Surrogate Models for Particle Accelerators

Andreas Adelmann

- Motivation
- Polynomial Chaos Expansion (PCE)
- Artificial Neural Nets (ANNs)
References

A. Adelmann, On Nonintrusive Uncertainty Quantification and Surrogate Model Construction in Particle Accelerator Modeling, SIAM/ASA Journal on Uncertainty Quantification, 7(3), 2019

N. Wiener, The homogenous chaos, Amer. J. Math. 30, 897-936, (1938)


Motivation

Motivation

\[ \frac{\partial}{\partial t} \circ \partial = 0 \]

Reduction mathematically

Polynomial Expansion \((n \to \infty)\)

High-Dimensional Model Representation (PCe, Sobol', ANOVA exact)

Artifical Neural Nets (ANNs)

Surrogate Models for Particle Accelerators
Motivation

\[ \frac{\partial o}{\partial t} = 0 \]

Reduction mathematically

Polynomial Expansion \( (n \rightarrow \infty) \)

High-Dimensional Model Representation (PCe, Sobol', ANOVA exact)

Artifical Neural Nets (ANNs)

3D $\rightarrow$ 2D
Motivation

Reduction mathematically

- Polynomial Expansion \((n \to \infty)\)
- High-Dimensional Model Representation (PCE, Sobol’, ANOVA exact)
- Artificial Neural Nets (ANNs)
Motivation

The diagram illustrates the trade-off between costs and error for high-fidelity models compared to low-fidelity models. The high-fidelity model is positioned at the top right, indicating higher costs and lower error. As we move down and to the right, the models become lower fidelity, with decreasing costs but increasing error. This visual representation helps in understanding the balance between accuracy and resource usage in modeling particle accelerators.
Motivation

Weak vs. Strong Scaling
Motivation
Weak vs. Strong Scaling

![Graph showing execution time vs. number of processors]
Surrogate Model a Simple Definition

Surrogate models (SMs) approximate a computationally expensive simulator $\eta$. Suppose

$$y(x) = \eta(x), \quad x \in \mathbb{R}^n, \quad y \in \mathbb{R}^m$$

then the SM is an approximation of the form

$$\hat{y}(x) = \hat{\eta}(x)$$

such that

$$y(x) = \hat{y}(x) + \varepsilon$$

and $\hat{y}(x)$ cheap to evaluate.
A Complicated Example (PCe)
[AA, On Nonintrusive UQ and SM Construction ... (2019)]

- **Goal:** model halo \( h_x = \frac{\langle q_x^4 \rangle}{\langle q_x^2 \rangle^2} - C & \tilde{x} \)

- **Simplification:** 3 design parameters
  1. initial condition: \( \langle xp_x \rangle \)
  2. collimator setting: \( \Delta C_1 \)
  3. rf phase setting: \( \phi_1 \)
A Complicated Example (PCe)

[AA, On Nonintrusive UQ and SM Construction ... (2019)]

**Goal:** model halo \( h_x = \frac{\langle q_x^4 \rangle}{\langle q_x^2 \rangle^2} - C & \tilde{x} \)

**Simplification:** 3 design parameters

1. initial condition: \( \langle xp_x \rangle \)
2. collimator setting: \( \Delta C_1 \)
3. rf phase setting: \( \phi_1 \).

This extensive search in the 3 dimensional parameter space requires PIC models with enough particles to estimate halo at a given location.
Illustration of the Basic Ideas (P Ce)

Let $\eta$ be the simulator

Accelerator Lattice $\xrightarrow{}$ $\eta(\cdot)$$\xrightarrow{}$ Qols

Random variables:
1. initial condition: $(x_{p0})$
2. collimator setting: $\Delta C_1$
3. rf phase setting: $\phi_1$

Design Variables (DVars)

$u^n = \eta(\xi^n)$
Polynomial Chaos Expansions (PCE) 1

All square integrable, second-order random processes with finite variance output, \( y(\xi) \in L_2(\Omega, \mathcal{F}, \mathcal{P}) \), can be written as [N. Wiener]

\[
y = \sum_{k=0}^{\infty} \alpha_k \Psi_k(\xi).
\]

- \( y \): Random Variable (RV)
- \( \alpha_k \): PC coefficients (deterministic)
- \( \Psi_k \): Hermite polynomial, \( \xi \): Gaussian RV

Expansion in terms of functions of random variables multiplied with deterministic PC coefficients.
Polynomial Chaos Expansions (PCE) I

[AA, On Nonintrusive UQ and SM Construction ... (2019)]

**Algorithm:** generate for each design variable, a PC surrogate model to order $K$

1. generate $N$ samples ($\xi^n$) according to the sampling strategy of interest
2. create the **deterministic** training points with high fidelity simulations (non-intrusive)

$$u^n = \eta(\xi^n).$$

3. solve for $\alpha_k$ via
   - orthogonal Galerkin-projection
   - regression methods
   - Bayesian Compressive Sensing
Given the computed $\alpha_k$ values one assembles $\hat{\eta}$

$$
\hat{\eta}(\xi) = \sum_{k=0}^{K-1} \alpha_k \Psi_k(\xi)
$$

$$
S(\xi) = \frac{\sum_{k \in I} \alpha^2_k}{\sum_{k=0}^{K-1} \alpha^2_k}
$$

**Surrogate Model** $\hat{\eta}$

**Global Sensitivity** $S$
Predictions - $h_x$

$I = 5$ mA

Polynomial Surrogate vs. High Fidelity OPAL Simulation

- $P=4$
- $P=3$
- $P=2$
- $y=x$
Predictions of $\tilde{x}$ with 95% CL
Predictions of $\tilde{x}$ with 95% CL
Sensitivities

![Diagram showing sensitivities to various parameters such as $E_{\text{kin}}$, $\tilde{E}$, $\Delta E$, $h_5$, and $h_{10}$ with corresponding color codes: blue for $x p_x$, green for $C_1$, and red for $\Delta \phi_{\text{rf}}$.](image)
MOGA for the Argonne Wakefield Accelerator
[N. Neveu, AA, et al. (2019)]

- Full 3D Start to End (S2E) needed
- OPAL Particle In Cell (PIC) model
- Very timeconsuming
- Parameter study / multi-objective optimisation expensive
MOGA for the Argonne Wakefield Accelerator
[N. Neveu, AA, et al. (2019)]

- One 3D medium fidelity S2E 3600 (s) on 32 cores
- 3...7 Qols, 6...15 Dvars
- Genetic Algorithm setup: \( G = 200, I = 100 \)
MOGA for the Argonne Wakefield Accelerator
[N. Neveu, AA, et al. (2019)]

OPAL MOGA: 24h on $\approx 5000$ cores
Machine Learning to Construct a cheap & accurate SM
[A. Edelen et al (2019)]

- optimise parameters at a given location
- One 3D S2E 300 (s) on 8 cores
- 7 Qols, 7 Dvars
- MOGA (in OPAL): $G = 200, I = 100 \Rightarrow$ ground truth
4 Step Process to Construct an ANN SM

1. generate random sample
2. split labeled data set (80%, 20%)
3. create ANN
4. understand quality
Artificial Neural Network

- Fully connected and feed forward
- Hyperparameters
  - A lot of different architectures
  - Learning rate
- Best results using
  - 6-12-24-48-96-8
  - Adam optimizer with 0.0001 learn rate, trained for 30k epochs
  - Tanh as activation, no activation after output layer
  - Weights inverse proportional to the estimated density likelihood

Figure: Neural Network scheme
https://towardsdatascience.com
Fidelity on the Test Data I

![Graph showing ΔE vs Sample](image-url)
When all comes together ....
Take Home Points

**OPAL MOGA**: 24h on ≈ 5000 cores
Take Home Points

OPAL MOGA: 24h on $\approx 5000$ cores

Train ANN once: $2-5$h on $\approx 128$ cores
Take Home Points

OPAL MOGA: 24h on $\approx 5000$ cores

Train ANN once: $2 - 5h$ on $\approx 128$ cores

ANN & MOGA: $\approx 30$ minutes $\Rightarrow$
Take Home Points

**OPAL MOGA:** 24h on $\approx 5000$ cores

Train ANN once: $2 - 5h$ on $\approx 128$ cores

ANN & MOGA: $\approx 30$ minutes $\Rightarrow$
Take Home Points

**OPAL MOGA**: 24h on $\approx 5000$ cores

Train ANN once: $2 - 5h$ on $\approx 128$ cores

ANN & MOGA: $\approx 30$ minutes $\Rightarrow$

**Speedup** $> 1\,000\,000$ & accurate
Take Home Points

- Surrogate Models are the only way to achieve real-time performance & accuracy in complicated system!

- ANN & PCe are wonderful tools to achieve this

- Much to learn robustness, training sizes, & accuracy