fractal and fractional

MDPI

Article

Parameter Estimation for Several Types of Linear Partial Differential Equations Based on Gaussian Processes

Wenbo Zhang and Wei Gu *

School of Statistics and Mathematics, Zhongnan University of Economics and Law, Wuhan 430073, China
* Correspondence: wei_gu@zuel.edu.cn

Abstract: This paper mainly considers the parameter estimation problem for several types of differential equations controlled by linear operators, which may be partial differential, integro-differential and fractional order operators. Under the idea of data-driven methods, the algorithms based on Gaussian processes are constructed to solve the inverse problem, where we encode the distribution information of the data into the kernels and construct an efficient data learning machine. We then estimate the unknown parameters of the partial differential Equations (PDEs), which include high-order partial differential equations, partial integro-differential equations, fractional partial differential equations and a system of partial differential equations. Finally, several numerical tests are provided. The results of the numerical experiments prove that the data-driven methods based on Gaussian processes not only estimate the parameters of the considered PDEs with high accuracy but also approximate the latent solutions and the inhomogeneous terms of the PDEs simultaneously.

Keywords: data-driven methods; Gaussian processes; inverse problems; partial integro-differential equations; fractional partial differential equations

Citation: Zhang, W.; Gu, W. Parameter Estimation for Several Types of Linear Partial Differential Equations Based on Gaussian Processes. *Fractal Fract.* **2022**, *6*, 433. https://doi.org/10.3390/fractalfract6080433

Academic Editor: Ricardo Almeida

Received: 5 July 2022
Accepted: 5 August 2022
Published: 8 August 2022

1. Introduction

In the era of big data, the study of data-driven methods and probabilistic machine-learning methods has increasingly attracted researchers [1–3]. Exploiting data-driven methods and machine-learning methods to solve the forward and inverse problem of partial differential equations (PDEs) has been valued [4–11], and (deep) neural networks are preferred [12–20].

Compared with other machine-learning methods, such as regularized least-squares classifiers (RLSCs) [21] and support vector machines (SVMs) [22], Gaussian processes possess a strict mathematical basis and coincide with the Bayesian estimation method in essence [23]. Raissi et al. (2017) [24] considered a new algorithm for the inverse problem of the PDEs controlled by linear operators based on Gaussian process regression. Different from classical methods, such as the Tikhonov regularization method [25], Gaussian processes solve inverse problems of the PDEs from the perspective of statistical inference.

In the Gaussian processes, the Bayesian method is introduced to encode the distribution information of the data into structured prior information [26], so as to construct an efficient data learning machine, estimate the unknown parameters of the PDEs, infer the solution of the considered equations and quantify the uncertainty of the prediction solution. The research on Gaussian processes solving the forward and inverse problem of the PDEs is a new branch of probabilistic numerics in numerical analysis [27–34].

In this paper, the inverse problem is to estimate the unknown parameters of the considered equations from (noisy) observation data. We extend the Gaussian processes method to deal with the inverse problem of several types of PDEs, which include high-order partial differential equations, partial integro-differential equations, fractional partial differential equations and the system of partial differential equations.

Finally, several numerical tests are provided. The results of the numerical experiments prove that the data-driven methods based on Gaussian processes not only estimate the parameters of the considered PDEs with high accuracy but also approximate the latent solutions and the inhomogeneous terms of the PDEs simultaneously.

The sections of this paper are organized as follows. In Section 2, the basic workflow of Gaussian processes solving the inverse problems of the PDEs is provided. Section 3 describes the estimation algorithm of the fractional PDEs and the linear PDEs system based on Gaussian processes in detail. In Section 4, numerical experiments are performed to prove the validity of the proposed methodology. Finally, our conclusions are given in Section 5.

2. Mathematical Model and Methodology

The following partial differential equations are considered in this paper,

$$\mathcal{L}_x^{\phi} u(x) = f(x), \tag{1}$$

where x represents a vector of D dimensions, $u(x)$ is the latent solution of (1), $f(x)$ is the inhomogeneous term, $\mathcal{L}_x^{\phi}$ is a linear operator, and ϕ represents the unknown parameter. For the sake of simplicity, we can introduce the following heat equation as an example,

$$\mathcal{L}_{(t,x)}^{\phi} u(t,x) := \frac{\partial}{\partial t} u(t,x) - \alpha \frac{\partial^2}{\partial x^2} u(t,x) = f(t,x), \tag{2}$$

where heat diffusivity α is the unknown parameter ϕ.

Assume that we obtain observations $\{x_u, y_u\}$ and $\left\{x_f, y_f\right\}$ from the latent solution $u(x)$ and the inhomogeneous term $f(x)$, respectively, then we can estimate the unknown parameter ϕ and approximate the latent solutions of the forward differential equations system according to the posterior results. However, one of the advantages of the methods used in this paper is that we do not need to consider the initial and boundary conditions of problems, because (noisy) observation data from the latent solution and the inhomogeneous term can give enough distribution information of the functions.

As with other machine-learning methods, Gaussian processes can be applied to solve regression problem and classification problem. Moreover Gaussian processes can be seen as a class of methods called kernel machines [35]. However, compared with other kernel machine methods, such as support vector machines (SVMs) and relevance vector machines (RVMs) [36], the strict probability theory of Gaussian processes limits the popularization of this method in industrial circles. Another drawback of Gaussian processes is that the computational cost may be expensive.

2.1. Gaussian Process Prior

Take a Gaussian process prior hypothesis as follows,

$$u(x) \sim \mathcal{GP}(0, k_{uu}(x, x'; \theta)), \tag{3}$$

and assume that covariance function $k_{uu}(x, x'; \theta)$ has the following squared exponential form,

$$k_{uu}(x, x'; \theta) = \sigma_u^2 \exp(-\frac{1}{2} \sum_{d=1}^{D} w_d (x_d - {x'}_d)^2), \tag{4}$$

where θ is the hyper-parameters of the kernel (covariance function) k_{uu}, for Equation (4), $\theta = (\sigma_u^2, (w_d)_{d=1}^D)$. Any prior information of $u(x)$, such as monotonicity and periodicity, can be encoded into the kernel k_{uu}.

Since the linear transformation of Gaussian processes, such as differentiation and integration, is still Gaussian, we can obtain

$$\mathcal{L}_x^{\phi} u(x) = f(x) \sim \mathcal{GP}(0, k_{ff}(x, x'; \theta, \phi)), \tag{5}$$

where the covariance function is $k_{ff}(x, x'; \theta, \phi) = \mathcal{L}_x^{\phi} \mathcal{L}_{x'}^{\phi} k_{uu}(x, x'; \theta)$.

Moreover, the covariance function between $u(x)$ and $f(x')$ is $k_{uf}(x, x'; \theta, \phi) = \mathcal{L}_{x'}^{\phi} k_{uu}(x, x'; \theta)$ and the covariance function between $f(x)$ and $u(x')$ is $k_{fu}(x, x'; \theta, \phi) = \mathcal{L}_x^{\phi} k_{uu}(x, x'; \theta)$. The proposal and reasoning of Equation (5) are the core of Gaussian processes to estimate the unknown parameters of the PDEs [24]. In this step, the parameter information of the PDEs is encoded into the kernels k_{ff}, k_{uf} and k_{fu}. Furthermore, we can utilize the joint density function of $u(\boldsymbol{x})$ and $f(\boldsymbol{x})$ for maximum likelihood estimation of parameters ϕ. The greatest contribution of Gaussian process regression is that we transform the unknown parameters ϕ of the linear operator $\mathcal{L}_x^{\phi}$ into the hyper-parameters of the kernels k_{ff}, k_{uf} and k_{fu}.

By Mercer's theorem [37], a positive definite covariance function $k(x, x')$ can be decomposed into $k(x, x') = \sum_{i=1}^{\infty} \lambda_i \varphi_i(x) \varphi_i(x')$, where λ_i and φ_i are eigenvalues and eigenfunctions, respectively, satisfying $\int k(x, x') \varphi_i(x) dx = \lambda_i \varphi_i(x')$. $\{\sqrt{\lambda_i} \varphi_i\}_{i=1}^{\infty}$ is treated as a set of orthogonal basis and a reproducing kernel Hilbert space (RKHS) is constructed [38], which can be seen as a kernel trick [38]. However, the covariance function in Equation (4) has no finite decomposition. Different covariance functions, such as rational quadratic covariance functions and Matérn covariance functions should be selected under different prior information [23]. The most important thing is that the kernel considered should cover the prior information [24].

2.2. Data Training

From the properties of the Gaussian processes [23], we find

$$\boldsymbol{y} \sim \mathcal{GP}(\boldsymbol{0}, \boldsymbol{K}), \tag{6}$$

where $\boldsymbol{y} = \begin{bmatrix} \boldsymbol{y}_u \\ \boldsymbol{y}_f \end{bmatrix}$, $\boldsymbol{K} = \begin{bmatrix} k_{uu}(\boldsymbol{x}_u, \boldsymbol{x}_u; \theta) + \sigma_u^2 I & k_{uf}(\boldsymbol{x}_u, \boldsymbol{x}_f; \theta, \phi) \\ k_{fu}(\boldsymbol{x}_f, \boldsymbol{x}_u; \theta, \phi) & k_{ff}(\boldsymbol{x}_f, \boldsymbol{x}_f; \theta, \phi) + \sigma_f^2 I \end{bmatrix}$.

According to (6), we train the parameters ϕ and the hyper-parameters θ by minimizing the following negative log marginal likelihood,

$$-\log p(\boldsymbol{y} | \boldsymbol{x}_u, \boldsymbol{x}_f, \phi, \theta, \sigma_{n_u}^2, \sigma_{n_f}^2) = \frac{1}{2} \log |\boldsymbol{K}| + \frac{1}{2} \boldsymbol{y}^T \boldsymbol{K}^{-1} \boldsymbol{y} + \frac{N}{2} \log 2\pi, \tag{7}$$

where N is the length of $\boldsymbol{y}$, and we consider a Quasi–Newton optimizer $L - BFGS - B$ for training [39].

Add noise on the observed data of formula (6), we introduce notations that $\boldsymbol{y}_u = u(\boldsymbol{x}_u) + \boldsymbol{\varepsilon}_u$ and $\boldsymbol{y}_f = u(\boldsymbol{x}_f) + \boldsymbol{\varepsilon}_f$, where $\boldsymbol{\varepsilon}_u \sim N(\boldsymbol{0}, \sigma_u^2 I)$ and $\boldsymbol{\varepsilon}_f \sim N(\boldsymbol{0}, \sigma_f^2 I)$. Assume that $\boldsymbol{\varepsilon}_u$ and $\boldsymbol{\varepsilon}_f$ are mutually independent additive noise.

The training procedure is the core of the algorithm, which reveals the "regression nature" of Gaussian processes. Furthermore, it is worth mentioning that the negative log marginal likelihood (7) is not only suitable for training the model, it automatically trades off between data-fit and model complexity. While minimizing the term $\frac{1}{2} y^T K^{-1} y$ in Equation (7), we use the term $\log |K|$ to penalize the model complexity [23]. This regularization-like mechanism is a key property of Gaussian process regression, which effectively prevents overfitting.

In truth, model training is solving parameters ϕ and hyper-parameters $\{\theta, \sigma_u^2, \sigma_f^2\}$ by minimizing Equation (7), and this defines a non-convex optimization problem, which is friendly to machine learning. Heuristic algorithms can be introduced to solve it, such as the whale optimization algorithm [40] and the ant colony optimization algorithm [41]. It is also worth noting that the computational cost of training has a cubic relationship with the amount of the training data due to the Cholesky decomposition of the covariance functions in (7), and the papers [42–44] considered this conundrum.

2.3. Gaussian Process Posterior

By the conditional distribution of Gaussian processes [23], the posterior distribution of the prediction $u(x^*)$ at point x^* can be directly written as

$$u(x^*)\left|\begin{bmatrix} \boldsymbol{y}_u \\ \boldsymbol{y}_f \end{bmatrix}\right. \sim \mathcal{GP}(\boldsymbol{q}_u^T \boldsymbol{K}^{-1} \boldsymbol{y}, k_{uu}(x^*, x^*) - \boldsymbol{q}_u^T \boldsymbol{K}^{-1} \boldsymbol{q}_u), \tag{8}$$

where $\boldsymbol{q}_u^T = \left[k_{uu}(x^*, \boldsymbol{x}_u),\ k_{uf}(x^*, \boldsymbol{x}_f)\right]$.

By the deduction of [11], the posterior distribution of the prediction $f(x^*)$ at the point x^* can be written as

$$f(x^*)\left|\begin{bmatrix} \boldsymbol{y}_u \\ \boldsymbol{y}_f \end{bmatrix}\right. \sim \mathcal{GP}(\boldsymbol{q}_f^T \boldsymbol{K}^{-1} \boldsymbol{y}, k_{ff}(x^*, x^*) - \boldsymbol{q}_f^T \boldsymbol{K}^{-1} \boldsymbol{q}_f), \tag{9}$$

where $\boldsymbol{q}_f^T = \left[k_{fu}(x^*, \boldsymbol{x}_u),\ k_{ff}(x^*, \boldsymbol{x}_f)\right]$.

The posterior mean of (8) and (9) can be seen as the predicted solution of $u(x)$ and $f(x)$, respectively. Furthermore, the posterior variance is the direct result of the Bayesian method, which can be used to measure the reliability of the prediction solution.

3. Inverse Problem for Fractional PDEs and the System of Linear PDEs

This section provides the process to estimate the unknown parameters of fractional PDEs and the system of linear PDEs in detail when Gaussian processes are used. However, the processing of these two types of PDEs is more complicated than the other two types of PDEs considered in this paper.

3.1. Processing of Fractional PDEs

The following fractional partial differential equations are considered,

$$\mathcal{L}^{\phi}_{(t,x)} u(t,x) := \frac{\partial^{\alpha}}{\partial t^{\alpha}} u(t,x) - Q \frac{\partial^2}{\partial x^2} u(t,x) = f(t,x), \tag{10}$$

where $\phi = Q$ and the fractional order α is a given parameter, $u(t,x)$ is a continuous real function absolutely integrable in $\mathbb{R}^2$, and its arbitrary-order partial derivatives are also continuous function absolutely integrable in $\mathbb{R}^2$. The fractional partial derivative in (10) is defined in the Caputo sense [45] as the following

$$\frac{\partial^{\alpha}}{\partial t^{\alpha}} u(t,x) = \frac{1}{\Gamma(n-\alpha)} \int_{-\infty}^{t} \frac{1}{(t-\tau)^{\alpha-n+1}} \frac{\partial^n u(\tau,x)}{\partial \tau^n} d\tau, \tag{11}$$

where α is a positive number ($n-1 < \alpha \leq n, n \in N_+$), and $\Gamma()$ is the gamma function.

The key step of solving (10) is the deriving of kernels with the fractional order operators. By [46,47], the four-dimensional Fourier transforms are applied with respect to the temproral variable (t, t') and with respect to the spatial variable (x, x') on $k_{uu}[(t, x), (t', x')]$ to obtain $\hat{k}_{uu}[(v, \omega), (v', \omega')]$, as the excellent properties of the squared exponential covariance function $k_{uu}[(t, x), (t', x')]$ assures the feasibility of the Fourier transform of derivatives of k_{uu}, on the basis of that k_{uu} and its partial derivatives vanish for $\sqrt{t^2 + x^2 + t'^2 + x'^2} \to +\infty$. We find the following intermediate kernels,

$$\begin{aligned}
&\hat{k}_{ff}[(v, \omega), (v', \omega')] = [(-iv)^{\alpha}(-iv')^{\alpha} - Q(-iv)^{\alpha}(-i\omega')^2 - \\
&Q(-i\omega)^2(-iv')^{\alpha} + Q^2(-i\omega)^2(-i\omega')^2]\hat{k}_{uu}[(v, \omega), (v', \omega')], \\
&\hat{k}_{uf}[(v, \omega), (v', \omega')] = [(-iv')^{\alpha} - Q(-i\omega')^2]\hat{k}_{uu}[(v, \omega), (v', \omega')], \\
&\hat{k}_{fu}[(v, \omega), (v', \omega')] = [(-iv)^{\alpha} - Q(-i\omega)^2]\hat{k}_{uu}[(v, \omega), (v', \omega')].
\end{aligned} \tag{12}$$

Then, we perform the inverse Fourier transform on $\hat{k}_{ff}[(v, \omega), (v', \omega')]$, $\hat{k}_{uf}[(v, \omega), (v', \omega')]$ and $\hat{k}_{fu}[(v, \omega), (v', \omega')]$ to obtain kernels $k_{ff}[(t, x), (t', x')]$, $k_{uf}[(t, x), (t', x')]$ and $k_{fu}[(t, x), (t', x')]$, respectively. Furthermore, the other steps are exactly the same as described in Section 2.

3.2. Processing of the System of Linear PDEs

Consider the systems of linear partial differential equations as follows,

$$\begin{cases} \mathcal{L}_x^{\phi,1} u_1(x) + \mathcal{L}_x^{\phi,2} u_2(x) = f_1(x), \\ \mathcal{L}_x^{\phi,3} u_1(x) + \mathcal{L}_x^{\phi,4} u_2(x) = f_2(x), \end{cases} \tag{13}$$

where x represents a vector of D dimensions, $\mathcal{L}_x^{\phi,1}$, $\mathcal{L}_x^{\phi,2}$, $\mathcal{L}_x^{\phi,3}$ and $\mathcal{L}_x^{\phi,4}$ are linear operators, and ϕ denotes the unkonwn parameters of (13).

Assume the following prior hypotheses

$$\begin{aligned}
u_1(x) &\sim \mathcal{GP}(0, k_{uu}^{11}(x, x'; \theta_1)), \\
u_2(x) &\sim \mathcal{GP}(0, k_{uu}^{22}(x, x'; \theta_2)),
\end{aligned} \tag{14}$$

to be two mutually independent Gaussian processes, and the covariance functions have a squared exponential form (4). Adding noise to the observed data, we introduce notations that $\boldsymbol{y}_{u_1} = u_1(\boldsymbol{x}_{u_1}) + \boldsymbol{\varepsilon}_{u_1}$, $\boldsymbol{y}_{u_2} = u_2(\boldsymbol{x}_{u_2}) + \boldsymbol{\varepsilon}_{u_2}$, $\boldsymbol{y}_{f_1} = f_1(\boldsymbol{x}_{f_1}) + \boldsymbol{\varepsilon}_{f_1}$ and $\boldsymbol{y}_{f_2} = f_2(\boldsymbol{x}_{f_2}) + \boldsymbol{\varepsilon}_{f_2}$, where $\boldsymbol{\varepsilon}_{u_1} \sim N(\boldsymbol{0}, \sigma_{u_1}^2 I)$, $\boldsymbol{\varepsilon}_{u_2} \sim N(\boldsymbol{0}, \sigma_{u_2}^2 I)$, $\boldsymbol{\varepsilon}_{f_1} \sim N(\boldsymbol{0}, \sigma_{f_1}^2 I)$ and $\boldsymbol{\varepsilon}_{f_2} \sim N(\boldsymbol{0}, \sigma_{f_2}^2 I)$.

According to the prior hypotheses, we find

$$\boldsymbol{y} \sim \mathcal{GP}(\boldsymbol{0}, \boldsymbol{K}), \tag{15}$$

where $\boldsymbol{y} = \begin{bmatrix} \boldsymbol{y}_{u_1} \\ \boldsymbol{y}_{u_2} \\ \boldsymbol{y}_{f_1} \\ \boldsymbol{y}_{f_2} \end{bmatrix}$, $\boldsymbol{K} = \begin{bmatrix} k_{uu}^{11} + \sigma_{n_{u_1}}^2 I_{n_{u_1}} & k_{uu}^{12} & k_{uf}^{11} & k_{uf}^{12} \\ k_{uu}^{21} & k_{uu}^{22} + \sigma_{n_{u_2}}^2 I_{n_{u_2}} & k_{uf}^{21} & k_{uf}^{22} \\ k_{fu}^{11} & k_{fu}^{12} & k_{ff}^{11} + \sigma_{n_{f_1}}^2 I_{n_{f_1}} & k_{ff}^{12} \\ k_{fu}^{21} & k_{fu}^{22} & k_{ff}^{21} & k_{ff}^{22} + \sigma_{n_{f_2}}^2 I_{n_{f_2}} \end{bmatrix}$,
$k_{uu}^{12} = 0$, $k_{uf}^{11} = \mathcal{L}_{x'}^{\phi,1} k_{uu}^{11}$, $k_{uf}^{12} = \mathcal{L}_{x'}^{\phi,3} k_{uu}^{11}$, $k_{uf}^{21} = \mathcal{L}_{x'}^{\phi,2} k_{uu}^{22}$, $k_{uf}^{22} = \mathcal{L}_{x'}^{\phi,4} k_{uu}^{22}, k_{uu}^{21} = 0$, $k_{fu}^{11} = \mathcal{L}_{x}^{\phi,1} k_{uu}^{11}$, $k_{fu}^{12} = \mathcal{L}_{x}^{\phi,2} k_{uu}^{22}$, $k_{fu}^{21} = \mathcal{L}_{x}^{\phi,3} k_{uu}^{11}$, $k_{fu}^{22} = \mathcal{L}_{x}^{\phi,4} k_{uu}^{22}, k_{ff}^{11} = \mathcal{L}_{x}^{\phi,1} \mathcal{L}_{x'}^{\phi,1} k_{uu}^{11} + \mathcal{L}_{x}^{\phi,2} \mathcal{L}_{x'}^{\phi,2} k_{uu}^{22}$, $k_{ff}^{12} = \mathcal{L}_{x}^{\phi,1} \mathcal{L}_{x'}^{\phi,3} k_{uu}^{11} + \mathcal{L}_{x}^{\phi,2} \mathcal{L}_{x'}^{\phi,4} k_{uu}^{22}, k_{ff}^{21} = \mathcal{L}_{x}^{\phi,3} \mathcal{L}_{x'}^{\phi,1} k_{uu}^{11} + \mathcal{L}_{x}^{\phi,4} \mathcal{L}_{x'}^{\phi,2} k_{uu}^{22}$, $k_{ff}^{22} = \mathcal{L}_{x}^{\phi,3} \mathcal{L}_{x'}^{\phi,3} k_{uu}^{11} + \mathcal{L}_{x}^{\phi,4} \mathcal{L}_{x'}^{\phi,4} k_{uu}^{22}$.

According to (15), parameters ϕ and hyper-parameters $\{\theta_1, \theta_2\}$ can be trained by minimizing the following negative log marginal likelihood

$$-\log p(\boldsymbol{y}|\boldsymbol{x}_{u_1}, \boldsymbol{x}_{u_2}, \boldsymbol{x}_{f_1}, \boldsymbol{x}_{f_2}, \phi, \theta_1, \theta_2, \sigma^2_{n_{u_1}}, \sigma^2_{n_{u_2}}, \sigma^2_{n_{f_1}}, \sigma^2_{n_{f_2}}) = \frac{1}{2}\log|\boldsymbol{K}| + \frac{1}{2}\boldsymbol{y}^T\boldsymbol{K}^{-1}\boldsymbol{y} + \frac{N}{2}\log 2\pi,$$

where N is the length of $\boldsymbol{y}$.

After the training step, we can write the posterior distribution of u_1, u_2, f_1 and f_2 as follows,

$$\begin{aligned}
u_1(x^*)|\boldsymbol{y} &\sim \mathcal{GP}(\boldsymbol{q}_{u_1}^T\boldsymbol{K}^{-1}\boldsymbol{y}, k_{uu}^{11}(x^*, x^*) - \boldsymbol{q}_{u_1}^T\boldsymbol{K}^{-1}\boldsymbol{q}_{u_1}),\\
u_2(x^*)|\boldsymbol{y} &\sim \mathcal{GP}(\boldsymbol{q}_{u_2}^T\boldsymbol{K}^{-1}\boldsymbol{y}, k_{uu}^{22}(x^*, x^*) - \boldsymbol{q}_{u_2}^T\boldsymbol{K}^{-1}\boldsymbol{q}_{u_2}),\\
f_1(x^*)|\boldsymbol{y} &\sim \mathcal{GP}(\boldsymbol{q}_{f_1}^T\boldsymbol{K}^{-1}\boldsymbol{y}, k_{ff}^{11}(x^*, x^*) - \boldsymbol{q}_{f_1}^T\boldsymbol{K}^{-1}\boldsymbol{q}_{f_1}),\\
f_2(x^*)|\boldsymbol{y} &\sim \mathcal{GP}(\boldsymbol{q}_{f_2}^T\boldsymbol{K}^{-1}\boldsymbol{y}, k_{ff}^{22}(x^*, x^*) - \boldsymbol{q}_{f_2}^T\boldsymbol{K}^{-1}\boldsymbol{q}_{f_2}),
\end{aligned} \tag{16}$$

where

$$\begin{aligned}
\boldsymbol{q}_{u_1}^T &= \left[k_{uu}^{11}(x^*, \boldsymbol{x}_{u_1}), k_{uu}^{12}(x^*, \boldsymbol{x}_{u_2}), k_{uf}^{11}(x^*, \boldsymbol{x}_{f_1}), k_{uf}^{12}(x^*, \boldsymbol{x}_{f_2})\right],\\
\boldsymbol{q}_{u_2}^T &= \left[k_{uu}^{21}(x^*, \boldsymbol{x}_{u_1}), k_{uu}^{22}(x^*, \boldsymbol{x}_{u_2}), k_{uf}^{21}(x^*, \boldsymbol{x}_{f_1}), k_{uf}^{22}(x^*, \boldsymbol{x}_{f_2})\right],\\
\boldsymbol{q}_{f_1}^T &= \left[k_{fu}^{11}(x^*, \boldsymbol{x}_{u_1}), k_{fu}^{12}(x^*, \boldsymbol{x}_{u_2}), k_{ff}^{11}(x^*, \boldsymbol{x}_{f_1}), k_{ff}^{12}(x^*, \boldsymbol{x}_{f_2})\right],\\
\boldsymbol{q}_{f_2}^T &= \left[k_{fu}^{21}(x^*, \boldsymbol{x}_{u_1}), k_{fu}^{22}(x^*, \boldsymbol{x}_{u_2}), k_{ff}^{21}(x^*, \boldsymbol{x}_{f_1}), k_{ff}^{22}(x^*, \boldsymbol{x}_{f_2})\right].
\end{aligned} \tag{17}$$

4. Numerical Tests

This section provides four examples to prove the validity of the methodology proposed. We consider the inverse problem of four types of PDEs, which include high-order partial differential equations, partial integro-differential equations, fractional partial differential equations and the system of partial differential equations.

Introduce the relative error $\mathcal{L}^2$ between the exact solution and the prediction solution to represent the prediction error of the algorithm,

$$\mathcal{L}^2 = \sqrt{\sum_{x^*}[u(x^*) - \bar{u}(x^*)]^2} \Big/ \sqrt{\sum_{x^*}[u(x^*)]^2}, \tag{18}$$

where x^* represents the predicted point, $u(x^*)$ is the exact solution at this point, and $\bar{u}(x^*)$ is the corresponding prediction solution.

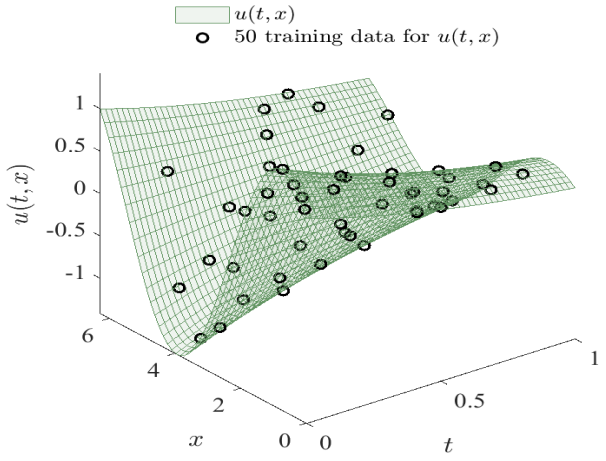
4.1. Simulation for a High-Order Partial Differential Equation

Example 1.

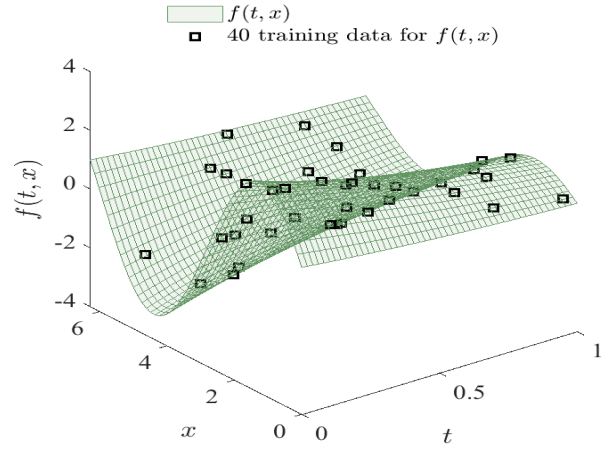
$$\mathcal{L}^{\phi}_{(t,x)}u(t,x) := u_{tttt} - \alpha u_{ttxx} + \beta u_{xxxxxx} = f, \tag{19}$$

where $(t, x) \in [0, 1] \times [0, 2\pi]$, $\phi = (\alpha, \beta)$. *The exact solution is* $u(t, x) = e^{-t}[\cos(x) + \sin(x)]$ *and the inhomogeneous term is* $f(t, x) = (-1 + \alpha - \beta)e^{-t}[\cos(x) + \sin(x)]$.

Numerical experiments are performed with the noiseless data of Example 1. We denote that the amount of the training data of $u(t, x)$ is N_u, and the number of the training data of $f(t, x)$ is N_f. Take $N_u = 50$ and $N_f = 40$ in this subsection. Fix (α, β) to be (1,1), and the estimated value $(\hat{\alpha}, \hat{\beta})$ is (1.014005, 1.014634). Figure 1 shows the distribution of the training data of $u(t, x)$ and $f(t, x)$. Figure 2 shows the estimation error $|\bar{u}(t, x) - u(t, x)|$ and $|\bar{f}(t, x) - f(t, x)|$ of $u(t, x)$ and $f(t, x)$, respectively. Figure 3 shows the posterior standard deviation of the corresponding prediction solution. It can be seen from Figures 2 and 3 that the posterior standard deviation is positively correlated with the prediction error. The posterior distribution of Gaussian processes can return a satisfactory numerical approximation to $u(t, x)$ and $f(t, x)$.



(**a**) Training data for $u(t,x)$



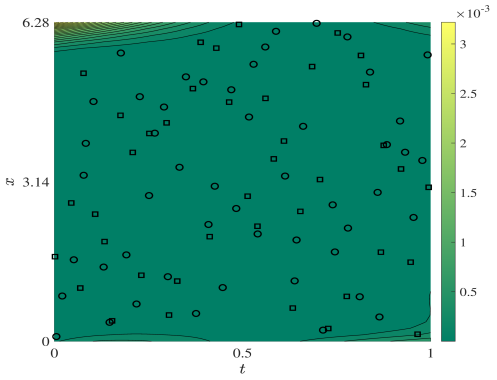
(**b**) Training data for $f(t,x)$

Figure 1. High-order partial differential equation: training data for $u(t,x)$ and $f(t,x)$.

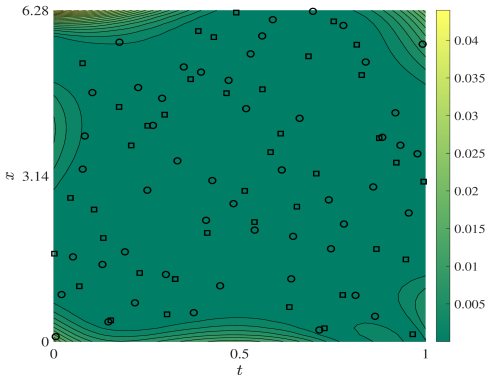
Table 1 shows the estimation values of (α,β) and the relative error $\mathcal{L}^2$ for $u(t,x)$ and $f(t,x)$, where (α,β) is fixed to be (1, 1), (1, 2), (2, 1) and (2, 2), in turn. Furthermore, the experimental results prove that the results of parameter estimation based on Gaussian processes have relatively high accuracy when high-order PDEs are considered.

Table 1. High-order partial differential equation: estimated values of (α,β) and the relative error $\mathcal{L}^2$ for $u(t,x)$ and $f(t,x)$, where $N_u = 50$, $N_f = 40$, and (α,β) is fixed to be (1, 1), (1, 2), (2, 1) and (2, 2) in turn.

$\hat{\alpha}$	1.014005	1.021724	2.006856	2.008363
$\hat{\beta}$	1.014634	2.032989	1.006408	2.020304
$\mathcal{L}^2$ for u	3.201×10^{-4}	5.615×10^{-4}	1.869×10^{-4}	3.319×10^{-4}
$\mathcal{L}^2$ for f	2.665×10^{-3}	1.825×10^{-3}	3.037×10^{-3}	2.823×10^{-3}

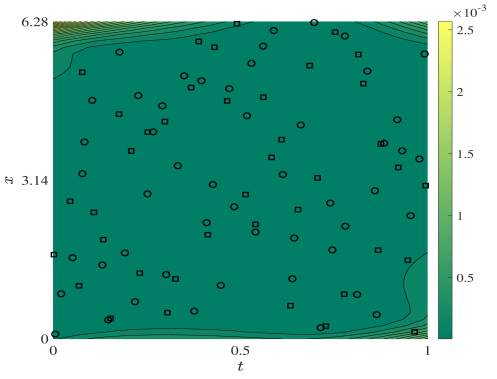


(**a**) Absolute error $|\bar{u}(t,x) - u(t,x)|$

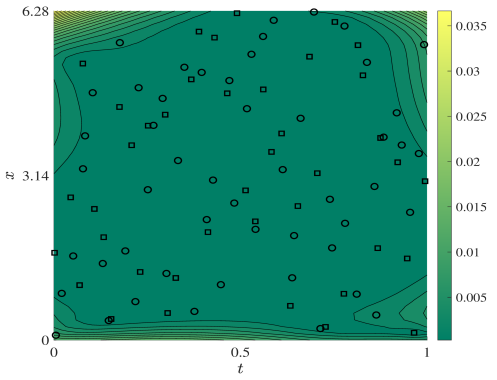


(**b**) Absolute error $|\bar{f}(t,x) - f(t,x)|$

Figure 2. High-order partial differential equation: prediction error for $u(t,x)$ and $f(t,x)$.



(**a**) Standard deviation for $u(t,x)$



(**b**) Standard deviation for $f(t,x)$

Figure 3. High-order partial differential equation: Standard deviation for $u(t,x)$ and $f(t,x)$.

4.2. Simulation for a Fractional Partial Differential Equation

Example 2.

$$\mathcal{L}^{\phi}_{(t,x)}u(t,x) := \frac{\partial^{\alpha}}{\partial t^{\alpha}}u(t,x) - Q\frac{\partial^{2}}{\partial x^{2}}u(t,x) = f(t,x), \tag{20}$$

where $(t,x) \in (-\infty,1] \times [0,1]$ $\phi = Q$ *and the fractional order* α *is a given parameter. Assume* $0 < \alpha \leq 1$*, and the fractional partial derivative in (20) is defined in the Caputo sense as the following,*

$$\frac{\partial^{\alpha}}{\partial t^{\alpha}}u(t,x) = \frac{1}{\Gamma(1-\alpha)}\int_{-\infty}^{t}\frac{1}{(t-\tau)^{\alpha}}\frac{\partial u(\tau,x)}{\partial \tau}d\tau, \tag{21}$$

The exact solution is $u(t,x) = e^{t}(1-x)^{2}x^{2}$ *and the inhomogeneous term is* $f(t,x) = \frac{(1-x)^{2}x^{2}}{\Gamma(1-\alpha)}\int_{-\infty}^{t}\frac{e^{\tau}}{(t-\tau)^{\alpha}}d\tau - 2Qe^{t}[1+6(-1+x)x]$.

Experiments are performed with noiseless data of Example 2. Table 2 shows the estimation values of Q and the relative error $\mathcal{L}^{2}$ for $u(t,x)$ and $f(t,x)$, where (α, Q) is fixed to be (0.25, 1), (0.5, 1), (0.75, 1), (0.25, 2), (0.5,2) and (0.75,2), in turn. The experimental results show that the results of parameter estimation based on Gaussian processes have relatively high accuracy when fractional partial differential equations are considered. Furthermore, the posterior distribution of Gaussian processes can return a satisfactory numerical approximation to $u(t,x)$ and $f(t,x)$.

Table 2. Fractional partial differential equation: estimated values of Q and the relative error $\mathcal{L}^{2}$ for $u(t,x)$ and $f(t,x)$ where $N_u = 15$, $N_f = 12$, (α, Q) is fixed to be (0.25, 1), (0.5, 1), (0.75, 1), (0.25, 2), (0.5, 2) and (0.75, 2), in turn.

(α, Q)	$(0.25, 1)$	$(0.5, 1)$	$(0.75, 1)$	$(0.25, 2)$	$(0.5, 2)$	$(0.75, 2)$
$\hat{Q}$	1.000262	1.000115	0.999436	2.000810	1.999836	1.995571
$\mathcal{L}^{2}$ for u	1.147×10^{-2}	1.075×10^{-2}	8.409×10^{-3}	8.935×10^{-3}	8.555×10^{-3}	7.244×10^{-3}
$\mathcal{L}^{2}$ for f	1.562×10^{-1}	1.552×10^{-1}	1.488×10^{-1}	1.462×10^{-1}	1.474×10^{-1}	1.524×10^{-1}

4.3. Simulation for a Partial Integro-Differential Equation

Example 3.

$$\mathcal{L}^{\phi}_{(t,x)}u(t,x) := \frac{\partial}{\partial t}u(t,x) - \alpha\Delta u(t,x) - \beta\int_{0}^{t}\Delta u(\tau,x)d\tau = f, \tag{22}$$

where $(t,x) \in [0,1] \times [0,1]$*,* $\phi = (\alpha, \beta)$*, and* $\Delta u(t,x) = \frac{\partial^{2}u}{\partial t^{2}} + \frac{\partial^{2}u}{\partial x^{2}}$*. The exact solution is* $u(t,x) = (1+t^{3})(2-x^{2})$*, and the inhomogeneous term is* $f(t,x) = \frac{1}{2}t[12t + \beta(4 - 12t + t^{3}) + 6(-1+\beta)tx^{2}] + 2\alpha[1 + t^{3} + 3t(-2+x^{2})]$.

Experiments are performed with the noiseless data of Example 3. Table 3 shows the estimation values of (α, β) and the relative error $\mathcal{L}^{2}$ for $u(t,x)$ and $f(t,x)$, where (α, β) is fixed to be (1, 1), (1, 2), (2, 1) and (2, 2), in turn. The experimental results show that the results of parameter the estimation based on Gaussian processes have relatively high accuracy when partial integro-differential equations are considered. Furthermore, the posterior distribution of Gaussian processes can return a satisfying numerical approximation to $u(t,x)$ and $f(t,x)$.

Table 3. Partial integro-differential equation: estimated values of (α, β) and the relative error $\mathcal{L}^2$ for $u(t,x)$ and $f(t,x)$, where $N_u = 20$, $N_f = 20$, (α, β) is fixed to be (1, 1), (1, 2), (2, 1) and (2, 2), in turn.

(α, β)	$(1,1)$	$(1,2)$	$(2,1)$	$(2,2)$
$\hat{\alpha}$	1.000103	0.999282	2.000769	2.000163
$\hat{\beta}$	0.999468	2.001618	0.996979	1.998910
$\mathcal{L}^2$ for u	3.077×10^{-5}	6.223×10^{-5}	4.726×10^{-5}	3.202×10^{-5}
$\mathcal{L}^2$ for f	6.331×10^{-3}	4.784×10^{-3}	5.294×10^{-3}	3.857×10^{-3}

Moreover, we investigate the impact of the amount of training data and the noise level on the accuracy of the estimation, where (α, β) is taken as (1, 1). Table 4 shows the estimation values of (α, β) and the relative error $\mathcal{L}^2$ for $u(t,x)$ and $f(t,x)$, where N_u and N_f are fixed to be 10, 20, 30, 40 and 50, in turn. The results show that the larger amount of training data, the higher the prediction accuracy in general. Table 5 shows the estimation values and the relative error $\mathcal{L}^2$ where $N_u = N_f = 20$, and the levels of addictive noise from the training data are taken at different values.

According to Table 5, we can conclude that the method can estimate parameters with relatively gratifying accuracy when noise-free data can not be obtained, although the estimation accuracy is sensitive to the noise of $u(t,x)$, which is likely due to the high complexity of the PDEs considered in this paper. Furthermore, we have to admit that this high sensitivity to the noise of the latent solution u greatly limits the scope of application of Gaussian processes. In summary, this problem deserves further research.

Table 4. Numerical results for Example 3. Impact of the amount of training data: estimated values of (α, β) and the relative error $\mathcal{L}^2$ for $u(t,x)$ and $f(t,x)$, where (α, β) is fixed to be (1,1) and noise-free data is used.

$N_u = N_f$	**10**	**20**	**30**	**40**	**50**
$\hat{\alpha}$	0.887239	1.000103	0.999763	1.000057	0.999051
$\hat{\beta}$	1.343989	0.999468	1.001156	0.999610	1.005044
$\mathcal{L}^2$ for u	1.979×10^{-2}	3.077×10^{-5}	6.761×10^{-6}	2.510×10^{-6}	2.752×10^{-5}
$\mathcal{L}^2$ for f	1.759×10^{-2}	6.331×10^{-3}	6.737×10^{-4}	4.779×10^{-4}	9.552×10^{-5}

Table 5. Numerical results for Example 3. Impact of the levels of addictive noise: estimated values of (α, β) and the relative error $\mathcal{L}^2$ for $u(t,x)$ and $f(t,x)$ where $N_u = N_f = 20$, (α, β) is fixed to be (1,1).

(σ_u^2, σ_f^2)	$(0,0)$	$(0, 0.5^2)$	$(0, 1.0^2)$	$(0.005^2, 0)$	$(0.005^2, 0.5^2)$	$(0.005^2, 1.0^2)$
$\hat{\alpha}$	1.000103	0.983124	0.977075	0.974645	0.941651	0.926553
$\hat{\beta}$	0.999468	1.125553	1.243469	1.450893	1.511768	1.582164
$\mathcal{L}^2$ for u	3.077×10^{-5}	1.467×10^{-3}	1.684×10^{-3}	4.594×10^{-3}	4.274×10^{-3}	5.241×10^{-3}
$\mathcal{L}^2$ for f	6.331×10^{-3}	1.134×10^{-1}	1.483×10^{-1}	3.814×10^{-3}	1.800×10^{-1}	2.734×10^{-1}

4.4. Simulation for a System of Partial Differential Equations

Example 4.

$$\begin{cases} \mathcal{L}^{\phi,1}_{(t,x,y)} u + \mathcal{L}^{\phi,2}_{(t,x,y)} v = u_t + a u_{xx} + b v_{xy} = f_1, \\ \mathcal{L}^{\phi,3}_{(t,x,y)} u + \mathcal{L}^{\phi,4}_{(t,x,y)} v = u_{tt} + c u_{yy} + v_t + d v_{xy} = f_2, \end{cases} \tag{23}$$

where $(t,x,y) \in [0,1]^3$, $\phi = (a,b,c,d)$. $u(t,x,y) = e^t x(x-1)y(y-1)$, $v(t,x,y) = e^t \sin(2\pi x)\cos(2\pi y)$, $f_1(t,x,y) = e^t[(x^2 - x + 2a)(y^2 - y) - 4b\pi^2 \cos(2\pi x)\sin(2\pi y)]$ *and* $f_2(t,x,y) = e^t[(x^2 - x)(y^2 - y + 2c) + \cos(2\pi y)\sin(2\pi x) - 4d\pi^2 \cos(2\pi x)\sin(2\pi y)]$ *satisfy Equation (23).*

Experiments are performed with noiseless data of Example 4. Denote that the amount of training data of $u(t,x,y)$ is N_u, the amount of training data $v(t,x,y)$ is N_v, the amount of training data $f_1(t,x,y)$ is N_{f_1}, and the amount of training data $f_2(t,x,y)$ is N_{f_2}. Take $N_u = N_v = 100$ and $N_{f_1} = N_{f_2} = 80$ in the experiments.

Table 6 shows the estimation values of (a, b, c, d) and the corresponding prediction errors for u, v, f_1 and f_2, where (a, b, c, d) is fixed to be (1, 1, 1, 1), (1, 1, 2, 2), (2, 2, 1, 1) and (2, 2, 2, 2), in turn. The experimental results show that the results of parameter estimation based on Gaussian processes have high accuracy, and the posterior distribution of Gaussian processes can return a satisfying numerical approximation to u, v, f_1 and f_2, when the system of linear PDEs is considered.

Table 6. A system of partial differential equations: estimated values of (a, b, c, d) and the relative error $\mathcal{L}^2$ for u, v, f_1 and f_2, where $N_{u_1} = N_{u_2} = 100$, $N_{f_1} = N_{f_2} = 80$, and (a, b, c, d) is fixed to be (1, 1, 1, 1), (1, 1, 2, 2), (2, 2, 1, 1) and (2, 2, 2, 2), in turn.

(a, b, c, d)	**(1, 1, 1, 1)**	**(1, 1, 2, 2)**	**(2, 2, 1, 1)**	**(2, 2, 2, 2)**
$\hat{a}$	1.000803	1.000828	2.001061	2.000685
$\hat{b}$	1.000120	1.000112	2.000256	2.000255
$\hat{c}$	0.999639	2.000954	0.998905	2.000404
$\hat{d}$	0.999927	1.999879	0.999935	1.999898
$\mathcal{L}^2$ for u	1.363×10^{-2}	1.409×10^{-2}	1.215×10^{-2}	1.230×10^{-2}
$\mathcal{L}^2$ for v	4.805×10^{-3}	4.601×10^{-3}	4.754×10^{-3}	4.395×10^{-3}
$\mathcal{L}^2$ for f_1	1.383×10^{-2}	1.368×10^{-2}	1.354×10^{-2}	1.335×10^{-2}
$\mathcal{L}^2$ for f_2	1.396×10^{-2}	1.374×10^{-2}	1.366×10^{-2}	1.339×10^{-2}

5. Conclusions

In this paper, we explored the possibility of using Gaussian processes in solving inverse problems of complex linear partial differential equations, which include high-order partial differential equations, partial integro-differential equations, fractional partial differential equations and a system of partial differential equations. The main points of the Gaussian processes method were to encode the distribution information of the data into kernels (covariance functions) of Gaussian process priors, transform unknown parameters of the linear operators into the hyper-parameters of the kernels, train the parameters and hyper-parameters through minimizing the negative log marginal likelihood and infer the solution of the considered equations.

Numerical experiments showed that the data-driven method based on Gaussian processes had high prediction accuracy when estimating the unknown parameters of the PDEs considered, which proved that Gaussian processes have impressive performance in dealing with linear problems. Furthermore, the posterior distribution of Gaussian processes can return a satisfactory numerical approximation to the latent solution and the inhomogeneous term of the PDEs. However, the estimation accuracy of unknown parameters was sensitive to the noise of the latent solution, which still deserves further research. In the future work, we may focus on how to exploit the Gaussian process to solve the inverse problem of nonlinear PDEs and on how to solve the problem of multi-parameter estimation for fractional partial differential equations.

Author Contributions: All the authors contributed equally to this paper. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by National Natural Science Foundation of China (No. 71974204).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Rudy, S.H.; Brunton, S.L.; Proctor, J.L.; Kutz, J.N. Data-driven discovery of partial differential equations. *Sci. Adv.* **2017**, *3*, e1602614. [CrossRef] [PubMed]
2. Ghahramani, Z. Probabilistic machine learning and artificial intelligence. *Nature* **2015**, *521*, 452–459. [CrossRef] [PubMed]
3. Chen, I.Y.; Joshi, S.; Ghassemi, M.; Ranganath, R. Probabilistic machine learning for healthcare. *Annu. Rev. Biomed. Data Sci.* **2021**, *4*, 393–415. [CrossRef]
4. Maslyaev, M.; Hvatov, A.; Kalyuzhnaya, A.V. Partial differential equations discovery with EPDE framework: application for real and synthetic data. *J. Comput. Sci.* **2021**, *53*, 101345. [CrossRef]
5. Lorin, E. From structured data to evolution linear partial differential equations. *J. Comput. Phys.*, **2019**, *393*, 162–185. [CrossRef]
6. Arbabi, H.; Bunder, J.E.; Samaey, G.; Roberts, A.J.; Kevrekidis, I.G. Linking machine learning with multiscale numerics: Data-driven discovery of homogenized equations. *JOM* **2020**, *72*, 4444–4457. [CrossRef]
7. Chang, H.; Zhang, D. Machine learning subsurface flow equations from data. *Comput. Geosci.* **2019**, *23*, 895–910. [CrossRef]
8. Martina-Perez, S.; Simpson, M.J.; Baker, R.E. Bayesian uncertainty quantification for data-driven equation learning. *Proc. R. Soc. A* **2021**, *477*, 20210426. [CrossRef]
9. Dal, Santo, N.; Deparis, S.; Pegolotti, L. Data driven approximation of parametrized PDEs by reduced basis and neural networks. *J. Comput. Phys.* **2020**, *416*, 109550. [CrossRef]
10. Raissi, M.; Perdikaris, P.; Karniadakis, G.E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* **2019**, *378*, 686–707. [CrossRef]
11. Kaipio, J.; Somersalo, E. *Statistical and Computational Inverse Problems*; Springer Science & Business Media: New York, NY, USA, 2006.
12. Kremsner, S.; Steinicke, A.; Szölgyenyi, M. A deep neural network algorithm for semilinear elliptic PDEs with applications in insurance mathematics. *Risks* **2020**, *8*, 136. [CrossRef]
13. Guo, Y.; Cao, X.; Liu, B.; Gao, M.. Soling partial differential equations using deep learning and physical constraints. *Appl. Sci.* **2020**, *10*, 5917. [CrossRef]
14. Chen, Z.; Liu, Y.; Sun, H. Physics-informed learning of governing equations from scarce data. *Nat. Commun.* **2021**, *12*, 6136. [CrossRef]
15. Gelbrecht, M.; Boers, N.; Kurths, J. Neural partial differential equations for chaotic systems. *New J. Phys.* **2021**, *23*, 43005. [CrossRef]
16. Cheung, K.C.; See, S. Recent advance in machine learning for partial differential equation. *CCF Trans. High Perform. Comput.* **2021**, *3*, 298–310. [CrossRef]
17. Omidi, M.; Arab, B.; Rasanan, A.H.; Rad, J.A.; Par, K. Learning nonlinear dynamics with behavior ordinary/partial/system of the differential equations: Looking through the lens of orthogonal neural networks. *Eng. Comput.* **2021**, *38*, 1635–1654. [CrossRef]
18. Lagergren, J.H.; Nardini, J.T.; Lavigne, G.M.; Rutter, E.M.; Flores, K.B. Learning partial differential equations for biological transport models from noisy spatio-temporal data. *Proc. R. Soc. A* **2020**, *476*, 20190800. [CrossRef]
19. Koyamada, K.; Long, Y.; Kawamura, T.; Konishi, K. Data-driven derivation of partial differential equations using neural network model. *Int. J. Model. Simul. Sci. Comput.* **2021**, *12*, 2140001. [CrossRef]
20. Kalogeris, I.; Papadopoulos, V. Diffusion maps-aided Neural Networks for the solution of parametrized PDEs. *Comput. Methods Appl. Mech. Eng.* **2021**, *376*, 113568. [CrossRef]
21. Rifkin, R.; Yeo, G.; Poggio, T. Regularized least-squares classification. *Nato Sci. Ser. Sub Ser. III Comput. Syst. Sci.* **2003**, *190*, 131–154.
22. Drucker, H.; Wu, D.; Vapnik, V.N. Support vector machines for spam categorization. *IEEE Trans. Neural Netw.* **2002**, *10*, 1048–1054. [CrossRef]
23. Williams, C.K.; Rasmussen, C.E. *Gaussian Processes for Machine Learning*; MIT Press: Cambridge, MA, USA, 2006.
24. Raissi, M.; Perdikaris, P.; Karniadakis, G.E. Machine learning of linear differential equations using Gaussian processes. *J. Comput. Phys.* **2017**, *348*, 683–693. [CrossRef]
25. Yang, S.; Xiong, X.; Nie, Y. Iterated fractional Tikhonov regularization method for solving the spherically symmetric backward time-fractional diffusion equation. *Appl. Numer. Math.* **2021**, *160*, 217–241. [CrossRef]
26. Bernardo, J.M.; Smith, A.F. *Bayesian Theory*; John Wiley & Sons: Hoboken, NJ, USA, 2009.
27. Oates, C.J.; Sullivan, T.J. A modern retrospective on probabilistic numerics. *Stat. Comput.* **2019**, *29*, 1335–1351. [CrossRef]
28. Hennig, P.; Osborne, M.A.; Girolami, M. Probabilistic numerics and uncertainty in computations. *Proc. R. Soc. A Math. Phys. Eng. Sci.* **2015**, *471*, 20150142. [CrossRef]
29. Conrad, P.R.; Girolami, M.; Särkkä, S.; Stuart, A.; Zygalakis, K. Statistical analysis of differential equations: introducing probability measures on numerical solutions. *Stat. Comput.* **2017**, *27*, 1065–1082. [CrossRef]
30. Hennig, P. Fast probabilistic optimization from noisy gradients. In Proceedings of the International Conference on Machine Learning PMLR, Atlanta, GA, USA, 17–19 June 2013; pp. 62–70.
31. Kersting, H.; Sullivan, T.J.; Hennig, P. Convergence rates of Gaussian ODE filters. *Stat. Comput.* **2020**, *30*, 1791–1816. [CrossRef]
32. Raissi, M.; Karniadakis, G.E. Hidden physics models: Machine learning of nonlinear partial differential equations. *J. Comput. Phys.* **2018**, *357*, 125–141. [CrossRef]
33. Raissi, M.; Perdikaris, P.; Karniadakis, G.E. Inferring solutions of differential equations using noisy multi-fidelity data. *J. Comput. Phys.* **2017**, *335*, 736–746. [CrossRef]
34. Raissi, M.; Perdikaris, P.; Karniadakis, G.E. Numerical Gaussian processes for time-dependent and nonlinear partial differential equations. *SIAM J. Sci. Comput.* **2018**, *40*, A172–A198. [CrossRef]

35. Scholkopf, B.; Smola, A.J.; Bach, F. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*; MIT Press: Cambridge, MA, USA, 2002.
36. Tipping, M.E. Sparse Bayesian learning and the relevance vector machine. *J. Mach. Learn. Res.* **2001**, *1*, 211–244.
37. Konig, H. *Eigenvalue Distribution of Compact Operators*; Birkhäuser: Basel, Switzerland, 2013.
38. Berlinet, A.; Thomas-Agnan, C. *Reproducing Kernel Hilbert Spaces in Probability and Statistics*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2011.
39. Zhu, C.; Byrd, R.H.; Lu, P.; Nocedal, J. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Trans. Math. Softw. (TOMS)* **1997**, *23*, 550–560. [CrossRef]
40. Mirjalili, S.; Lewis, A. The whale optimization algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [CrossRef]
41. Dorigo, M.; Stützle, T. Ant colony optimization: Overview and recent advances. In *Handbook of Metaheuristics*; Springer: Cham, Switzerland, 2019.
42. Raissi, M.; Babaee, H.; Karniadakis, G.E. Parametric Gaussian process regression for big data. *Comput. Mech.* **2019**, *64*, 409–416. [CrossRef]
43. Snelson, E.; Ghahramani, Z. Sparse Gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems 18 (NIPS 2005)*; MIT Press: Cambridge, MA, USA, 2005.
44. Liu, H.; Ong, Y.S.; Shen, X.; Cai, J. When Gaussian process meets big data: A review of scalable GPs. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *31*, 4405–4423. [CrossRef]
45. Milici, C.; Draganescu, G.; Machado, J.T. *Introduction to Fractional Differential Equations*; Springer: Cham, Switzerland, 2018.
46. Podlubny, I. *Fractional Differential Equations*; Academic Press: San Diego, CA, USA, 1998.
47. Povstenko, Y. *Linear Fractional Diffusion-Wave Equation for Scientists and Engineers*; Birkhäuser: New York, NY, USA, 2015.