Global Sensitivity Analysis in High Dimensional Parameter Spaces

A Tensor-Network Approach

Hongli Zhao, University of Chicago CCAM, IL, USA

Daniel Tartakovsky, Stanford ERE, CA, USA

May 2023

Global Sensitivity Analysis

- Forward model:

$$\mathbf{x} \mapsto f(\mathbf{x}) \tag{1}$$

  where:
  - $\mathbf{x} = (x_1, \ldots, x_d) \in K^d$ unit hypercube. Components are assumed to be uniformly random and independent.
  - f is assumed to be real-valued and square-integrable.
- Would like to quantify relative importance of components of $\mathbf{x}$.
  - Guide design of numerical simulations
  - Identify chaotic parameter regimes in dynamical systems

- f admits a unique decomposition (Sobol' 1993) :

$$f(x_1,\ldots,x_d) = f_0 + \sum_{i=1}^{d} f_i(x_i) + \sum_{1\le i<j\le d} f_{ij}(x_i,x_j) + \cdots + f_{1,\ldots,d}(x_1,\ldots,x_d) \quad (2)$$

with:

$$\begin{cases} f_0 = \mathbb{E}\left[f(\mathbf{x})\right] \\ f_i(x_i) = \mathbb{E}\left[f(\mathbf{x})|x_i\right] - f_0 \\ f_{ij}(x_i,x_j) = \mathbb{E}\left[f(\mathbf{x})|x_i,x_j\right] - f_i(x_i) - f_j(x_j) - f_0 \\ \ldots \end{cases} \quad (3)$$

- By construction for multi-indices $\mathcal{I},\mathcal{J}$:

$$\mathbb{E}\left[f_{\mathcal{I}}\right] = 0, \mathbb{E}\left[f_{\mathcal{I}} f_{\mathcal{J}}\right] = 0 \quad (4)$$

such that:

$$\mathrm{Var}[f] = \sum_{i=1}^{d} \mathrm{Var}[f_i] + \sum_{1\le i<j\le d} \mathrm{Var}[f_{ij}] + \cdots + \mathrm{Var}[f_{1,\ldots,d}] \quad (5)$$

and:

$$1 = \sum_{i=1}^{d} \frac{\mathrm{Var}[f_i]}{\mathrm{Var}[f]} + \sum_{1\le i<j\le d} \frac{\mathrm{Var}[f_{ij}]}{\mathrm{Var}[f]} + \cdots + \frac{\mathrm{Var}[f_{1,\ldots,d}]}{\mathrm{Var}[f]} \quad (6)$$

- Define: $S_{\mathcal{I}} = \frac{\mathrm{Var}[f_{\mathcal{I}}]}{\mathrm{Var}[f]}$.

Sobol' indices via Polynomial Chaos Expansion

- In the absence of an analytic model f, we must resort to Monte Carlo integration to approximate $S_{\mathcal{I}}$, which is computationally demanding when $|\mathcal{I}|$ is large.
- As a Monte Carlo approximation, $\widehat{S_{\mathcal{I}}}$ could be negative.
- (Karniadakis 2003) Approximate $y = f(\mathbf{x})$ with a truncated series of orthonormal basis functions:

$$y_{\text{PCE}} = \sum_{i_1,\ldots,i_d} \mathcal{C}_{i_1,\ldots,i_d} \Phi_{i_1,\ldots,i_d}(x_1,\ldots,x_d) \tag{7}$$

where the multivariate basis can be constructed as a product of 1d basis functions (e.g. Legendre polynomials):

$$\Phi_{i_1,\ldots,i_d}(x_1,\ldots,x_d) = \prod_{j=1}^{d} \phi_{i_j}(x_j)$$

in particular:

$$\mathbb{E}\left[\Phi_{\mathcal{I}}\Phi_{\mathcal{J}}\right] = 0, \mathbb{E}\left[\Phi_{\mathcal{I}}^2\right] = 1$$

- (Sudret 2007) May establish connections between the PCE expansion and Sobol' indices.

$$\mathbb{E}[y_{\text{PCE}}] = \sum_{i_1,\dots,i_d} \mathcal{C}_{i_1,\dots,i_d} \mathbb{E}[\Phi_{i_1,\dots,i_d}] = \mathcal{C}_{0,\dots,0} \tag{8}$$

$$\begin{aligned} \text{Var}[y_{\text{PCE}}] &= \mathbb{E}[y_{\text{PCE}}^2] - \mathbb{E}[y_{\text{PCE}}]^2 \\ = \sum_{i_1,\dots,i_d} \mathcal{C}^2_{i_1,\dots,i_d} \int_{K^d} \Phi^2_{i_1,\dots,i_d} d\mathbf{x} &- \mathcal{C}^2_{0,\dots,0} \\ = \sum_{i_1,\dots,i_d} \mathcal{C}^2_{i_1,\dots,i_d} &- \mathcal{C}^2_{0,\dots,0} \end{aligned}$$

$$\begin{gathered} \mathbb{E}[y_{\text{PCE}}|x_j] = \sum_{i_1,\dots i_d} \mathcal{C}_{i_1,\dots,i_d} \int_{K^{d-1}} \Phi_{i_1,\dots,i_d}(\mathbf{x}_{\setminus j}, x_j) d\mathbf{x}_{\setminus j} = \sum_{i_j} \mathcal{C}_{0,\dots,0,i_j,0,\dots,0} \phi_{i_j}(x_j) \\ \text{Var}[\mathbb{E}[y_{\text{PCE}}|x_j]] = \sum_{i_j} \mathcal{C}^2_{0,\dots,i_j,0,\dots,0} - \mathcal{C}^2_{0,\dots,0} \end{gathered} \tag{9}$$

- Likewise:

$$\text{Var}[\mathbb{E}[y_{\text{PCE}}|\mathbf{x}_{\mathcal{I}}]] = \sum_{\mathcal{I}} \mathcal{C}^2_{0,\dots,\mathcal{I},\dots,0} - \mathcal{C}^2_{0,\dots,0}$$

- Variable dependence is captured in $\mathcal{C}$ $\rightarrow$ How to find $\mathcal{C}$?

- Galerkin projection:

$$\mathcal{C}_{i_1,\ldots,i_d} = \mathbb{E}\left[f\Phi_{i_1,\ldots,i_d}\right] \approx \sum_{j_1=1}^{n_1} \cdots \sum_{j_d=1}^{n_d} w_{i_1} \cdots w_{i_d} f(x_{j_1}, \ldots, x_{j_d}) \Phi_{i_1,\ldots,i_d}(x_{j_1}, \ldots, x_{j_d})$$

- Regression:

$$\widehat{\mathcal{C}} = \underset{\mathcal{C}}{\operatorname{argmin}} \frac{1}{M} \sum_{k=1}^{M} \left(y_k - \sum_{i_1,\ldots,i_d} \mathcal{C}_{i_1,\ldots,i_d} \Phi_{i_1,\ldots,i_d}(\mathbf{x}_k) \right)^2 + \lambda \|\mathcal{C}\|_F^2$$

  for queried points $\{(\mathbf{x}_k, y_k)\}_{k=1}^{M}$.
- In either case, $O(n^d)$ complexity is incurred.

Tensor-Train (TT) Format

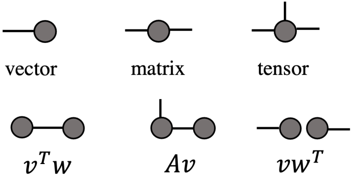
- $\mathcal{C}$ is a d-dimensional tensor, the *tensor-train* format gives the following tensor decomposition:

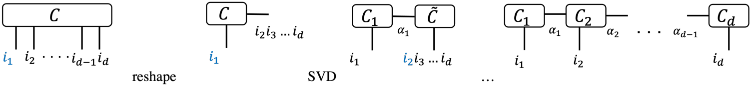
$$\mathcal{C}[i_1, \dots, i_d] \approx \mathcal{C}_1[1, i_1, :] \cdot \mathcal{C}_2[:, i_2, :] \cdots \mathcal{C}_d[:, i_d, 1]$$

$$= \sum_{\alpha_1=1}^{r_1} \sum_{\alpha_2=1}^{r_2} \cdots \sum_{\alpha_{d-1}=1}^{r_{d-1}} \mathcal{C}_1[\alpha_0, i_1, \alpha_1] \mathcal{C}_2[\alpha_1, i_2, \alpha_2] \cdots \mathcal{C}_d[\alpha_{d-1}, i_d, \alpha_d]$$

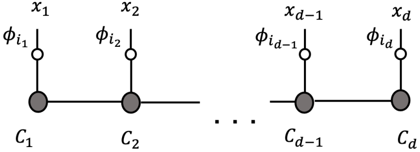
with $\alpha_0 = \alpha_d = 1$. $(r_1, \dots, r_{d-1})$ are the TT ranks.

- Tensor diagrams:





- With the TT decomposition of $\mathcal{C}$, the PCE model is now:



- Allows continuous evaluations as a surrogate model.
- If r is low, the complexity is now $O(dnr^2)$.

Gradient-based optimization

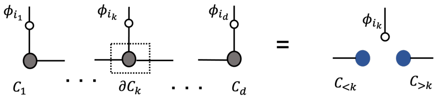
- Define loss function:

$$\mathcal{L}(\mathcal{C}_1,\ldots,\mathcal{C}_k) = \frac{1}{M}\sum_{i=1}^{M}\left(y_i - \sum_{i_1,\ldots,i_d}[\mathcal{C}_1[1,i_1,:]\cdots\mathcal{C}_d[:,i_d,1]\Phi_{i_1,\ldots,i_d}(\mathbf{x}_k)\right)^2 \quad (10)$$

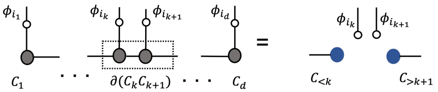
- Initialize with prespecified ranks
- In each iteration, compute $\frac{\partial\mathcal{L}}{\partial\mathcal{C}_k}$ and optimize each TT core $\mathcal{C}_k$ by:

$$\mathcal{C}_k^{(t+1)} \leftarrow \mathcal{C}_k^{(t)} - \eta\left(\frac{\partial\mathcal{L}}{\partial\mathcal{C}_k^{(t)}}\right)$$

- Gradient-descent:



- Two-site strategy (Stoudemire 2016, NeurIPS):

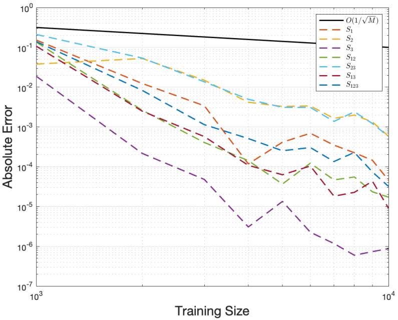


- TT ranks can be adapted by applying SVD after every k iterations

Example 1: Ishigami Function

The Ishigami function is defined in 3 dimensions as:

$$y = f(\mathbf{x}) = \sin(x_1) + 7\sin^2(x_2) + 0.1x_3^4\sin(x_1), \mathbf{x} \in [-\pi, \pi]^3$$



The detailed comparison of first-order indices is as follows:

Index	Analytic	FTT
S_1	0.3138	0.3139
S_2	0.4424	0.4423
S_3	0	2.163×10^{-6}

S_{12}	0	8.731×10^{-7}
S_{23}	0	1.716×10^{-5}
S_{13}	0.2431	0.2437
S_{123}	0	3.204×10^{-5}

Example 2: Sobol' Function ($d = 8$)

The Sobol' function is a well-known test problem in GSA with decaying first-order indices, defined as the following:

$$y = f(\mathbf{x}) = \prod_{i=1}^{d} \frac{|4x_i + 2| + a_i}{1 + a_i}$$

where $\mathbf{a} = [a_1, \cdots, a_8] = [1, 2, 5, 10, 20, 50, 100, 500]$, and supported on $[0, 1]^8$. The Sobol' indices can be determined from the following formulae:

$$D = \prod_{i=1}^{d} (D_i + 1) - 1$$

where $D_i = \frac{1}{3(a_i+1)^2}$, and $S_i = D_i / D$.

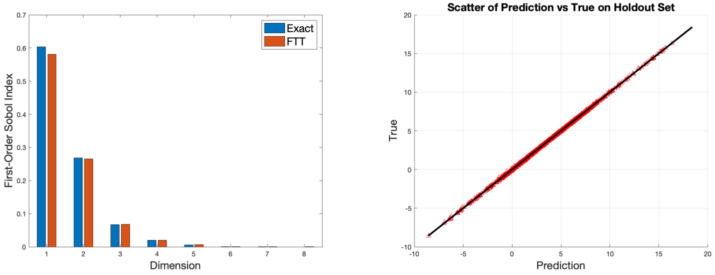


Figure: 5×10^3 data points, final training $MSE = 4.1914 \times 10^{-6}$, in 4324 iterations.

The detailed comparison of first-order indices is as follows:

Index	Analytic	FTT
S_1	0.6037	0.5814
S_2	0.2683	0.2650
S_3	0.0671	0.0677
S_4	0.02	0.0197
S_5	0.0055	0.00631
S_6	0.0009	0.0010
S_7	0.0002	0.00025
S_8	0	2.238×10^{-5}

Example 3: Doyle-Fuller-Newman battery discharge time ($d = 14$)

- Baseline parameters are taken from Chen et al. (2020)

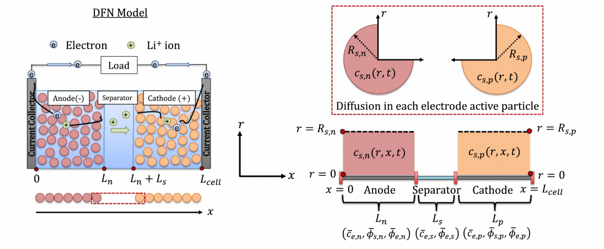


Figure: Visual chart of DFN model (Onori 2019)

- 14 parameters were investigated by varying around baseline values $\pm 5\%$, with cutoff voltage at $2.7V$ and discharge time recorded. Data points were simulated using the COMSOL framework.

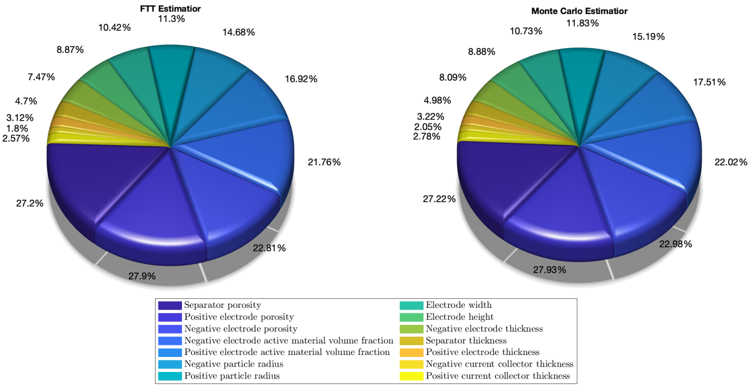
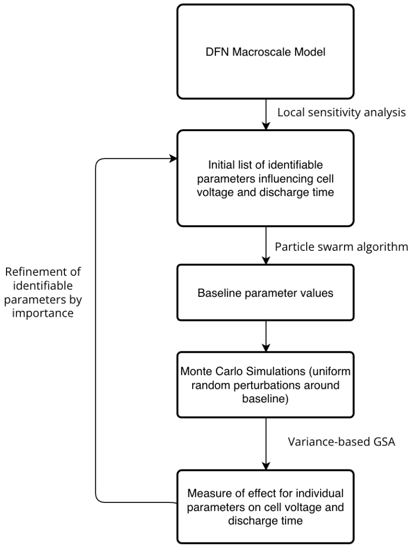


Figure: Left: FTT emulator fitted with 2×10^4 data points. Right: MC estimator using 1.3×10^5 points.


DFN Macroscale Model
Local sensitivity analysis
Initial list of identifiable parameters influencing cell voltage and discharge time
Particle swarm algorithm
Baseline parameter values
Monte Carlo Simulations (uniform random perturbations around baseline)
Variance-based GSA
Measure of effect for individual parameters on cell voltage and discharge time
Refinement of identifiable parameters by importance

Future Directions

- Ensemble estimator
  - Divide data into $\lfloor M/P \rfloor$-sized partitions and compute P emulators
    $$f^{(i)}(\mathbf{x}) = \sum_{\mathcal{I}} \mathcal{C}_{\mathcal{I}}^{(i)} \Phi_{\mathcal{I}}(\mathbf{x})$$
    and form:
    $$f(\mathbf{x}) = \frac{1}{P} \sum_{\mathcal{I}} f^{(i)}(\mathbf{x})$$
  - If computed separately:
    $$S_{\mathcal{I}} = \frac{\sum_{i=1}^{P} D_{\mathcal{I}}^{(i)}}{\sum_{i=1}^{P} D^{(i)}}$$
- Density estimation and time-dependent processes.

Thank you for your attention!

Hongli Zhao, honglizhaobob@uchicago.edu