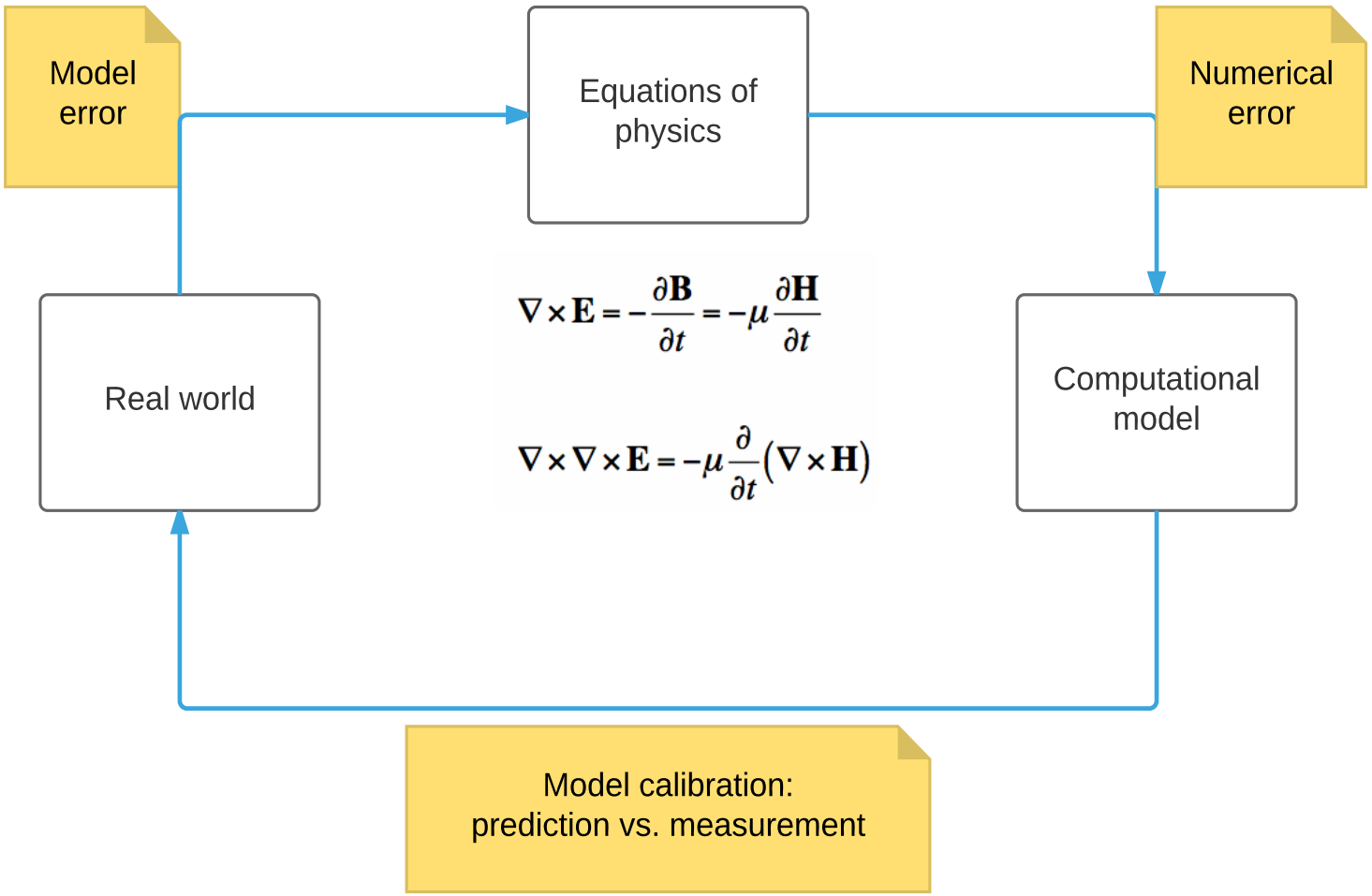
Uncertainty Quantification of the FD-FDTD Computation for Human Body

Runze Hu [1], Vikass Monebhurrun[3] Fumie Costen [1,2]
[1]University of Manchester, UK
[2]RIKEN,Wako, Saitama,351-0198, Japan
[3]CentraleSupélec, France

November 23, 2019

Outline

Modeling techniques to perfectly match reality ?



The finite-difference time-domain (FDTD) method numerically solves Maxwell equations.

Uncertainty Quantification

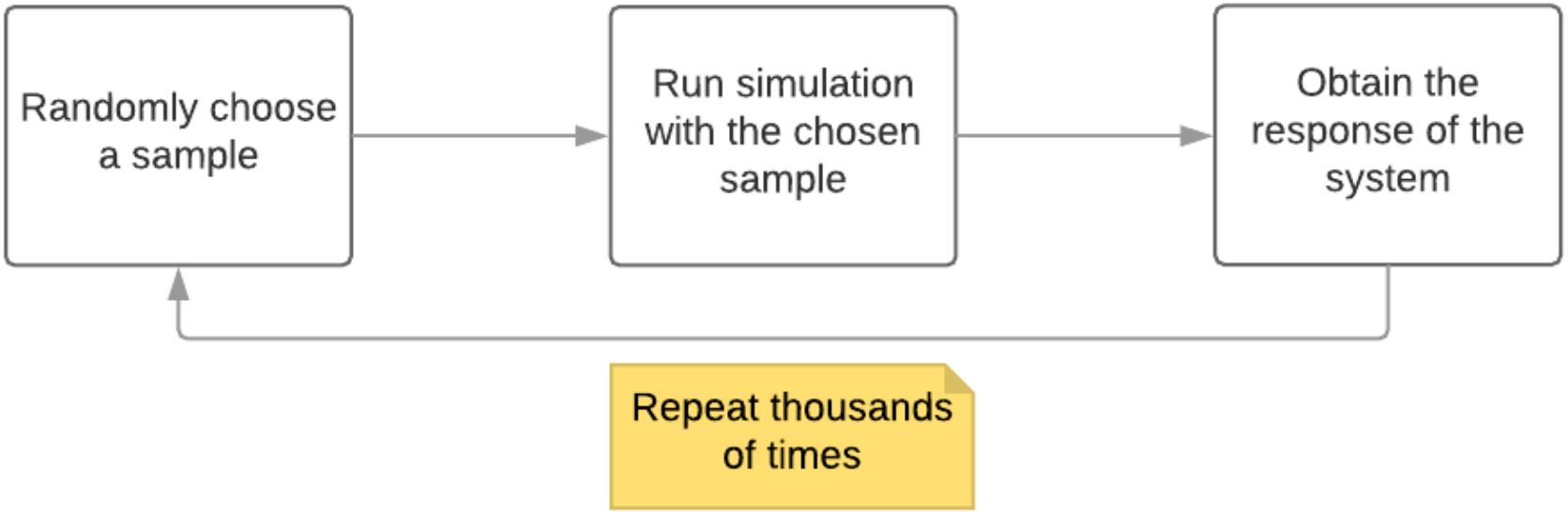
1. Two types of uncertainty
   - Aleatory uncertainty: natural randomness.
   - Epistemic uncertainty: arisen due to lack of exact knowledge, i.e.,the uncertainty of input parameters.
2. Uncertainty Quantification(UQ) : estimating
   mean,
   standard deviation,
   and probability density function of the system response.
3. UQ of the system response (FDTD observation $|E|^2$) from epistemic uncertainty
4. Input parameters: complex permittivity of human tissue

Methods for uncertainty quantification

- **Monte Carlo method (MCM)**
- **Non-intrusive polynomial chaos (NIPC) expansion method**
- **Least angle regression (LARS) method**
- **Artificial neural network (ANN)**

Key ideas of MCM

Use random samples of parameters to explore the behaviour of a complex system.



Estimate ν and σ using the first $\mathcal{M}$ system responses

$$\nu(\mathcal{M}) = \frac{1}{\mathcal{M}} \sum_{m=1}^{\mathcal{M}} \mathcal{Y}(\xi^{(m)})$$

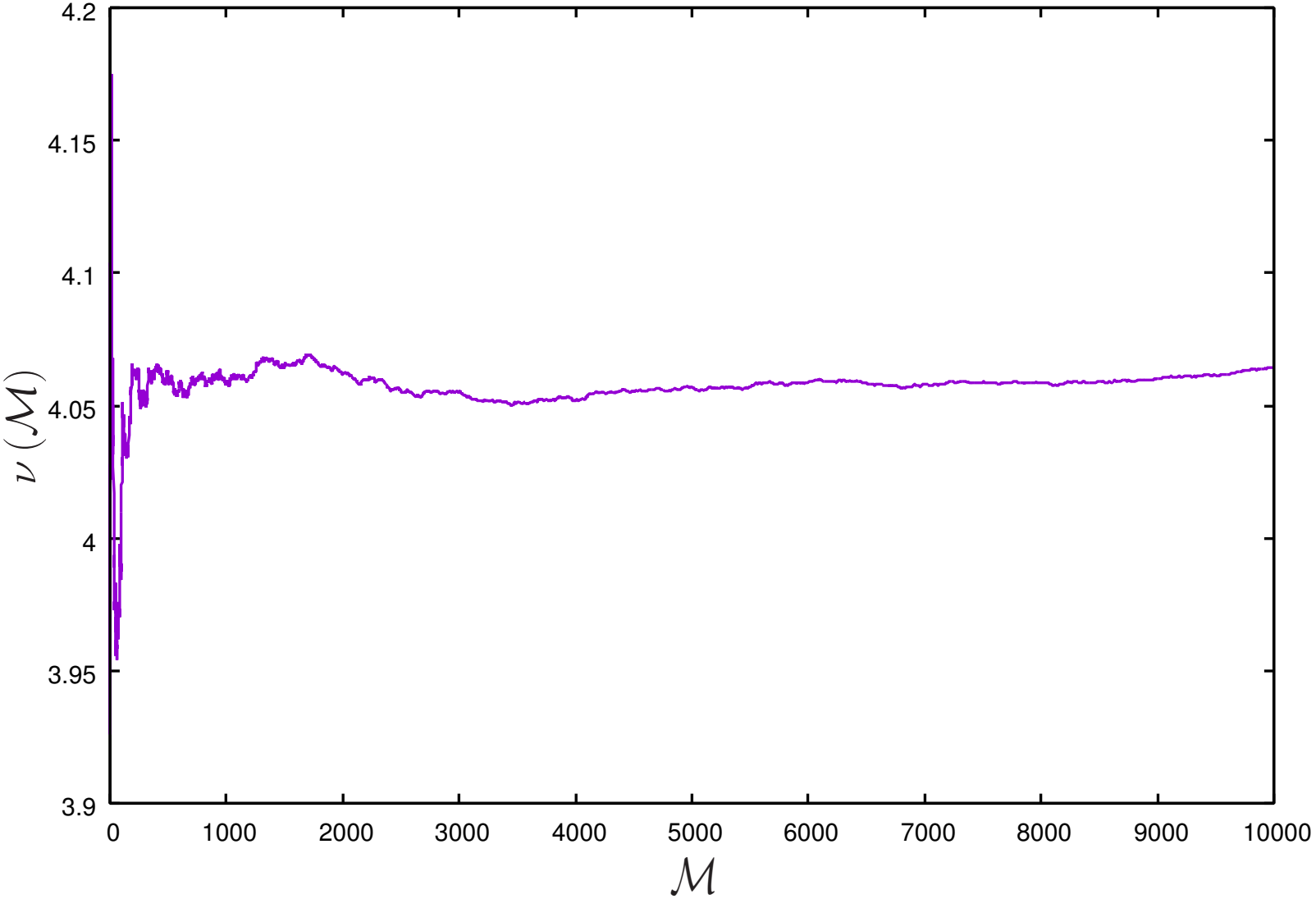
$$\sigma(\mathcal{M})^2 = \frac{1}{\mathcal{M}-1} \sum_{m=1}^{\mathcal{M}} (\mathcal{Y}(\xi^{(m)}) - \nu)^2$$

$\mathcal{M}$	**total number of samples**
ξ	**variables (Debye parameters)**
$\xi^{(m)}$	**m-th sample for ξ**
$\mathcal{Y}(\xi^{(m)})$	**system response**
$\nu(\mathcal{M})$	**mean of $\mathcal{Y}(\xi^{(m)})$**
$\sigma(\mathcal{M})^2$	**variance of $\mathcal{Y}(\xi^{(m)})$**

Example of MCM for UQ

Calculate ν of $\mathcal{Y}(\xi^{(m)})$ using the first $\mathcal{M}$ $\mathcal{Y}(\xi^{(m)})$.

$\mathcal{M} \in [2, 10000]$



Non-intrusive polynomial chaos (NIPC) expansion

- **Approximate the outputs of a system by a series of polynomials.**

$$\mathcal{T}(\xi) \approx \sum_{\ell=0}^{\mathcal{L}} a_\ell \psi_\ell(\xi) = a_0\psi_0(\xi) + a_1\psi_1(\xi) + \cdots + a_{\mathcal{L}}\psi_{\mathcal{L}}(\xi),$$

$\mathcal{T}(\xi)$: **the approximation of system response.**
$\psi_\ell(\xi)$: ℓ**-th polynomial basis for** $\ell \in [0, \mathcal{L}]$
a_ℓ : **the coefficient of** $\psi_\ell(\xi)$.

- **Calculate the coefficients based on a set of input values and the corresponding system response.**

NIPC for uncertainty quantification

- **The mean ν : $\nu = a_0$.**
- **The variance σ^2 : $\sigma^2 = \sum_{\ell=1}^{\mathcal{L}} {a_\ell}^2 \psi_\ell^2$.**
- **Ex. Hermite polynomial ψ_ℓ when random input variables to be normal distribution**

$$\psi_0(\xi_1) = 1$$
$$\psi_1(\xi_1) = \xi_1$$
$$\psi_2(\xi_1) = \xi_1^2 - 1$$

Note: Type of polynomials depends on the probability distribution of random input variables
Note: Values of ν and σ^2 depend on values of a_ℓ

Polynomial bases for multiple variables

- **Define the polynomial basis as $\psi_{\alpha_1\alpha_2\ldots\alpha_{\mathcal{K}}}(\xi_1, \xi_2, \cdots, \xi_{\mathcal{K}})$.**
- **$\alpha_1\alpha_2\cdots\alpha_{\mathcal{K}}$ is the index of the polynomial basis.**
- **The total number of polynomial bases $\mathcal{L}$ is calculated as in**

$$\mathcal{L} = \frac{(r+\mathcal{K})\,!}{r\,!\mathcal{K}\,!},$$

r: the highest order of polynomial bases, satisfying

$$\sum_{k=1}^{\mathcal{K}} \alpha_k \leq r, \alpha_k \geq 0$$

For example, when $r = 2$ and $\mathcal{K} = 2$, $\mathcal{L} = 6$. These 6 $\psi_{\alpha_1\alpha_2}(\xi_1, \xi_2)$ are presented as.

Hermite polynomials	The order of ψ
$\psi_{00}(\xi_1, \xi_2) = \psi_0(\xi_1)\psi_0(\xi_2) = 1$	0
$\psi_{10}(\xi_1, \xi_2) = \psi_1(\xi_1)\psi_0(\xi_2) = \xi_1$	1
$\psi_{01}(\xi_1, \xi_2) = \psi_0(\xi_1)\psi_1(\xi_2) = \xi_2$	
$\psi_{20}(\xi_1, \xi_2) = \psi_2(\xi_1)\psi_0(\xi_2) = \xi_1^2 - 1$	2
$\psi_{02}(\xi_1, \xi_2) = \psi_0(\xi_1)\psi_2(\xi_2) = \xi_2^2 - 1$	
$\psi_{11}(\xi_1, \xi_2) = \psi_1(\xi_1)\psi_1(\xi_2) = \xi_1\xi_2$	

Calculation of a_ℓ by regression method

Calculate a_ℓ which gives minimum difference between $\mathcal{Y}$ and $\mathcal{T}(\xi)$ based on the least squares regression method.

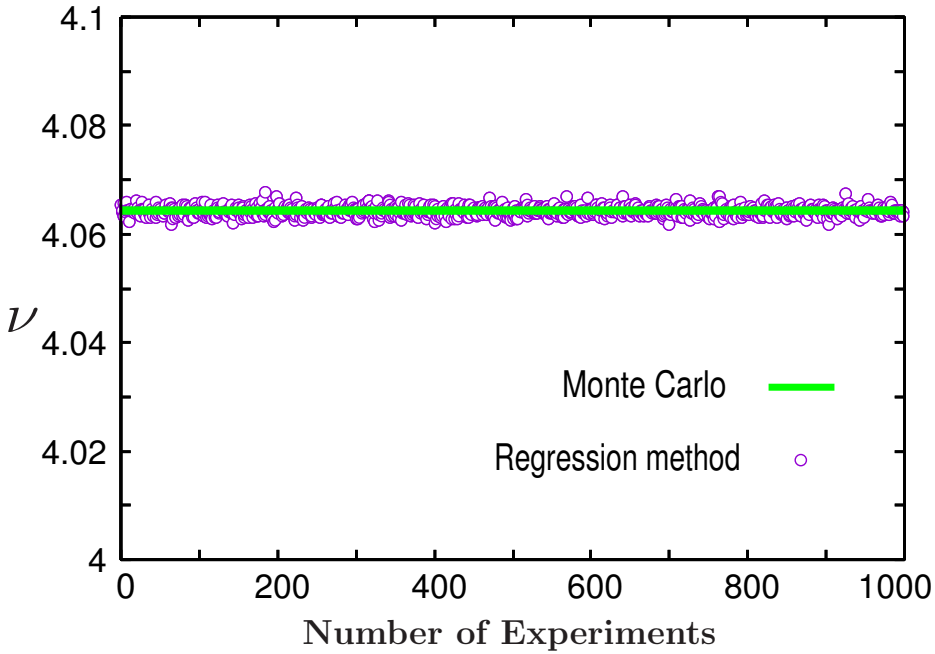
$$\boldsymbol{a} = \left(\boldsymbol{\Psi}^T\boldsymbol{\Psi}\right)^{-1}\boldsymbol{\Psi}^T\boldsymbol{\mathcal{Y}}$$

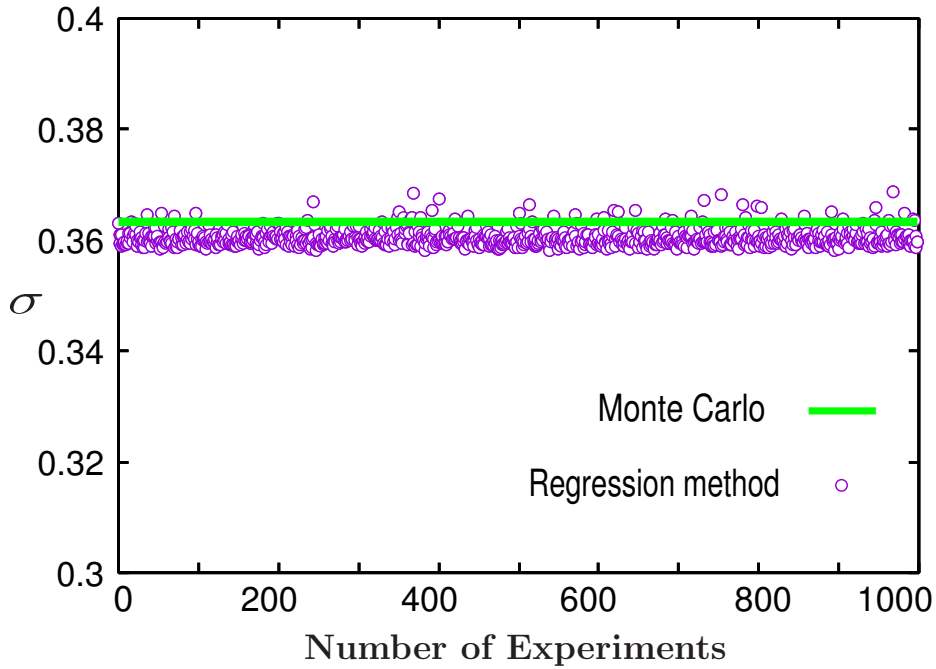
- **Coefficient vector $\boldsymbol{a} = [a_0, ..., a_\mathcal{L}]^T$.**
- **$\mathcal{K}$ random variables $\boldsymbol{\xi} = [\xi_1, ..., \xi_\mathcal{K}]^T$.**
- **$\mathcal{M}$ system responses $\boldsymbol{\mathcal{Y}} = \left[\mathcal{Y}(\boldsymbol{\xi}^{(1)}), \mathcal{Y}(\boldsymbol{\xi}^{(2)}), ..., \mathcal{Y}(\boldsymbol{\xi}^{(\mathcal{M})})\right]^T$ obtained from FDTD simulations using $\mathcal{M}$ sets of input sample $\boldsymbol{\xi}^{(m)}$.**
- **Polynomial basis matrix $\boldsymbol{\Psi} = (\psi_\ell(\boldsymbol{\xi}^{(m)}), m = 1 \sim \mathcal{M}, \ell = 0 \sim \mathcal{L})$.**

$$\boldsymbol{\Psi} = \begin{bmatrix} \psi_0(\boldsymbol{\xi}^{(1)}) & \psi_1(\boldsymbol{\xi}^{(1)}) & \dots & \psi_\mathcal{L}(\boldsymbol{\xi}^{(1)}) \\ \psi_0(\boldsymbol{\xi}^{(2)}) & \psi_1(\boldsymbol{\xi}^{(2)}) & \dots & \psi_\mathcal{L}(\boldsymbol{\xi}^{(2)}) \\ \vdots & \vdots & \ddots & \vdots \\ \psi_0(\boldsymbol{\xi}^{(\mathcal{M})}) & \psi_1(\boldsymbol{\xi}^{(\mathcal{M})}) & \dots & \psi_\mathcal{L}(\boldsymbol{\xi}^{(\mathcal{M})}) \end{bmatrix}$$

Comparison of NIPC with MCM for $\mathcal{K} = 1$

Estimates of ν and σ from NIPC and MCM

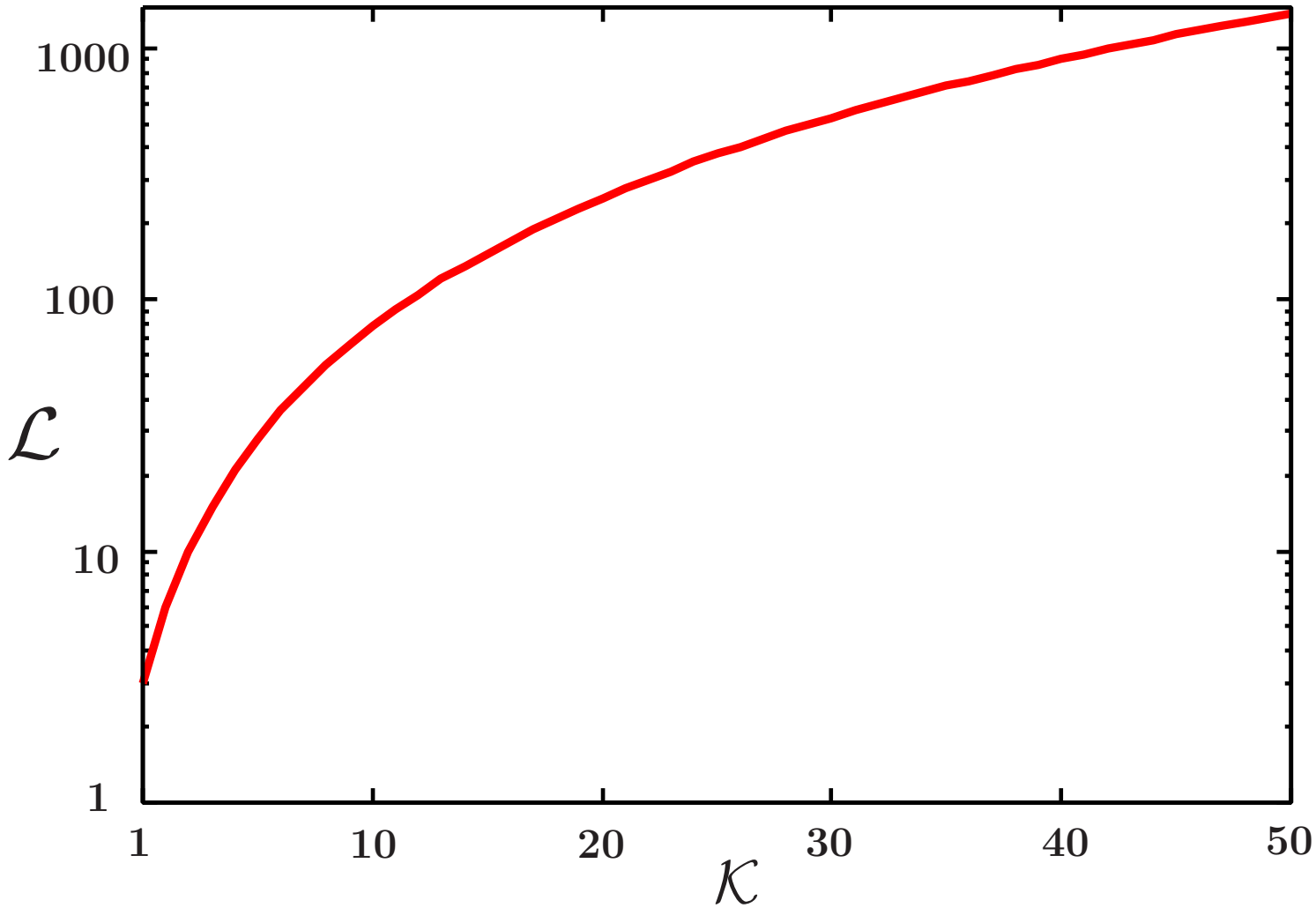




10000 FDTD simulations for MCM to obtain one ν and one σ
3 FDTD simulations for NIPC to obtain one ν and one σ
1000 experiments of NIPC method to obtain 1000 ν and 1000 σ

Curse of dimensionality

- **The regression method would be valid only if $\mathcal{L} \leq \mathcal{M}$.**
- **The number of required FDTD simulations $\propto$ the number of random variables $\mathcal{K}$.**



Least angle regression (LARS) method

- **NIPC expansion**

$$\mathcal{T}(\boldsymbol{\xi}) \approx \sum_{\ell=0}^{\mathcal{L}} a_\ell \psi_\ell(\boldsymbol{\xi}) = a_0\psi_0(\boldsymbol{\xi}) + a_1\psi_1(\boldsymbol{\xi}) + \cdots + a_{\mathcal{L}}\psi_{\mathcal{L}}(\boldsymbol{\xi})$$

  has a collection of polynomial bases $\{\psi_0(\boldsymbol{\xi}), \psi_1(\boldsymbol{\xi}), \ldots, \psi_{\mathcal{L}}(\boldsymbol{\xi})\}$

- **The LARS method selects those ψ which have significant correlation to system response.**
- **The less number of ψ, the less number of FDTD simulations.**

Procedures of LARS method

1. **Initialize** $\boldsymbol{a} = [a_1, a_2, \ldots, a_{\mathcal{L}}]^T = [0, 0, \ldots, 0]^T$.
2. **Calculate the correlation between** ψ **and the residual vector** $(\mathcal{Y} - \mathcal{T})$.

$$\mathcal{C} = \boldsymbol{\Psi}^T(\mathcal{Y} - \mathcal{T})$$

   $\mathcal{C} = [\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_{\mathcal{L}}]^T$. **Each element in** $\mathcal{C}$**, such as** $\mathcal{C}_\ell$**, indicates the correlation between** ψ_ℓ **and the residual vector.**
3. **Find the most correlated** ψ **with the residual vector.**

$$|\mathcal{C}_{\widetilde{k}}| = \max_{1 \le \ell \le \mathcal{L}} |\mathcal{C}_\ell|.$$

4. **Calculate** $a_{\widetilde{k}}$.

$$\boldsymbol{a} = (\hat{\boldsymbol{\Psi}}^T \hat{\boldsymbol{\Psi}})^{-1} \hat{\boldsymbol{\Psi}}^T \mathcal{Y} = \left[0, \ldots, 0, a_{\widetilde{k}}, 0, \ldots, 0\right]^T$$

   $\hat{\boldsymbol{\Psi}}$**: matrix with an additional** $\widetilde{k}$**-th column of** $\boldsymbol{\Psi}$ **at each iteration.** $\hat{\boldsymbol{\Psi}} = (\psi_{\widetilde{k}}(\boldsymbol{\xi}^{(m)}), m = 1 \sim \mathcal{M}, \widetilde{k} \in [1, \mathcal{L}])$.

Procedure of the LARS method

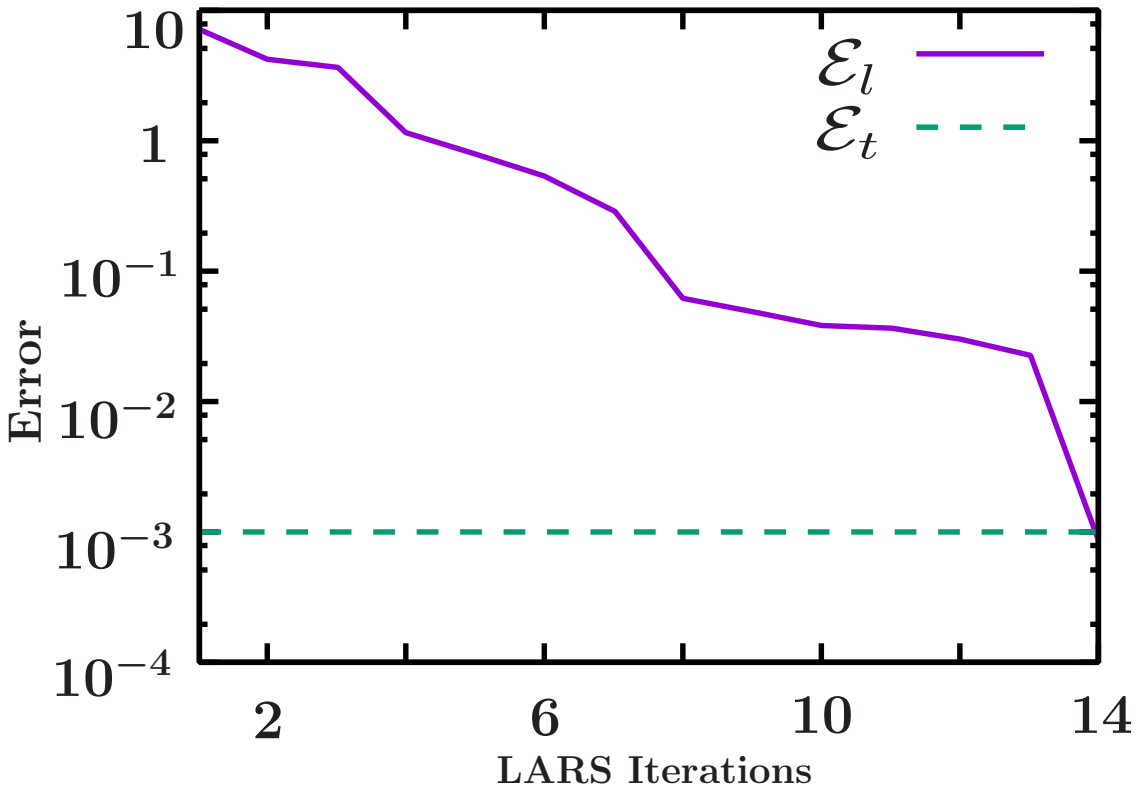
5. **Calculate $\mathcal{T}(\xi)$, and evaluate the error ε_l between $\mathcal{Y}$ and $\mathcal{T}$ using Leave-one-out cross-validation method. ε_l is calculated as in**

$$\varepsilon_l = \frac{1}{\mathcal{M}} \sum_{m=1}^{\mathcal{M}} \left(\frac{\mathcal{Y}(\boldsymbol{\xi}^{(m)}) - \mathcal{T}(\boldsymbol{\xi}^{(m)})}{1 - h_m} \right)^2,$$

h_m : the m-th diagonal element of the square matrix $\hat{\boldsymbol{\Psi}}(\hat{\boldsymbol{\Psi}}^T \hat{\boldsymbol{\Psi}})^{-1} \hat{\boldsymbol{\Psi}}^T$.

Procedure of the LARS method

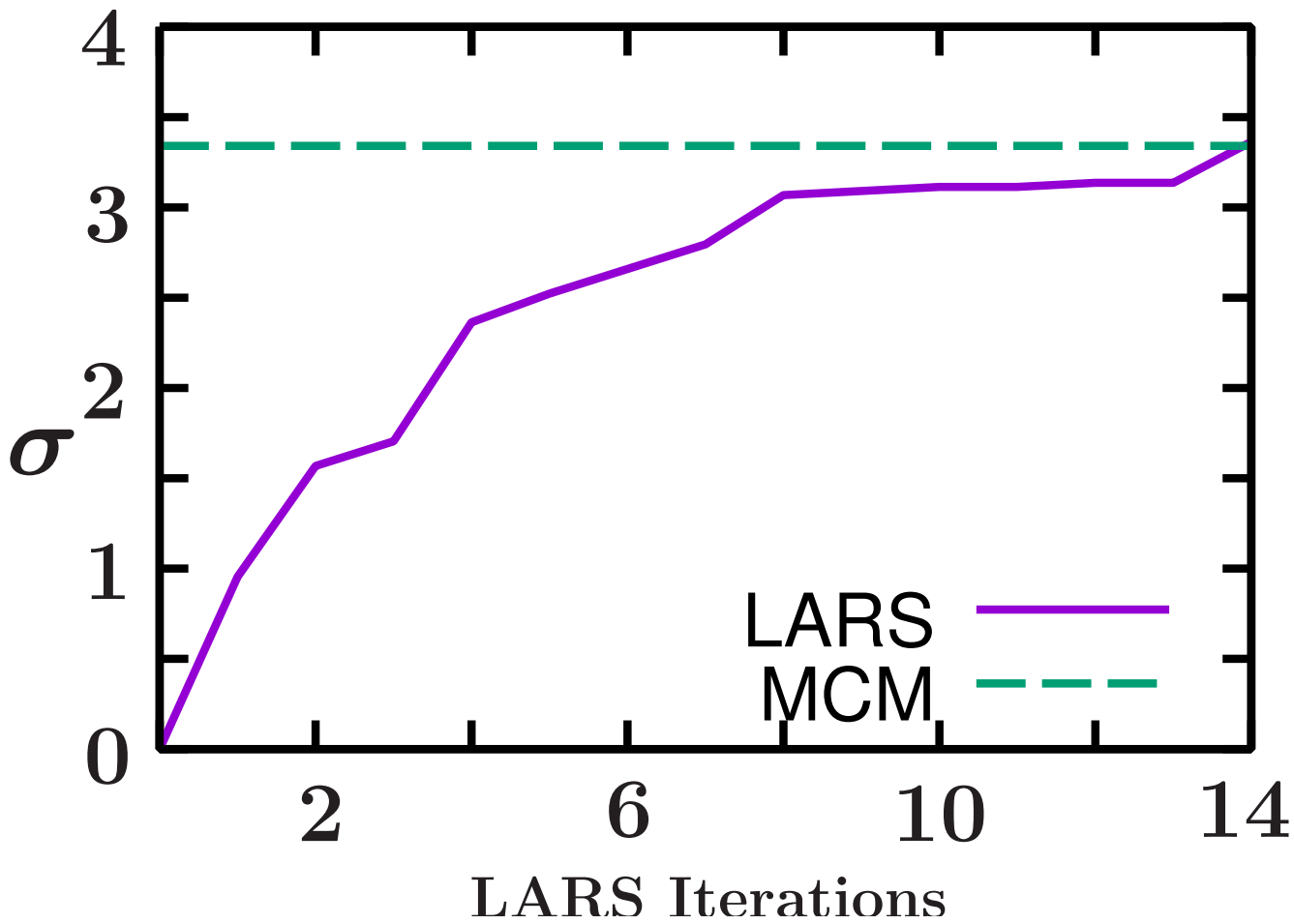
6. **Repeat from step 2 to Step 5, until $\varepsilon_l \leq$ the target error ε_t.**



- **The setting of ε_t is application-dependent.**
- **Empirically, $\varepsilon_t = 10^{-3}$ is a reasonable value in our scenario enabling the LARS method to have a stable performance.**

Procedure of the LARS method

7. **Estimate σ using $\alpha_{\tilde{k}}$ of the chosen polynomials as** $\sigma^2 = \sum \alpha_{\tilde{k}}^2 \psi_{\tilde{k}}^2$.



Pros and cons of LARS method

Advantages of LARS method

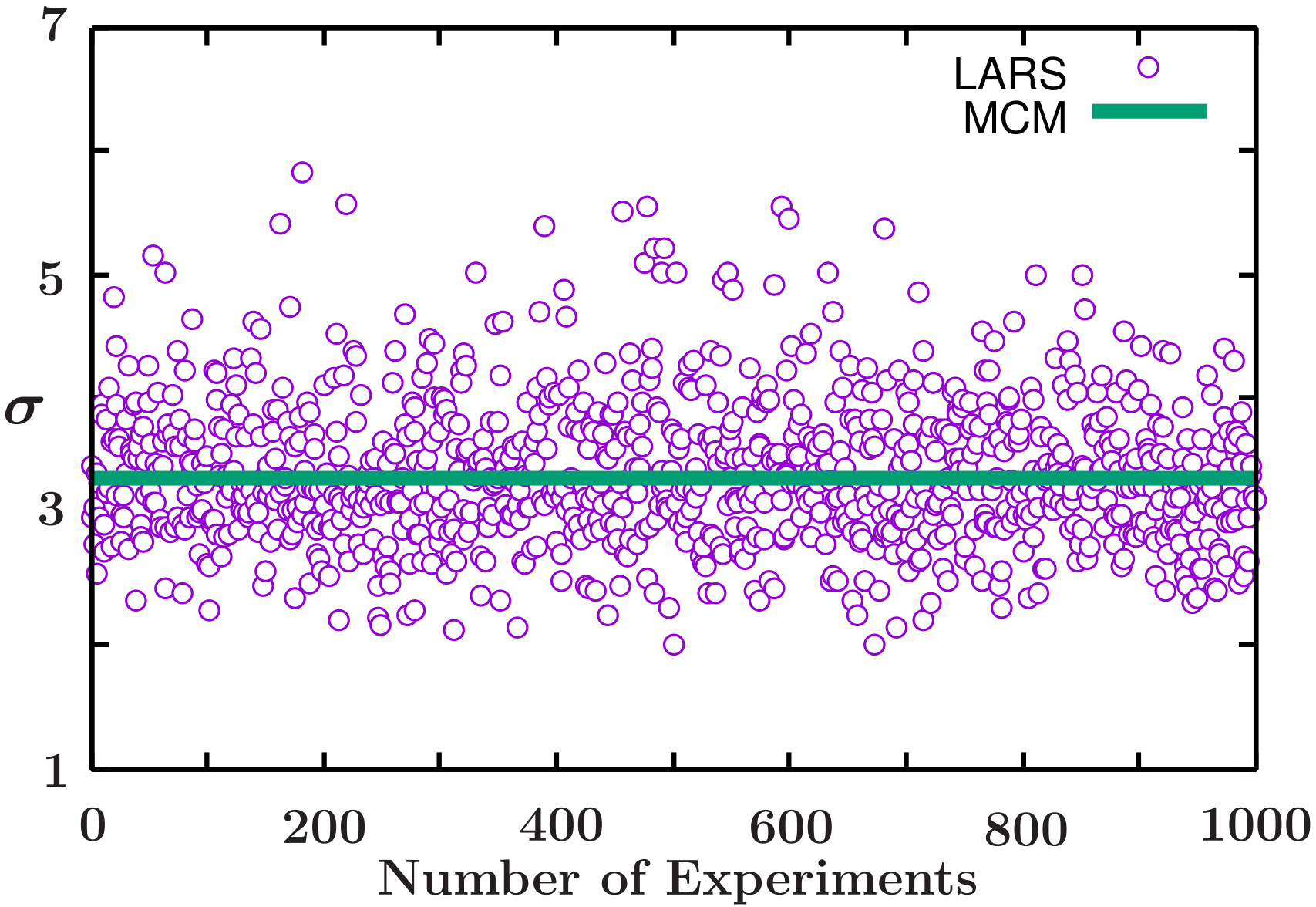
1. **Computationally efficient.**
   Only use significant polynomial bases.
2. **More stable than standard NIPC method.**
   The less number of polynomial bases, more stable computation of $(\hat{\Psi}^T\hat{\Psi})^{-1}$.

Pros and cons of LARS method

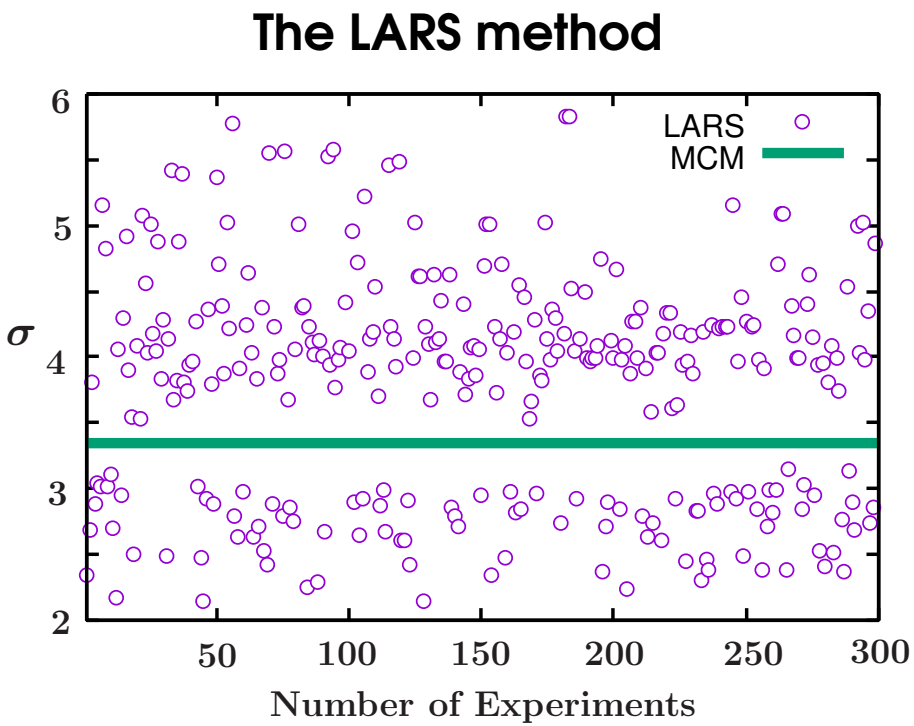
Adaptive LARS method to overcome demerits of LARS

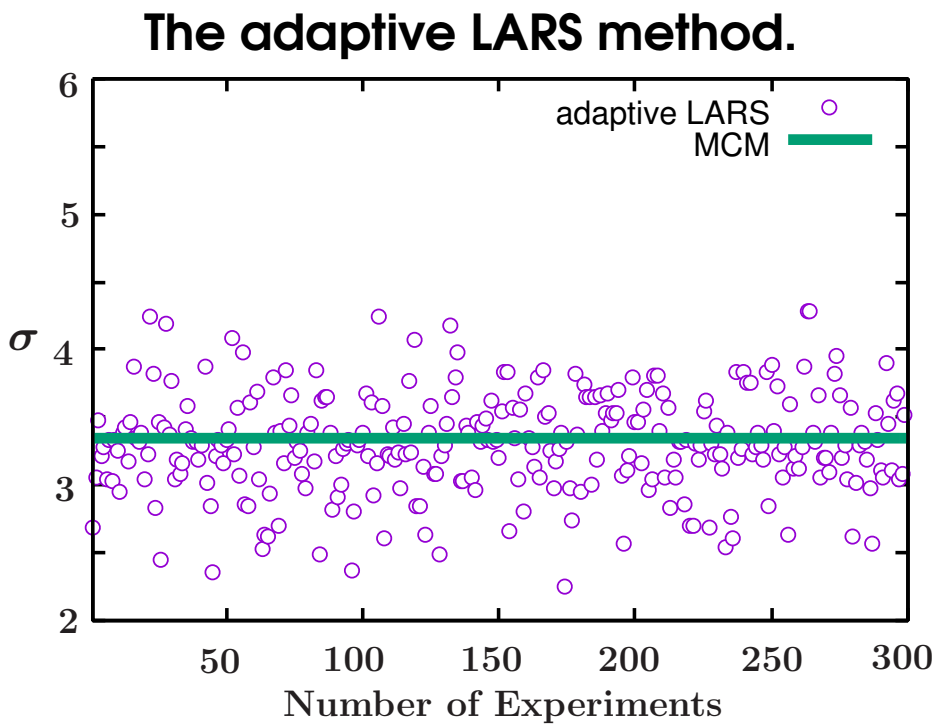
Issues	LARS	Adaptive LARS method
Setting of ε_t	LARS procedure stops too early when $\varepsilon_l < \varepsilon_t$ after a few number of iterations.	Set minimum number of the LARS iterations to prevent the LARS procedure from being terminated too early.
Lack of flexibility	$\mathcal{M}$ should exceed the number of chosen ψ.	Introduce the L_2 regularisation to the LARS method to obtain a pseudo-inverse of $(\hat{\boldsymbol{\Psi}}^T \hat{\boldsymbol{\Psi}})^{-1}$.

Experimental results from LARS method with 15 $\mathcal{M}$ in case of $\mathcal{K} = 10$



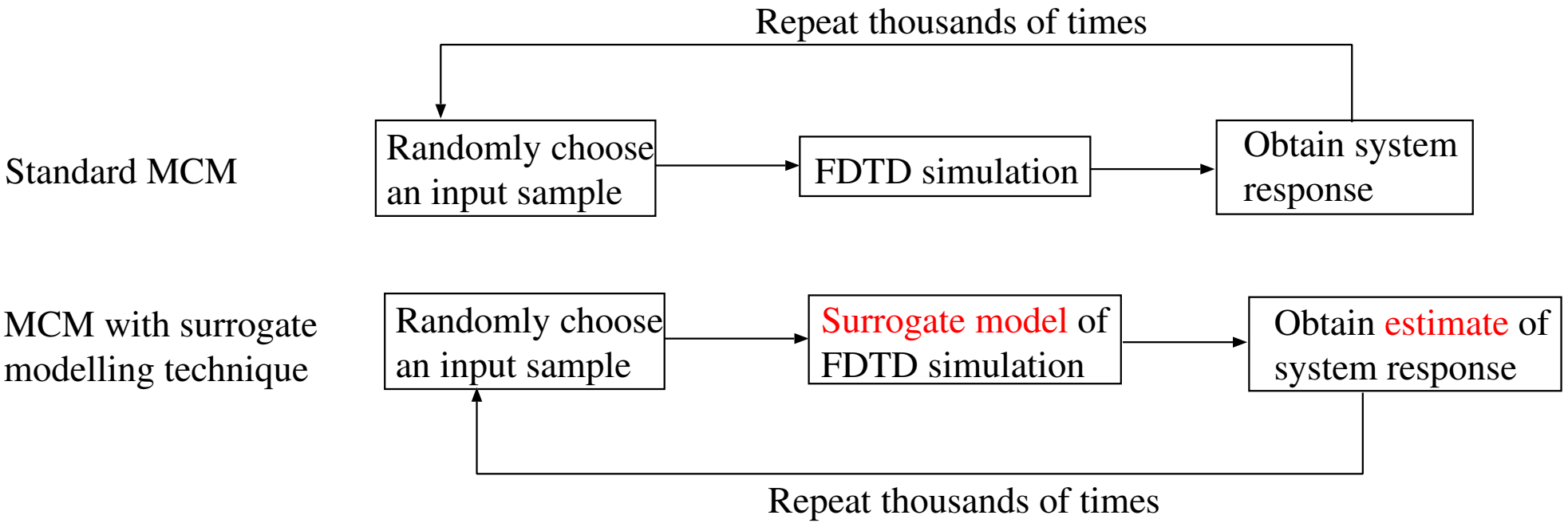
LARS vs adaptive LARS





UQ using artificial neural network (ANN)

Machine learning algorithms to model the underlying relationships between the input variables and the system output.



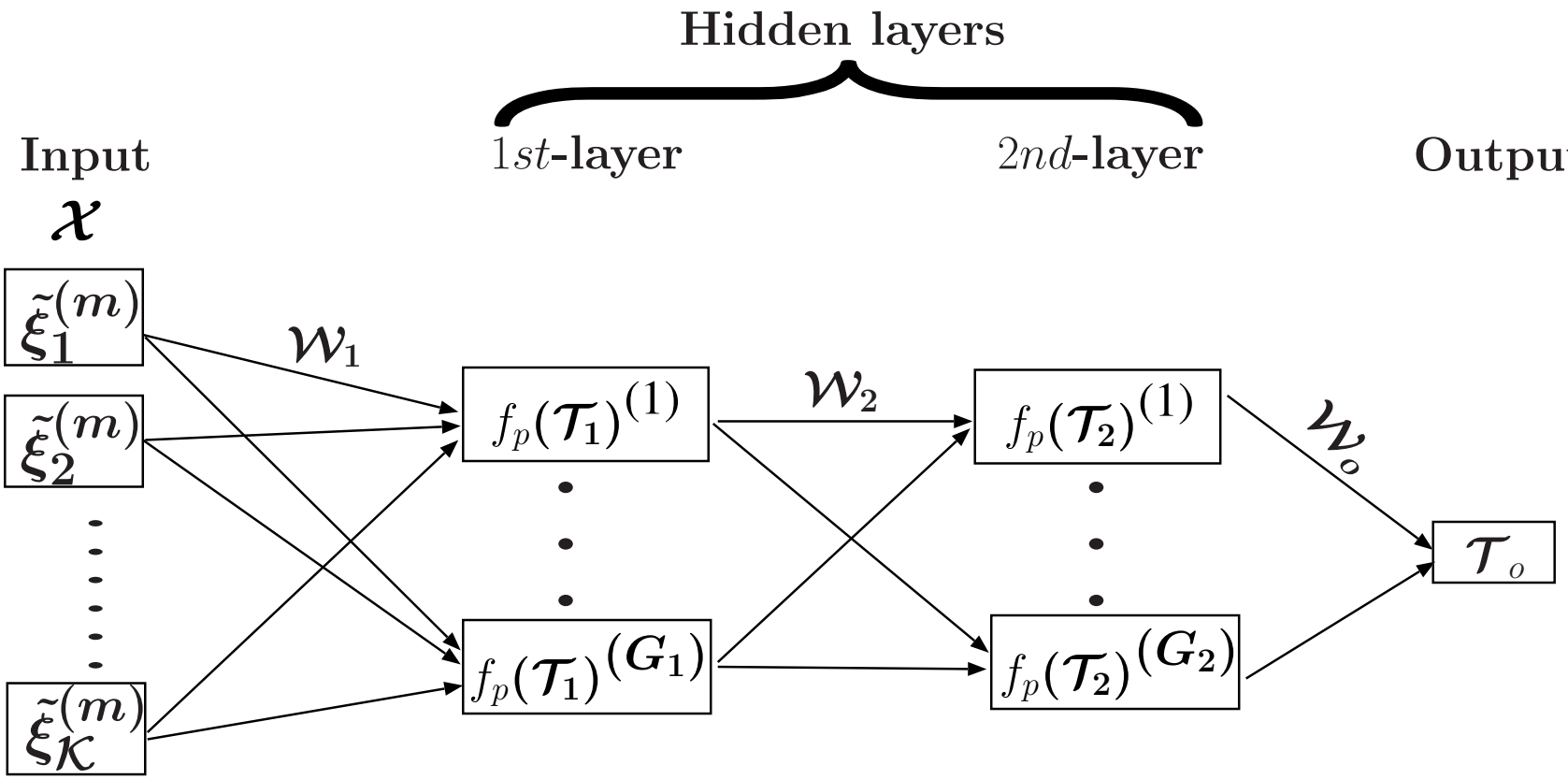
ANN to build a surrogate model for the FDTD simulation

UQ using artificial neural network (ANN)

Surrogate models

- **replace FDTD simulations required in the MCM.**
- **are computationally efficient compared with FDTD simulations.**
- **improve the computational efficiency of MCM.**

The architecture of ANN



- **Three types of layers of input, hidden and output layers.**
- **Two hidden layers of the $1st$-hidden layer with G_1 nodes and the $2nd$-hidden layer with G_2 nodes.**

The architecture of ANN

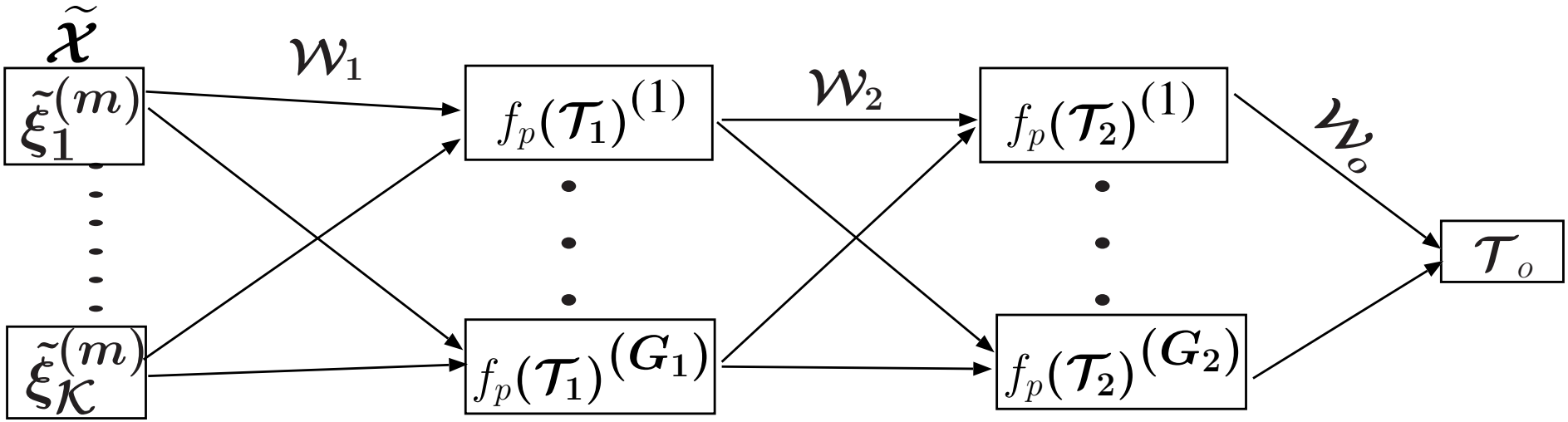
- **Three sets of weights**

$$\mathcal{W}_1 = \{\mathcal{W}_{1_{(kg_1)}}, k = 1 \sim \mathcal{K}, g_1 = 1 \sim G_1\}$$
$$\mathcal{W}_2 = \{\mathcal{W}_{2_{(g_1 g_2)}}, g_1 = 1 \sim G_1, g_2 = 1 \sim G_2\}$$
$$\mathcal{W}_o = \left[\mathcal{W}_{o_{(1)}}, \mathcal{W}_{o_{(2)}}, \ldots, \mathcal{W}_{o_{(G_2)}}\right]^T$$

- **The input and output of each neuron in hidden layers :** $\mathcal{T}$ **and** $f_p(\mathcal{T}) = \mathcal{T} + \mathcal{T}^2$ **respectively.**
- $f_p(\mathcal{T})$ **: our original activation function, inspired from NIPC expansion method.**
  When a system has one random variable,
  $\psi_0(\xi_k) = 1 \quad ; \quad \psi_1(\xi_k) = \xi_k \quad ; \quad \psi_2(\xi_k) = {\xi_k}^2 - 1$
  are the Hermite polynomial bases with the highest order equalling 2. Replacing ξ_k **with** $\mathcal{T}$**, we design** $f_p(\mathcal{T})$ **as in**

$$f_p(\mathcal{T}) = \sum_{\alpha=0}^{2} \psi_\alpha(\mathcal{T}).$$

Training process - Forward propagation



1. $\mathcal{T}_1 = \tilde{\mathcal{X}}(\mathcal{M})\mathcal{W}_1 \rightarrow f_p(\mathcal{T}_1) \rightarrow f_p(\mathcal{T}_1)\mathcal{W}_2 = \mathcal{T}_2$

   $\rightarrow f_p(\mathcal{T}_2) \rightarrow f_p(\mathcal{T}_2)\mathcal{W}_o = \mathcal{T}_o$
2. $\mathcal{T}_o = f_p\left(f_p\left(\tilde{\mathcal{X}}(\mathcal{M})\mathcal{W}_1\right)\cdot\mathcal{W}_2\right)\cdot\mathcal{W}_o$

Training process - Backpropagation

1. **Quantify the influence of weights on the loss function of**

$$L_o = \frac{1}{2}(\mathcal{T}_o - \mathcal{Y})^2,$$

where $\mathcal{Y} = \left[\mathcal{Y}(\xi^{(1)}), \mathcal{Y}(\xi^{(2)}), ..., \mathcal{Y}(\xi^{(\mathcal{M})})\right]^T$ **obtained from** $\mathcal{M}$ **FDTD simulations.**

2. **Use error signals to measure how much** L_o **varies with the changes of** $\mathcal{T}_o$, $\mathcal{T}_2$, **and** $\mathcal{T}_1$.
The error signal at the output layer:

$$\delta_o = \frac{\partial L_o}{\partial \mathcal{T}_o} = \frac{\partial \frac{1}{2}(\mathcal{T}_o - \mathcal{Y})^2}{\partial \mathcal{T}_o} = (\mathcal{T}_o - \mathcal{Y})\frac{\partial(\mathcal{T}_o - \mathcal{Y})}{\partial \mathcal{T}_o} = \mathcal{T}_o - \mathcal{Y}.$$

The error signal at the $2nd$**- and** $1st$**- hidden layer:**

$$\delta_2 = \frac{\partial f_p(\mathcal{T}_2)}{\partial \mathcal{T}_2} \odot \left(\delta_o \mathcal{W}_o{}^T\right); \quad \delta_1 = \frac{\partial f_p(\mathcal{T}_1)}{\partial \mathcal{T}_1} \odot \left(\delta_2 \mathcal{W}_2{}^T\right).$$

Derivation of $\boldsymbol{\delta}_2$ and $\boldsymbol{\delta}_1$

Assuming we have 1 $(= \mathcal{M})$ set of input samples to an ANN, and $G_1 = G_2 = 1$. The error signal δ_o is written by $\delta_o = (\mathcal{T}_o - \mathcal{Y})$. δ_2 is calculated as in

$$\begin{aligned}\delta_2 = \frac{\partial L_o}{\partial \mathcal{T}_2} &= \frac{\partial L_o}{\partial f_p(\mathcal{T}_2)} \cdot \frac{\partial f_p(\mathcal{T}_2)}{\partial \mathcal{T}_2} = \frac{\partial \frac{1}{2}(\mathcal{T}_o - \mathcal{Y})^2}{\partial f_p(\mathcal{T}_2)} \cdot \frac{\partial f_p(\mathcal{T}_2)}{\partial \mathcal{T}_2} \\ &= (\mathcal{T}_o - \mathcal{Y}) \cdot \frac{\partial(\mathcal{T}_o - \mathcal{Y})}{\partial f_p(\mathcal{T}_2)} \cdot \frac{\partial f_p(\mathcal{T}_2)}{\partial \mathcal{T}_2} = \delta_o \cdot \frac{\partial \mathcal{T}_o}{\partial f_p(\mathcal{T}_2)} \cdot \frac{\partial f_p(\mathcal{T}_2)}{\partial \mathcal{T}_2} \\ &= \delta_o \cdot \frac{\partial f_p(\mathcal{T}_2)\mathcal{W}_o}{\partial f_p(\mathcal{T}_2)} \cdot \frac{\partial f_p(\mathcal{T}_2)}{\partial \mathcal{T}_2} = \delta_o \cdot \mathcal{W}_o \cdot \frac{\partial f_p(\mathcal{T}_2)}{\partial \mathcal{T}_2}.\end{aligned}$$

When $G_1 = G_2 > 1$, $\boldsymbol{\delta}_2$ can be written as in $\boldsymbol{\delta}_2 = \left[\delta_o \mathcal{W}_{o(1)} \frac{\partial f_p\left(\mathcal{T}_{2(1)}\right)}{\partial \mathcal{T}_{2(1)}}, \cdots, \delta_o \mathcal{W}_{o(G_2)} \frac{\partial f_p\left(\mathcal{T}_{2(G_2)}\right)}{\partial \mathcal{T}_{2(G_2)}}\right] = \delta_o \boldsymbol{\mathcal{W}}_o^T \odot \frac{\partial f_p(\boldsymbol{\mathcal{T}}_2)}{\partial \boldsymbol{\mathcal{T}}_2}$**. For $\mathcal{M}$ sets of input samples to an ANN,**

$\boldsymbol{\delta}_2$ is presented as in $\boldsymbol{\delta}_2 = \left\{\delta_o{}^{(m)} \mathcal{W}_{o(g_2)} \frac{\partial f_p\left(\mathcal{T}_{2(mg_2)}\right)}{\partial \mathcal{T}_{2(mg_2)}}, m = 1 \sim \mathcal{M}, g_2 = 1 \sim G_2\right\}$**.**

Training process - Weights update

1. **Weight update to minimise L_o.**

$$\mathcal{W}_o{}^{(\vartheta+1)} = \mathcal{W}_o{}^{(\vartheta)} - \eta f_p\left(\mathcal{T}_2\right)^{(\vartheta)^T} \boldsymbol{\delta}_o{}^{(\vartheta)}$$

$$\mathcal{W}_2{}^{(\vartheta+1)} = \mathcal{W}_2{}^{(\vartheta)} - \eta f_p\left(\mathcal{T}_1\right)^{(\vartheta)^T} \boldsymbol{\delta}_2{}^{(\vartheta)}$$

$$\mathcal{W}_1{}^{(\vartheta+1)} = \mathcal{W}_1{}^{(\vartheta)} - \eta \widetilde{\boldsymbol{\mathcal{X}}}(\mathcal{M})^T \boldsymbol{\delta}_1{}^{(\vartheta)}.$$

   ϑ indicates the ϑ-th ANN iteration(gradient descent).

2. **The ANN iteration, each with updated weights.**
3. **Termination of the ANN iteration when the accuracy of the trained ANN model reaches our expectation.**

The gradient descent method

- **An optimisation algorithm to minimise a function.**
- **The linear regression model**

$$\mathcal{T}(\xi_1, \ldots, \xi_K, \mathcal{W}_1, \cdots, \mathcal{W}_K) = \mathcal{W}_1\xi_1 + \cdots + \mathcal{W}_K\xi_K$$
$$= \mathcal{T}(\boldsymbol{\xi}, \boldsymbol{\mathcal{W}}) = \boldsymbol{\xi}^T\boldsymbol{\mathcal{W}},$$

$\boldsymbol{\mathcal{W}} = [\mathcal{W}_1, \cdots, \mathcal{W}_K]^T$
$\mathcal{T}(\boldsymbol{\xi}, \boldsymbol{\mathcal{W}})$ **: the prediction of** $\mathcal{Y}(\boldsymbol{\xi})$**.**

- **Find the optimal** $\mathcal{W}$ **enabling the cost function of**

$$Q(\boldsymbol{\mathcal{W}}) = \frac{1}{\mathcal{M}} \sum_{m=1}^{\mathcal{M}} \left(\mathcal{T}(\boldsymbol{\xi}^{(m)}, \boldsymbol{\mathcal{W}}) - \mathcal{Y}(\boldsymbol{\xi}^{(m)}) \right)^2$$

to reach local minimum.

The gradient descent method

- **Derive the gradient of $Q(\mathcal{W})$ with respect to $\mathcal{W}_k$**

$$\frac{\partial}{\partial \mathcal{W}_k} Q(\mathcal{W}) = \frac{1}{\mathcal{M}} \sum_{m=1}^{\mathcal{M}} 2 \left(\mathcal{T}(\boldsymbol{\xi}^{(m)}, \mathcal{W}) - \mathcal{Y}(\boldsymbol{\xi}^{(m)}) \right) \xi_k{}^{(m)}.$$

- **Update $\mathcal{W}$**

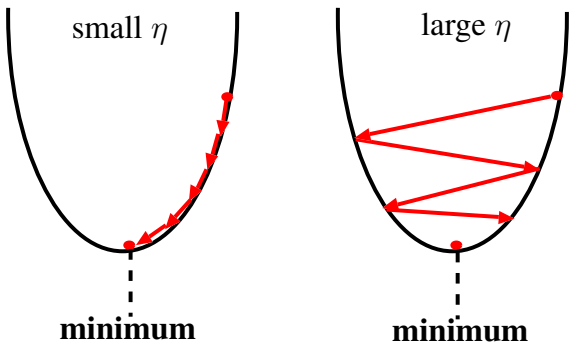
$$\mathcal{W}^{(i+1)} = \mathcal{W}^{(i)} - \eta \nabla_{\mathcal{W}^{(i)}} Q,$$

η **: the learning rate.**

$$\nabla_{\mathcal{W}} Q = \left[\frac{\partial}{\partial \mathcal{W}_1} Q(\mathcal{W}), \cdots, \frac{\partial}{\partial \mathcal{W}_K} Q(\mathcal{W}) \right]^T$$

$\mathcal{W}^{(i)}$ **:** $\mathcal{W}$ **at** i**-th gradient descent iteration.**

- **Repeat gradient descent iteration until $\mathcal{W}$ converges.**



Detection of overfitting in ANN iteration

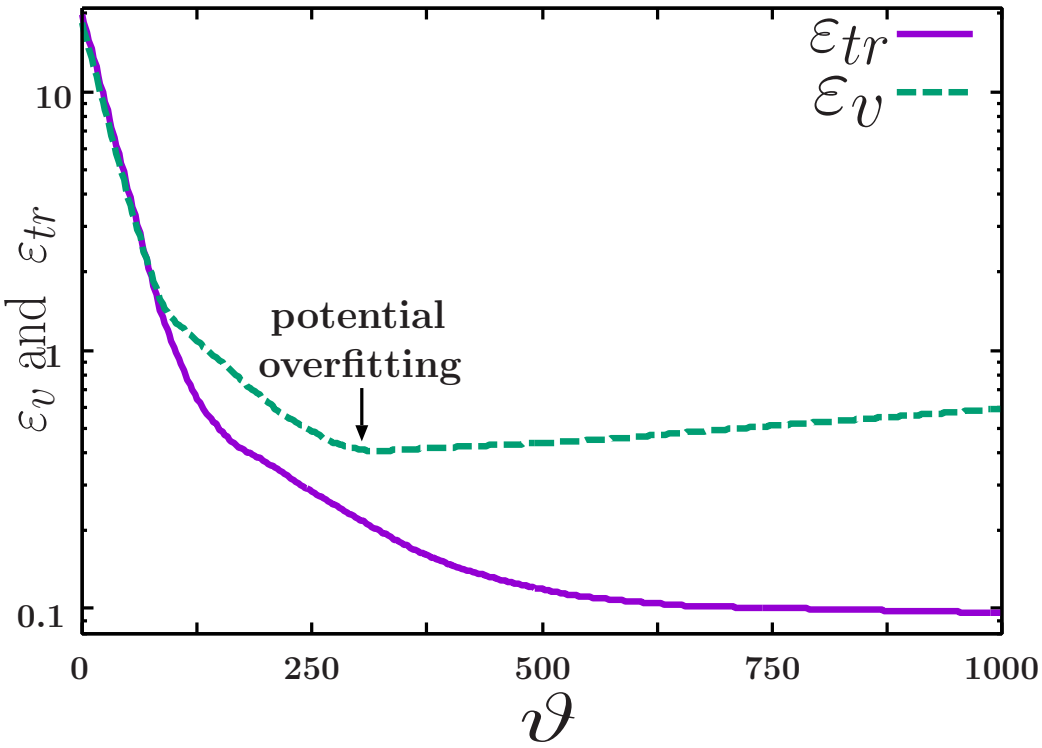
Overfitting: The *trained* ANN is able to accurately predict the outputs for the *training* data, whereas the predictive accuracy for the *test / validation* data is substantially lower than those for the *training* data.

1. **Split the dataset into three parts: *training* data $\widetilde{\mathcal{X}}(\mathcal{M}_{tr})$, *validation* data $\widetilde{\mathcal{X}}(\mathcal{M}_{vl})$, and *test* data $\widetilde{\mathcal{X}}(\mathcal{M}_{ts})$, where $\mathcal{M} = \mathcal{M}_{tr} + \mathcal{M}_{vl} + \mathcal{M}_{ts}$.**
2. **At each ANN iteration, update $\mathcal{W}$ using $\widetilde{\mathcal{X}}(\mathcal{M}_{tr})$ and utilise the updated $\mathcal{W}$ to calculate the output of ANN for $\widetilde{\mathcal{X}}(\mathcal{M}_{vl})$ of the ϑ-th ANN iteration as**

$$\mathcal{T}_{vlo}{}^{(\vartheta)} = f_p\left(f_p\left(\widetilde{\mathcal{X}}(\mathcal{M}_{vl})\mathcal{W}_1{}^{(\vartheta)}\right)\cdot\mathcal{W}_2{}^{(\vartheta)}\right)\cdot\mathcal{W}_o{}^{(\vartheta)},$$

where $\mathcal{T}_{vlo}{}^{(\vartheta)} = \left[\mathcal{T}_{vlo}{}^{(\vartheta)(1)},\cdots,\mathcal{T}_{vlo}{}^{(\vartheta)(\mathcal{M}_{vl})}\right]^T$.

Detection of overfitting in ANN iteration



3. **Calculate the *validation* error ε_V of the ϑ-th ANN iteration.**

$$\varepsilon_V{}^{(\vartheta)} = \frac{1}{\mathcal{M}_{Vl}} \sum_{m=1}^{\mathcal{M}_{Vl}} \left(\mathcal{T}_{Vlo}{}^{(\vartheta)(m)} - \mathcal{Y}(\xi^{(m)}) \right)^2 ,$$

Detection of overfitting in ANN iteration

4. **Detection of overfitting at when $\frac{\varepsilon_V{}^{(\vartheta+1)}}{\varepsilon_V{}^{(\vartheta)}} \geq \varpi$, where $\varpi > 1$.**

Usage of *validation* error

- **Hyperparameters set to minimize the local minimum of $\varepsilon_V{}^{(\vartheta)}$**
- **Termination of the ANN iteration when ample data available**

Criteria to terminate ANN iteration

Use the entire dataset as *training* data to maximise the number of training data, and terminate ANN iteration when ε_{tr} reaches stable status (convergence), judged by the conditions of

$$\frac{\left|\varepsilon_{tr}{}^{(\vartheta)} - \varepsilon_{tr}{}^{(\vartheta-1)}\right|}{\varepsilon_{tr}{}^{(\vartheta-1)}} \leq b$$

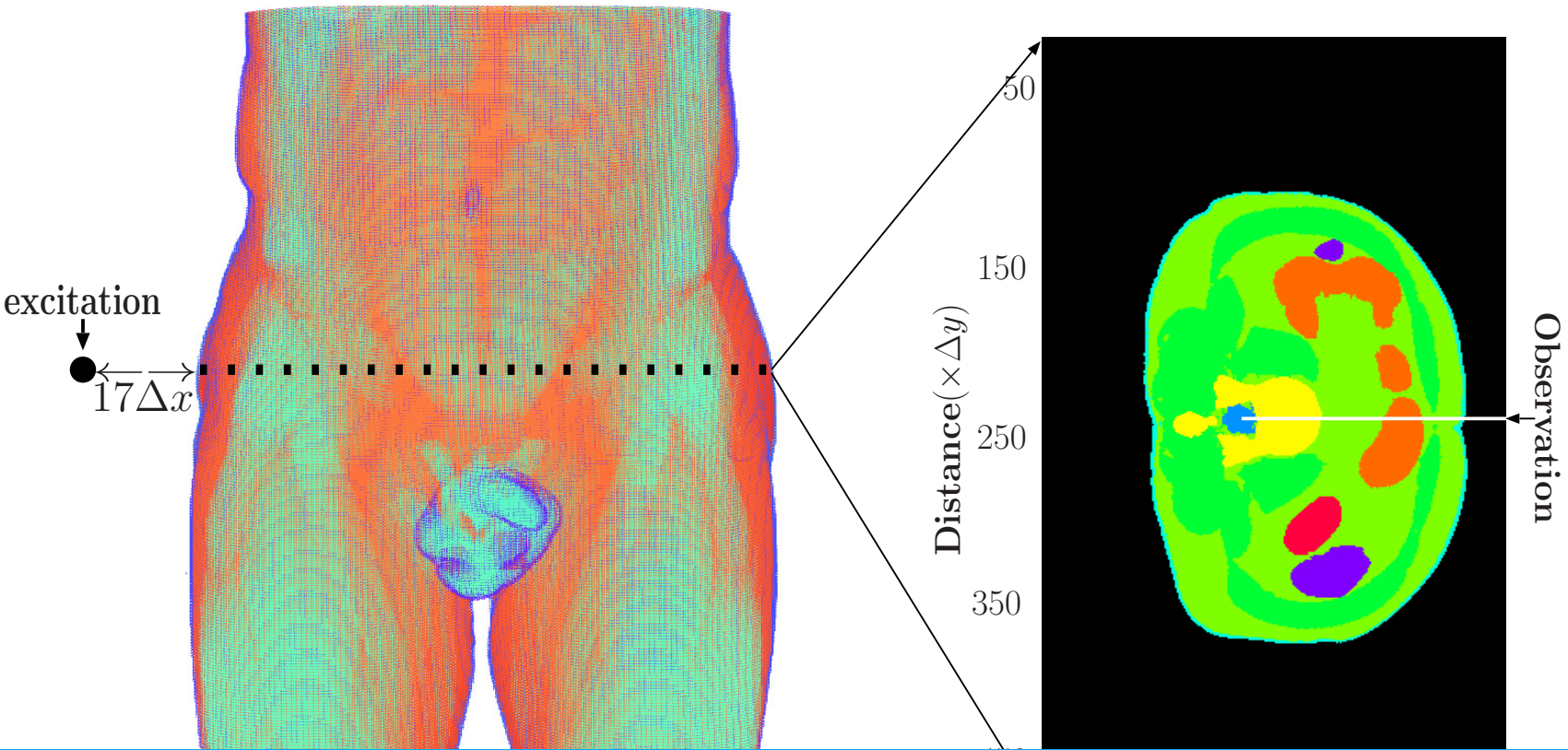
and

$$\frac{\left|\varepsilon_{tr}{}^{(\vartheta-1)} - \varepsilon_{tr}{}^{(\vartheta-2)}\right|}{\varepsilon_{tr}{}^{(\vartheta-2)}} \leq b,$$

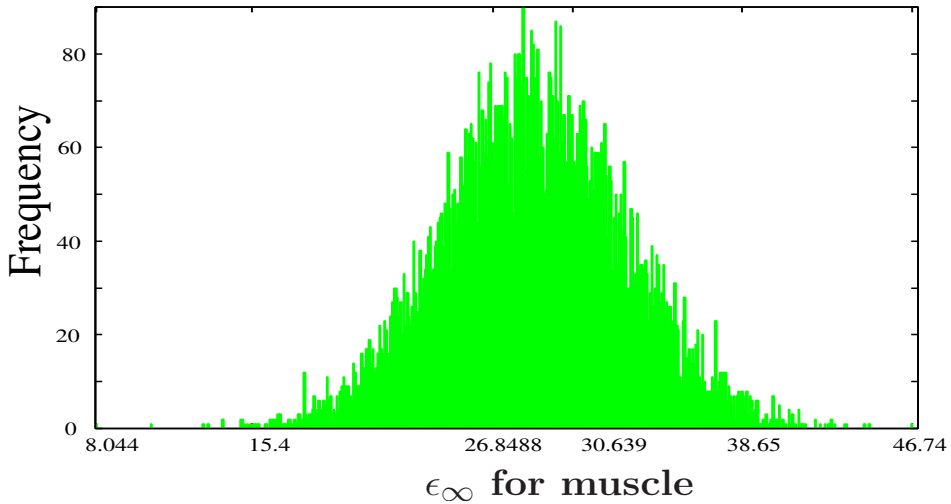
$b > 0$ **: a small constant , say 0.01**

Numerical simulation setup

- **Five influential tissues (fat, skin, muscle, bone and prostate).**
- **Two Debye parameters of interest for each influential tissue.**
- **10($= \mathcal{K}$) Debye parameters as uncertain inputs. $|\boldsymbol{E}|^2$ is the output of the FDTD simulation.**



Each Debye parameter of interest yielding the normal distribution ($\mathcal{K} = 10$)



ξ	Meaning of ξ	Mean	Standard deviation
ξ_1	ϵ_∞ of fat	6.80	0.68
ξ_2	$\Delta\epsilon$ of fat	7.37	0.74
ξ_3	ϵ_∞ of skin	18.07	1.81
ξ_4	$\Delta\epsilon$ of skin	29.87	2.99
ξ_5	ϵ_∞ of muscle	28.93	2.88
ξ_6	$\Delta\epsilon$ of muscle	28.02	2.81
ξ_7	ϵ_∞ of bone	1.53	0.15
ξ_8	$\Delta\epsilon$ of bone	4.01	0.40
ξ_9	ϵ_∞ of prostate	32.82	3.28
ξ_{10}	$\Delta\epsilon$ of prostate	27.73	2.78

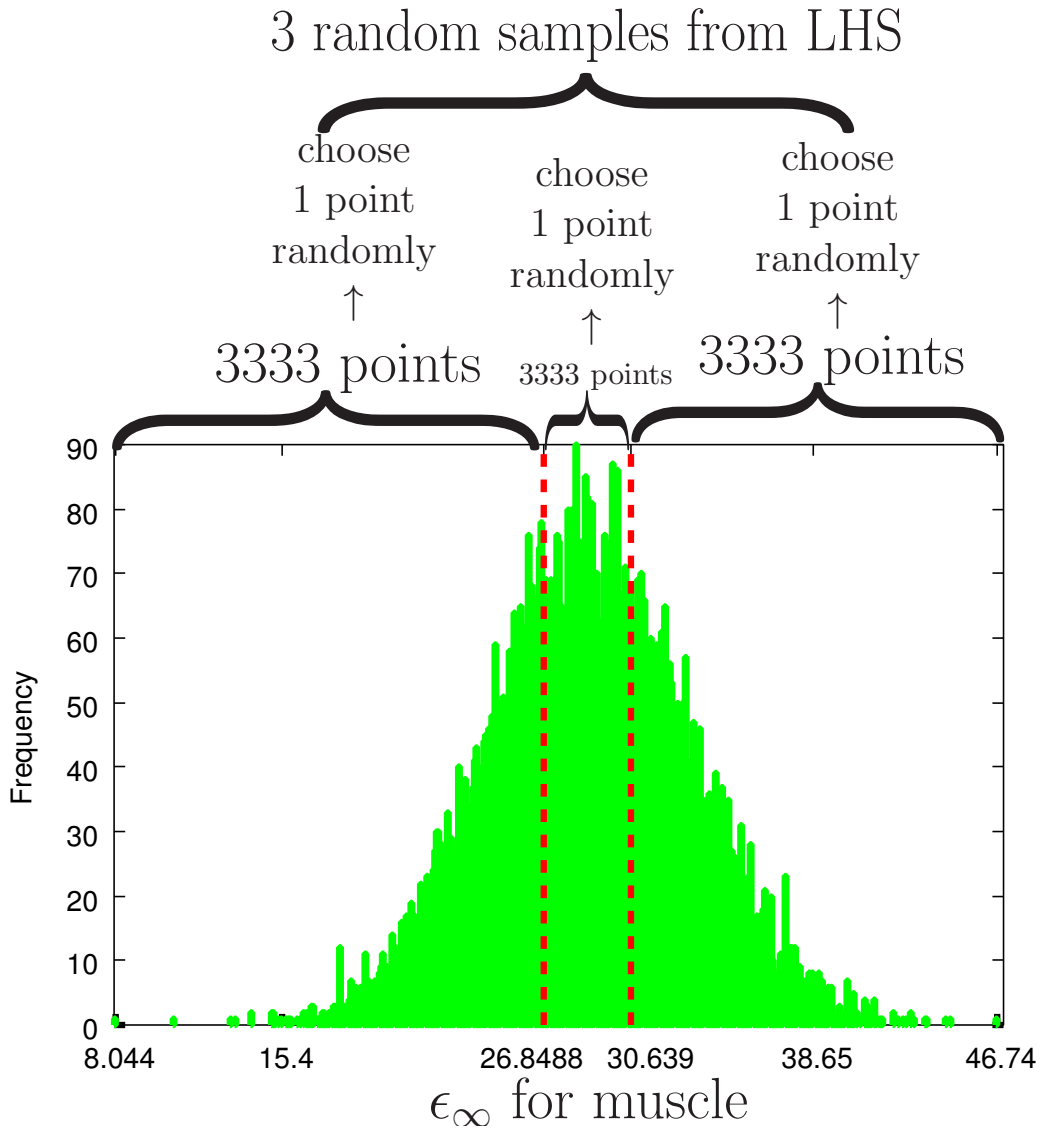
Generation of input dataset

1. **Generate 10^4 samples for each Debye parameter.**
2.
   - **MCM with FDTD runs: Randomly pick one sample out of 10^4 samples for each Debye parameter and combine those chosen samples to produce 1 combination**
   - **MCM with ANN: Use the Latin hypercube sampling method for $\mathcal{M}$ inputs for each Debye parameter and combine those chosen samples to produce $\mathcal{M}$ combination**

   of 10 Debye parameters $\xi = [\xi_1, \xi_2, \cdots, \xi_{10}]^T$
3. **Repeat 10^4 times to produce 10^4 ξ for MCM with FDTD runs. We do not choose the same samples as those chosen earlier**

The Latin hypercube sampling method. Ex. $\mathcal{M} = 3$

A sampling method to generate random samples based on the probability distribution of the random variable.



UQ using synthetic FDTD outputs

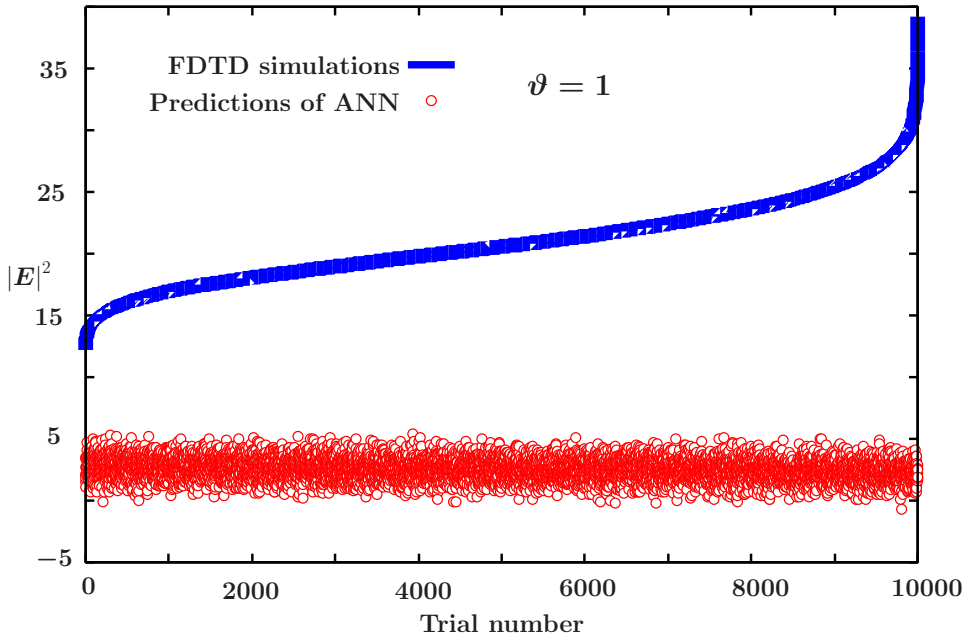
ANN

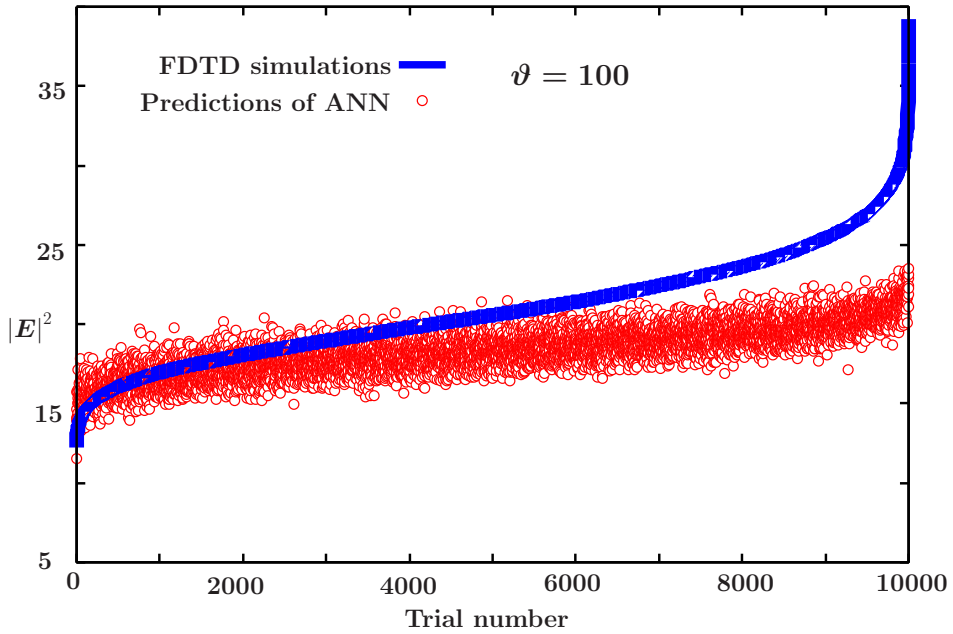
1. **Form 10 $\xi^{(m)}$ out of the 10^4 ξ based on LHS**
2. **Obtain 10 system responses $|\boldsymbol{E}_m|^2$ by the FDTD simulations using for $\xi^{(m)}$.**
3. **Train ANN with 10 sets of data pair of $\xi^{(m)}$ and $|\boldsymbol{E}_m|^2$**
4. **Use the trained ANN model to make predictions $\mathcal{T}_o^{(n)}$ for 10^4 ξ, where $n = 1 \sim 10^4$.**

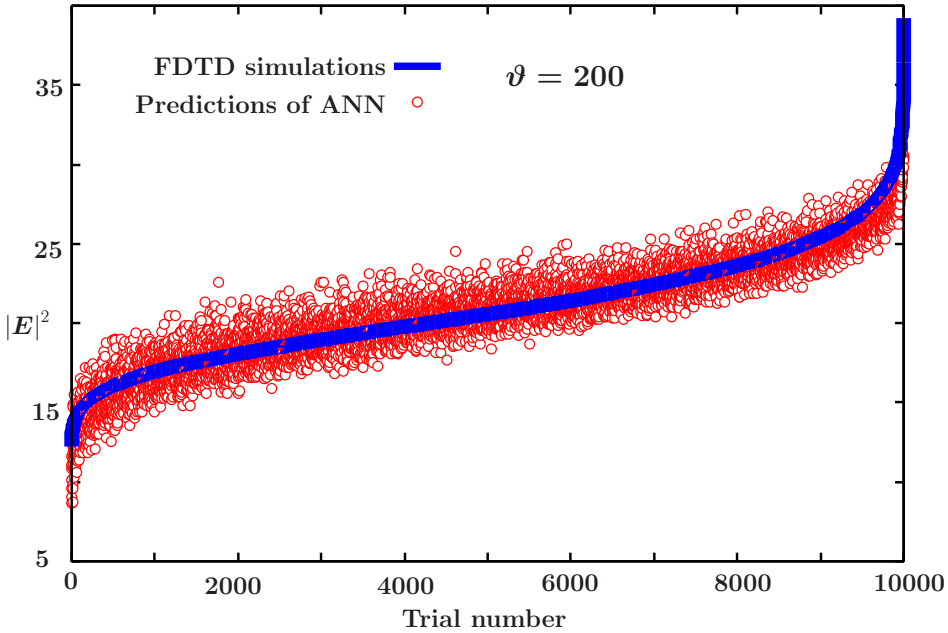
MCM

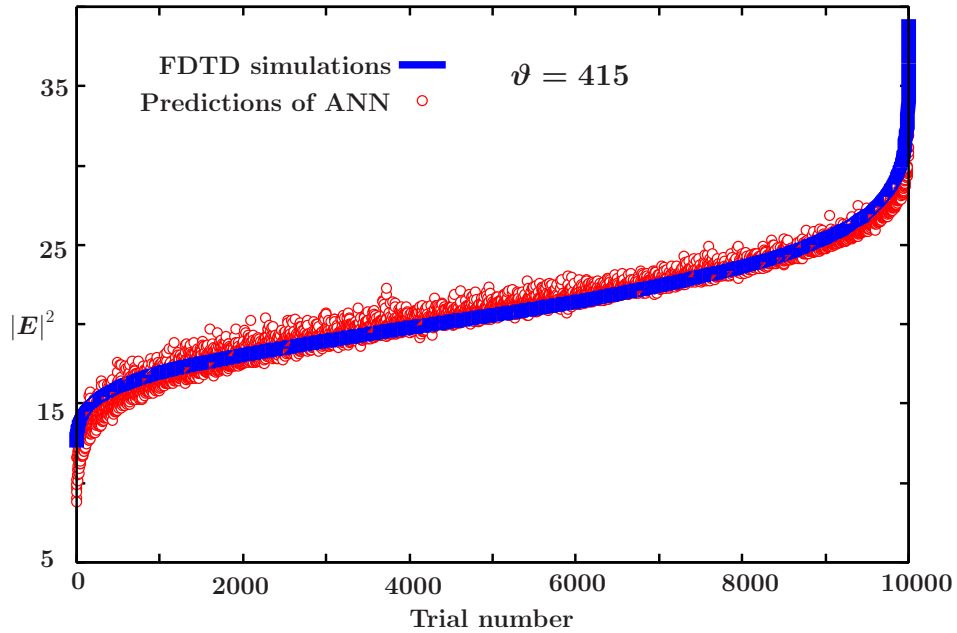
1. **Run 10^4 FDTD simulations to obtain 10^4 $|\boldsymbol{E}_m|^2$ using 10^4 $\xi^{(m)}$.**
2. **Estimate the uncertainty of system outputs by calculating the mean and standard deviation of 10^4 $|\boldsymbol{E}_m|^2$.**

ANN prediction varying ϑ

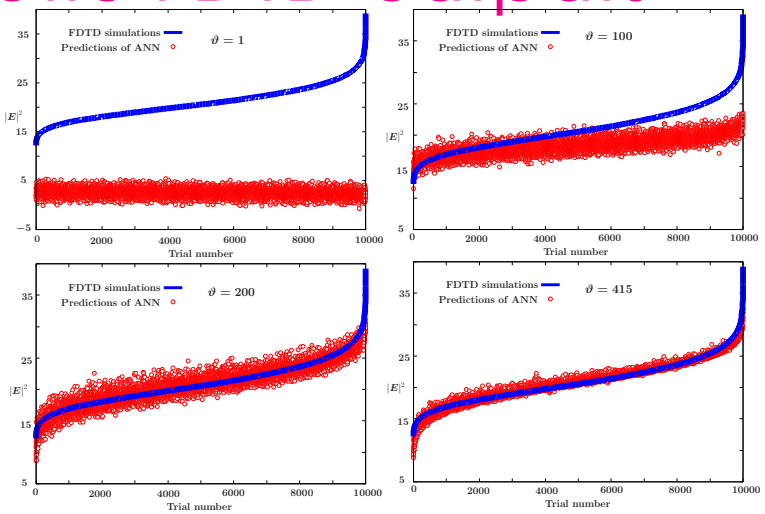








UQ using synthetic FDTD outputs



Computational time for each element

Activity	Time
1 FDTD simulation	5.5 hours
415 ANN iterations	less than 1 second

Computational efficiency comparison

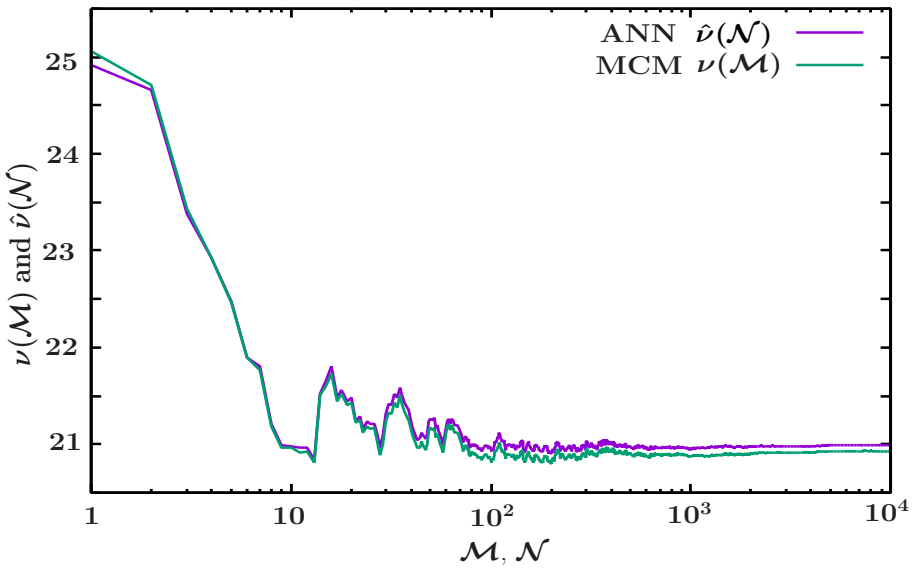
Approach	Activity elements
MCM with FDTD	10000 FDTD simulations
MCM with ANN	10 FDTD simulations + 415 ANN iterations

UQ using synthetic FDTD outputs

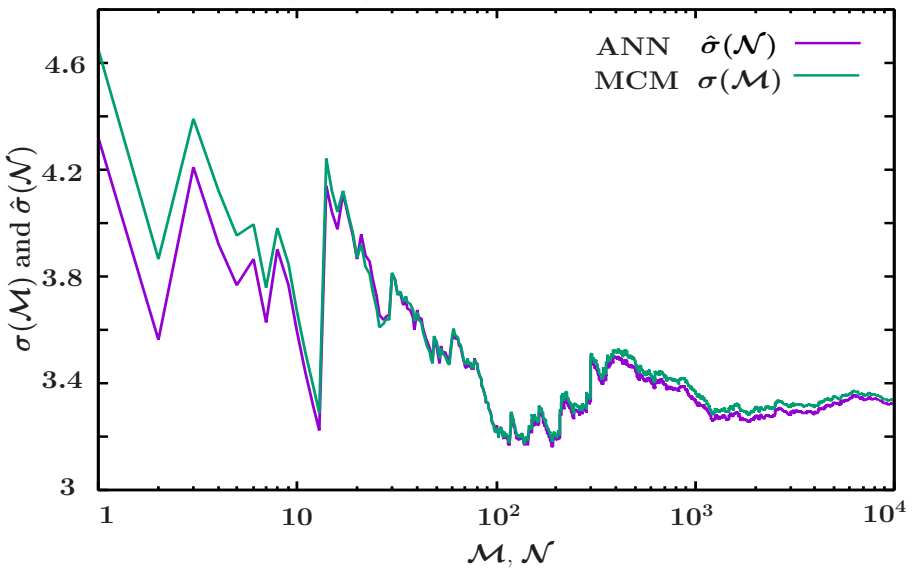
10^4 $\mathcal{T}_o^{(n)}$ at $\vartheta = 415$ to estimate the mean $\hat{\nu}(\mathcal{N})$ and standard deviation $\hat{\sigma}\,(\mathcal{N})$ of $|\boldsymbol{E}|^2$. $\mathcal{N}$: the number of samples used as inputs to the trained ANN model to predict $|\boldsymbol{E}|^2$.

$$\hat{\nu}(\mathcal{N}) = \frac{1}{\mathcal{N}} \sum_{n=1}^{\mathcal{N}} \mathcal{T}_o^{(n)} \qquad \hat{\sigma}\,(\mathcal{N})^2 = \frac{1}{\mathcal{N}-1} \sum_{n=1}^{\mathcal{N}} \left(\mathcal{T}_o^{(n)} - \hat{\nu}(\mathcal{N}) \right)^2$$

$\nu(\mathcal{M})$ **from MCM**
$\hat{\nu}(\mathcal{N})$ **from ANN**



$\sigma(\mathcal{M})$ **from MCM**
$\hat{\sigma}\,(\mathcal{N})$ **from ANN**

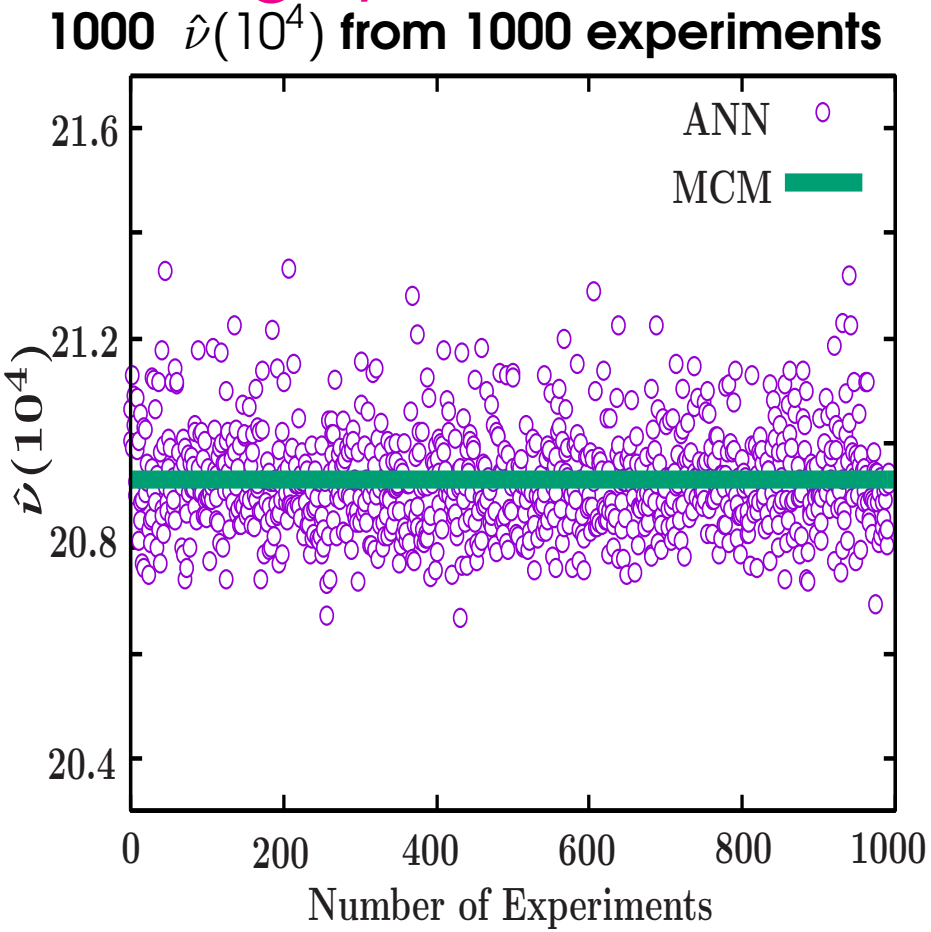


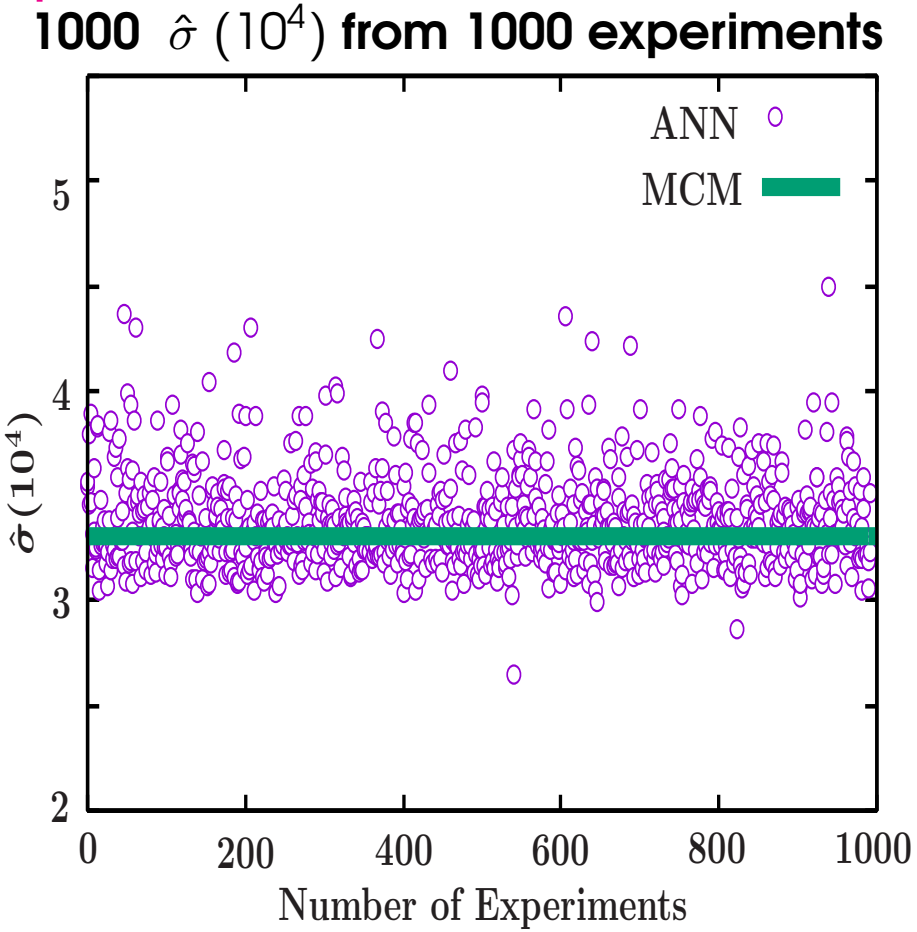
Comparisons of the proposed method with others

UQ methods	FDTD runs	ν	Accuracy of ν estimation	σ	Accuracy of σ estimation
NIPC method	65	20.942	99.92%	4.083	78.11%
Work in (1)	221	21.154	98.91%	3.267	97.52%
Work in (2)	30	20.720	99.02%	4.027	79.79%
Proposed ANN	10	20.849	99.63%	3.311	98.83%
Standard ANN	10	20.835	99.57%	2.983	89.06%

- **The standard ANN for regression analysis uses $f(\mathcal{T}) = \mathcal{T}$ as its activation function and stops ANN iteration based on preset iteration number.**
- **Work 1: J. S. Ochoa and A. C. Cangellaris. Random-space dimensionality reduction for expedient yield estimation of passive microwave structures.** *IEEE Trans. Microw. Theory Tech.*, **2013.**
- **Work 2: X. Cheng and V. Monebhurrun. Application of different methods to quantify uncertainty in specific absorption rate calculation using a cad-based mobile phone model.** *IEEE Trans. Electromagn. Compat.*, **2017.**

UQ using synthetic FDTD outputs





1. **The mean and standard deviation of the 1000** $\hat{\nu}(10^4)$ **are** 20.927 ± 0.098**, while** $\nu(10^4)$ **from the MCM is** 20.925**.**
2. **The mean and standard deviation of the 1000** $\hat{\sigma}(10^4)$ **are** 3.375 ± 0.222**, while** $\sigma(10^4)$ **from the MCM is** 3.348**.**

Conclusions and future plan

Background

- NIPC expansion method is
  - an ideal alternative to MCM.
  - incapable of handling the high-dimensional UQ problem due to the curse of dimensionality.
- The LARS method alleviates the curse of dimensionality.

Proposals

- An adaptive LARS method to improve the accuracy of LARS method.
- An ANN based technique to quantify the uncertainty of the FDTD simulation in the human body.

The proposed ANN outperforms other UQ techniques in terms of accuracy and computational efficiency.

Conclusions and future plan

Future work

1. Extend the proposed UQ techniques to handle the data whose correlated uncertainties satisfy non-Gaussian distribution.
2. Explore effective sample selection techniques to select most informative simulation samples.

Finally.....

Thank you for your attention

Any questions ?