Mitigating the Cost of PDE-constrained Bayesian Inverse Problems Using Dimensionality Reduction and Machine Learning

Sheroze Sheriffdeen, Tan Bui-Thanh

The Oden Institute for Computational Engineering and Sciences
The University of Texas at Austin

February 27, 2019

Overview

Sampling-based Bayesian Inference

$$y = \mathcal{F}(x(z)) + \epsilon \tag{1}$$

where,

y represents the observables or the quantity of interest

$\epsilon \sim \mathcal{N}(0, \Sigma_y)$ is the observation noise

$\mathcal{F}$ is the parameter-to-observable map (deterministic)

$x(z)$ represents the state and z represents the parameters

Sampling-based Bayesian Inference

$$y = \mathcal{F}(x(z)) + \epsilon$$

Bayes Rule:

$$p(z|y) = \frac{p(y|z)p(z)}{p(y)} \tag{2}$$

Likelihood function takes the form:

$$p(y|z) = \mathcal{N}(\mathcal{F}(x(z)), \Sigma_y) \tag{3}$$

For sampling-based Bayesian inference, many evaluations of the costly $\mathcal{F}$ operator is necessary.

Idea: Approximate $\mathcal{F}$ in a computationally cheaper manner

General Nonlinear Dynamical System

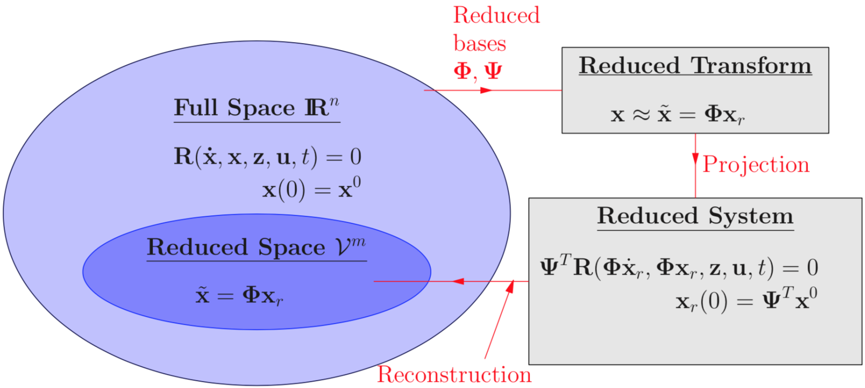


Figure: general projection-based model order reduction[1]

[1]Bui-Thanh, Tan. *Model-constrained optimization methods for reduction of parameterized large-scale systems.* Diss. Massachusetts Institute of Technology, 2007.

Constructing a trial reduced basis Φ

Given current basis Φ, find the location in the parameter space of maximum QoI error by solving:

$$\max_{x, x_r, z} \mathcal{G} = \frac{1}{2} ||y - y_r||_O^2 \tag{4}$$

subject to

$$R(\dot{x}, x, z, u(t), t) = 0 \tag{5}$$

$$x(0) = x^0 \tag{6}$$

$$y = \mathcal{P}(x, z, u(t), t) \tag{7}$$

$$\Psi^T R(\Phi \dot{x}_r, \Phi x_r, z, u(t), t) = 0 \tag{8}$$

$$x_r(0) = \Psi^T x^0 \tag{9}$$

$$y_r = \mathcal{P}(\Phi x_r, z, u, u(t), t) \tag{10}$$

$$z_{\min} \leq z \leq z_{\max} \tag{11}$$

Constructing a trial reduced basis Φ

Algorithm: Model-Constrained Adaptive Sampling Procedure [2]

1. Given a reduced basis Φ and initial guess z^0, find $z^* = \text{argmax}\mathcal{G}(z)$.
2. If $\mathcal{G}(z^*) < \epsilon$, where ϵ is the desired level of accuracy, then terminate then algorithm.
3. Else, with $z = z^*$, solve full system to compute state $x(z^*, t)$ and use span of these solutions to update Φ. Go to step 1.

[2] Bui-Thanh, Tan. *Model-constrained optimization methods for reduction of parameterized large-scale systems.* Diss. Massachusetts Institute of Technology, 2007.

Full steady system

$$A(z)x = B(z), \quad y = C(z)x \tag{12}$$

Define the residual as,

$$R(\Phi x_r, z) = B(z) - A(z)\Phi x_r \tag{13}$$

and this projection-based model order reduction technique will yield the reduced system of the form

$$A_r(z)x_r = B_r(z), \quad y_r = C_r(z)x_r \tag{14}$$

where

$$A_r(z) = \Psi^T A(z)\Phi$$
$$B_r(z) = \Psi^T B(z)$$
$$C_r(z) = C(z)\Phi$$

Reduced Order Model Error

Error in the quantity of interest between the full order model and the reduced order model

$$\begin{aligned}\epsilon_{\text{true}}(z, \Phi) &= y(z) - y_r(z, \Phi) \\ &= C(z)x - C_r(z)x_r \\ &= C(z)(x - \Phi x_r) \\ &= C(z)(x - \tilde{x})\end{aligned}$$

Idea: Predict this error using a deep learning model. $\epsilon_{\text{true}} \approx \epsilon_{\text{NN}}$

$$\tilde{y} = y_r(z, \Phi) + \epsilon_{\text{NN}} \tag{15}$$

Steady Thermal Fin Heat Conduction

Problem Definition: The steady-state temperature distribution within the fin, w, is governed by the following elliptic PDE:

$$-\kappa \nabla^2 w = 0 \quad \text{in} \quad \Omega \tag{16}$$

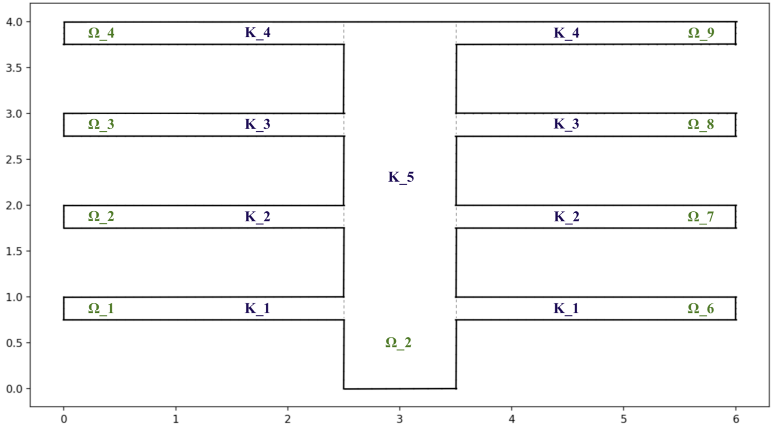
$$-\kappa(\nabla w \cdot \hat{\mathbf{n}}) = \mathrm{Bi}\, w \quad \text{on} \quad \Gamma^{\text{ext}} \setminus \Gamma^{\text{root}} \tag{17}$$

$$-\kappa(\nabla w \cdot \hat{\mathbf{n}}) = -1 \quad \text{on} \quad \Gamma^{\text{root}} \tag{18}$$

- κ denotes the thermal heat conductivity
- Bi is the Biot number
- Ω is the physical domain describing the thermal fin
- Γ^{root} is the bottom edge of the fin
- Γ^{ext} is the exterior edges of the fin
- Equation (17) model convective heat losses to the external surface
- Equation (18) model the heat source at the root

Parameter Setup

The problem is parametrized by $z = \{k_1, k_2, k_3, k_4, k_5\}$ denoting thermal conductivities of sub-fin regions. Assume that $0.1 \leq k_i \leq 10$.



Weak Form

The temperature distribution w belongs to $H^1(\overline{\Omega})$, where $\overline{\Omega} = \sum_{i=1}^{9} \overline{\Omega_i}$, and satisfies the following weak form.

$$a(w, v) = l(v), \forall v \in H^1(\overline{\Omega}) \tag{19}$$

where the bilinear form a is given as,

$$a(w, v) = \int_{\overline{\Omega}} k \nabla w \cdot \nabla v \, \mathrm{d}\overline{\Omega} + \mathrm{Bi} \int_{\overline{\Gamma}^{\mathrm{ext}} \setminus \Gamma^{\mathrm{root}}} wv \, \mathrm{d}\overline{\Gamma} \tag{20}$$

and the linear form l is given as

$$l(v) = \int_{\Gamma^{\mathrm{root}}} v \, \mathrm{d}\overline{\Gamma} \tag{21}$$

Matrix form and quantity of interest

The quantity of interest is the average temperature over the thermal fin:

$$y = \frac{\int_\Omega w \,\mathrm{d}\Omega}{\int_\Omega \mathrm{d}\Omega} \tag{22}$$

The weak form can be written in the matrix form as:

$$A(z)x = B(z), \quad y = C(z)x \tag{23}$$

where x is the nodal temperature value vector.

Deep Feed Forward Neural Network

$$y = W_n^T \sigma(W_{n-1}^T \ldots \sigma(W_1^T x) \ldots)$$

Parameters:

$$\theta = \{W_1, W_2, \ldots W_n\}$$

Error:

$$J(\theta) = y_{\text{true}} - y_{\text{pred}}(\theta)$$

Loss function (e.g. mean square error):

$$loss(\theta) = \frac{1}{N} \sum_{i=1}^{N} |y^i_{\text{true}} - y^i_{\text{pred}}(\theta)|^2$$

Update weights: (e.g. SGD)

$$\theta = \theta - \eta \nabla_\theta J(\theta; y^i_{\text{true}}, y^i_{\text{pred}})$$

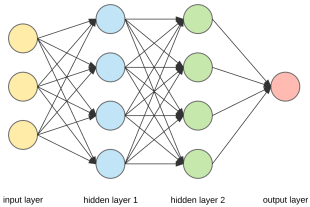


Figure: dense feed forward neural network (https://towardsdatascience.com)

Structure of the neural network

Input: $z = \{k_1, k_2, k_3, k_4, k_5\}$ (thermal conductivity of sub-fins)

Output: $\epsilon_{\text{NN}} \approx y - y_r$

Data: $(z^i, y^i - y_r^i)$ obtained by simultaneously running full order model and reduced order model for the same parameters z_i.

Hyperparameters

Number of hidden layers

Number of weights per hidden layer

Choice of activation function

Choice of optimizer and learning rate

Batch size

Number of epochs

Bayesian optimization of hyperparameters

Parametrize validation error

Let θ^H be the selected hyperparameters.

Let θ be the associated trained weights of the neural network.

$$loss_{\text{val}}(\theta^H) = \frac{1}{N_{\text{val}}} \sum_{i=1}^{N_{\text{val}}} \frac{|\epsilon^i_{\text{true}} - \epsilon^i_{\text{NN}}(\theta, \theta^H)|}{|\epsilon^i_{\text{true}}|}$$

where

$$\epsilon_{\text{true}} = y(z) - y_r(z, \Phi)$$

Bayesian optimization of hyperparameters

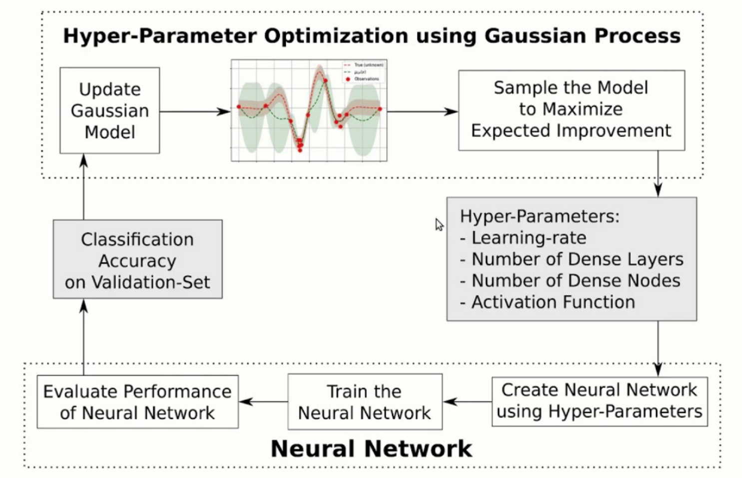


Figure: Flow of Bayesian optimization[3]

[3]https://github.com/Hvass-Labs

Hyperparameters

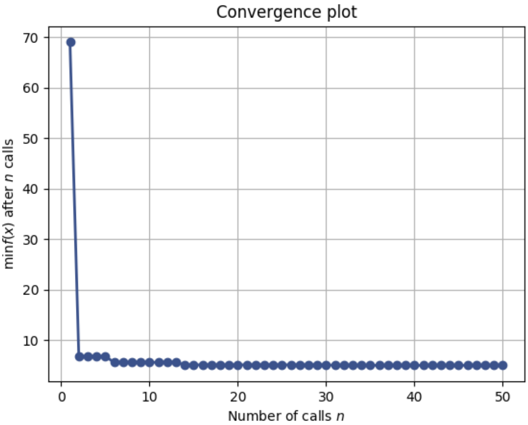
```
space = [Categorical(['relu', 'sigmoid', 'tanh'], name='activation'),
         Categorical([Adam, RMSprop, Adadelta], name='optimizer'),
         Real(1e-4, 1, prior="log-uniform", name='lr'),
         Integer(1, 6, name='n_hidden_layers'),
         Integer(10, 100, name='n_weights'),
         Integer(10, 200, name='batch_size'),
         Integer(100, 400, name='n_epochs')]

res_gp = gp_minimize(objective, space, n_calls=50, random_state=0)
```

Simple implementation using `scikit-optimize`.[4] `objective` function maps given choices of hyperparameters to the average relative validation error.

---

[4]https://scikit-optimize.github.io/

Hyperparameters



A few important hyperparameters for this problem (epoch size in this case).

Deep Neural Network Architecture

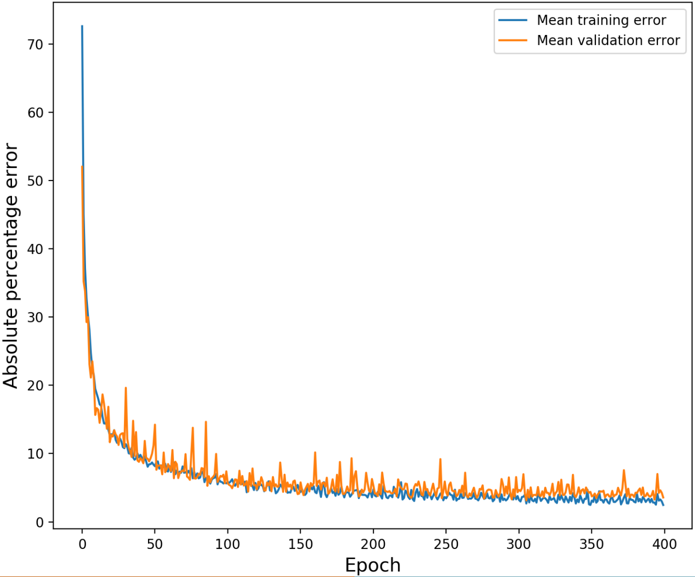
Hyperparameters for the deep neural network after 50 train and evaluate cycles (3.5% average validation error):

- Number of hidden layers: 6
- Number of neurons per hidden layer: 100
- Optimizer: Adam [5]
- Learning rate: 0.0001
- Activation function: Rectified Linear Unit
- Number of epochs: 400
- Batch size: 10

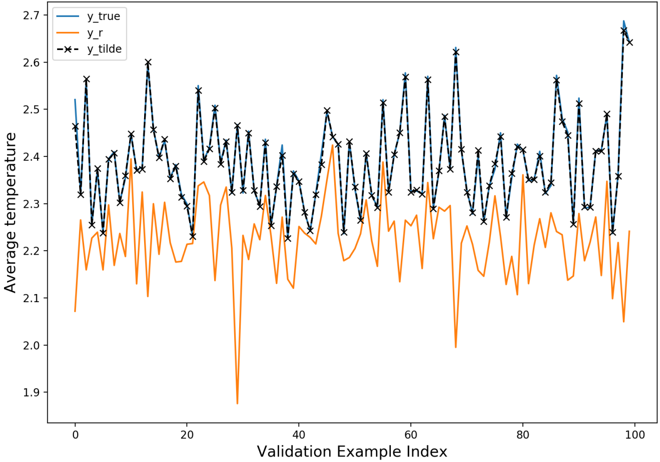
---

[5] Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).

Training error and validation error

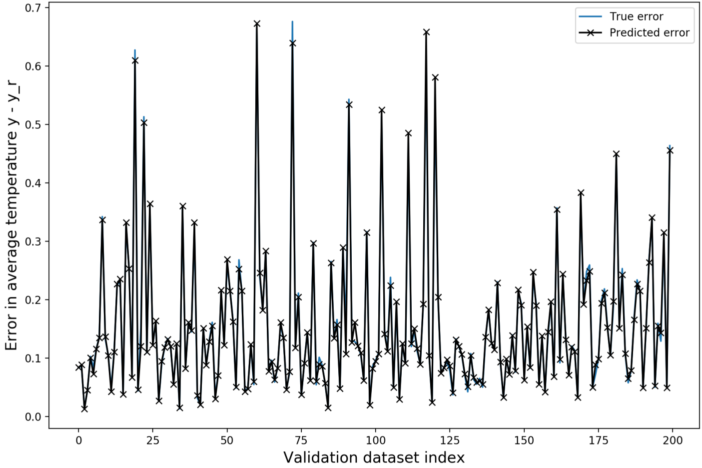


Improvement over reduced order model



True error vs predicted error

The deep neural net has a 3.5% average relative error over the validation dataset.



Summary

- Model order reduction coupled with deep learning can provide computationally efficient and accurate predictions for quantities of interest given expensive offline training.
- Increased efficiency and accuracy mitigates the cost of performing forward solves for sampling-based Bayesian inference.
- In the future, improve deep learning error model by incorporating physics as opposed to the purely data-driven approach shown today.