N_f/N_g = 32$ collocation points. Panel (a) shows the true solution where the points show the location of the virtual observations, and panels (b)-(d) show error density plots for three different PIML models, with predictions denoted $\hat{u}$.

Here we examined the inverse problem of learning the parameter θ from data, over the domain $x \in [0, 1], t \in [0.5, 1]$. We assumed the initial temperature distribution $u(x, 0.5)$ and lower boundary value of $u(0, t) = 0$ were observed, while the solution value at the upper boundary of $x = 1$ was unobserved. To examine the effect of data size on the results, we performed the inverse problem for $N_u = 10, 25, 50, 100$ function space observations using both PIGPs and PSGPs. For the PIGPs, we introduced $N_f = N_u$ virtual observation of the PDE at the same input locations, i.e. $\mathbf{X}_f = \mathbf{X}_u$. For the PSGPs, we additionally incorporated $N_g = N_u$ virtual observations, again at the same locations. For each data set size, we performed the inverse problem under 20 randomly generated datasets, where both the input locations of the data and observation noise were resampled in each case. Noise levels were set to 1% in each case - see Appendix B for results under different levels of noise.

PIGP (blue) and PSGP (green) results are displayed in the top row of Figure 2, using density plots to capture the variation under dataset resampling. The plots show that the PSGP clearly outperforms the PIGP in terms of parameter inference accuracy ($|\theta - \hat{\theta}|$), prediction accuracy (MAE) and calibration of posterior predictive intervals, as measured by the continuous rank probability score (CRPS) (Gneiting & Raftery, 2007). The PSGP is able to obtain impressive accuracy even for very small datasets. For example, with only $N_u = 25$ data points, the worst parameter estimate is with two decimal places of the true value. Similarly, the PSGP can recover the true solution almost perfectly with $N_u = 10$ data points in contrast to the PIGP, which is clear from panels (e) and (f) of Figure 2. Also of note is the efficiency with which the PSGP can process the noisy data, allowing for monotonically increasing performance with more data points. By contrast, the PIGP results oscillate in this noisy, low data regime.

4.4. Wave Equation

Our final set of numerical experiments involved the wave equation

$$u_{tt} - u_{xx} = f = 0. \tag{28}$$

Eq. (28) admits a Lie symmetry (Márquez & Bruzón, 2021)

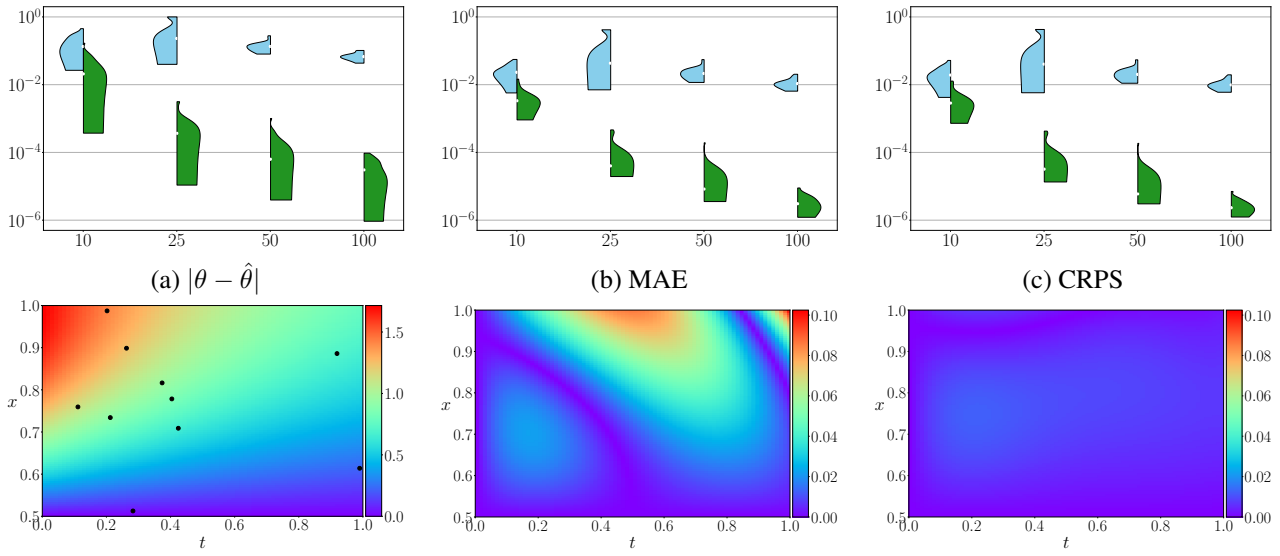
$$\mathcal{X}_{wave} = \partial_x + \partial_t, \tag{29}$$

with associated ISC (see Eq. (15)) of the form

$$u_t + u_x = g = 0. \tag{30}$$

Here we considered the forward problem of learning the form of the underlying solution from datasets of $N_u \in \{16, 32, 64, 128, 256\}$ noise-corrupted observations. We used the particular solution $u(x, t) = sn(x - t \mid 0.5)$ with sn the Jacobi elliptic sine function for $x \in [0, 10]$ and $t \in [0, 10]$ (see Figure 3 (a)). Predictive performance was evaluated against an independent grid of test points over the same input ranges.

The PIGP and PINN models incorporated the PDE by using $N_f = 512$ virtual observations of Eq. (28) from across the spatio-temporal domain. For the PSGP, PDE and ISC information could have been incorporated in the same manner. However, in this instance, it is possible to deduce from the ISC that $u(x, t) = w(z)$ for some function w, with $z = x - t$ known as the *canonical coordinate* (Márquez & Bruzón, 2021, Reduction 1.). We leveraged this reduction by defining the PSGP's kernel directly on the canonical coordinate system. In Appendix C.2, we prove that doing so ensured that both the ISC and the PDE were explicitly satisfied with probability one. Therefore, virtual observations $\boldsymbol{y}_f$ and $\mathbf{0}_g$ were not required in this case, and the PSGP reduced to a standard GP in which only the noisy u-space observations considered. In exactly the same manner, we defined a PSNN on the canonical coordinates to also explicitly enforce the PDE and ISC. We remark that applying the method introduced by Härkönen et al. (2023) for kernel design under linear PDE constraints to the Wave PDE and ISC considered here yields the exact same form of kernel on



(a) $|\theta - \hat{\theta}|$ (b) MAE (c) CRPS

(d) True solution (e) PIGP - $|u - \hat{u}|$ (f) PSGP - $|u - \hat{u}|$

Figure 2. Results for the **heat equation**. The top row shows density plots for the PIGP (blue) and PSGP (green) models under three evaluation metrics, which were obtained from the variation observed under dataset resampling. The bottom row shows the true solution and error density plots for the PIGP and PSGP models when $N_u = 10$. The points in panel (d) show the locations of the observed data.

the canonical coordinates - for further details, see Example 4.3 of their paper.

Table 2 reports test set MAE for each model against the size of the training data. Once again, the sample efficiency of the PSGP is clear, as it obtained the lowest errors with $N_u \leq 128$. For $N_u = 128$, the errors of the PSNN were lowest, while for $N_u = 256$ observations, the results of all four models began to converge. Note that for $N_u \leq 128$, the PIGP learned the trivial solution of $\hat{u} \approx 0$. This is a well-known problem in PIML in the case of sparse observational data (Leiteritz & Pflüger, 2021; Krishnapriyan et al., 2021). Counterintuitively, using more collocation points here actually yields worse accuracy for the PIGP, as it makes the attractor domain of the trivial solution cover the entire parameter space. Only once N_u exceeds approximately 25-30% of N_f is the influence of the observation data strong enough to recover the true solution accurately.

Table 2. Test-set MAE ($\times 10^2$) for Wave Equation, for different number N_u of noisy function observations.

MODEL	16	32	64	128	256
PINN	24.5	11.2	7.1	3.5	3.1
PSNN	20.1	10.3	4.3	**2.1**	2.8
PIGP	66.7	66.9	66.9	51.3	3.1
PSGP	**14.9**	**4.4**	**4.0**	3.2	**1.5**

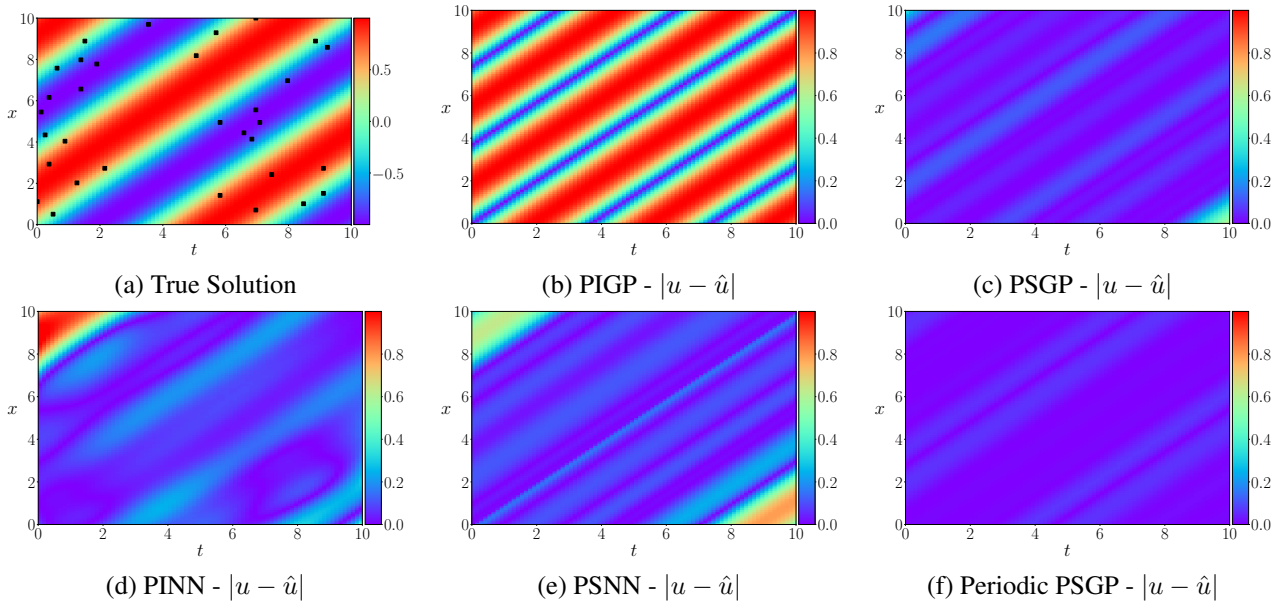
Density plots of prediction errors with $N_u = 32$ observations are displayed in Figure 3. For the PINN, PSNN and PSGP models, the highest errors are incurred at the upper left and bottom right of the spatio-temporal domain, which correspond to the lower and upper regions of the canonical coordinate space respectively. In order for better accuracy to be obtained in these regions, additional prior information can be used. For instance, the use of a periodic kernel in the specification of the PSGP allows for the solution to be recovered almost perfectly across the domain, as is illustrated in panel (f) of the figure.

5. Conclusion

We have introduced PSGPs, which leverage the ISC induced by a Lie symmetry to improve GP-based modelling of physical systems governed by PDEs. Using numerical experiments involving three different PDEs, we have shown that incorporating a known Lie symmetry improves accuracy both in the context of forward and inverse problems. Furthermore, comparisons with neural networks demonstrate the superior performance of PSGPs in the presence of sparse data.

The clear limitation of our approach is its restriction to low to medium-sized datasets, due to the cubic complexity of GP inference with respect to the size of the training data. However, several well established methods exist for overcoming this computational bottleneck, e.g. (Hensman et al., 2013), which could be made use of in this context. It would also be of interest to evaluate the performance of deep GPs (Damianou & Lawrence, 2013) for problems involving physics and symmetry information.

We have additionally assumed that the Lie symmetry is



(a) True Solution (b) PIGP - $|u - \hat{u}|$ (c) PSGP - $|u - \hat{u}|$

(d) PINN - $|u - \hat{u}|$ (e) PSNN - $|u - \hat{u}|$ (f) Periodic PSGP - $|u - \hat{u}|$

Figure 3. Results for the **wave equation** with $N_u = 32$ training observations. Panel (a) shows the true solution and panels (b)-(f) show error density plots for the different PIML models considered, where $\hat{u}$ denotes the predicted value. The points in panel (a) indicate the locations of the observed data.

known *a-priori*. This assumption could be relaxed however to allow the symmetry to be learned, as in (Dehmamy et al., 2021; Moskalev et al., 2022; Gabel et al., 2023). Finally, we have restricted our analysis to classical Lie symmetries, however the ISCs generated by non-classical symmetries (see Gandarias Bruzon (2009), for instance) could also be considered in our framework.

Acknowledgements

This work has been funded by the Engineering and Physical Sciences Research Council (EPSRC) of the United Kingdom, grant reference numbers EP/T017899/1, EP/S030875/1 and EP/S020950/1. HG would also like to thank the funding provided by the British Heart Foundation (PG/22/10930).

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Adler, R. J. *The Geometry of Random Fields*. Society for Industrial and Applied Mathematics, 2010.

Akhound-Sadegh, T., Perreault-Levasseur, L., Brandstetter, J., Welling, M., and Ravanbakhsh, S. Lie point symmetry and physics-informed networks. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Al-Nassar, S. and Nadjafikhah, M. Lie symmetry analysis and some new exact solutions of the fokker–planck equation. *Arabian Journal of Mathematics*, 12:467–482, 2023.

Alvarez, M. A., Luengo, D., and Lawrence, N. D. Linear latent force models using gaussian processes. *IEEE transactions on pattern analysis and machine intelligence*, 35 (11):2693–2705, 2013.

Bakhshandeh-Chamazkoti, R. and Alipour, M. Lie symmetries reduction and spectral methods on the fractional two-dimensional heat equation. *Mathematics and Computers in Simulation*, 200:97–107, 2022.

Barber, D. and Wang, Y. Gaussian processes for bayesian estimation in ordinary differential equations. In *International conference on machine learning*, pp. 1485–1493. PMLR, 2014.

Berman, P. *Introductory Quantum Mechanics: A Traditional Approach Emphasizing Connections with Classical Physics*. Springer International Publishing, 2017.

Bishop, C. M. and Nasrabadi, N. M. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

Bluman, G. W. and Anco, S. C. *Symmetry and Integration Methods for Differential Equations*, volume 154 of *Ap-*

plied Mathematical Sciences. Springer, New York, NY, 2002.

Bluman, G. W. and Kumei, S. *Symmetries and differential equations*, volume 81. Springer Science & Business Media, 2013.

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

Brandstetter, J., Welling, M., and Worrall, D. E. Lie point symmetry data augmentation for neural pde solvers. In *Thirty-ninth International Conference on Machine Learning*, 2022.

Calderhead, B., Girolami, M., and Lawrence, N. Accelerating bayesian inference over nonlinear differential equations with gaussian processes. *Advances in neural information processing systems*, 21, 2008.

Champala, R., Jamal, S., and Khan, S. Fractional pricing models: Transformations to a heat equation and lie symmetries. *Fractal and Fractional*, 7(8), 2023.

Chen, J., Chen, Z., Zhang, C., and Wu, C. F. J. APIK: Active Physics-Informed Kriging Model with Partial Differential Equations, 2020. arXiv:2012.11798 [cs, stat].

Cherniha, R. and Kovalenko, S. Lie symmetries of nonlinear boundary value problems. *Communications in Nonlinear Science and Numerical Simulation*, 17(1):71–84, 2012.

Damianou, A. and Lawrence, N. D. Deep Gaussian processes. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, 2013.

Dehmamy, N., Walters, R., Liu, Y., Wang, D., and Yu, R. Automatic symmetry discovery with lie algebra convolutional network. *Advances in Neural Information Processing Systems*, 34:2503–2515, 2021.

Dondelinger, F., Filippone, M., Rogers, S., and Husmeier, D. ODE parameter inference using adaptive gradient matching with Gaussian processes. In *Proceedings of The 16th International Conference on Artificial Intelligence and Statistics*, pp. 216–228, 2013.

Duvenaud, D. *Automatic Model Construction with Gaussian Processes*. PhD thesis, University of Cambridge, 2014.

Fuchs, F., Worrall, D. E., Fischer, V., and Welling, M. Se(3)-transformers: 3d roto-translation equivariant attention networks. In *Thirty-fourth Conference on Neural Information Processing Systems*, 2020.

Gabel, A., Klein, V., Valperga, R., Lamb, J. S., Webster, K., Quax, R., and Gavves, E. Learning lie group symmetry transformations with neural networks. In *Topological, Algebraic and Geometric Learning Workshops 2023*, pp. 50–59. PMLR, 2023.

Gandarias, M. and Bruzon, M. Nonclassical potential symmetries for the burgers equation. *Nonlinear Analysis: Theory, Methods & Applications*, 71(12):e1826–e1834, 2009.

Gneiting, T. and Raftery, A. E. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

Goard, J. Finding symmetries by incorporating initial conditions as side conditions. *European Journal of Applied Mathematics*, 19:701–715, 2008.

Hao, Z., Liu, S., Zhang, Y., Ying, C., Feng, Y., Su, H., and Zhu, J. Physics-informed machine learning: A survey on problems, methods and applications, 2022.

Hensman, J., Fusi, N., and Lawrence, N. Gaussian processes for big data. In *Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, 2013.

Hydon, P. E. *Symmetry methods for differential equations: a beginner's guide*. Number 22. Cambridge University Press, 2000.

Härkönen, M., Lange-Hegermann, M., and Raita, B. Gaussian process priors for systems of linear partial differential equations with constant coefficients. In *International Conference on Machine Learning*, pp. 12587–12615. PMLR, 2023.

Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., and Yang, L. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021. Number: 6 Publisher: Nature Publishing Group.

Kharazmi, E., Zhang, Z., and Karniadakis, G. E. hp-vpinns: Variational physics-informed neural networks with domain decomposition. *Computer Methods in Applied Mechanics and Engineering*, 374:113547, 2021. ISSN 0045-7825.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization, 2017.

Krishnapriyan, A. S., Gholami, A., Zhe, S., Kirby, R., and Mahoney, M. W. Characterizing possible failure modes in physics-informed neural networks. *Advances in Neural Information Processing Systems*, 34, 2021.

Leiteritz, R. and Pflüger, D. How to Avoid Trivial Solutions in Physics-Informed Neural Networks, 2021. arXiv:2112.05620 [cs, stat].

Li, Z. and Tan, M. H. Y. Improving Gaussian Process Emulators with Boundary Information. In Steland, A. and Tsui, K.-L. (eds.), *Artificial Intelligence, Big Data and Data Science in Statistics: Challenges and Solutions in Environmetrics, the Natural Sciences and Technology*, pp. 171–192. Springer International Publishing, Cham, 2022.

Li, Z., Qiao, Z., and Tang, T. *Numerical Solution of Differential Equations: Introduction to Finite Difference and Finite Element Methods*. Cambridge University Press, 2017.

Liu, S., Hao, Z., Ying, C., Su, H., Zhu, J., and Cheng, Z. A Unified Hard-Constraint Framework for Solving Geometrically Complex PDEs, 2022. arXiv:2210.03526 [cs].

Long, D., Wang, Z., Krishnapriyan, A., Kirby, R., Zhe, S., and Mahoney, M. AutoIP: A United Framework to Integrate Physics into Gaussian Processes. In *Thirty-ninth International Conference on Machine Learning*, 2022.

Macdonald, B., Higham, C., and Husmeier, D. Controversy in mechanistic modelling with gaussian processes. In *International conference on machine learning*, pp. 1539–1547. PMLR, 2015.

Meng, X., Li, Z., Zhang, D., and Karniadakis, G. E. Ppinn: Parareal physics-informed neural network for time-dependent pdes. *Computer Methods in Applied Mechanics and Engineering*, 370:113250, 2020.

Mialon, G., Garrido, Q., Lawrence, H., Rehman, D., LeCun, Y., and Kiani, B. Self-supervised learning with lie symmetries for partial differential equations. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Moskalev, A., Sepliarskaia, A., Sosnovik, I., and Smeulders, A. Liegg: Studying learned lie group generators. *Advances in Neural Information Processing Systems*, 35: 25212–25223, 2022.

Murray, I. and Adams, R. P. Slice sampling covariance hyperparameters of latent gaussian models. In *Advances in Neural Information Processing Systems*, volume 23, 2010.

Márquez, A. P. and Bruzón, M. S. Lie point symmetries, traveling wave solutions and conservation laws of a nonlinear viscoelastic wave equation. *Mathematics*, 9(17), 2021.

Najim, K., Ikonen, E., and Daoud, A.-K. *Chapter 1 - Stochastic Processes*. Kogan Page Science, 2004.

Nevin, J. W., Vaquero-Caballero, F. J., Ives, D. J., and Savory, S. J. Physics-informed gaussian process regression for optical fiber communication systems. *J. Lightwave Technol.*, 39(21):6833–6844, Nov 2021.

Nguyen-Thanh, V. M., Zhuang, X., and Rabczuk, T. A deep energy method for finite deformation hyperelasticity. *European Journal of Mechanics - A/Solids*, 80:103874, 2020.

Pan, R., Gu, M., and Wu, J. Physics-informed gaussian process regression of in operando capacitance for carbon supercapacitors. *Energy Advances*, 2:843–853, 2023.

Pateras, J., Rana, P., and Ghosh, P. A taxonomic survey of physics-informed machine learning. *Applied Sciences*, 13(12), 2023.

Pförtner, M., Steinwart, I., Hennig, P., and Wenger, J. Physics-informed gaussian process regression generalizes linear pde solvers, 2024.

Raissi, M., Perdikaris, P., and Karniadakis, G. E. Machine learning of linear differential equations using Gaussian processes. *Journal of Computational Physics*, 348:683–693, 2017.

Raissi, M., Perdikaris, P., and Karniadakis, G. E. Numerical Gaussian Processes for Time-Dependent and Nonlinear Partial Differential Equations. *SIAM Journal on Scientific Computing*, 40(1):A172–A198, 2018. Publisher: Society for Industrial and Applied Mathematics.

Raissi, M., Perdikaris, P., and Karniadakis, G. E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

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

Satorras, V. G., Hoogeboom, E., and Welling, M. E(n) equivariant graph neural network. In *Thirty-eight International Conference on Machine Learning*, 2021.

Sheng, H. and Yang, C. PFNN: A Penalty-Free Neural Network Method for Solving a Class of Second-Order Boundary-Value Problems on Complex Geometries. *Journal of Computational Physics*, 428:110085, 2021. arXiv:2004.06490 [cs, math].

Tan, M. Gaussian process modeling with boundary information. *Statistica Sinica*, 2016.

Tartakovsky, A. M., Ma, T., Barajas-Solano, D. A., and Tipireddy, R. Physics-informed gaussian process regression for states estimation and forecasting in power

grids. *International Journal of Forecasting*, 39(2):967–980, 2023.

Thomas, N., Smidt, T., Kearnes, S., Yang, L., Li, L., Kohlhoff, K., and Riley, P. Tensor field networks: Rotation- and translation-equivariant neural networks for 3d point clouds, 2018.

Verma, A., Jiwari, R., and Koksal, M. E. Analytic and numerical solutions of nonlinear diffusion equations via symmetry reductions. *Advances in Difference Equations*, (1):229, 2014.

Wang, R., Walters, R., and Yu, R. Incorporating symmetry into deep dynamics models for improved generalization. In *International Conference on Learning Representations*, 2021.

Zhang, H., Cai, S.-J., Li, J.-Y., Liu, Y., and Zhang, Z.-Y. Enforcing generalized conditional symmetry in physics-informed neural network for solving the kdv-like equation with robin initial/boundary conditions. *Nonlinear Dynamics*, 111, 2023a.

Zhang, Y., Pan, J., Li, L. K., Liu, W., Chen, Z., Liu, X., and Wang, J. On the properties of kullback-leibler divergence between multivariate gaussian distributions. *Advances in Neural Information Processing Systems*, 36, 2024.

Zhang, Z. and Chen, Y. Classical and nonclassical symmetries analysis for initial value problems. *Physics Letters A*, 374(9):1117–1120, 2010.

Zhang, Z.-Y., Zhang, H., Liu, Y., Li, J.-Y., and Liu, C.-B. Generalized conditional symmetry enhanced physics-informed neural network and application to the forward and inverse problems of nonlinear diffusion equations. *Chaos, Solitons & Fractals*, 168:113169, 2023b.

Zhang, Z.-Y., Zhang, H., Zhang, L.-S., and Guo, L.-L. Enforcing continuous symmetries in physics-informed neural network for solving forward and inverse problems of partial differential equations. *Journal of Computational Physics*, 492:112415, 2023c.

A. Illustrative example of the Lie symmetry method

Given a physical system represented by some function $u(x, y)$, we consider the Poisson equation on the unit disc $\Omega = \left\{(x, y) : x^2 + y^2 < 1\right\}$ with homogeneous Dirichlet boundary conditions, i.e.

$$u_{xx} - u_{yy} = -4 \text{ in } \Omega, \tag{31}$$
$$u = 0 \text{ on } \partial\Omega. \tag{32}$$

This BVP is rotationally invariant, or equivalently, it satisfies rotational symmetry. To see this, recall that the Laplacian operator (equal to $u_{xx} - u_{yy}$ in this case) is rotationally invariant, and note that the circular domain Ω and homogeneous boundary conditions are also left invariant under rotation. In more general applications, invariant solutions describe the behaviour of the system "far away" from boundary conditions (Hydon, 2000, Chapter 9), however it is possible in some cases to construct a solution to a BVP by a composition of invariant solutions (Bluman & Kumei, 2013).

Given the existence of rotational symmetry, we present below the two different derivations of the induced invariant surface condition (ISC) (see Eq. (15). The first derivation makes use of a constraint which the specific assumption of rotational invariance places on the directional derivative of u. The second derivation simply follows step by step the Lie symmetry method from Section 2.4. As is clear below, both derivations yield the same result. The beauty and power of the Lie symmetry method, however, is its generality to other symmetry transformations beyond rotation.

A.1. Derivation of ISC given rotational invariance

If u is rotationally invariant, then this means that, for each point $(x, y) \in \Omega$, the rate of change of u is *equal to zero*, with respect to changes in any direction *orthogonal* to (x, y). We can formulate this mathematically using the directional derivative as

$$\boldsymbol{v} \cdot \left[u_x(x, y)\ u_y(x, y)\right]^\top = 0, \tag{33}$$

for any $\boldsymbol{v} \in \mathbb{R}^2$ orthogonal to (x, y). Note that $(-y, x)$ is orthogonal to (x, y). Plugging $\boldsymbol{v} = (-y, x)$ into Eq. (33) and expanding out yields the following PDE

$$-yu_x(x, y) + xu_y(x, y) = 0 \ \forall\ (x, y) \in \Omega, \tag{34}$$

or in shorthand notation,

$$-yu_x + xu_y = 0. \tag{35}$$

This PDE mathematically encodes the assumption of rotational invariance.

A.2. Derivation of ISC following Section 2.4

The assumption that $u(x, y)$ is rotationally invariant means that $u(x, y)$ satisfies symmetry under rotation of the 2D plane. When equipped with a binary operator defined to be the sequential application of two rotations, the set of all such 2D rotations forms what is called the *special orthogonal group*, or SO(2), or the equivalently (two-dimensional) *rotation group*. Note that each 2D rotation of the plane can be parametrised by a single number $\varepsilon \in [0, 2\pi)$, namely the angle of rotation. This allows us to express the group as

$$\begin{aligned} x^* &= x\cos\varepsilon + y\sin\varepsilon, \\ y^* &= -x\sin\varepsilon + y\cos\varepsilon. \end{aligned} \tag{36}$$

For details on this equation, we refer the reader to (Bluman & Anco, 2002, page 45). Furthermore, the rotation group is in fact a *Lie group*, i.e. a group which is also a smooth manifold. To see this, note that due to the periodicity of rotations, the space of all possible rotation parameters ε can be represented as a circle, which is a (1D) manifold in embedded $\mathbb{R}^2$.

As discussed in Section 2.4, in order to find the infinitesimal generator associated with a Lie group, only the first order Taylor expansion of x^* and y^* around $\varepsilon = 0$ is required. Applying this expansion to Eq. (36) and neglecting terms above order ε^2 yields the following representation of the rotation group

$$\begin{aligned} x^* &= x + \varepsilon y + O\left(\varepsilon^2\right), \\ y^* &= y - \varepsilon x + O\left(\varepsilon^2\right), \end{aligned} \tag{37}$$

where again further details can be found in (Bluman & Anco, 2002, page 45). Recall that we are assuming u is invariant under the rotation group. This means that the PDE itself will continue to be satisfied after a rotation is applied - therefore, we have that the following Lie group of transformations leaves the PDE invariant:

$$\begin{aligned} x^* &= x + \varepsilon y + O\left(\varepsilon^2\right), \\ y^* &= y - \varepsilon x + O\left(\varepsilon^2\right), \\ u^* &= u. \end{aligned} \tag{38}$$

This is the form of Eq. (13) in the particular case of rotational symmetry. From the Lie group in Eq. (38), we can apply Definition 2.3 to derive the infinitesimal generator of the group as

$$\mathcal{X} = y\partial_x - x\partial_y, \tag{39}$$

where we have replaced t in with y in the notation from Section 2.4. Finally, Definition 2.4 can be used to derive the form of the induced ISC for this example:

$$-yu_x + xu_y = 0, \tag{40}$$

which is the same as Eq. (35) above.

A.3. Exact solution

The Poisson BVP stated in Eq. (31) can be solved to yield

$$u(x, y) = 1 - x^2 - y^2. \tag{41}$$

The form of the ISC derived above can be validated by plugging in this form of u and showing that it equals zero.

B. Additional Experimental Results

The experiments involving the heat equation in Section 4.3 assumed fixed observation noise levels of 1%, measured with respect to signal variance. Figure 4 below displays results for two additional levels of noise: noise-free (i.e. 0%) and 2.5%. In all cases, we see that the PSGP outperforms the PIGP. Furthermore, it is clear that the results obtained using a PSGP are significantly more robust to increasing noise than those obtained with a PIGP.

C. Proofs

C.1. Proof of Dirichlet boundary condition enforcement

Proof. From Eq. (22), we know that for each $\boldsymbol{x}_s^{(l)}$, the PSGP posterior $\hat{p}$ over the associated function value $u(\boldsymbol{x}_s^{(l)})$ has the form

$$\hat{p}(u(\boldsymbol{x}_s^{(l)})) = \mathcal{N}(u(\boldsymbol{x}_s^{(l)}) \mid \hat{\mu}_s^{(l)}, \hat{\Sigma}_s^{(l)}), \text{ with} \tag{42}$$

$$\hat{\mu}_s^{(l)} = \tilde{m}_u(\boldsymbol{x}_s^{(l)}) + \boldsymbol{k}_{sh}^{(l)}\boldsymbol{b} \tag{43}$$

$$\hat{\Sigma}_s^{(l)} = \tilde{k}_{uu}(\boldsymbol{x}_s^{(l)}, \boldsymbol{x}_s^{(l)}) + \boldsymbol{k}_{sh}^{(l)}\mathbf{B}\boldsymbol{k}_{hs}^{(l)}, \tag{44}$$

where $[\boldsymbol{k}_{sh}^{(l)}]^{(1,j)} = \text{Cov}\left(u(\boldsymbol{x}_s^{(l)}), h^{(j)}\right)$, $\boldsymbol{b} = \mathbf{K}_{hh}^{-1}(\boldsymbol{a} - \boldsymbol{m}_h)$ and $\mathbf{B} = \mathbf{K}_{hh}^{-1} - \mathbf{K}_{hh}^{-1}\mathbf{A}\mathbf{A}^\top\mathbf{K}_{hh}^{-1}$.

We first remark that, since the mean function is both continuous and satisfies the boundary conditions by assumption, we have

$$\lim_{l\to\infty} \tilde{m}_u(\boldsymbol{x}_s^{(l)}) = \tilde{m}_u\left(\lim_{l\to\infty} \boldsymbol{x}_s^{(l)}\right) = \tilde{m}_u(\boldsymbol{x}_b) = \mathcal{B}(\boldsymbol{x}_b). \tag{45}$$

Similarly, the continuity of the distance function ϕ means

$$\lim_{l\to\infty} \phi(\boldsymbol{x}_s^{(l)}) = \phi\left(\lim_{l\to\infty} \boldsymbol{x}_s^{(l)}\right) = \phi(\boldsymbol{x}_b) = 0, \tag{46}$$

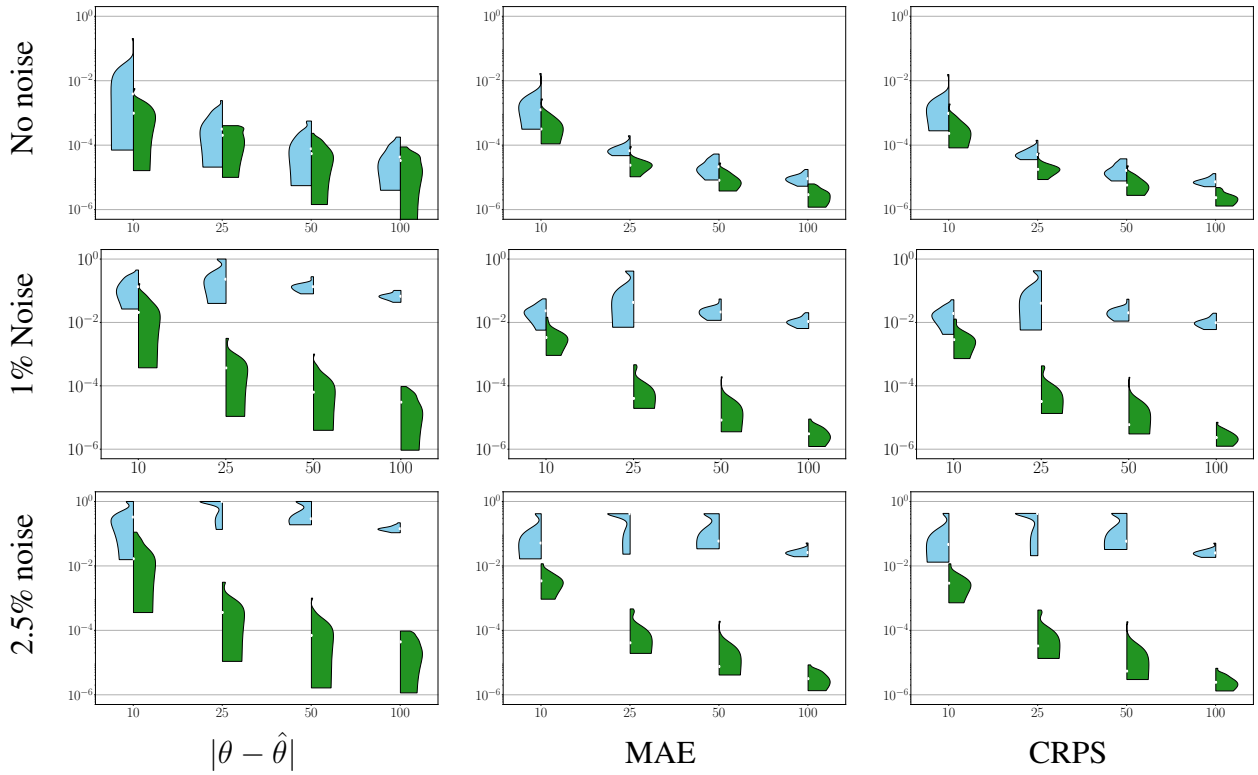


Figure 4. Additional results for the **heat equation** displayed as density plots, which were obtained from the variation observed under dataset resampling. Blue represents the PIGP results, and green the PSGP results. Each row corresponds to a different observation noise level. The left column shows parameter estimation error ($|\theta - \hat{\theta}|$), the centre prediction error (MAE), and the right column measures calibration of the prediction uncertainty intervals (CRPS).

which in turn implies

$$\lim_{l\to\infty} \tilde{k}_{uu}(\boldsymbol{x}_s^{(l)}, \boldsymbol{x}_s^{(l)}) = \lim_{l\to\infty} \phi(\boldsymbol{x}_s^{(l)})^2 k_{uu}(\boldsymbol{x}_s^{(l)}, \boldsymbol{x}_s^{(l)}) = 0, \tag{47}$$

since k_{uu} is continuous and hence bounded on Ω.

By Eq. (11), $\mathrm{Cov}\left(u(\boldsymbol{x}_s^{(l)}), h^{(j)}\right)$ takes the form

$$\begin{aligned}
\mathrm{Cov}\left(u(\boldsymbol{x}_s^{(l)}), h^{(j)}\right) &= \mathcal{L}_{\boldsymbol{x}'}^{(j)} \tilde{k}_{uu}\left(\boldsymbol{x}_s^{(l)}, \boldsymbol{x}_h^{(j)}\right) &(48)\\
&= \mathcal{L}_{\boldsymbol{x}'}^{(j)} \phi(\boldsymbol{x}_s^{(l)}) \phi(\boldsymbol{x}_h^{(j)}) k_{uu}\left(\boldsymbol{x}_s^{(l)}, \boldsymbol{x}_h^{(j)}\right) &(49)\\
&= \phi(\boldsymbol{x}_s^{(l)}) \mathcal{L}_{\boldsymbol{x}'}^{(j)} \phi(\boldsymbol{x}_h^{(j)}) k_{uu}\left(\boldsymbol{x}_s^{(l)}, \boldsymbol{x}_h^{(j)}\right), &(50)
\end{aligned}$$

where $\mathcal{L}_{\boldsymbol{x}'}^{(j)}$ is the linear operator associated with the j^{th} element of $\boldsymbol{h}$ (this is equal to the identity mapping for observations in u-space), and $\boldsymbol{x}_h^{(j)}$ is the location in input space which corresponds to the j^{th} element. Recall that we assume k_{uu} is sufficiently smooth that the result of applying $\mathcal{L}_{\boldsymbol{x}'}^{(j)}$ yields a function which is continuous. Since the domain Ω is both bounded and closed, this implies that this function is bounded. Coupled with the limiting form of ϕ (see Eq. (46)) this means

$$\lim_{l\to\infty} [\boldsymbol{k}_{sh}^{(l)}]^{(1,j)} = 0 \tag{51}$$

We now consider the form of the approximate posterior mean as $l \to \infty$:

$$\lim_{l\to\infty} \hat{\mu}_s^{(l)} = \lim_{l\to\infty} \left[\tilde{m}_u(\boldsymbol{x}_s^{(l)}) + \boldsymbol{k}_{sh}^{(l)}\boldsymbol{b}\right] \tag{52}$$

$$= \lim_{l\to\infty} \left[\tilde{m}_u(\boldsymbol{x}_s^{(l)}) + \sum_{j=1}^{H} [\boldsymbol{k}_{sh}^{(l)}]^{(1,j)}[\boldsymbol{b}]^{(j)}\right] \tag{53}$$

$$= \lim_{l\to\infty} \tilde{m}_u(\boldsymbol{x}_s^{(l)}) + \lim_{l\to\infty} \sum_{j=1}^{H} [\boldsymbol{k}_{sh}^{(l)}]^{(1,j)}[\boldsymbol{b}]^{(j)} \tag{54}$$

$$= \lim_{l\to\infty} \tilde{m}_u(\boldsymbol{x}_s^{(l)}) + \sum_{j=1}^{H} [\boldsymbol{b}]^{(j)} \lim_{l\to\infty} [\boldsymbol{k}_{sh}^{(l)}]^{(1,j)} \tag{55}$$

$$= \mathcal{B}(\boldsymbol{x}_b) + \sum_{j=1}^{H} [\boldsymbol{b}]^{(j)} \cdot 0 \tag{56}$$

$$= \mathcal{B}(\boldsymbol{x}_b). \tag{57}$$

Similarly, the limiting form of the approximate posterior variance is

$$\lim_{l\to\infty} \hat{\Sigma}_s^{(l)} = \lim_{l\to\infty} \left[\tilde{k}_{uu}(\boldsymbol{x}_s^{(l)}, \boldsymbol{x}_s^{(l)}) + \boldsymbol{k}_{sh}^{(l)}\mathbf{B}\boldsymbol{k}_{hs}^{(l)}\right] \tag{58}$$

$$= \lim_{l\to\infty} \left[\tilde{k}_{uu}(\boldsymbol{x}_s^{(l)}, \boldsymbol{x}_s^{(l)}) + \sum_{i=1}^{H}\sum_{j=1}^{H} [\boldsymbol{k}_{sh}^{(l)}]^{(1,j)}[\mathbf{B}]^{(i,j)}[\boldsymbol{k}_{hs}^{(l)}]^{(1,i)}\right] \tag{59}$$

$$= \lim_{l\to\infty} \tilde{k}_{uu}(\boldsymbol{x}_s^{(l)}, \boldsymbol{x}_s^{(l)}) + \sum_{i=1}^{H}\sum_{j=1}^{H} [\mathbf{B}]^{(i,j)} \lim_{l\to\infty} [\boldsymbol{k}_{sh}^{(l)}]^{(1,j)} \lim_{l\to\infty} [\boldsymbol{k}_{hs}^{(l)}]^{(i,1)} \tag{60}$$

$$= 0. \tag{61}$$

Let $\epsilon > 0$ and $\Delta > 0$ be arbitrary. To show convergence in probability, we need to show there exists a number L such that

$$\mathbb{P}\left(\left|u(\boldsymbol{x}_s^{(l)}) - \mathcal{B}(\boldsymbol{x}_b)\right| > \epsilon\right) < \Delta, \tag{62}$$

for all $l > L$ (Najim et al., 2004, Definition 16). Since $\lim_{l\to\infty} \hat{\mu}_s^{(l)} = \mathcal{B}(\boldsymbol{x}_b)$, there exists L_1 such that for all $l > L_1$, we have

$$\left|\hat{\mu}_s^{(l)} - \mathcal{B}(\boldsymbol{x}_b)\right| < \frac{\epsilon}{2}. \tag{63}$$

Similarly, since $\lim_{l\to\infty} \hat{\Sigma}_s^{(l)} = 0$, there exists L_2 such that

$$\left|\hat{\Sigma}_s^{(l)}\right| < \frac{\epsilon^2}{4}\Delta, \tag{64}$$

for all $l > L_2$. Let $L = \max(L_1, L_2)$. Then, for all $l > L$, we have

$$\mathbb{P}\left(\left|u(\boldsymbol{x}_s^{(l)}) + \mathcal{B}(\boldsymbol{x}_b)\right| > \epsilon\right) = \mathbb{P}\left(\left|u(\boldsymbol{x}_s^{(l)}) - \hat{\mu}_s^{(l)} + \hat{\mu}_s^{(l)} - \mathcal{B}(\boldsymbol{x}_b)\right| > \epsilon\right) \tag{65}$$

$$\leq \mathbb{P}\left(\left|u(\boldsymbol{x}_s^{(l)}) - \hat{\mu}_s^{(l)}\right| + \left|\hat{\mu}_s^{(l)} - \mathcal{B}(\boldsymbol{x}_b)\right| > \epsilon\right) \tag{66}$$

$$< \mathbb{P}\left(\left|u(\boldsymbol{x}_s^{(l)}) - \hat{\mu}_s^{(l)}\right| + \frac{\epsilon}{2} > \epsilon\right) \tag{67}$$

$$= \mathbb{P}\left(\left|u(\boldsymbol{x}_s^{(l)}) - \hat{\mu}_s^{(l)}\right| > \frac{\epsilon}{2}\right) \tag{68}$$

$$< \frac{\hat{\Sigma}_s^{(l)}}{(\varepsilon/2)^2} \tag{69}$$

$$< \frac{(\varepsilon/2)^2\Delta}{(\varepsilon/2)^2} \tag{70}$$

$$= \Delta. \tag{71}$$

where the second last inequality holds by Chebyshev's inequality:

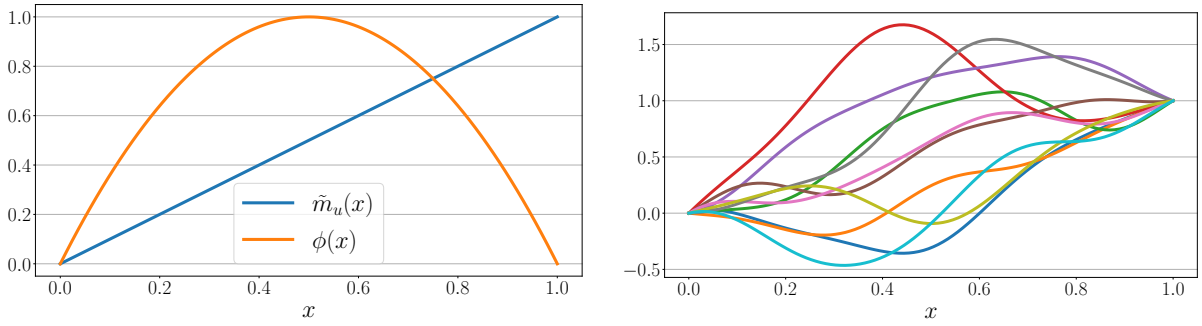
$$\mathbb{P}\left(\left|u(\boldsymbol{x}_s^{(l)}) - \hat{\mu}_s^{(l)}\right| > \lambda\right) < \frac{\hat{\Sigma}_s^{(l)}}{\lambda^2} \text{ for any } \lambda > 0 \tag{72}$$

by setting $\lambda = \varepsilon/2$.

□

C.1.1. One dimensional example

If we consider the one dimensional case where $u : [0, 1] \to \mathbb{R}$ with Dirichlet boundary conditions $u(0) = 0$ and $u(1) = 1$, constructing the mean function $\tilde{m}_u$ and distance function ϕ is straightforward - we simply set $\tilde{m}_u(x) = x$ and $\phi(x) = 4x(1 - x)$. These functions are plotted in Figure 5 (a). Panel (b) shows 10 samples from a GP with mean function $\tilde{m}_u(x)$ and kernel $\phi(x)\phi(x')k_{uu}(x, x')$ with k_{uu} the rational-quadratic kernel. All samples satisfy the Dirichlet conditions.



(a) (b)

Figure 5. Panel (a) shows mean function $\tilde{m}_u(x) = x$ and distance function $\phi(x) = 4x(1 - x)$ discussed in Section C.1.1. Panel (b) shows 10 samples from a GP with mean function $\tilde{m}_u(x)$ and kernel $\phi(x)\phi(x')k_{uu}(x, x')$ with k_{uu} the rational-quadratic kernel.

C.2. Proof of exact enforcement of Wave PDE and ISC

Theorem C.1. *Let $u : \mathbb{R}^2 \to \mathbb{R}$ follow a Gaussian process of the form*

$$u \sim \mathcal{GP}\left(m_u(x - t), k_{uu}(x - t, x' - t')\right), \tag{73}$$

Consider the linear differential operators $\mathcal{L}_{x,t}^{ISC}$ and $\mathcal{L}_{x,t}^{PDE}$ corresponding to the Wave ISC (Eq. (30)) and Wave PDE (Eq. (28)) respectively, which take the form

$$\mathcal{L}_{x,t}^{ISC}[\cdot] = \partial_t[\cdot] + \partial_x[\cdot], \tag{74}$$

$$\mathcal{L}_{x,t}^{PDE}[\cdot] = \partial_{tt}[\cdot] - \partial_{xx}[\cdot]. \tag{75}$$

Then we have

$$\mathcal{L}_{x,t}^{ISC}[u] = g = 0, \tag{76}$$
$$\mathcal{L}_{x,t}^{PDE}[u] = f = 0 \tag{77}$$

where the final equalities hold with probability one.

Proof. Note that we assume u, m_f and k_{uu} are all sufficiently smooth for both differential operators $\mathcal{L}_{x,t}^{ISC}$ and $\mathcal{L}_{x,t}^{ISC}$ to be applied.

$\mathcal{L}_{x,t}^{ISC}$ is a linear differential operator. Therefore, as discussed in Section 2.3 (see Eqs. (9) and (10)), we have

$$g \sim \mathcal{GP}(m_g(x,t), k_{gg}([x,t],[x',t'])), \tag{78}$$

with

$$m_g(x,t) = \mathcal{L}_{x,t}^{ISC} m_u(x-t), \tag{79}$$
$$k_{gg}([x,t],[x',t']) = \mathcal{L}_{x,t}^{ISC}\mathcal{L}_{x',t'}^{ISC} k_{uu}(x-t, x'-t'). \tag{80}$$

Firstly evaluating the mean function, we have

$$m_g(x,t) = \mathcal{L}_{x,t}^{ISC} m_u(x-t) \tag{81}$$
$$= \frac{\partial}{\partial t} m_u(x-t) + \frac{\partial}{\partial x} m_u(x-t) \tag{82}$$

Using the canonical coordinates $z = x - t$, this can be re-written by the chain rule as

$$m_g(x,t) = \frac{\partial z}{\partial t}\frac{d}{dz} m_u(z) + \frac{\partial z}{\partial x}\frac{d}{dz} m_u(z) \tag{83}$$
$$= -\frac{d}{dz} m_u(z) + \frac{d}{dz} m_u(z) \tag{84}$$
$$= 0, \tag{85}$$

since $\frac{\partial z}{\partial t} = -1$ and $\frac{\partial z}{\partial x} = 1$.

To find k_{gg}, we first evaluate

$$\mathcal{L}_{x',t'}^{ISC} k_{uu}(x-t, x'-t') = \frac{\partial}{\partial t'} k_{uu}(x-t, x'-t') + \frac{\partial}{\partial x'} k_{uu}(x-t, x'-t') \tag{86}$$
$$= \frac{\partial z'}{\partial t'}\frac{\partial}{\partial z'} k_{uu}(z, z') + \frac{\partial z'}{\partial x'}\frac{\partial}{\partial z'} k_{uu}(z, z') \tag{87}$$
$$= -\frac{\partial}{\partial z'} k_{uu}(z, z') + \frac{\partial}{\partial z'} k_{uu}(z, z') \tag{88}$$
$$= 0, \tag{89}$$

from which it follows that $k_{gg}([x,t],[x',t']) = 0$.

Now let $(x,t) \in \mathbb{R}^2$ be arbitrary. By the definition of a GP, this implies that $g(x,t) \sim \mathcal{N}(0,0)$, i.e. a zero variance normal distribution. Recall that in the limit of infinite precision, a Gaussian distribution becomes a Dirac delta function δ centred with respect to its mean (Berman, 2017, Section 2.6), which is also zero in this case. This means that

$$\mathbb{P}(|g(x,t)| > \epsilon) = \int_{-\infty}^{-\epsilon} \delta(g-0)dg + \int_{\epsilon}^{\infty} \delta(g-0)dg \tag{90}$$
$$= 0 + 0 = 0 \tag{91}$$

for all $\epsilon > 0$, and therefore $g(x,t) = 0$ with probability one.

$\mathcal{L}^{PDE}_{x,t}$ is also a linear differential operator. Therefore, we have

$$f \sim \mathcal{GP}(m_f(x,t), k_f([x,t],[x',t']), \tag{92}$$

with

$$m_f(x,t) = \mathcal{L}^{PDE}_{x,t} m_u(x-t), \tag{93}$$
$$k_{ff}([x,t],[x',t']) = \mathcal{L}^{PDE}_{x,t}\mathcal{L}^{PDE}_{x',t'} k_{uu}(x-t, x'-t'). \tag{94}$$

The mean function can once again be found using the canonical coordinates to be

$$\begin{aligned}
m_f(x,t) &= \mathcal{L}^{PDE}_{x,t} m_u(x-t) && (95)\\
&= \frac{\partial^2}{\partial t^2} m_u(x-t) - \frac{\partial^2}{\partial x^2} m_u(x-t) && (96)\\
&= \frac{\partial}{\partial t}\left[\frac{\partial z}{\partial t}\frac{d}{dz} m_u(z)\right] + \frac{\partial}{\partial x}\left[\frac{\partial z}{\partial x}\frac{d}{dz} m_u(z)\right] && (97)\\
&= -\frac{\partial}{\partial t}\left[\frac{d}{dz} m_u(z)\right] - \frac{\partial}{\partial x}\left[\frac{d}{dz} m_u(z)\right] && (98)\\
&= -\frac{\partial z}{\partial t}\frac{d^2}{dz^2} m_u(z) - \frac{\partial z}{\partial x}\frac{d^2}{dz^2} m_u(z) && (99)\\
&= \frac{d^2}{dz^2} m_u(z) - \frac{d^2}{dz^2} m_u(z) && (100)\\
&= 0. && (101)
\end{aligned}$$

To find k_{ff}, we first evaluate

$$\begin{aligned}
\mathcal{L}^{PDE}_{x',t'} k_{uu}(x-t, x'-t') &= \frac{\partial^2}{(\partial t')^2} k_{uu}(x-t, x'-t') - \frac{\partial^2}{(\partial x')^2} k_{uu}(x-t, x'-t') && (102)\\
&= \frac{\partial}{\partial t'}\left[\frac{\partial z'}{\partial t'}\frac{\partial}{\partial z'} k_{uu}(z,z')\right] - \frac{\partial}{\partial x'}\left[\frac{\partial z'}{\partial x'}\frac{\partial}{\partial z'} k_{uu}(z,z')\right] && (103)\\
&= -\frac{\partial}{\partial t'}\frac{\partial}{\partial z'} k_{uu}(z,z') - \frac{\partial}{\partial x'}\frac{\partial}{\partial z'} k_{uu}(z,z') && (104)\\
&= -\frac{\partial z'}{\partial t'}\frac{\partial}{(\partial z')^2} k_{uu}(z,z') - \frac{\partial z'}{\partial x'}\frac{\partial}{(\partial z')^2} k_{uu}(z,z') && (105)\\
&= \frac{\partial}{(\partial z')^2} k_{uu}(z,z') - \frac{\partial}{(\partial z')^2} k_{uu}(z,z') && (106)\\
&= 0, && (107)
\end{aligned}$$

where once again we have made use of the fact that $\frac{\partial z}{\partial t} = -1$ and $\frac{\partial z}{\partial x} = 1$. The above result implies that implies $k_{ff}([x,t],[x',t']) = 0$ in turn, as in the case of k_{gg}.

This means that $f(x,t) \in \mathbb{R} \sim \mathcal{N}(0,0)$ for arbitrary $(x,t) \in \mathbb{R}^2$, and therefore, as with g above, $f = 0$ with probability one.

□

D. Exact form of ELBO (Eq. (21)) and posterior predictive distribution (Eq. (22))

D.1. ELBO

This section presents the steps required to compute each individual term in the ELBO from Eq.(21). Using the shorthand notation $\boldsymbol{d}_{0:m} = [\boldsymbol{d}_0; \boldsymbol{d}_1; \ldots; \boldsymbol{d}_m]$, we denote the vector of latent variables $\boldsymbol{h}$ as

$$h = [\boldsymbol{u}; \boldsymbol{d}_{0:m}; \boldsymbol{g}]. \tag{108}$$

For more details on $\boldsymbol{h}$, see the "Prior" subsection of Section 3.1 above. Denoting $\mathbf{S} = \mathbf{A}\mathbf{A}^\top$, the variational posterior $q(\boldsymbol{h})$ takes the form

$$q(\boldsymbol{h}) = \mathcal{N}(\boldsymbol{h} \mid \boldsymbol{a}, \mathbf{S}). \tag{109}$$

For more details on q and the variational parameters $\boldsymbol{a}$ and $\mathbf{A}$, see the "Inference" subsection of Section 3.1. In practice, we use the whitening trick (Murray & Adams, 2010) when evaluating the ELBO. For clarity of exposition here, however, we present the computations in un-whitened form.

In evaluating the terms of the ELBO relating to the respective log likelihoods of the real observations $\boldsymbol{y}_u$ and virtual observations $\boldsymbol{y}_f$ and $\mathbf{0}_g$, it is useful to split the mean and covariance of q into the block form given in Eq. (117), where the blocks correspond to the observations from each of the three different function spaces we consider (i.e. u, f and g space respectively).

$$q(\boldsymbol{h}) = q\left(\begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{d}_{0:m} \\ \boldsymbol{g} \end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{a}_u \\ \boldsymbol{a}_f \\ \boldsymbol{a}_g \end{bmatrix}, \begin{bmatrix} \mathbf{S}_{uu} & \mathbf{S}_{uf} & \mathbf{S}_{ug} \\ \mathbf{S}_{fu} & \mathbf{S}_{ff} & \mathbf{S}_{fg} \\ \mathbf{S}_{gu} & \mathbf{S}_{gf} & \mathbf{S}_{gg} \end{bmatrix}\right). \tag{110}$$

D.1.1. KL DIVERGENCE TERM

The first term in the ELBO is the KL divergence between the variational distribution over the latent vector $q(\boldsymbol{h})$ and the prior distribution $p(\boldsymbol{h})$. Recall that both q and p are Gaussian - the form of q is given in Eq. (109), while the form of p is given in Eq. (16). The KL divergence between two Gaussians can be evaluated to yield the below closed form expression (Zhang et al., 2024, Eq. (1))

$$\mathrm{KL}(q(\boldsymbol{h}) \| p(\boldsymbol{h})) = \frac{1}{2}\left(\log \frac{|\mathbf{K}_{hh}|}{|\mathbf{S}|} + \mathrm{Tr}\left(\mathbf{K}_{hh}^{-1}\mathbf{S}\right) + (\boldsymbol{m}_h - \boldsymbol{a})^\top \mathbf{K}_{hh}^{-1}(\boldsymbol{m}_h - \boldsymbol{a}) - n\right), \tag{111}$$

where n is the dimensionality of the latent vector $\boldsymbol{h}$.

D.1.2. EXPECTED LOG PROBABILITY OF $\boldsymbol{y}_f$ TERM

The second term in the ELBO is the expected log likelihood of the virtual PDE observations $\boldsymbol{y}_f$ under the variational distribution q. From the form of $p(\boldsymbol{y}_f \mid \boldsymbol{d}_{0:m})$ given in Eq. (18), this term can be expressed as:

$$\begin{aligned}
\mathbb{E}_q[\log p(\boldsymbol{y}_f \mid \boldsymbol{d}_{0:m})] &= \int q(\boldsymbol{h}) \log p(\boldsymbol{y}_f \mid \boldsymbol{d}_{0:m}) d\boldsymbol{h} \\
&= \int \mathcal{N}(\boldsymbol{h} \mid \boldsymbol{a}, \mathbf{S}) \log \mathcal{N}(\boldsymbol{y}_f \mid F_{\boldsymbol{\theta}}(\boldsymbol{d}_{0:m}), \sigma_f^2 \mathbf{I}_f) d\boldsymbol{h} \\
&= \int \mathcal{N}(\boldsymbol{d}_{0:m} \mid \boldsymbol{a}_f, \mathbf{S}_{ff}) \log \mathcal{N}(\boldsymbol{y}_f \mid F_{\boldsymbol{\theta}}(\boldsymbol{d}_{0:m}), \sigma_f^2 \mathbf{I}_f) d\boldsymbol{d}_{0:m} \\
&= \int \mathcal{N}(\boldsymbol{d}_{0:m} \mid \boldsymbol{a}_f, \mathbf{S}_{ff}) \log \prod_{i=1}^{N_f} \mathcal{N}(y_f^{(i)} \mid F_{\boldsymbol{\theta}}([\boldsymbol{d}_{0:m}]^{(i)}), \sigma_f^2) d\boldsymbol{d}_{0:m} \\
&= \int \mathcal{N}(\boldsymbol{d}_{0:m} \mid \boldsymbol{a}_f, \mathbf{S}_{ff}) \sum_{i=1}^{N_f} \log \mathcal{N}(y_f^{(i)} \mid F_{\boldsymbol{\theta}}([\boldsymbol{d}_{0:m}]^{(i)}), \sigma_f^2) d\boldsymbol{d}_{0:m} \\
&= \sum_{i=1}^{N_f} \int \mathcal{N}(\boldsymbol{d}_{0:m} \mid \boldsymbol{a}_f, \mathbf{S}_{ff}) \log \mathcal{N}(y_f^{(i)} \mid F_{\boldsymbol{\theta}}([\boldsymbol{d}_{0:m}]^{(i)}), \sigma_f^2) d\boldsymbol{d}_{0:m}
\end{aligned} \tag{112}$$

Due to the assumed non-linearity of $F_{\boldsymbol{\theta}}$, each of the N_f integrals above are analytically intractable, and we therefore instead use a Monte-Carlo sample to evaluate them.

D.1.3. EXPECTED LOG PROBABILITY OF $\mathbf{0}_g$ TERM

The third term in the ELBO is the expected log likelihood of the virtual ISC observations $\mathbf{0}_g$ under the variational distribution q. From the form of $p(\mathbf{0}_g \mid \boldsymbol{g})$ given in Eq. (19), this term can be expressed as:

$$\begin{aligned}\mathbb{E}_q[\log p(\mathbf{0}_g \mid \boldsymbol{g})] &= \int q(\boldsymbol{h}) \log p(\mathbf{0}_g \mid \boldsymbol{g}) d\boldsymbol{h} \\ &= \int \mathcal{N}(\boldsymbol{h} \mid \boldsymbol{a}, \mathbf{S}) \log \mathcal{N}(\mathbf{0}_g \mid \boldsymbol{g}, \sigma_g^2 \mathbf{I}_g) d\boldsymbol{h} \\ &= \int \mathcal{N}(\boldsymbol{g} \mid \boldsymbol{a}_g, \mathbf{S}_{gg}) \log \mathcal{N}(\mathbf{0}_g \mid \boldsymbol{g}, \sigma_g^2 \mathbf{I}_g) d\boldsymbol{g}, \end{aligned} \tag{113}$$

where $\boldsymbol{a}_g$ and $\mathbf{S}_{gg}$ come from the block-decomposition of $q(\boldsymbol{h})$ given in Eq. (117). The final integral above can be evaluated according to the result of Lemma D.2 to give:

$$\begin{aligned}\mathbb{E}_q[\log p(\mathbf{0}_g \mid \boldsymbol{g})] &= \log \mathcal{N}(\mathbf{0}_g \mid \boldsymbol{a}_g, \sigma_g^2 \mathbf{I}_g) - \frac{1}{2\sigma_g^2} \operatorname{Tr}[\mathbf{S}_{gg}] \\ &= \log \prod_{i=1}^{N_g} \mathcal{N}(0 \mid a_g^{(i)}, \sigma_g^2) - \frac{1}{2\sigma_g^2} \sum_{i=1}^{N_g} S_{gg}^{(i,i)} \\ &= \sum_{i=1}^{N_g} \log \mathcal{N}(0 \mid a_g^{(i)}, \sigma_g^2) - \frac{1}{2\sigma_g^2} \sum_{i=1}^{N_g} S_{gg}^{(i,i)} \\ &= \sum_{i=1}^{N_g} \left[\log \mathcal{N}(0 \mid a_g^{(i)}, \sigma_g^2) - \frac{S_{gg}^{(i,i)}}{2\sigma_g^2} \right]. \end{aligned} \tag{114}$$

D.1.4. EXPECTED LOG PROBABILITY OF $\boldsymbol{y}_u$ TERM

The fourth term of the ELBO is the expected log likelihood of solution-space observations $\boldsymbol{y}_u$ under the variational distribution q. Given the form of $p(\boldsymbol{y}_u \mid \boldsymbol{u})$ stated in Eq. (17), this term can be evaluated in exactly the same manner as the term in the ELBO involving the ISC observations, yielding:

$$\mathbb{E}_q[\log p(\boldsymbol{y}_u \mid \boldsymbol{u})] = \sum_{i=1}^{N_u} \left[\log \mathcal{N}(y_u^{(i)} \mid a_u^{(i)}, \sigma_u^2) - \frac{S_{uu}^{(i,i)}}{2\sigma_u^2} \right]. \tag{115}$$

D.2. Posterior predictive distribution

From Eq. (22), the (approximate) posterior predictive distribution over a vector of test inputs $\boldsymbol{u}_s$ is found as

$$\hat{p}(\boldsymbol{u}_s) = \int p(\boldsymbol{u}_s \mid \boldsymbol{h}) q(\boldsymbol{h}) d\boldsymbol{h}. \tag{116}$$

Since, by construction, the latent vector $\boldsymbol{h}$ is jointly Gaussian with solution space (i.e. u-space) observations, we have

$$p\left(\begin{bmatrix} \boldsymbol{u}_s \\ \boldsymbol{h} \end{bmatrix} \right) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{m}_s \\ \boldsymbol{m}_h \end{bmatrix}, \begin{bmatrix} \mathbf{K}_{ss} & \mathbf{K}_{sh} \\ \mathbf{K}_{hs} & \mathbf{K}_{hh} \end{bmatrix} \right). \tag{117}$$

The well-known formula for the form of a conditional multivariate Gaussian distribution (see for example Section 2.3.2 of Bishop (2006)) can then be applied to show that $p(\boldsymbol{u}_s \mid \boldsymbol{h})$ is a Gaussian of the form

$$p(\boldsymbol{u}_s | \boldsymbol{h}) = \mathcal{N}(\boldsymbol{u}_s | \boldsymbol{m}_s + \mathbf{K}_{sh} \mathbf{K}_{hh}^{-1} (\boldsymbol{a} - \boldsymbol{m}_h), \mathbf{K}_{ss} - \mathbf{K}_{sh} \mathbf{K}_{hh}^{-1} \mathbf{K}_{hs}). \tag{118}$$

Recall that $q(\boldsymbol{h})$ is also Gaussian (see Eq. (109) above). Therefore, if we let

$$\mathbf{W} = \mathbf{K}_{sh} \mathbf{K}_{hh}^{-1} \tag{119}$$

$$\boldsymbol{b} = \boldsymbol{m}_s - \mathbf{W} \boldsymbol{m}_h, \tag{120}$$

then Lemma D.1 can be applied to solve the integral in Eq. (116), yielding

$$\hat{p}(\boldsymbol{u}_s) = \mathcal{N}(\boldsymbol{u}_s \mid \hat{\boldsymbol{\mu}}_s, \hat{\boldsymbol{\Sigma}}_s), \tag{121}$$

where

$$\begin{aligned}\hat{\boldsymbol{\mu}}_s &= \mathbf{W}\boldsymbol{a} + \boldsymbol{b} \\ &= \mathbf{W}\boldsymbol{a} + \boldsymbol{m}_s - \mathbf{W}\boldsymbol{m}_h \\ &= \boldsymbol{m}_s + \mathbf{W}(\boldsymbol{a} - \boldsymbol{m}_h)\end{aligned} \tag{122}$$

and

$$\begin{aligned}\hat{\boldsymbol{\Sigma}}_s &= \mathbf{K}_{ss} - \mathbf{K}_{sh}\mathbf{K}_{hh}^{-1}\mathbf{K}_{hs} + \mathbf{W}\mathbf{S}\mathbf{W}^\top \\ &= \mathbf{K}_{ss} - \mathbf{W}\left(\mathbf{K}_{hs} - \mathbf{S}\mathbf{W}^\top\right) \qquad (123)\\ & \qquad (124)\end{aligned}$$

D.3. Useful Gaussian integration results

Lemma D.1. *Let $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ be random vectors, where*

$$\begin{aligned}p(\boldsymbol{x}_1) &= \mathcal{N}\left(\boldsymbol{x}_1 \mid \boldsymbol{\mu}_1, \mathbf{K}_1\right) \\ p(\boldsymbol{x}_2 \mid \boldsymbol{x}_1) &= \mathcal{N}\left(\boldsymbol{x}_2 \mid \mathbf{W}\boldsymbol{x}_1 + \boldsymbol{b}, \mathbf{K}_2\right).\end{aligned}$$

Then

$$p(\boldsymbol{x}_2) = \int p(\boldsymbol{x}_2 \mid \boldsymbol{x}_1)p(\boldsymbol{x}_1)d\boldsymbol{x}_1 = \mathcal{N}\left(\boldsymbol{x}_2 \mid \mathbf{W}\boldsymbol{\mu}_1 + \boldsymbol{b}, \mathbf{K}_2 + \mathbf{W}\mathbf{K}_1\mathbf{W}^\top\right)$$

Proof. See Section 2.3.3 of Bishop (2006). □

Lemma D.2. *Let $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ be random vectors where*

$$\begin{aligned}p(\boldsymbol{x}_1) &= \mathcal{N}\left(\boldsymbol{x}_1 \mid \boldsymbol{\mu}_1, \mathbf{K}_1\right) \\ p(\boldsymbol{x}_2 \mid \boldsymbol{x}_1) &= \mathcal{N}\left(\boldsymbol{x}_2 \mid \mathbf{W}\boldsymbol{x}_1 + \boldsymbol{b}, \mathbf{K}_2\right).\end{aligned}$$

Then

$$\mathbb{E}_{p(\boldsymbol{x}_1)}\left[\log p(\boldsymbol{x}_2 \mid \boldsymbol{x}_1)\right] = \log \mathcal{N}(\boldsymbol{x}_2 \mid \mathbf{W}\boldsymbol{\mu}_1 + \boldsymbol{b}, \mathbf{K}_2) - \frac{1}{2}\operatorname{Tr}\left[\mathbf{W}^\top\mathbf{K}_2^{-1}\mathbf{W}\mathbf{K}_1\right]$$

Proof.

$$
\begin{aligned}
\mathbb{E}_{p(\boldsymbol{x}_1)}\left[\log p(\boldsymbol{x}_2 \mid \boldsymbol{x}_1)\right] &= \mathbb{E}_{p(\boldsymbol{x}_1)}\left[\log\left(\det(2\pi\mathbf{K}_2)^{-\frac{1}{2}}\exp\left(-\frac{1}{2}(\boldsymbol{x}_2-(\mathbf{W}\boldsymbol{x}_1+\boldsymbol{b}))^\top\mathbf{K}_2^{-1}(\boldsymbol{x}_2-(\mathbf{W}\boldsymbol{x}_1+\boldsymbol{b}))\right)\right)\right] \\
&= -\frac{1}{2}\mathbb{E}_{p(\boldsymbol{x}_1)}\left[\log\det(2\pi\mathbf{K}_2)\right] - \frac{1}{2}\mathbb{E}_{p(\boldsymbol{x}_1)}\left[((\boldsymbol{x}_2-\boldsymbol{b})-\mathbf{W}\boldsymbol{x}_1)^\top\mathbf{K}_2^{-1}((\boldsymbol{x}_2-\boldsymbol{b})-\mathbf{W}\boldsymbol{x}_1)\right] \\
&= -\frac{1}{2}\log\det(2\pi\mathbf{K}_2) - \frac{1}{2}\mathbb{E}_{p(\boldsymbol{x}_1)}\left[(\boldsymbol{x}_2-\boldsymbol{b})^\top\mathbf{K}_2^{-1}(\boldsymbol{x}_2-\boldsymbol{b})\right] \\
&\quad - \frac{1}{2}\mathbb{E}_{p(\boldsymbol{x}_1)}\left[-2(\boldsymbol{x}_2-\boldsymbol{b})^\top\mathbf{K}_2^{-1}\mathbf{W}\boldsymbol{x}_1\right] - \frac{1}{2}\mathbb{E}_{p(\boldsymbol{x}_1)}\left[\boldsymbol{x}_1^\top\mathbf{W}^\top\mathbf{K}_2^{-1}\mathbf{W}\boldsymbol{x}_1\right] \\
&= -\frac{1}{2}\log\det(2\pi\mathbf{K}_2) - \frac{1}{2}(\boldsymbol{x}_2-\boldsymbol{b})^\top\mathbf{K}_2^{-1}(\boldsymbol{x}_2-\boldsymbol{b}) \\
&\quad - \frac{1}{2}\left[-2(\boldsymbol{x}_2-\boldsymbol{b})^\top\mathbf{K}_2^{-1}\mathbf{W}\boldsymbol{\mu}_1\right] - \frac{1}{2}\boldsymbol{\mu}_1^\top\mathbf{W}^\top\mathbf{K}_2^{-1}\mathbf{W}\boldsymbol{\mu}_1 - \frac{1}{2}\operatorname{Tr}\left[\mathbf{W}^\top\mathbf{K}_2^{-1}\mathbf{W}\mathbf{K}_1\right] \\
&= -\frac{1}{2}\log\det(2\pi\mathbf{K}_2) \\
&\quad - \frac{1}{2}\left[(\boldsymbol{x}_2-\boldsymbol{b})^\top\mathbf{K}_2^{-1}(\boldsymbol{x}_2-\boldsymbol{b}) - 2(\boldsymbol{x}_2-\boldsymbol{b})^\top\mathbf{K}_2^{-1}\mathbf{W}\boldsymbol{\mu}_1 + \boldsymbol{\mu}_1^\top\mathbf{W}^\top\mathbf{K}_2^{-1}\mathbf{W}\boldsymbol{\mu}_1\right] \\
&\quad - \frac{1}{2}\operatorname{Tr}\left[\mathbf{W}^\top\mathbf{K}_2^{-1}\mathbf{W}\mathbf{K}_1\right] \\
&= \log\mathcal{N}(\boldsymbol{x}_2 \mid \mathbf{W}\boldsymbol{\mu}_1+\boldsymbol{b}, \mathbf{K}_2) - \frac{1}{2}\operatorname{Tr}\left[\mathbf{W}^\top\mathbf{K}_2^{-1}\mathbf{W}\mathbf{K}_1\right]
\end{aligned}
$$

□