Numerical Gaussian Processes

(Physics Informed Learning Machines)

Maziar Raissi

Division of Applied Mathematics,
Brown University, Providence, RI, USA
maziar_raissi@brown.edu

June 7, 2017

Probabilistic Numerics v.s. Numerical Gaussian Processes

Probabilistic numerics aim to capitalize on the recent developments in probabilistic machine learning to revisit classical methods in numerical analysis and mathematical physics from a statistical inference point of view.

Probabilistic Numerics v.s. Numerical Gaussian Processes

Probabilistic numerics aim to capitalize on the recent developments in probabilistic machine learning to revisit classical methods in numerical analysis and mathematical physics from a statistical inference point of view.

This is exciting. However, it would be even more exciting if we could do the exact opposite.

Probabilistic Numerics v.s. Numerical Gaussian Processes

Probabilistic numerics aim to capitalize on the recent developments in probabilistic machine learning to revisit classical methods in numerical analysis and mathematical physics from a statistical inference point of view.

This is exciting. However, it would be even more exciting if we could do the exact opposite.

Numerical Gaussian processes aim to capitalize on the long-standing developments of classical methods in numerical analysis and revisits machine leaning from a mathematical physics point of view.

Physics Informed Learning Machines

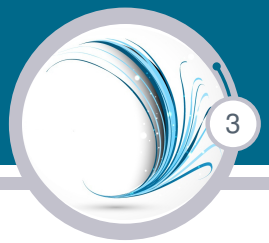
Numerical Gaussian processes enable the construction of data-efficient learning machines that can encode physical conservation laws as structured prior information.

Physics Informed Learning Machines

Numerical Gaussian processes enable the construction of data-efficient learning machines that can encode physical conservation laws as structured prior information.

Numerical Gaussian processes are essentially physics informed learning machines.

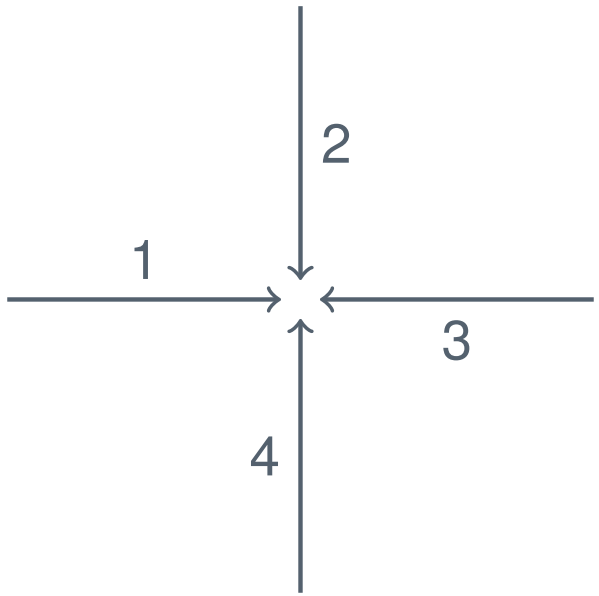
Content



Motivating Example

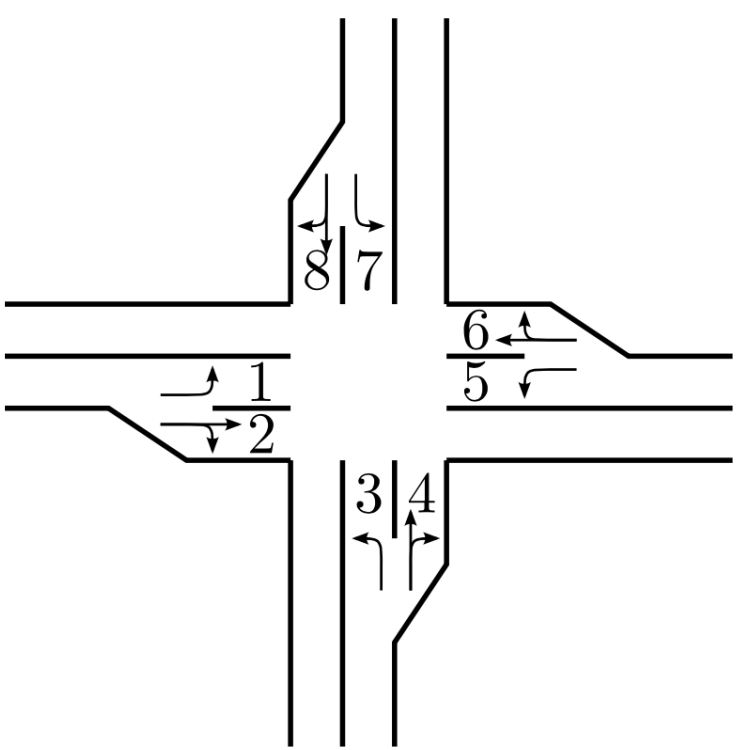
Road Networks

Consider a 2×2 junction as shown below.

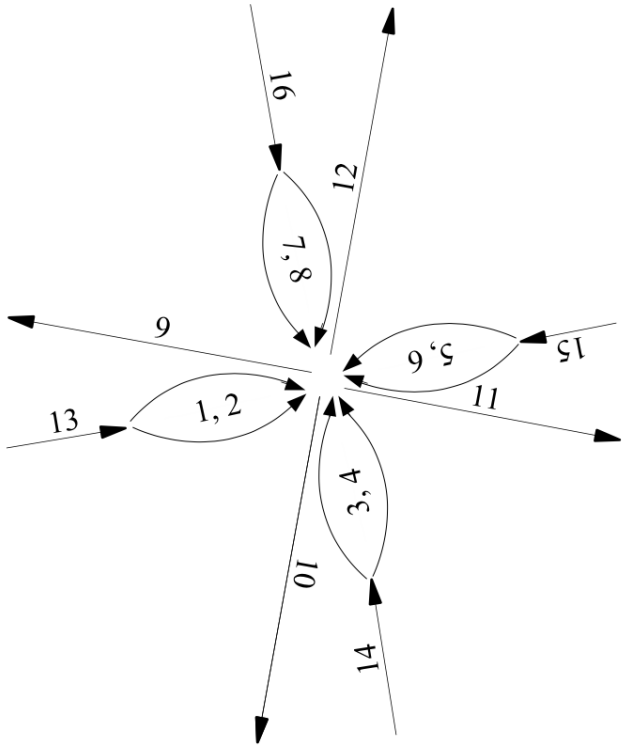


Roads have length $L_i,\ i = 1, 2, 3, 4.$

Road Networks



(a)



(b)

Hyperbolic Conservation Law

The road traffic densities $\rho_i(t, x) \in [0, 1]$ satisfy the one-dimensional hyperbolic conservation law

$$\partial_t \rho_i + \partial_x f(\rho_i) = 0, \quad \text{on } [0, T] \times [0, L_i].$$

Here, $f(\rho) = \rho(1 - \rho)$.

Black Box Initial Conditions

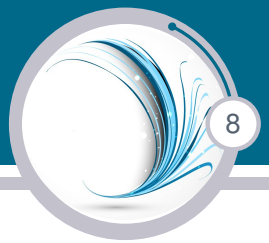
The densities must satisfy the initial conditions

$$\rho_i(0, x) = \rho_i^0(x),$$

where $\rho_i^0(x)$ are black-box functions. This means that $\rho_i^0(x)$ are observable only through noisy measurements $\{\boldsymbol{x}_i^0, \boldsymbol{\rho}_i^0\}$.

Introduction to Gaussian Processes

Gaussian Processes



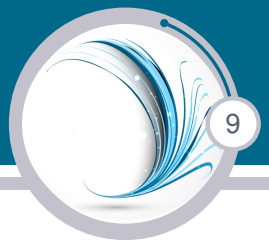
A Gaussian process

$$f(x) \sim \mathcal{GP}(0, k(x, x'; \theta)),$$

is just a shorthand notation for

$$\begin{bmatrix} f(x) \\ f(x') \end{bmatrix} \sim \mathcal{N}(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} k(x,x;\theta) & k(x,x';\theta) \\ k(x',x;\theta) & k(x',x';\theta) \end{bmatrix}.$$

Gaussian Processes



A Gaussian process

$$f(x) \sim \mathcal{GP}(0, k(x, x'; \theta)),$$

is just a shorthand notation for

$$\begin{bmatrix} f(x) \\ f(x') \end{bmatrix} \sim \mathcal{N}(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} k(x,x;\theta) & k(x,x';\theta) \\ k(x',x;\theta) & k(x',x';\theta) \end{bmatrix}.$$

Squared Exponential Covariance Function

A typical example for the kernel $k(x, x'; \theta)$ is the squared exponential covariance function, i.e.,

$$k(x, x'; \theta) = \gamma^2 \exp\left(-\frac{1}{2} w^2 (x - x')^2\right),$$

where $\theta = (\gamma, w)$ are the hyper-parameters of the kernel.

Training

Given a dataset $\{\boldsymbol{x}, \boldsymbol{y}\}$ of size N, the hyper-parameters θ and the noise variance parameter σ^2 can be trained by minimizing the negative log marginal likelihood

$$\mathcal{NLML}(\theta, \sigma) = \frac{1}{2}\boldsymbol{y}^T\boldsymbol{K}^{-1}\boldsymbol{y} + \frac{1}{2}\log|\boldsymbol{K}| + \frac{N}{2}\log(2\pi),$$

resulting from

$$\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K}),$$

where $\boldsymbol{K} = k(\boldsymbol{x}, \boldsymbol{x}; \theta) + \sigma^2\boldsymbol{I}$.

Prediction

Having trained the hyper-parameters and parameters of the model, one can use the posterior distribution

$$f(x^*)|\boldsymbol{y} \sim \mathcal{N}(k(x^*,\boldsymbol{x})\boldsymbol{K}^{-1}\boldsymbol{y}, k(x^*,x^*) - k(x^*,\boldsymbol{x})\boldsymbol{K}^{-1}k(\boldsymbol{x},x^*)).$$

to make predictions at a new test point x^*.

Prediction

Having trained the hyper-parameters and parameters of the model, one can use the posterior distribution

$$f(x^*)|\boldsymbol{y} \sim \mathcal{N}(k(x^*,\boldsymbol{x})\boldsymbol{K}^{-1}\boldsymbol{y}, k(x^*,x^*) - k(x^*,\boldsymbol{x})\boldsymbol{K}^{-1}k(\boldsymbol{x},x^*)).$$

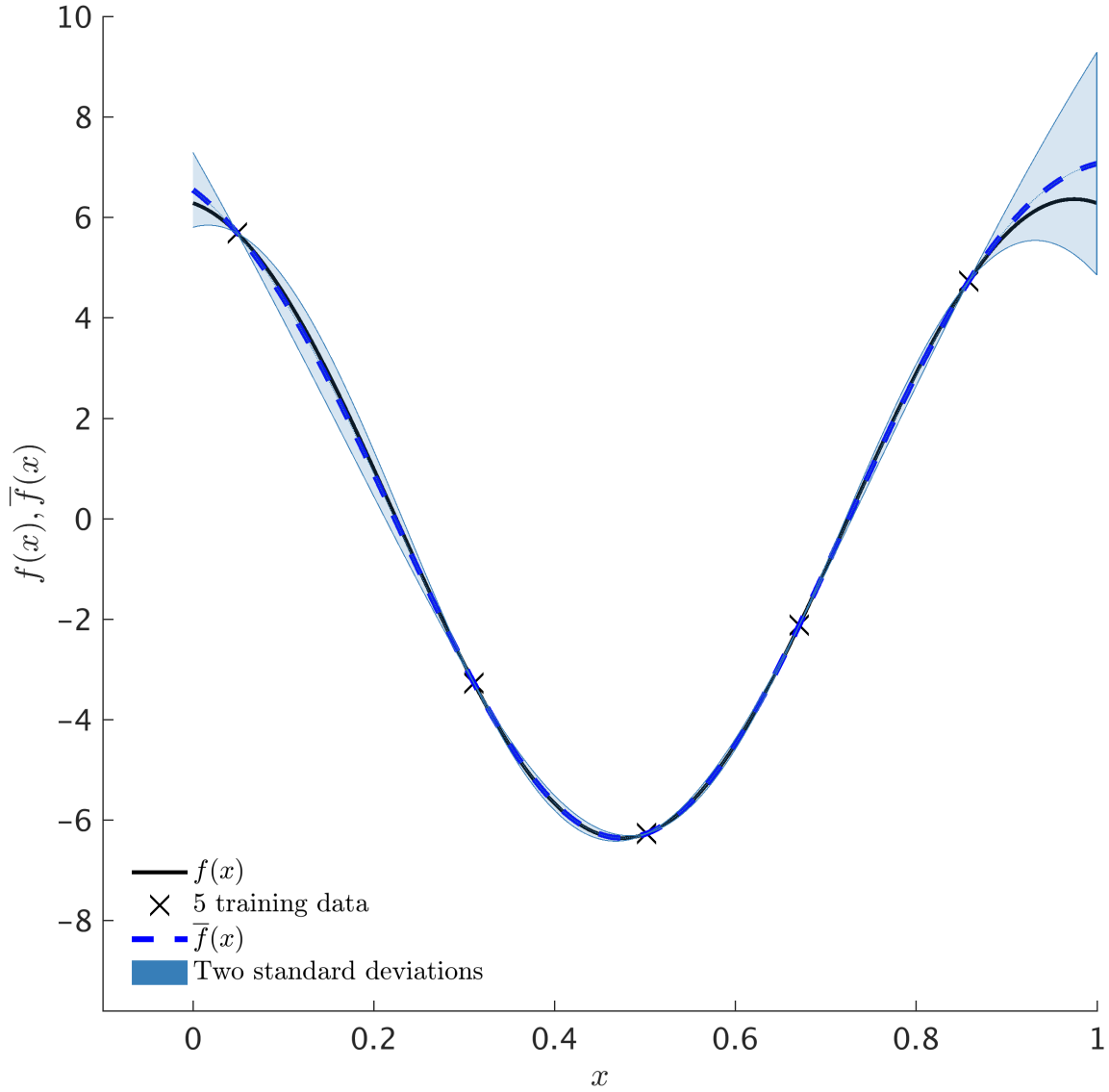
to make predictions at a new test point x^*.

This is obtained by writing the joint distribution

$$\begin{bmatrix} f(x^*) \\ \boldsymbol{y} \end{bmatrix} \sim \mathcal{N}(\begin{bmatrix} 0 \\ \boldsymbol{0} \end{bmatrix}, \begin{bmatrix} k(x^*,\boldsymbol{x}) & k(x^*,\boldsymbol{x}) \\ k(\boldsymbol{x},x^*) & \boldsymbol{K} \end{bmatrix}.$$

Example

- Code



Numerical Gaussian Processes

Numerical Gaussian Processes
Definition

Numerical Gaussian processes are Gaussian processes with covariance functions resulting from temporal discretization of time-dependent partial differential equations.

Example: Burgers' Equation

Burgers' equation is a fundamental non-linear partial differential equation arising in various areas of applied mathematics, including fluid mechanics, nonlinear acoustics, gas dynamics, and traffic flow.

Example: Burgers' Equation

Burgers' equation is a fundamental non-linear partial differential equation arising in various areas of applied mathematics, including fluid mechanics, nonlinear acoustics, gas dynamics, and traffic flow.

In one space dimension the Burgers' equation reads as

$$u_t + uu_x = \nu u_{xx},$$

along with Dirichlet boundary conditions $u(t,-1) = u(t,1) = 0$, where $u(t,x)$ denotes the unknown solution and $\nu = 0.01/\pi$ is a viscosity parameter.

Problem Setup
Burgers' Equation

Let us assume that all we observe are noisy measurements

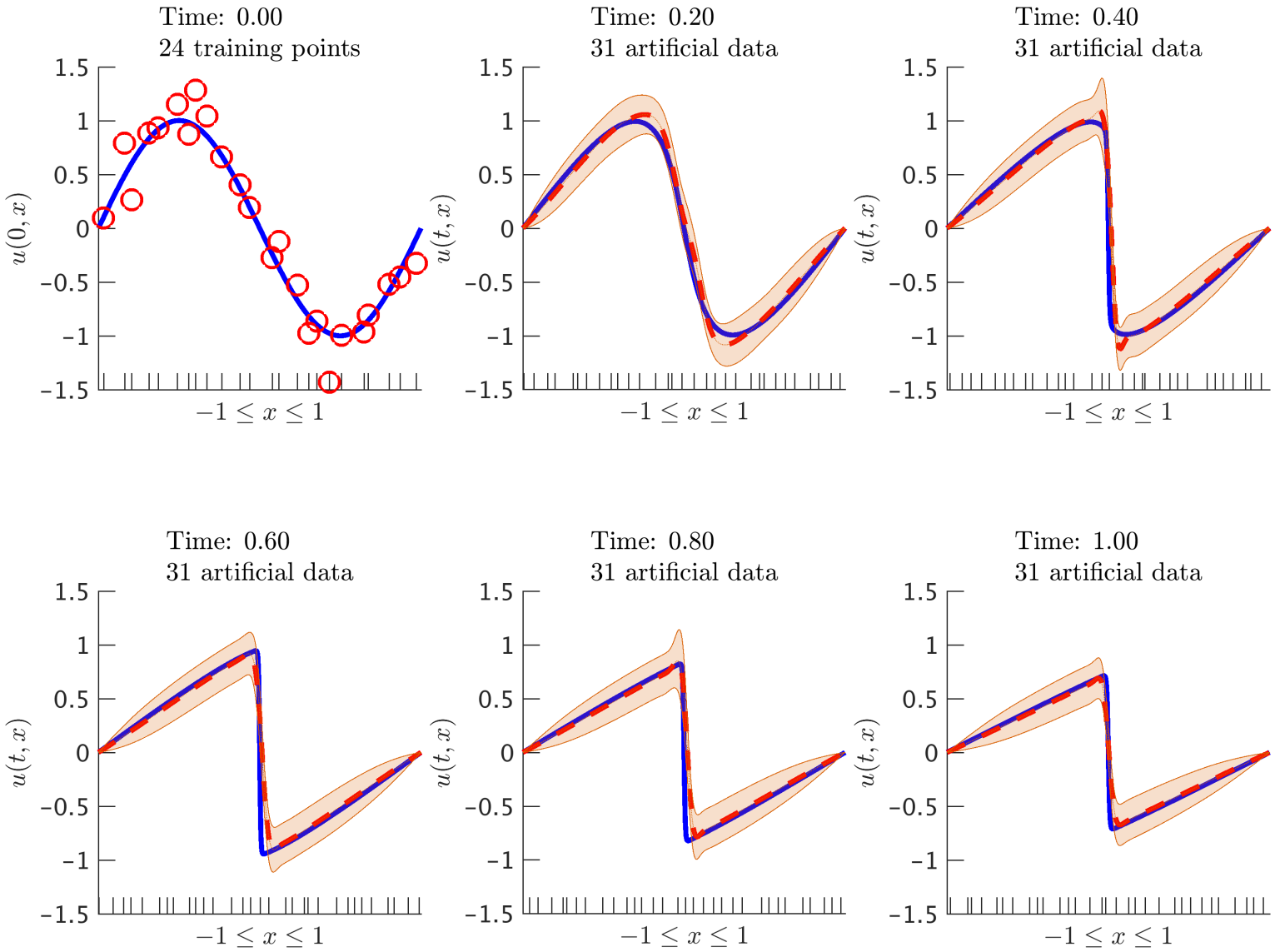
$$\{\boldsymbol{x}^0, \boldsymbol{u}^0\}$$

of the *black-box* initial function $u(0, x)$.

Given such measurements, we would like to solve the Burgers' equation while propagating through time the uncertainty associated with the noisy initial data.

Burgers' equation

- Movie
- Code



Burgers' equation

It is remarkable that the proposed methodology can effectively propagate an infinite collection of correlated Gaussian random variables (i.e., a Gaussian process) through the complex nonlinear dynamics of the Burgers' equation.

Let us apply the backward Euler scheme to the Burgers' equation. This can be written as

$$u^n + \Delta t u^n \frac{d}{dx} u^n - \nu \Delta t \frac{d^2}{dx^2} u^n = u^{n-1}.$$

Let us apply the backward Euler scheme to the Burgers' equation. This can be written as

$$u^n + \Delta t \mu^{n-1} \frac{d}{dx} u^n - \nu \Delta t \frac{d^2}{dx^2} u^n = u^{n-1}.$$

Let us make the prior assumption that

$$u^n(x) \sim \mathcal{GP}(0, k(x, x'; \theta)),$$

is a Gaussian process with a neural network covariance function

$$k(x, x'; \theta) = \frac{2}{\pi} \sin^{-1}\left(\frac{2(\sigma_0^2 + \sigma^2 x x')}{\sqrt{\left(1 + 2\left(\sigma_0^2 + \sigma^2 x^2\right)\right)\left(1 + 2\left(\sigma_0^2 + \sigma^2 x'^2\right)\right)}}\right),$$

where $\theta = \left(\sigma_0^2, \sigma^2\right)$ denotes the hyper-parameters.

This enables us to obtain the following *Numerical Gaussian Process*

$$\begin{bmatrix} u^n \\ u^{n-1} \end{bmatrix} \sim \mathcal{GP}\left(0, \begin{bmatrix} k_{u,u}^{n,n} & k_{u,u}^{n,n-1} \\ & k_{u,u}^{n-1,n-1} \end{bmatrix}\right).$$

The covariance functions for the Burgers' equation example are given by $k_{u,u}^{n,n} = k$,

$$k_{u,u}^{n,n-1} = k + \Delta t \mu^{n-1}(x') \frac{d}{dx'} k - \nu \Delta t \frac{d^2}{dx'^2} k.$$

The covariance functions for the Burgers' equation example are given by $k_{u,u}^{n,n} = k$,

$$k_{u,u}^{n,n-1} = k + \Delta t \mu^{n-1}(x') \frac{d}{dx'} k - \nu \Delta t \frac{d^2}{dx'^2} k.$$

Compare this with

$$u^n + \Delta t \mu^{n-1} \frac{d}{dx} u^n - \nu \Delta t \frac{d^2}{dx^2} u^n = u^{n-1}.$$

$$
\begin{aligned}
k_{u,u}^{n-1,n-1} &= k + \Delta t \mu^{n-1}(x') \frac{d}{dx'} k - \nu \Delta t \frac{d^2}{dx'^2} k, \\
&+ \Delta t \mu^{n-1}(x) \frac{d}{dx} k + \Delta t^2 \mu^{n-1}(x) \mu^{n-1}(x') \frac{d}{dx} \frac{d}{dx'} k \\
&- \nu \Delta t^2 \mu^{n-1}(x) \frac{d}{dx} \frac{d^2}{dx'^2} k - \nu \Delta t \frac{d^2}{dx^2} k \\
&- \nu \Delta t^2 \mu^{n-1}(x') \frac{d^2}{dx^2} \frac{d}{dx'} k + \nu^2 \Delta t^2 \frac{d^2}{dx^2} \frac{d^2}{dx'^2} k.
\end{aligned}
$$

The hyper-parameters θ and the noise parameters $\sigma_n^2, \sigma_{n-1}^2$ can be trained by employing the *Negative Log Marginal Likelihood* resulting from

$$\begin{bmatrix} \boldsymbol{u}_b^n \\ \boldsymbol{u}^{n-1} \end{bmatrix} \sim \mathcal{N}(0, \boldsymbol{K}),$$

where $\{\boldsymbol{x}_b^n, \boldsymbol{u}_b^n\}$ are the (noisy) data on the boundary and $\{\boldsymbol{x}^{n-1}, \boldsymbol{u}^{n-1}\}$ are artificially generated data. Here,

$$\boldsymbol{K} = \begin{bmatrix} k_{u,u}^{n,n}(\boldsymbol{x}_b^n, \boldsymbol{x}_b^n; \theta) + \sigma_n^2 \boldsymbol{I} & k_{u,u}^{n,n-1}(\boldsymbol{x}_b^n, \boldsymbol{x}^{n-1}; \theta) \\ k_{u,u}^{n-1,n}(\boldsymbol{x}^{n-1}, \boldsymbol{x}_b^n; \theta) & k_{u,u}^{n,n-1}(\boldsymbol{x}^{n-1}, \boldsymbol{x}^{n-1}; \theta) + \sigma_{n-1}^2 \boldsymbol{I} \end{bmatrix}$$

In order to predict $u^n(x_*^n)$ at a new test point x_*^n, we use the following conditional distribution

$$u^n(x_*^n) \mid \boldsymbol{u}_b^n \sim \mathcal{N}\left(\mu^n(x_*^n), \Sigma^{n,n}(x_*^n, x_*^n)\right),$$

where

$$\mu^n(x_*^n) = \boldsymbol{q}^T \boldsymbol{K}^{-1} \begin{bmatrix} \boldsymbol{u}_b^n \\ \boldsymbol{\mu}^{n-1} \end{bmatrix},$$

and

$$\Sigma^{n,n}(x_*^n, x_*^n) = k_{u,u}^{n,n}(x_*^n, x_*^n) - \boldsymbol{q}^T \boldsymbol{K}^{-1} \boldsymbol{q} + \boldsymbol{q}^T \boldsymbol{K}^{-1} \begin{bmatrix} 0 & 0 \\ 0 & \boldsymbol{\Sigma}^{n-1,n-1} \end{bmatrix} \boldsymbol{K}^{-1} \boldsymbol{q}.$$

Here, $\boldsymbol{q}^T = \begin{bmatrix} k_{u,u}^{n,n}(x_*^n, \boldsymbol{x}_b^n) & k_{u,u}^{n,n-1}(x_*^n, \boldsymbol{x}^{n-1}) \end{bmatrix}$.

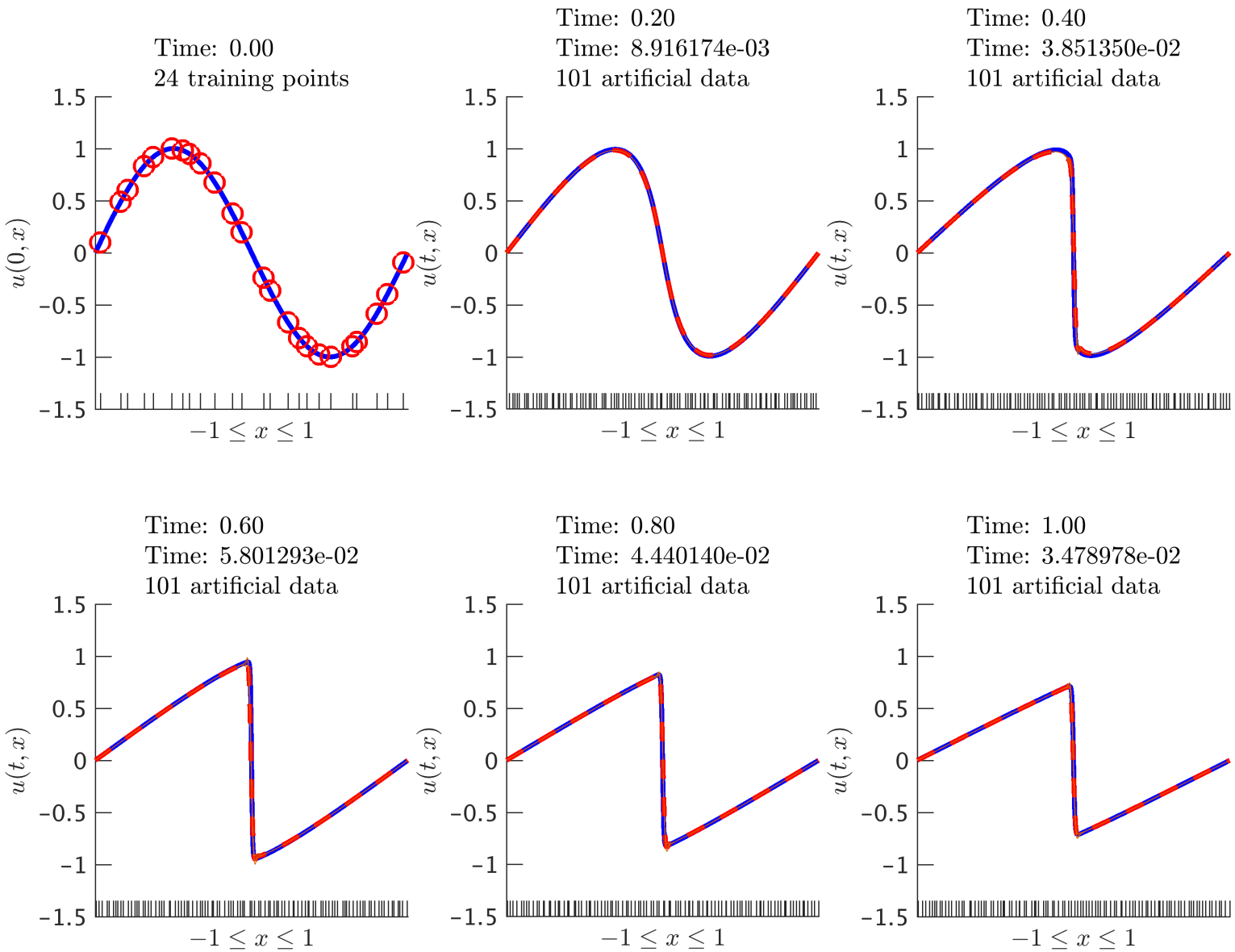
Now, one can use the resulting posterior distribution to obtain the *artificially generated data* $\{\boldsymbol{x}^n, \boldsymbol{u}^n\}$ for the next time step with

$$\boldsymbol{u}^n \sim \mathcal{N}\left(\boldsymbol{\mu}^n, \Sigma^{n,n}\right).$$

Here, $\boldsymbol{\mu}^n = \mu^n(\boldsymbol{x}^n)$ and $\Sigma^{n,n} = \Sigma^{n,n}(\boldsymbol{x}^n, \boldsymbol{x}^n)$.

Noiseless data

- Movie
- Code

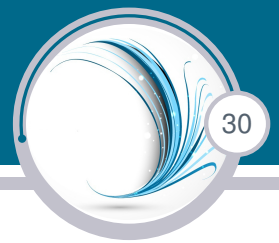


General Framework

Numerical Gaussian Processes

It must be emphasized that *numerical Gaussian processes*, by construction, are designed to deal with cases where:

- (1) all we observe is noisy data on *black-box* initial conditions, and
- (2) we are interested in *quantifying the uncertainty* associated with such noisy data in our solutions to time-dependent partial differential equations.



Let us consider linear partial differential equations of the form

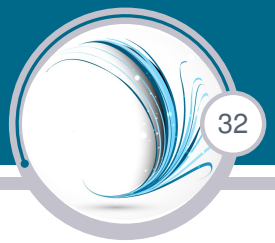
$$u_t = \mathcal{L}_x u, \;\; x \in \Omega, \;\; t \in [0, T],$$

where $\mathcal{L}_x$ is a linear operator and $u(t, x)$ denotes the latent solution.

Linear Multi-step Methods

The trapezoidal time-stepping scheme can be written as

$$u^n - \frac{1}{2}\Delta t \mathcal{L}_x u^n = u^{n-1} + \frac{1}{2}\Delta t \mathcal{L}_x u^{n-1}$$



The trapezoidal time-stepping scheme can be written as

$$u^n - \frac{1}{2}\Delta t \mathcal{L}_x u^n =: u^{n-1/2} := u^{n-1} + \frac{1}{2}\Delta t \mathcal{L}_x u^{n-1}$$

By assuming

$$u^{n-1/2}(x) \sim \mathcal{GP}(0, k(x, x'; \theta)),$$

we can capture the entire structure of the trapezoidal rule in the resulting joint distribution of u^n and u^{n-1}.

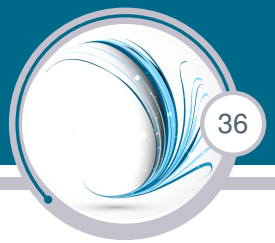
Runge-Kutta Methods

The trapezoidal time-stepping scheme can be written as

$$\begin{aligned} u^n &= u^{n-1} + \frac{1}{2}\Delta t \mathcal{L}_x u^{n-1} + \frac{1}{2}\Delta t \mathcal{L}_x u^n \\ u^n &= u^n. \end{aligned}$$

The trapezoidal time-stepping scheme can be written as

$$\begin{aligned} u_2^n &= u^{n-1} + \frac{1}{2}\Delta t \mathcal{L}_x u^{n-1} + \frac{1}{2}\Delta t \mathcal{L}_x u^n \\ u_1^n &= u^n. \end{aligned}$$



By assuming

$$\begin{aligned} u^n(x) &\sim \mathcal{GP}(0, k^{n,n}(x, x'; \theta_n)), \\ u^{n-1}(x) &\sim \mathcal{GP}(0, k^{n+1,n+1}(x, x'; \theta_{n+1})), \end{aligned}$$

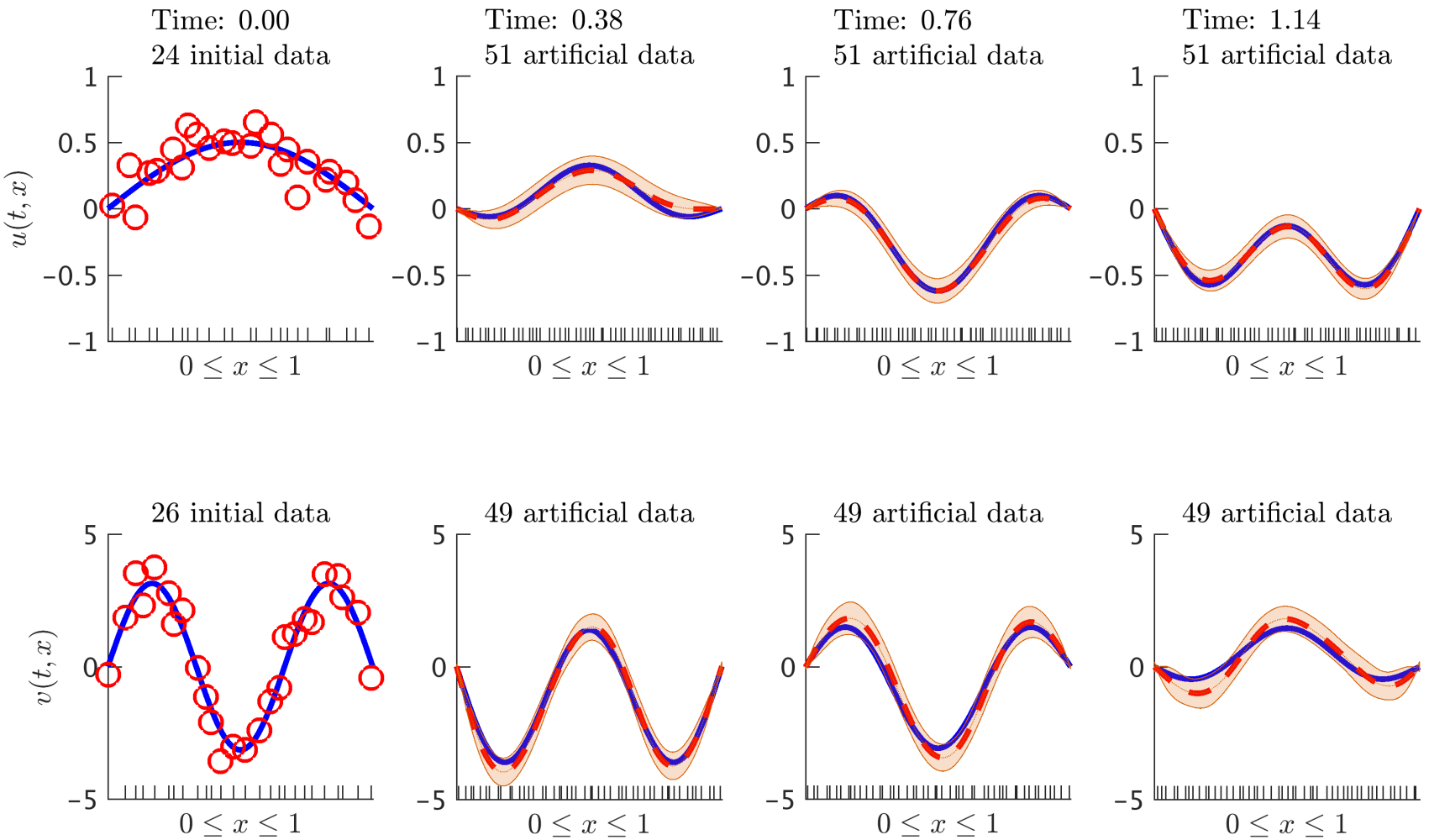
we can capture the entire structure of the trapezoidal rule in the resulting joint distribution of u^n, u^{n-1}, u_2^n, and u_1^n. Here,

$$u_2^n = u_1^n = u^n.$$

Experiments

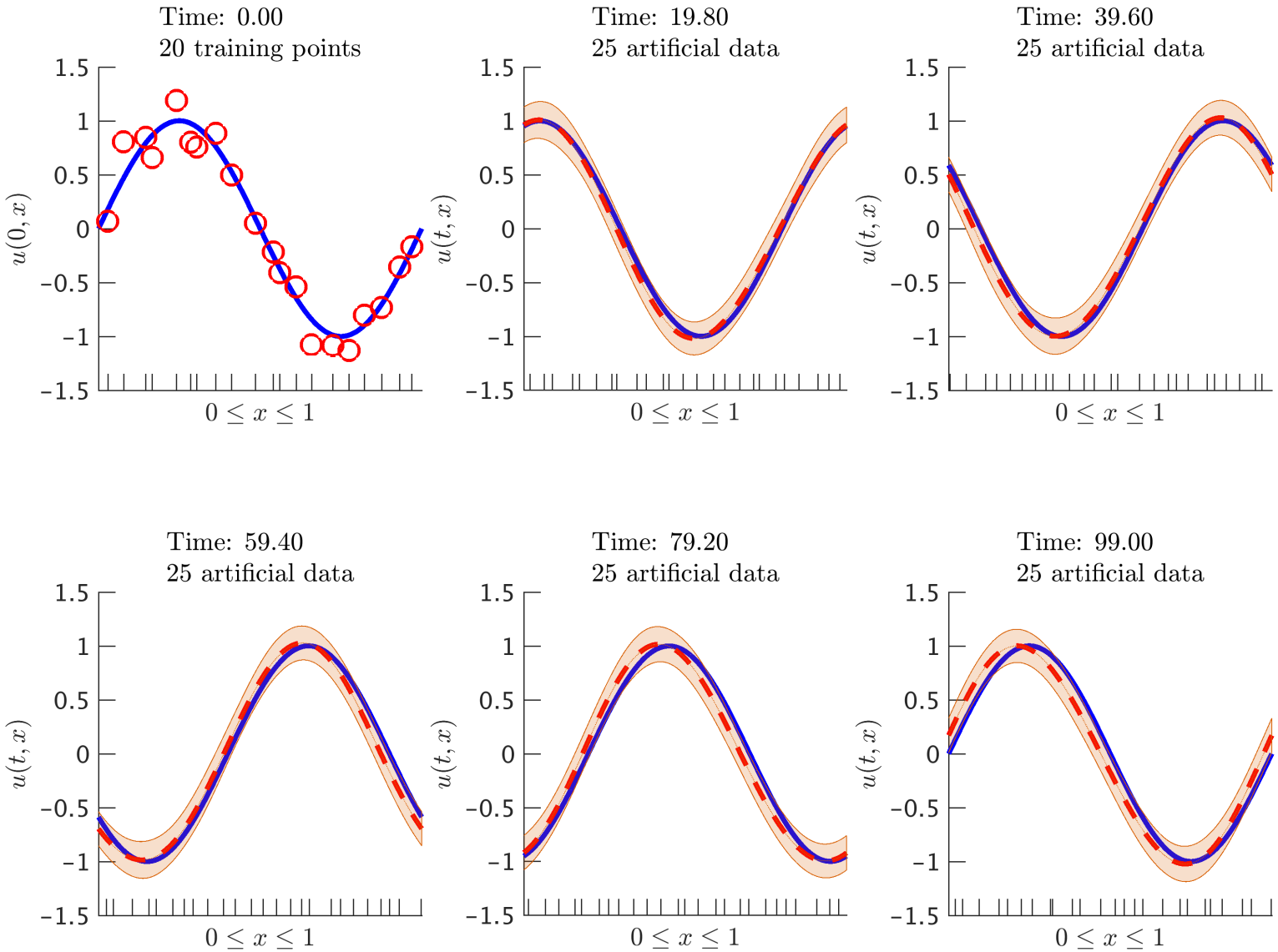
Wave Equation – Trapezoidal Rule

- Movie
- Code



Advection Equation – Gauss-Legendre

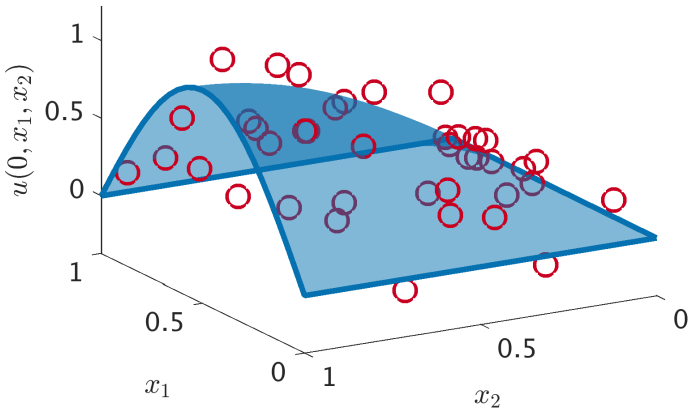
- Movie
- Code



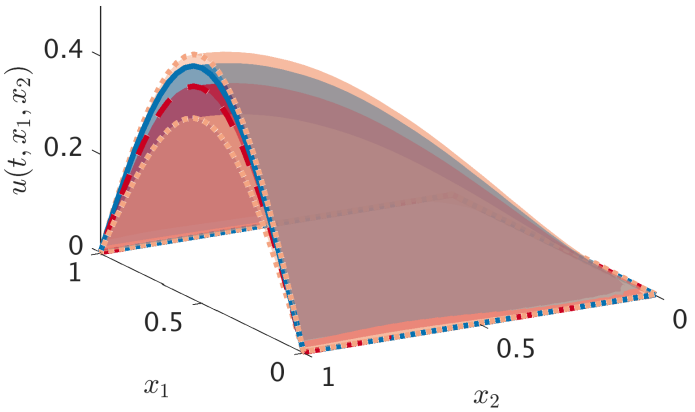
Heat Equation – Trapezoidal Rule

- Movie
- Code

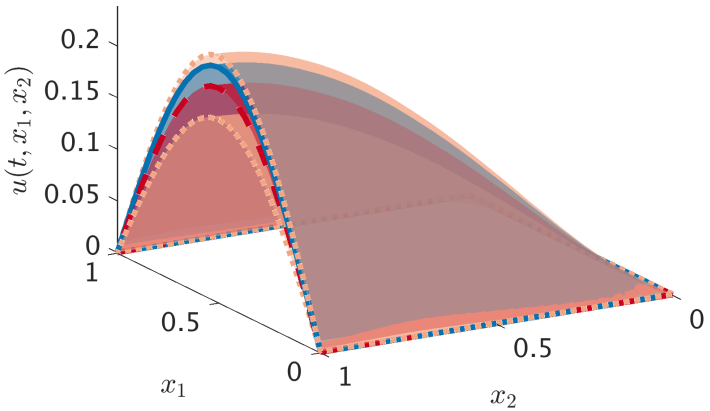
Time: 0.00
40 training points



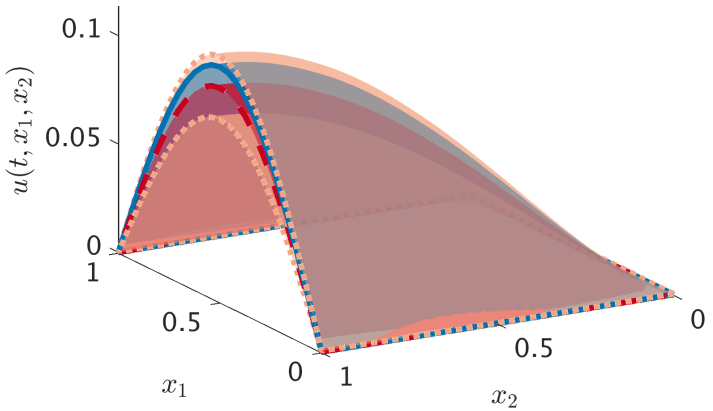
Time: 0.06
20 artificial data



Time: 0.12
20 artificial data

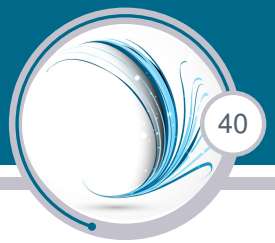


Time: 0.18
20 artificial data



Navier-Stokes Equations

Navier-Stokes Equations in 2D



Let us consider the Navier-Stokes equations in 2D given explicitly by

$$u_t + uu_x + vu_y = -p_x + \frac{1}{Re}(u_{xx} + u_{yy}),$$
$$v_t + uv_x + vv_y = -p_y + \frac{1}{Re}(v_{xx} + v_{yy}),$$

where the unknowns are the 2-dimensional velocity field $(u(t, x, y), v(t, x, y))$ and the pressure $p(t, x, y)$. Here, Re denotes the Reynolds number.

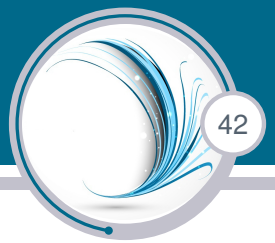
Continuity Equation

Solutions to the Navier-Stokes equations are searched in the set of divergence-free functions; i.e.,

$$u_x + v_y = 0.$$

This extra equation is the continuity equation for incompressible fluids that describes the conservation of mass of the fluid.

Backward Euler



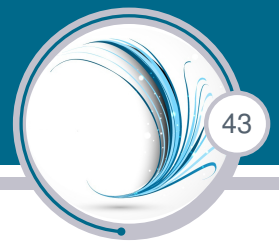
Applying the backward Euler time stepping scheme to the Navier-Stokes equations we obtain

$$u^n + \Delta t \overline{u}^{n-1} u_x^n + \Delta t \overline{v}^{n-1} u_y^n + \Delta t p_x^n - \frac{\Delta t}{Re}(u_{xx}^n + u_{yy}^n) = u^{n-1},$$

$$v^n + \Delta t \overline{u}^{n-1} v_x^n + \Delta t \overline{v}^{n-1} v_y^n + \Delta t p_y^n - \frac{\Delta t}{Re}(v_{xx}^n + v_{yy}^n) = v^{n-1},$$

where $u^n(x, y) = u(t^n, x, y)$.

Divergence Free



We make the assumption that

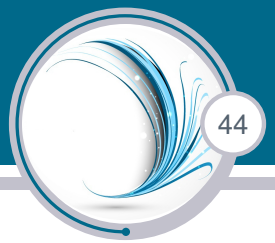
$$u^n = \psi^n_y, \quad v^n = -\psi^n_x,$$

for some latent function $\psi^n(x, y)$. Under this assumption, the continuity equation will be automatically satisfied. We proceed by placing a Gaussian process prior on

$$\psi^n(x, y) \sim \mathcal{GP}\left(0, k((x, y), (x', y'); \theta)\right),$$

where θ are the hyper-parameters of the kernel $k((x, y), (x', y'); \theta)$.

Divergence Free Prior



This will result in the following multi-output Gaussian process

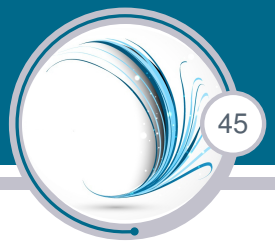
$$\begin{bmatrix} u^n \\ v^n \end{bmatrix} \sim \mathcal{GP}\left(0, \begin{bmatrix} k_{u,u}^{n,n} & k_{u,v}^{n,n} \\ k_{v,u}^{n,n} & k_{v,v}^{n,n} \end{bmatrix}\right),$$

where

$$k_{u,u}^{n,n} = \frac{\partial}{\partial y}\frac{\partial}{\partial y'}k, \qquad k_{u,v}^{n,n} = -\frac{\partial}{\partial y}\frac{\partial}{\partial x'}k,$$
$$k_{v,u}^{n,n} = -\frac{\partial}{\partial x}\frac{\partial}{\partial y'}k, \qquad k_{v,v}^{n,n} = \frac{\partial}{\partial x}\frac{\partial}{\partial x'}k.$$

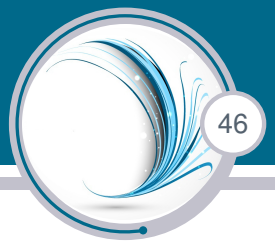
Any samples generated from this multi-output Gaussian process will satisfy the continuity equation.

Pressure



Moreover, independent from $\psi^n(x, y)$, we will place a Gaussian process prior on $p^n(x, y)$; i.e.,

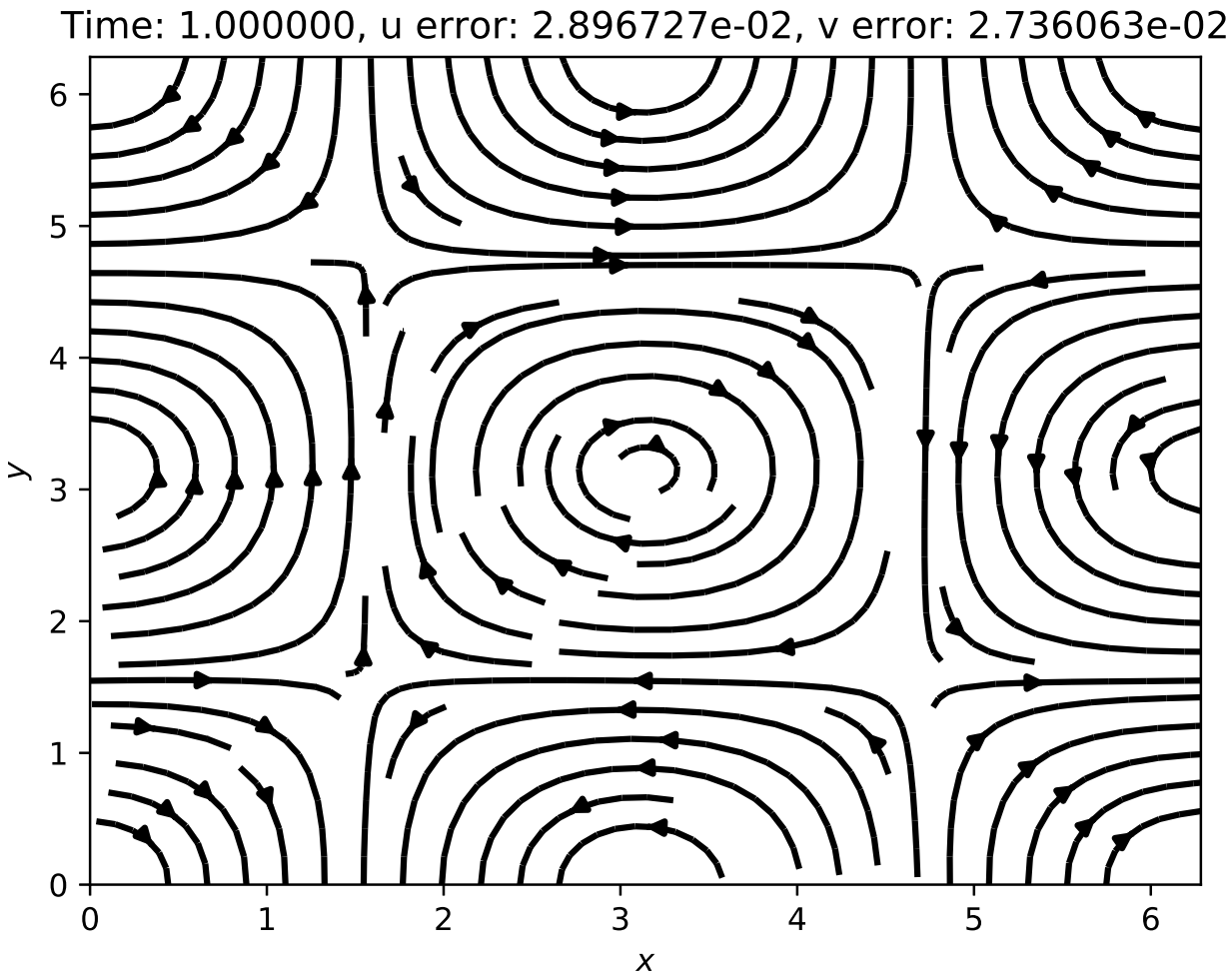
$$p^n(x, y) \sim \mathcal{GP}(0, k_{p,p}^{n,n}((x, y), (x', y'); \theta_p)).$$



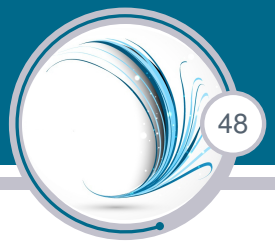
This will allow us to obtain the following *numerical Gaussian process* encoding the structure of the Navier-Stokes equations and the backward Euler time stepping scheme in its kernels; i.e.,

$$\begin{bmatrix} u^n \\ v^n \\ p^n \\ u^{n-1} \\ v^{n-1} \end{bmatrix} \sim \mathcal{GP}\left(0, \begin{bmatrix} k_{u,u}^{n,n} & k_{u,v}^{n,n} & 0 & k_{u,u}^{n,n-1} & k_{u,v}^{n,n-1} \\ & k_{v,v}^{n,n} & 0 & k_{v,u}^{n,n-1} & k_{v,v}^{n,n-1} \\ & & k_{p,p}^{n,n} & k_{p,u}^{n,n-1} & k_{p,v}^{n,n-1} \\ & & & k_{u,u}^{n-1,n-1} & k_{u,v}^{n-1,n-1} \\ & & & & k_{v,v}^{n-1,n-1} \end{bmatrix}\right).$$

Taylor-Green Vortex



Concluding Remarks



We have presented a novel machine learning framework for encoding physical laws described by partial differential equations into Gaussian process priors for nonparametric Bayesian regression.

The proposed algorithms can be used to infer solutions to time-dependent and nonlinear partial differential equations, and effectively quantify and propagate uncertainty due to noisy initial or boundary data.


Thank you!