Uncertainty Quantification in Physics-informed Neural Networks using Laplace Approximations

Master Thesis

Thomas Albrecht

December 9, 2022

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Faculty of Science
Department of Computer Science
Chair for the Methods of Machine Learning

MAX-PLANCK-GESELLSCHAFT
erc
ellis
CyberValley
DFG
Federal Ministry of Education and Research
Baden-Württemberg
MINISTERIUM FÜR WISSENSCHAFT, FORSCHUNG UND KUNST
imprs-is

Problem Setup

- General partial differential equation given by:

$$\mathcal{F}[u](x) = 0, \quad x \in \Omega \tag{1}$$

$$\mathcal{B}[u](x) = 0, \quad x \in \partial\Omega \tag{2}$$

  $u \colon \Omega \to \mathbb{R}^m$ arbitrary function and $\Omega \in \mathbb{R}^d$ bounded domain.
  $\mathcal{F}$ general differential operator and $\mathcal{B}$ boundary condition operator
- Goal: Find a solution function $u(x)$ to the PDE

Physics-informed Neural Networks (PINNs)

- Choose $u(\cdot)$ to be a neural network u_θ, minimize the loss:

$$L = MSE_b + MSE_f = \frac{1}{N_b}\sum_{i=1}^{N_b} |\mathcal{B}[u_\theta](x_b^i)|^2 + \frac{1}{N_f}\sum_{i=1}^{N_f} |\mathcal{F}[u_\theta](x_f^i)|^2 \tag{3}$$

- For initial and boundary collocation points $\{x_b^i\}$ and grid collocation points $\{x_f^i\}$.

Probabilistic Model

- Likelihoods:

$$p(z^f \mid x^f, \theta) = \mathcal{N}(\mathcal{F}[u_\theta](x^f), \sigma_f^2) \tag{4}$$

$$p(z^b \mid x^b, \theta) = \mathcal{N}(\mathcal{B}[u_\theta](x^b), \sigma_b^2) \tag{5}$$

- Prior over parameters: $p(\theta) \sim \mathcal{N}(0, \sigma^2 I)$
- Data: $\mathcal{D}_f = \{x_i^f, (z_i^f)\}_{i=1}^{N_f}$, $\mathcal{D}_b = \{x_i^b, (z_i^b)\}_{i=1}^{N_b}$, $\mathcal{D} = \mathcal{D}_f \cup \mathcal{D}_b$
- Posterior: $p(\theta \mid \mathcal{D}) \propto p(\mathcal{D}_f \mid \theta) p(\mathcal{D}_b \mid \theta) p(\theta)$

Laplace Approximations

$$p(\theta \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \theta)p(\theta)}{\int_\theta p(\mathcal{D} \mid \theta)p(\theta)d\theta} = \frac{p(\mathcal{D}_f \mid \theta)p(\mathcal{D}_b \mid \theta)p(\theta)}{\int_\theta p(\mathcal{D}_f \mid \theta)\, p(\mathcal{D}_b \mid \theta)p(\theta)d\theta} =: \frac{1}{Z}h(\theta) \tag{6}$$

$$\log h(\theta) \approx \log\left[p(\mathcal{D}_f \mid \theta_{MAP})p(\mathcal{D}_b \mid \theta_{MAP})p(\theta_{MAP})\right] - \frac{1}{2}((\theta - \theta_{MAP})^T \Lambda(\theta - \theta_{MAP})) \tag{7}$$

$$\text{with } \Lambda = -\nabla^2 \left[\log p(\mathcal{D}_f \mid \theta_{MAP}) + \log p(\mathcal{D}_b \mid \theta_{MAP}) + \log p(\theta_{MAP})\right]\Big|_{\theta_{MAP}}$$

Reminder:

$$\log p(\mathcal{D}_f \mid \theta_{MAP}) = \sum_{i=1}^{N} \log\left(\frac{1}{\sqrt{2\pi\sigma_f^2}}\right) + \left(-\frac{\mathcal{F}[u_{\theta_{MAP}}](x_i^f)^T \mathcal{F}[u_{\theta_{MAP}}](x_i^f)}{2\sigma_f^2}\right) \tag{8}$$

Posterior

Laplace approximated posterior: $p(\theta \mid \mathcal{D})$, but we want $p(u \mid \mathcal{D})$

$$u_\theta(x) \approx u_{\theta_{MAP}}(x) + J^T(\theta - \theta_{MAP}), \quad with\, J := \nabla_\theta u_\theta(x)\big|_{\theta_{MAP}} \tag{9}$$

This way the marginal distribution over the output is again Gaussian (given that the approximate posterior is Gaussian):

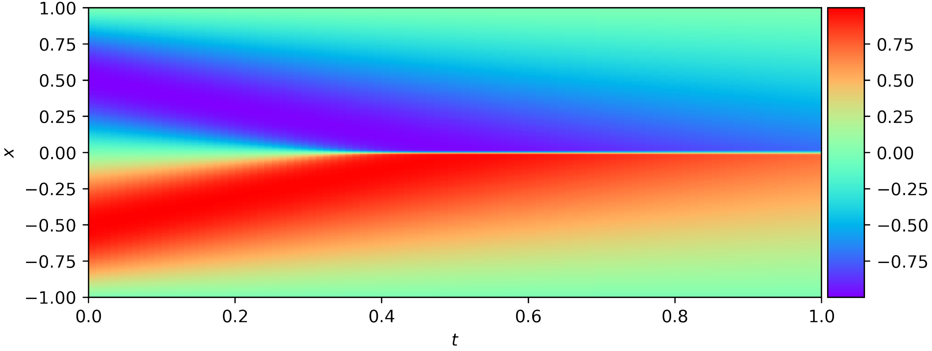
$$p(u(x) \mid \mathcal{D}) = \int \delta(u(x) - u_\theta(x))q(\theta \mid \mathcal{D})d\theta \tag{10}$$

$$\approx \mathcal{N}(u(x) \mid u_{\theta_{MAP}}(x), J^T \Lambda^{-1} J) \tag{11}$$

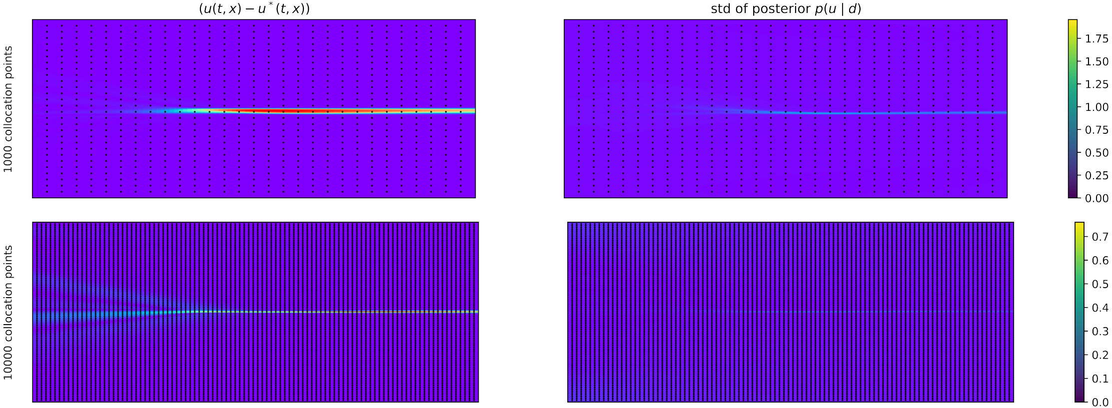
Experiment

- Burgers' equation:

$$u_t + uu_x - (0.01/\pi)u_{xx} = 0,\ x \in [-1, 1],\ t \in [0, 1]$$
$$u(0, x) = -sin(\pi x)$$
$$u(t, -1) = u(t, 1) = 0$$

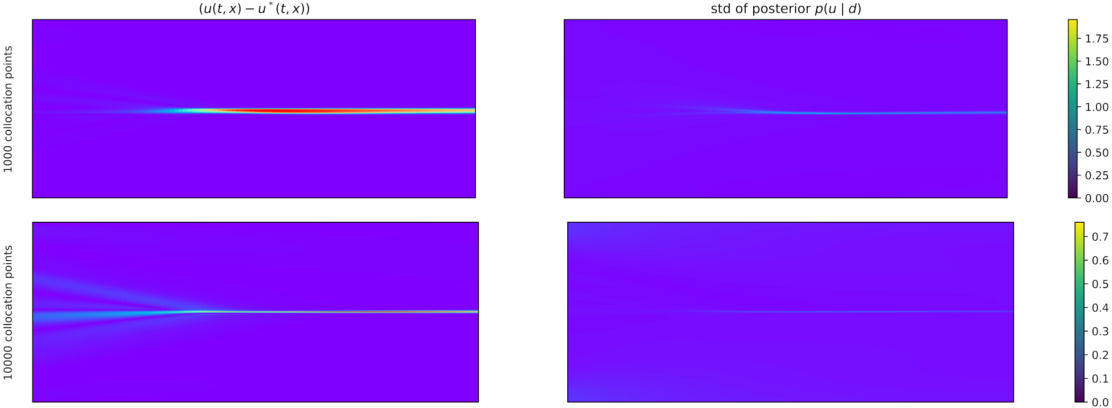


Plots



$(u(t,x) - u^*(t,x))$
std of posterior $p(u \mid d)$
1000 collocation points
10000 collocation points
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7

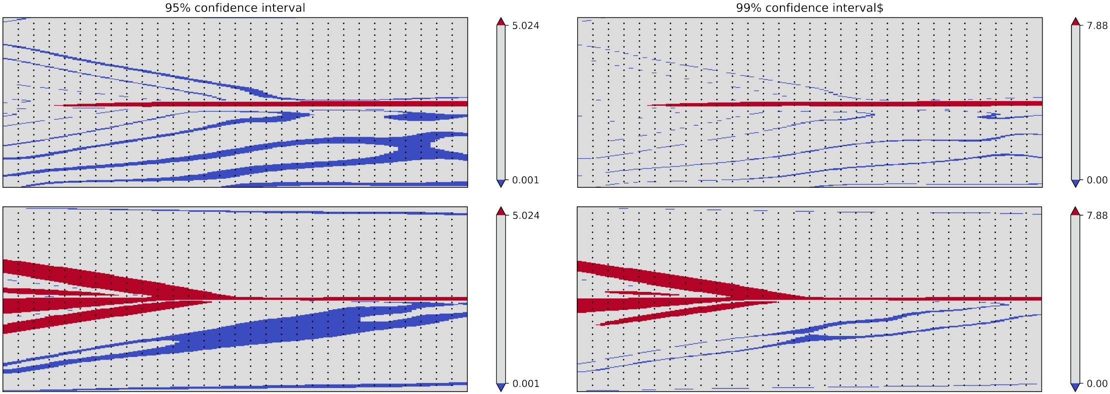
Plots



$(u(t,x) - u^*(t,x))$
std of posterior $p(u \mid d)$
1000 collocation points
10000 collocation points
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7

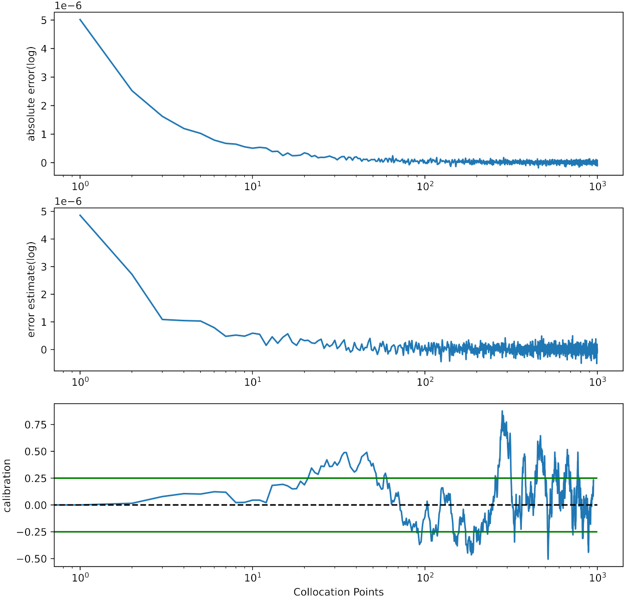
Plots

$$\frac{(u(x) - u^*(x))^2}{\sigma^2_{u(x)}} \sim \chi^2_1$$



Project Status

- Timeline: Only two months left!
- Code for two dimensional PDE works, could be extended to more dimensions if needed
- Other PDEs with same dimensions should work with little work, more complicated PDEs would be interesting but might be impossible in the given timeframe
- Next steps: Experiments to evaluate the method for Burgers' Equation with different grid sizes/network architectures



1e−6
5
4
3
2
1
0
absolute error(log)
10^0
10^1
10^2
10^3
1e−6
5
4
3
2
1
0
error estimate(log)
10^0
10^1
10^2
10^3
0.75
0.50
0.25
0.00
−0.25
−0.50
calibration
10^0
10^1
10^2
10^3
Collocation Points

Questions

- χ^2 confidence intervals with 1 degree of freedom useful?
- Metrics for fit of posterior/calibration
- Other experiments?