Uncertainty Analysis of Mathematical Models

Machine Learning Approach

Vilho Halonen
Joint work with Sergey Repin and Ilkka Pölönen

University of Jyväskylä
Faculty of Information Technology

May 31, 2021

1. Uncertain mathematical models
2. Machine learning model
3. Results

Introduction

Uncertainty quantification is an important task for analysis of real-life models. In physical reality problems with fully known data do not exist.

We want to demonstrate that it is feasible to use neural networks trained with data generated from pure computation to quantify errors related to uncertain data.

Why do this? Because there are cases where analytical methods are unreliable or fully unknown.

Uncertainty definitions

Definition

Consider an abstract differential problem: Find $u \in V$ such that

$$\mathcal{A}u = f.$$

The parameters $\mathcal{A}$ and f are usually not fully known but we only know that $(\mathcal{A}, f) \in \mathcal{D}$. The set $\mathcal{D}$ is called the *set of admissible data*.

Definition

We define a solution mapping $\mathcal{S} : \mathcal{D} \to V$. This mapping takes an input $(\mathcal{A}_x, f_x) \in \mathcal{D}$ and outputs the exact solution $u_x \in V$ of the related problem $\mathcal{A}_x u = f_x$. The *Solution Set* of the problem is then the image $\mathcal{S}(\mathcal{D})$.

Example: Stationary Diffusion

Find u such that

$$\begin{aligned} \text{div } A\nabla u + f = 0, &\quad \text{in } \Omega, \\ u = u_0 &\quad \text{on } \Gamma_1, \\ n \cdot A\nabla u = F &\quad \text{on } \Gamma_2, \end{aligned}$$

where the coefficients A and f are not fully known but rather belong to admissible sets

$$\begin{aligned} A \in \mathcal{D}_A &:= \{A = A_\circ + \delta_1 g_1 \ : \ ||g_1||_\infty \leq 1\}, \\ f \in \mathcal{D}_f &:= \{f = f_\circ + \delta_2 g_2 \ : \ ||g_2||_\infty \leq 1\}, \end{aligned}$$

where $A_\circ$ and $f_\circ$ are some given "central" elements of the datasets and $\delta_i \geq 0$ are the maximum perturbations from them. The full dataset can be denoted $\mathcal{D} := \mathcal{D}_A \times \mathcal{D}_f$.

Diameter of the Solution Set

We want to somehow quantify the ”size” of the solution set. For this purpose we define the norm diameter.

Definition

The diameter of the solution set $\mathcal{S}(\mathcal{D})$ is

$$\text{Diam}(\mathcal{S}(\mathcal{D})) := \sup_{u_1,u_2\in\mathcal{S}(\mathcal{D})} |||u_1 - u_2|||,$$

where $|||\cdot|||$ is a suitable norm in the (Banach) space V.

Diameter of the Solution Set

$\text{Diam}(\mathcal{S}(\mathcal{D}))$ is an important quantity for two reasons.

1. It gives us an idea of whether or not our measurements are sufficiently accurate.
2. When we approximate an exact solution u by some v and have a way of estimating the approximation error $|||u - v|||$ it gives us an accuracy limit on this error.

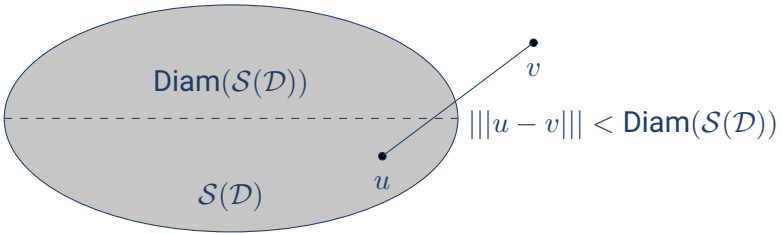


Figure: If v is the approximation of u, in this case v could already be inside the solution set! There is no point in sharpening approximation.

Analytical methods for quantifying Diam$(\mathcal{S}(\mathcal{D}))$

In the past uncertainty in models has been controlled in various ways such as the probabilistic method [1] and the worst case scenario method [2].

In recent decades analytical upper and lower bounds for Diam$(\mathcal{S}(\mathcal{D}))$ have been derived for many ordinary and partial differential equations ([3] and [4]). In suitable situations these bounds can be used to effectively quantify the diameter. However there are still many problems for which such bounds are very coarse or fully unknown.

[1] G.I. Schueller, *A state-of-the-art report on computational stochastic mechanics*, Probab. Eng. Mech. 1997

[2] I. Hlaváécek, J. Chleboun, and I. Babuéska , *Uncertain input data problems and the worst scenario method*, Elsevier, Amsterdam, 2004

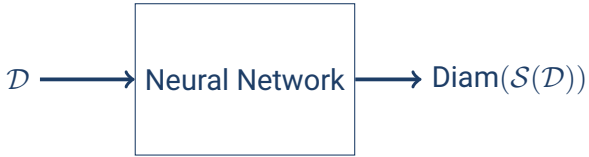
[3] Mali, Neittaanmäki, Repin, *Accuracy Verification Methods*, Springer, 2014.

[4] Repin, *A Posteriori Estimates for Partial Differential Equations*, De Gruyter, 2008.

Machine learning model for estimating Diam$(\mathcal{S}(\mathcal{D}))$

Instead of relying on analytical methods, we want to create a machine learning model which can approximate Diam$(\mathcal{S}(\mathcal{D}))$ by using training data generated by brute force computations. We want the model to take as input the admissible dataset $\mathcal{D}$ and output the diameter Diam$(\mathcal{S}(\mathcal{D}))$.

We will show that a dense neural network can perform in quantifying the diameter as well as analytical bounds in our example case.



Problem setting for our model

The model is trained to approximate Diam$(\mathcal{S}(\mathcal{D}))$ of uncertain problems with the form

$$(\alpha(x)u'(x))' - \beta(x)u(x) = f(x),$$
$$u(0) = 0, \quad u(1) = 0, \quad 0 < \alpha(x) < \overline{\alpha}, \quad 0 < \beta(x) < \overline{\beta}, \quad f \in L^2(0,1)$$

where we have three sources of indeterminacy

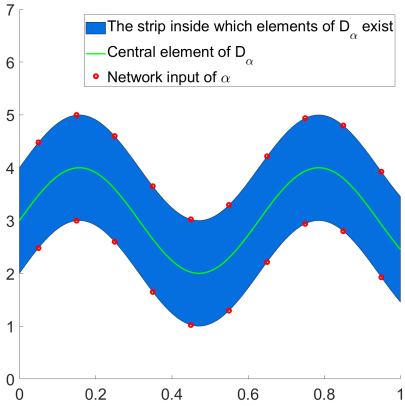
$$\alpha \in \mathcal{D}_\alpha = \{\alpha = \alpha_\circ + \delta_1 g_1 \ : \ ||g_1||_\infty \leq 1\},$$
$$\beta \in \mathcal{D}_\beta = \{\beta = \beta_\circ + \delta_2 g_2 \ : \ ||g_2||_\infty \leq 1\},$$
$$f \in \mathcal{D}_f = \{f = f_\circ + \delta_3 g_3 \ : \ ||g_3||_\infty \leq 1\},$$

where $\alpha_\circ, \beta_\circ$ and $f_\circ$ are the central elements and $\delta_i \geq 0$ are the maximum perturbations. Now $\mathcal{D} := \mathcal{D}_\alpha \times \mathcal{D}_\beta \times \mathcal{D}_f$. The norm used in this case is the energy norm

$$|||u||| = \left(\int_0^1 \alpha_\circ |u'|^2 + \beta_\circ |u|^2 \, dx \right)^{\frac{1}{2}}$$

Generating training data: input description

We describe $\mathcal{D}$ by a vector $\hat{\mathcal{D}}$ which acts as a set of input parameters. The vector $\hat{\mathcal{D}}$ is formed by the maximum and minimum values of the coefficients α, β and f at a given number of points. Figure below illustrates looking at $\mathcal{D}_\alpha$ at 10 points in the domain.



Generating training data: input description

For the full set $\mathcal{D}$ we must do the same for $\mathcal{D}_\beta$ and $\mathcal{D}_f$ so we would have in total 60 inputs. Our neural network would have 60 neurons on the input layer. Note that the amount of "looking points" is completely at our disposal. For our example model in this presentation we have used 4 points.

Generating training data: calculating output

For a given input $\mathcal{D}$ we numerically approximate the correct output $\mathrm{Diam}(\mathcal{S}(\mathcal{D}))$. We call this the **brute-force method**.

Numerical approximation is done by taking a subset of 300 random elements (which are weighted towards high indeterminacies) $U := \{u_1, u_2, \cdots, u_{300}\} \in \mathcal{S}(\mathcal{D})$ and computing the quantity

$$\mathrm{Diam}(U) := \sup_{v,w \in U} |||v - w||| \approx \mathrm{Diam}(\mathcal{S}(\mathcal{D})).$$

This is generally a very expensive way to compute the diameter.
When we calculate the quantity Approx we could also use any different norm we like. In analytical methods we are restricted to specific norms (namely the H^1-norm).

Constraints for our training examples

We use a special case of possible training inputs $\mathcal{D}$. The central functions and indeterminacy parameters are piecewise constant such that

$$\begin{aligned} \alpha_\circ(x) &= \alpha_{\circ i}, \quad \text{when} \quad x \in [x_{i-1}, x_i], \\ \beta_\circ(x) &= \beta_{\circ i}, \quad \text{when} \quad x \in [x_{i-1}, x_i], \\ f_\circ(x) &= f_{\circ i}, \quad \text{when} \quad x \in [x_{i-1}, x_i], \\ \delta_j(x) &= \delta_{ji}, \quad \text{when} \quad x \in [x_{i-1}, x_i]. \end{aligned} \tag{1}$$

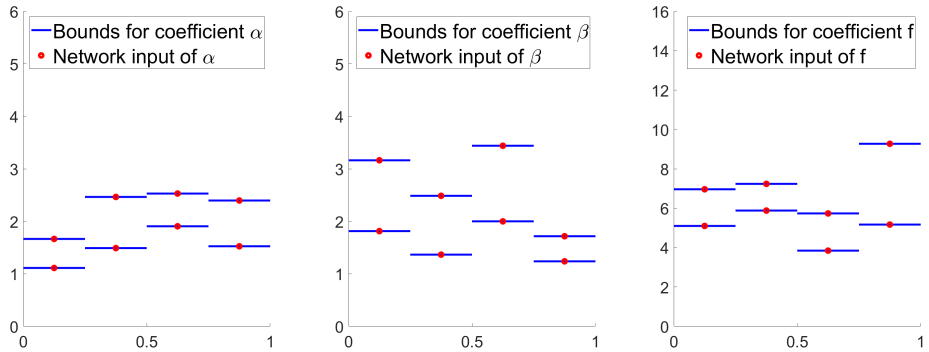
where $[x_0, x_1, x_2, x_3, x_4] = [0, 0.25, 0.5, 0.75, 1]$. The mean elements and indeterminacies are given range constraints

$$\alpha_\circ(x) \in [1, 3], \quad \beta_\circ(x) \in [1, 3], \quad f_\circ(x) \in [3, 10] \tag{2}$$

$$\delta_1(x) \leq 0.3\alpha_\circ(x), \quad \delta_2(x) \leq 0.3\beta_\circ(x), \quad \delta_3(x) \leq 0.3f_\circ(x)$$

Training data examples: input 1

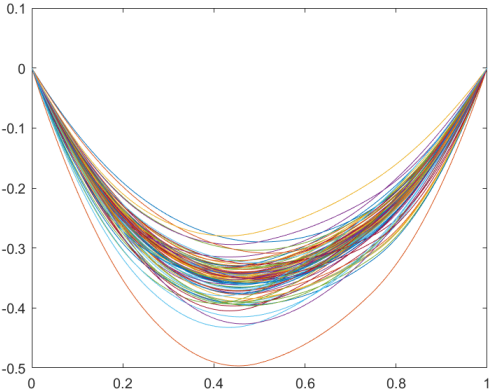
Admissible Dataset $\mathcal{D}_1$



Network Input: $\hat{\mathcal{D}}_1 = 1.1, 1.5, 1.9, 1.5, 1.7, 2.5, 2.5, 2.4, 1.8, 1.4, 2.0, 1.2,$
$3.2, 2.5, 3.4, 1.7, 5.1, 5.9, 3.8, 5.2, 7.0, 7.2, 5.7, 9.3$

Training data examples: output 1

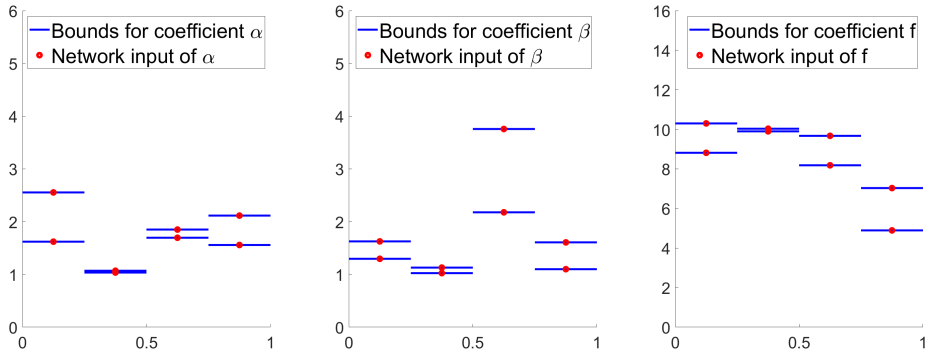
Part of the solution set $\mathcal{S}(\mathcal{D}_1)$



Target Output 1: $0.7556 \approx \text{Diam}(\mathcal{S}(\mathcal{D}_1))$.

Training data examples: input 2

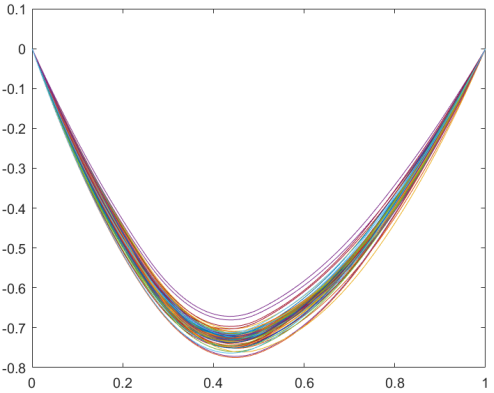
Dataset $\mathcal{D}_2$



Network Input 2: $\hat{\mathcal{D}}_1 = 1.6, 1.0, 1.7, 1.6, 2.6, 1.1, 1.9, 2.1, 1.3, 1.0, 2.2, 1.1,$
$1.6, 1.1, 3.8, 1.6, 8.8, 9.9, 8.2, 4.9, 10.3, 10.0, 9.7, 7.0$

Training data examples: output 2

Solution cloud $\mathcal{S}(\mathcal{D}_2)$



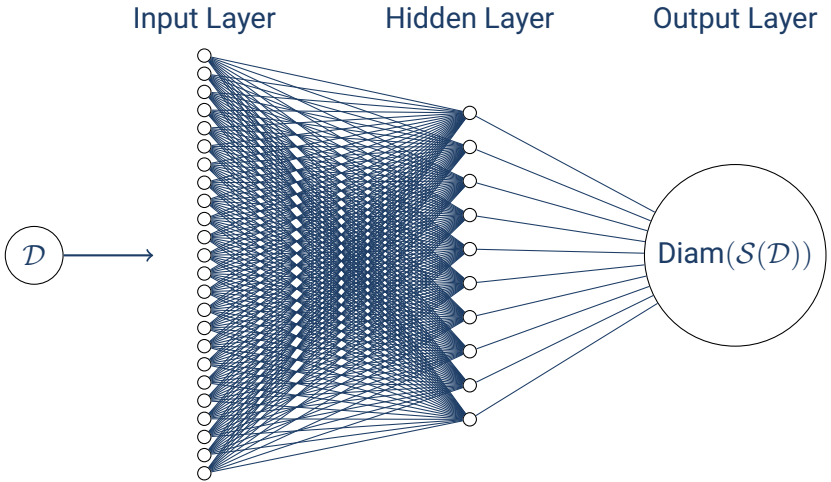
Target Output 2: $0.4190 \approx \text{Diam}(\mathcal{S}(\mathcal{D}_2))$.

Model architecture

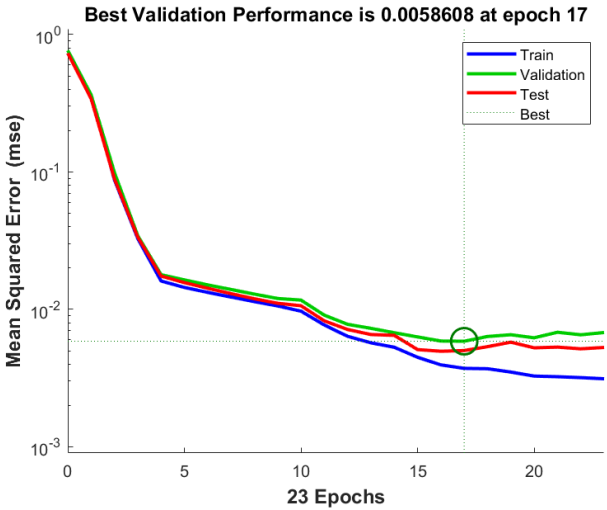
We used the Deep Learning toolbox of Matlab for training. The architechture that was found to work well for this example is a very small network containing one hidden layer with 10 neurons using sigmoid activation. The training algorithm that was used is the Levenberg-Marquardt algorithm.

For our training data we generated 2000 examples of which 70% was used for training, 15% for validation and 15% for testing.

Model architecture

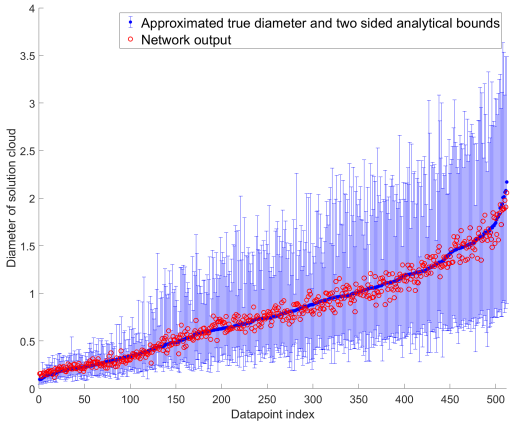


Model training progression



Neural network versus analytical bounds

Using a separate set of 500 test examples we find that the network produces results which are mostly closer to the approximated diameter then the analytical bounds.



Conclusion

- With powerful computational resources we can train a neural network to approximate the expensive brute force method.
- Analytical methods can be used in simple problems but still require understanding of mathematics.
- The objective is to advance this method into more complicated PDE's for which analytical methods are not an option.

Thank you for attention!