Advances in Active Learning and Emulation for Multi-Fidelity Simulations

Chih-Li Sung

Department of Statistics and Probability
Michigan State University

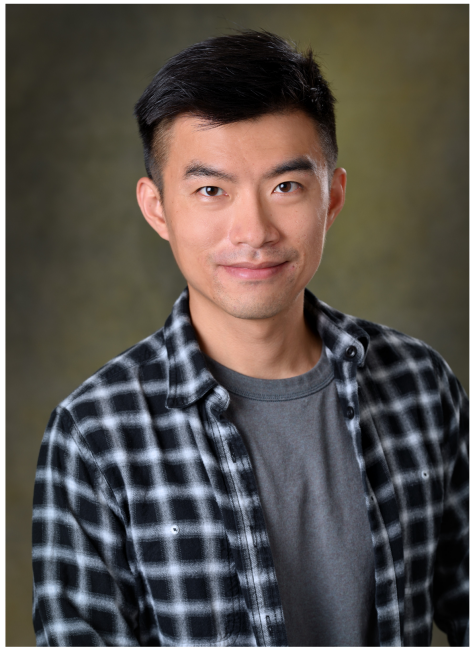
IMSI Workshop
Kernel Methods in Uncertainty Quantification and Experimental Design
April 4, 2025

MICHIGAN STATE UNIVERSITY

Outline

Junoh Heo

Romain Boutelet



Chih-Li Sung

Multi-Fidelity Simulations

- Computer models have been widely adopted to understand a real-world feature, phenomenon or event.
- Computer simulations are used to solve these models (e.g., finite element / finite difference) .

Multi-Fidelity Simulations

- Computer models have been widely adopted to understand a real-world feature, phenomenon or event.
- Computer simulations are used to solve these models (e.g., finite element / finite difference) .
- The simulation can be either
  - High-fidelity simulation: costly but close to the truth

Multi-Fidelity Simulations

- Computer models have been widely adopted to understand a real-world feature, phenomenon or event.
- Computer simulations are used to solve these models (e.g., finite element / finite difference) .
- The simulation can be either
  - High-fidelity simulation: costly but close to the truth
  - Low-fidelity simulation: cheaper but less accurate

Multi-Fidelity Simulations

- Computer models have been widely adopted to understand a real-world feature, phenomenon or event.
- Computer simulations are used to solve these models (e.g., finite element / finite difference) .
- The simulation can be either
  - High-fidelity simulation: costly but close to the truth
  - Low-fidelity simulation: cheaper but less accurate
  - (intermediate-fidelity simulation)

Motivated Example: Finite Element Simulations

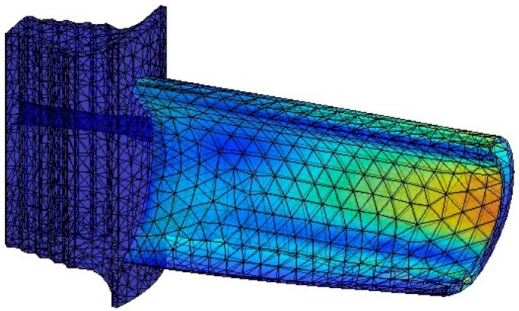
- Thermal stress of jet engine turbine blade can be analyzed through a static structural computer model.
- The model can be *numerically* solved via finite element method.

Motivated Example: Finite Element Simulations

- Thermal stress of jet engine turbine blade can be analyzed through a static structural computer model.
- The model can be *numerically* solved via finite element method.
- **Input**: $\mathbf{x} = (x_1, x_2) = (\text{pressure}, \text{suction})$

Motivated Example: Finite Element Simulations

- Thermal stress of jet engine turbine blade can be analyzed through a static structural computer model.
- The model can be *numerically* solved via finite element method.
- **Input**: $\mathbf{x} = (x_1, x_2) = (\text{pressure}, \text{suction})$
- **Output**: $f(\mathbf{x})$: **maximum** of thermal stress profile
- e.g., $\mathbf{x} = (0.23, 0.71)$



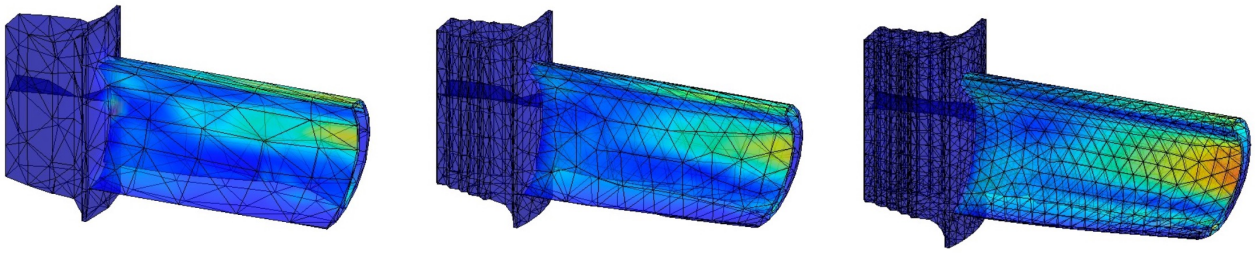
maximum of thermal stress profile $f(0.23, 0.71) = 20.3$

Multi-Fidelity Simulations

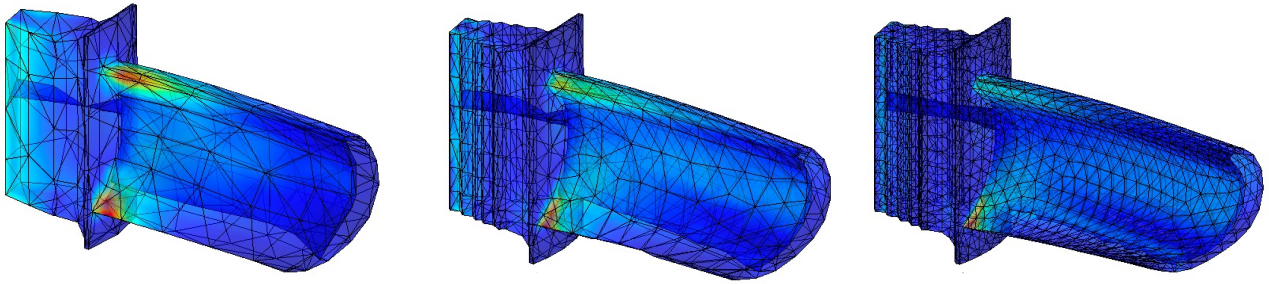
less accurate but cheaper accurate but expensive

Simulation accuracy

$\mathbf{x} = (0.50, 0.50)$



$\mathbf{x} = (0.23, 0.71)$



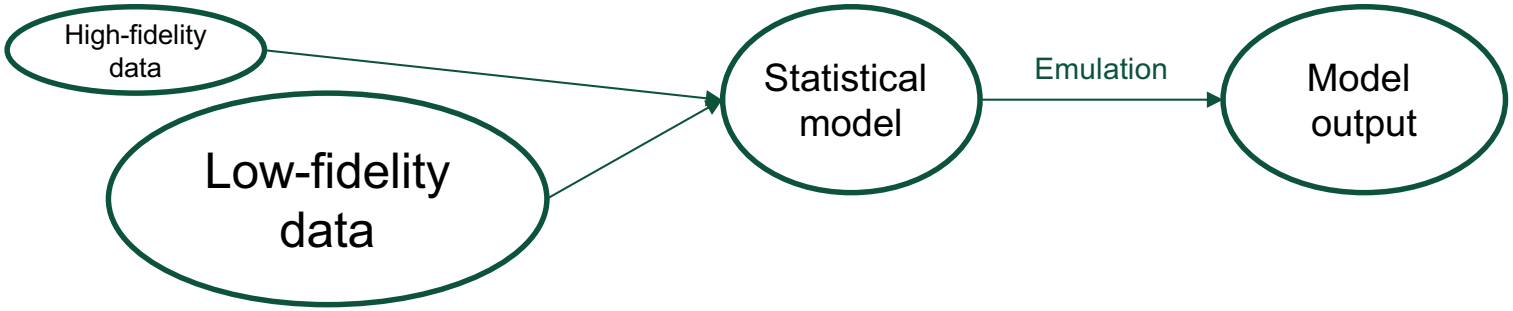
Simulation cost

Multi-Fidelity Emulation

- Can we leverage both low- and high-fidelity simulations in order to
  - build a statistical emulator, $\hat{f}(\mathbf{x})$, also known as a surrogate model, to approximate the output of a **high-fidelity complex simulator**:

$$\hat{f}(\mathbf{x}) \approx f(\mathbf{x}),$$

    where $f(\mathbf{x})$ represents the true simulator, with $\mathbf{x}$ as the input.
  - while minimizing the cost associated with the simulations?



Notation

fidelity level	1		2		3
output	$f_1(\mathbf{x})$		$f_2(\mathbf{x})$		$f_3(\mathbf{x})$
simulation cost	C_1	$<$	C_2	$<$	C_3

Notation

fidelity level	1		2		3
output	$f_1(\mathbf{x})$		$f_2(\mathbf{x})$		$f_3(\mathbf{x})$
simulation cost	C_1	$<$	C_2	$<$	C_3

- Goal: Emulate $f_L(\mathbf{x})$.
- Input: $\mathcal{X}_l = \{\mathbf{x}_i^{[l]}\}_{i=1}^{n_l}$ for $l = 1, \ldots, L$.
- Output: $\mathbf{y}_l := (f_l(\mathbf{x}))_{\mathbf{x} \in \mathcal{X}_l}$ for $l = 1, \ldots, L$

Existing Methods

- The canonical approach is auto-regressive (AR) model (Kennedy and O'Hagan, 2000).
- AR model assumes additive structure of Gaussian processes (GPs).

$$
\begin{aligned}
f_1(\mathbf{x}) &= Z_1(\mathbf{x}), \\
f_l(\mathbf{x}) &= \rho_{l-1} f_{l-1}(\mathbf{x}) + Z_l(\mathbf{x}), \quad \text{for} \quad 2 \leq l \leq L.
\end{aligned}
$$

- Several extensions including Qian et al. (2006); Qian and Wu (2008); Le Gratiet (2013); Le Gratiet and Garnier (2014); Perdikaris et al. (2017).

Existing Methods

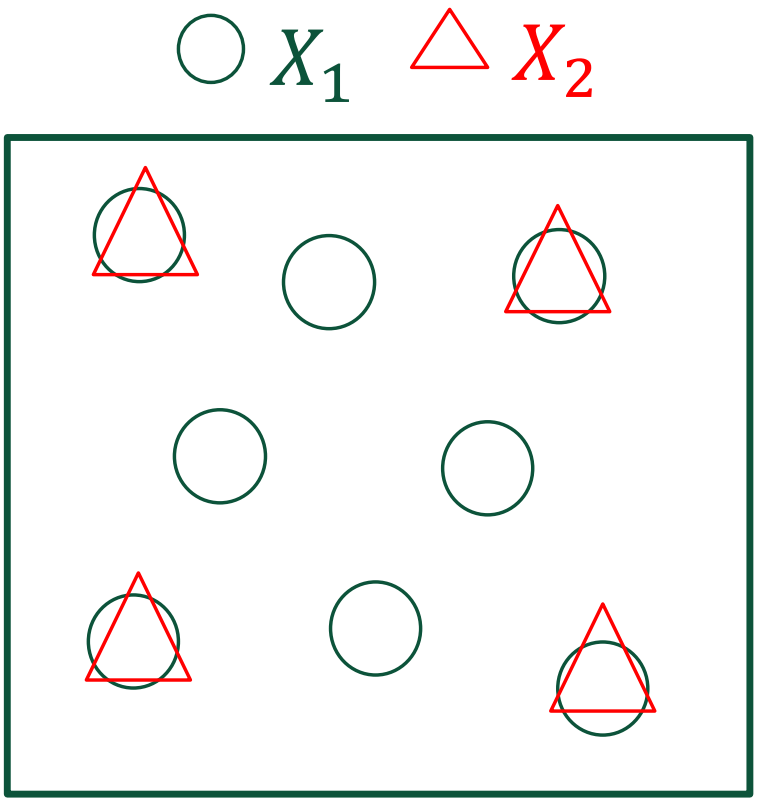
- Nested design, i.e.,

$$\mathcal{X}_L \subseteq \mathcal{X}_{L-1} \subseteq \cdots \subseteq \mathcal{X}_1 \subseteq \Omega,$$

  and $\mathbf{x}_i^{[l]} = \mathbf{x}_i^{[l-1]}$ for $i = 1, \ldots, n_l$.

- The nested property leads to more efficient inference in various multi-fidelity emulation approaches (Qian et al., 2009; Qian, 2009).

Nested Design

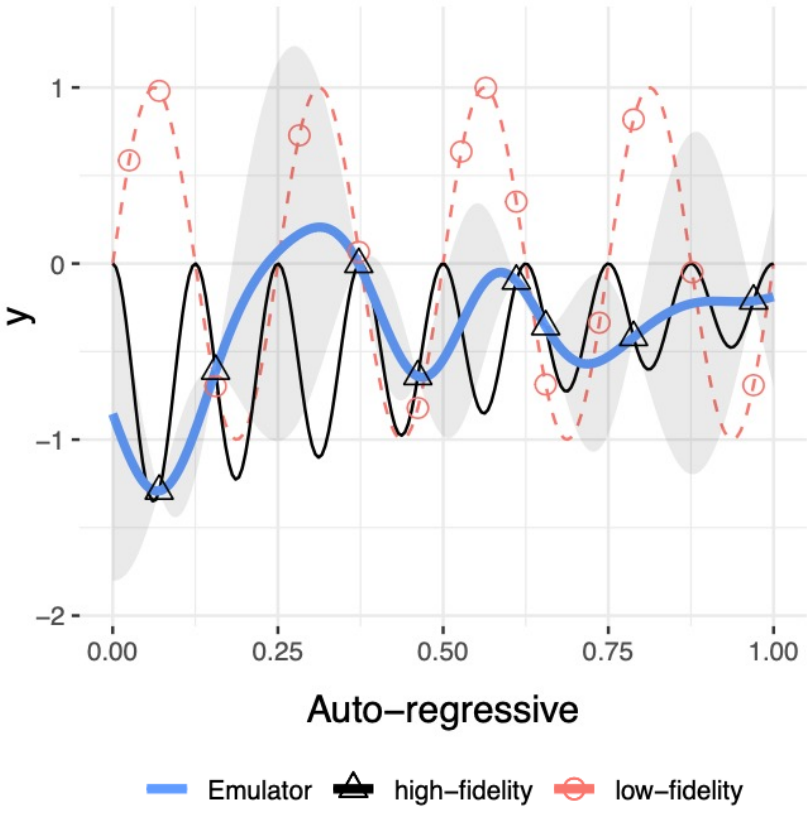


Existing Methods

- Q: Would it always follow an additive structure?

Existing Methods

- Q: Would it always follow an additive structure?



An example from Perdikaris et al. (2017), where $n_1 = 13$, $n_2 = 8$, $f_1(x) = \sin(8\pi x)$, and $f_2(x) = (x - \sqrt{2})f_1^2(x)$.

Heo, J. and **Sung, C.-L.** (2025)

Active learning for a recursive non-additive emulator for multi-fidelity computer experiments, *Technometrics*, 67(1), 58-72.

MICHIGAN STATE
UNIVERSITY

RNA Emulator

- We propose a Recursive Non-Additive emulator (RNA emulator) to overcome this limitation in a recursive fashion:

$$f_1(\mathbf{x}) = W_1(\mathbf{x}),$$

$$f_l(\mathbf{x}) = W_l(\mathbf{x}, f_{l-1}(\mathbf{x})), \quad l = 2, \cdots, L,$$

- The auto-regressive model ($f_l(\mathbf{x}) = \rho_{l-1} f_{l-1}(\mathbf{x}) + Z_l(\mathbf{x})$) becomes a special case!

RNA Emulator

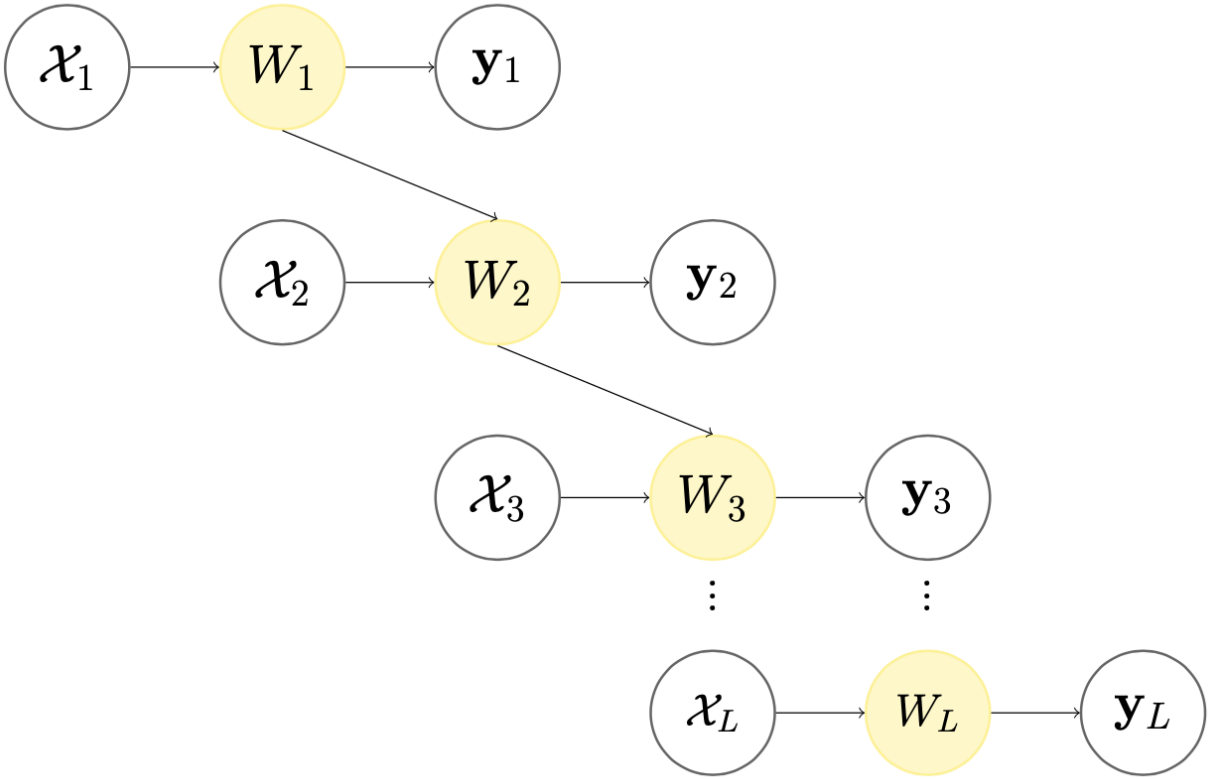
- We propose a Recursive Non-Additive emulator (RNA emulator) to overcome this limitation in a recursive fashion:

$$f_1(\mathbf{x}) = W_1(\mathbf{x}),$$

$$f_l(\mathbf{x}) = W_l(\mathbf{x}, f_{l-1}(\mathbf{x})), \quad l = 2, \cdots, L,$$

- The auto-regressive model ($f_l(\mathbf{x}) = \rho_{l-1} f_{l-1}(\mathbf{x}) + Z_l(\mathbf{x})$) becomes a special case!
- Model the relationship $\{W_l\}_{l=1}^{L}$ using independent GP priors

RNA Emulator



RNA Emulator

RNA Emulator

$$W_1(\mathbf{x}) \sim \mathcal{GP}(\alpha_1, \tau_1^2 K_1(\mathbf{x}, \mathbf{x}')),$$

$$W_l(\mathbf{z}) \sim \mathcal{GP}(\alpha_l, \tau_l^2 K_l(\mathbf{z}, \mathbf{z}')), \quad l = 2, \cdots, L,$$

where $\mathbf{z} = (\mathbf{x}, y)$, and $K_1(\mathbf{x}, \mathbf{x}')$ and $K_l(\mathbf{z}, \mathbf{z}')$ are a positive definite kernel.

RNA Emulator

RNA Emulator

$$W_1(\mathbf{x}) \sim \mathcal{GP}(\alpha_1, \tau_1^2 K_1(\mathbf{x}, \mathbf{x}')),$$

$$W_l(\mathbf{z}) \sim \mathcal{GP}(\alpha_l, \tau_l^2 K_l(\mathbf{z}, \mathbf{z}')), \quad l = 2, \cdots, L,$$

where $\mathbf{z} = (\mathbf{x}, y)$, and $K_1(\mathbf{x}, \mathbf{x}')$ and $K_l(\mathbf{z}, \mathbf{z}')$ are a positive definite kernel.

- e.g., squared exponential kernel:

$$K_1(\mathbf{x}, \mathbf{x}') = \prod_{j=1}^{d} \exp\left(-\frac{(x_j - x_j')^2}{\theta_{1j}}\right)$$

$$K_l(\mathbf{z}, \mathbf{z}') = \exp\left(-\frac{(y - y')^2}{\theta_{ly}}\right) \prod_{j=1}^{d} \exp\left(-\frac{(x_j - x_j')^2}{\theta_{lj}}\right)$$

Gaussian Process (GP)

- The observed simulations $\mathbf{y}_l$ follow a multivariate normal distribution:

$$\mathbf{y}_1 = W_1(\mathcal{X}_1) \sim \mathcal{N}_{n_1}(\alpha_1 \mathbf{1}_{n_1}, \tau_1^2 K_1(\mathcal{X}_1)) \quad \text{and}$$
$$\mathbf{y}_l = W_l(\mathcal{X}_l, f_{l-1}(\mathcal{X}_l)) \sim \mathcal{N}_{n_l}(\alpha_l \mathbf{1}_{n_l}, \tau_l^2 K_l(\mathcal{X}_l, f_{l-1}(\mathcal{X}_l))),$$

for $l = 2, \ldots, L$.

Gaussian Process (GP)

- The observed simulations $\mathbf{y}_l$ follow a multivariate normal distribution:

$$\mathbf{y}_1 = W_1(\mathcal{X}_1) \sim \mathcal{N}_{n_1}(\alpha_1 \mathbf{1}_{n_1}, \tau_1^2 K_1(\mathcal{X}_1)) \quad \text{and}$$
$$\mathbf{y}_l = W_l(\mathcal{X}_l, f_{l-1}(\mathcal{X}_l)) \sim \mathcal{N}_{n_l}(\alpha_l \mathbf{1}_{n_l}, \tau_l^2 K_l(\mathcal{X}_l, f_{l-1}(\mathcal{X}_l))),$$

  for $l = 2, \ldots, L$.

- $\{K_1(\mathcal{X}_1)\}_{ij} = K_1(\mathbf{x}_i^{[1]}, \mathbf{x}_j^{[1]})$
- $\{K_l(\mathcal{X}_l, f_{l-1}(\mathcal{X}_l))\}_{ij} = K_l((\mathbf{x}_i^{[l]}, f_{l-1}(\mathbf{x}_i^{[l]})), (\mathbf{x}_i^{[l]}, f_{l-1}(\mathbf{x}_i^{[l]})))$
- $f_{l-1}(\mathcal{X}_l) = (\mathbf{y}_{l-1})_{1:n_l}$ because of the nested assumption!
- The parameters $\{\alpha_l, \tau_l^2, \boldsymbol{\theta}_l\}_{l=1}^L$ can be estimated by maximum likelihood estimation

Posterior of $f_L(\mathbf{x})$ for a new input x

- Based on the properties of conditional multivariate normal distribution, it follows that

$$f_l(\mathbf{x})|\mathbf{y}_l, f_{l-1}(\mathbf{x}) \sim \mathcal{N}(\mu_l(\mathbf{x}, f_{l-1}(\mathbf{x})), \sigma_l^2(\mathbf{x}, f_{l-1}(\mathbf{x})))$$

for $l = 2, \ldots, L$ with

$$\mu_l(\mathbf{x}, f_{l-1}(\mathbf{x})) = \alpha_l \mathbf{1}_{n_l} + \mathbf{k}_l^T(\mathbf{x}, f_{l-1}(\mathbf{x})) K_l(\mathcal{X}_l, f_{l-1}(\mathcal{X}_l))^{-1}(\mathbf{y}_l - \alpha_l \mathbf{1}_{n_l}),$$
$$\sigma_l^2(\mathbf{x}, f_{l-1}(\mathbf{x})) = \tau_l^2(1 - \mathbf{k}_l(\mathbf{x}, f_{l-1}(\mathbf{x}))^T K_l(\mathcal{X}_l, f_{l-1}(\mathcal{X}_l))^{-1}\mathbf{k}_l(\mathbf{x}, f_{l-1}(\mathbf{x}))).$$

Posterior of $f_L(\mathbf{x})$ for a new input $\mathbf{x}$

- Based on the properties of conditional multivariate normal distribution, it follows that

$$f_l(\mathbf{x})|\mathbf{y}_l, f_{l-1}(\mathbf{x}) \sim \mathcal{N}(\mu_l(\mathbf{x}, f_{l-1}(\mathbf{x})), \sigma_l^2(\mathbf{x}, f_{l-1}(\mathbf{x})))$$

for $l = 2, \ldots, L$ with

$$\mu_l(\mathbf{x}, f_{l-1}(\mathbf{x})) = \alpha_l \mathbf{1}_{n_l} + \mathbf{k}_l^T(\mathbf{x}, f_{l-1}(\mathbf{x})) K_l(\mathcal{X}_l, f_{l-1}(\mathcal{X}_l))^{-1}(\mathbf{y}_l - \alpha_l \mathbf{1}_{n_l}),$$
$$\sigma_l^2(\mathbf{x}, f_{l-1}(\mathbf{x})) = \tau_l^2(1 - \mathbf{k}_l(\mathbf{x}, f_{l-1}(\mathbf{x}))^T K_l(\mathcal{X}_l, f_{l-1}(\mathcal{X}_l))^{-1}\mathbf{k}_l(\mathbf{x}, f_{l-1}(\mathbf{x}))).$$

- Intractable posterior distribution $p(f_L(\mathbf{x})|\mathbf{y}_1, \ldots, \mathbf{y}_L)$ can be approximated by Monte Carlo integration.
- However, it can be computationally demanding!

The closed form expression of RNA emulator

Proposition 1: The closed-form expressions

- Under the **squared exponential kernel**, the posterior mean and variance can be obtained as follows (Kyzyurova et al., 2018; Ming and Guillas, 2021):

$$\mu_l^*(\mathbf{x}) := \mathbb{E}[f_l(\mathbf{x})|\mathbf{y}_1,\ldots,\mathbf{y}_l]$$
$$= \alpha_l + \sum_{i=1}^{n_l} r_i \prod_{j=1}^{d} \exp\left(-\frac{(x_j - x_{ij}^{[l]})^2}{\theta_{lj}}\right) \frac{1}{\sqrt{1+2\frac{\sigma_{l-1}^{*2}(\mathbf{x})}{\theta_{ly}}}} \exp\left(-\frac{(y_i^{[l-1]} - \mu_{l-1}^*(\mathbf{x}))^2}{\theta_{ly} + 2\sigma_{l-1}^{*2}(\mathbf{x})}\right),$$

$$\sigma_l^{*2}(\mathbf{x}) := \mathbb{V}[f_l(x)|\mathbf{y}_1,\ldots,\mathbf{y}_l] = \tau_l^2 - (\mu_l^*(\mathbf{x}) - \alpha_l)^2 +$$
$$\left(\sum_{i,k=1}^{n_l} \zeta_{ik}\left(r_i r_k - \tau_l^2(\mathbf{K}_l^{-1})_{ik}\right) \prod_{j=1}^{d} \exp\left(-\frac{(x_j - x_{ij}^{[l]})^2 + (x_j - x_{kj}^{[l]})^2}{\theta_{lj}}\right)\right).$$

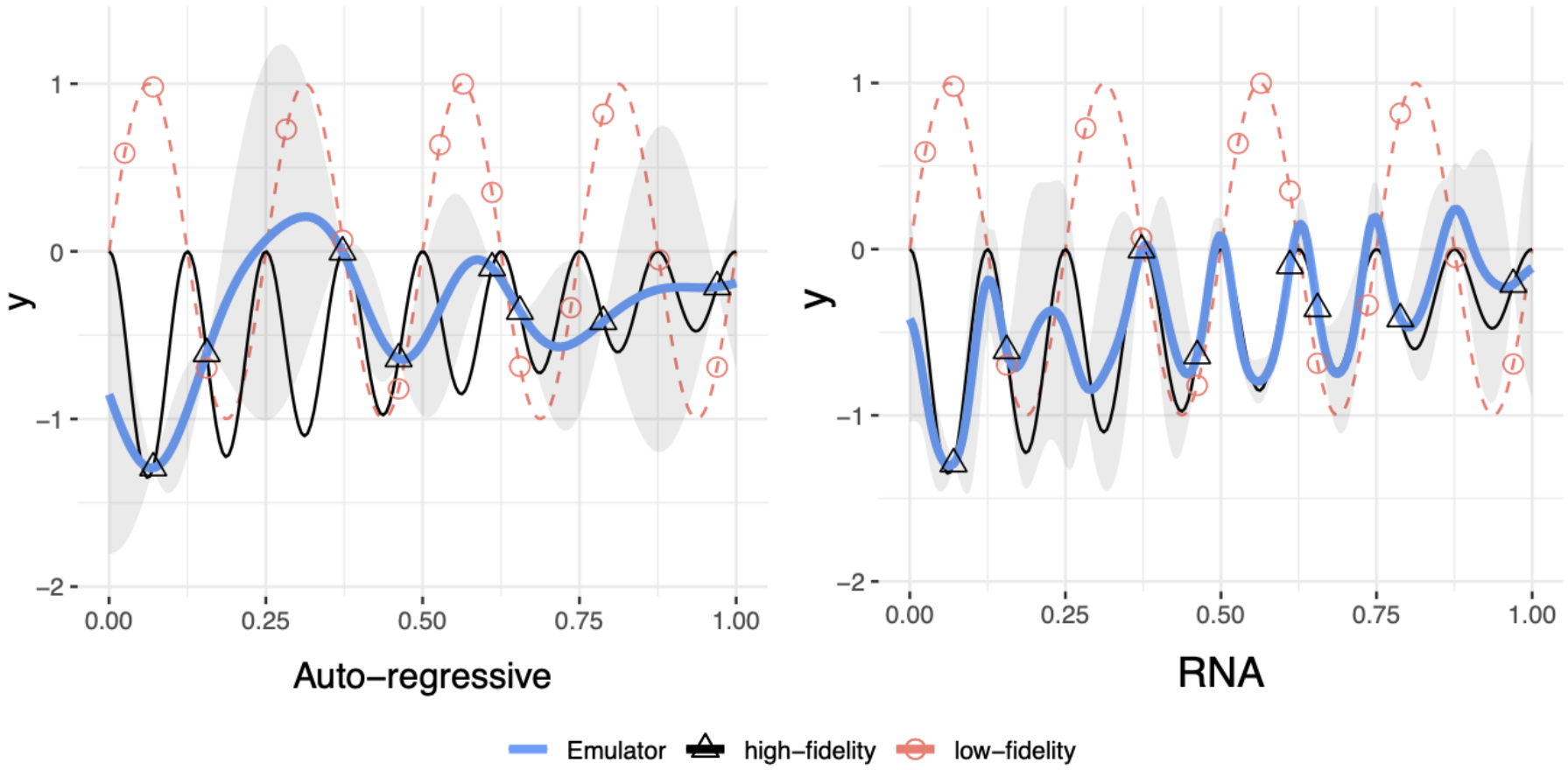
The closed form expression of RNA emulator

- The closed form expressions can be derived under a Matérn kernel with the smoothness parameters $\nu = 1.5$ and $\nu = 2.5$ as well.

The closed form expression of RNA emulator

- The closed form expressions can be derived under a Matérn kernel with the smoothness parameters $\nu = 1.5$ and $\nu = 2.5$ as well.
- Adopt the moment matching method to approximate the posterior distribution. That is, $f_L(\mathbf{x})|\mathbf{y}_1, \ldots, \mathbf{y}_L \sim \mathcal{N}(\mu_L^*(\mathbf{x}), \sigma_L^{*2}(\mathbf{x}))$.
- R package called RNAmf is available.

RNA emulator



Active Learning for RNA emulator

- Active learning (also known as Sequential Design)
  - sequentially searches for and acquires new data points at optimal location by a given criterion.
- In multi-fidelity simulation, active learning requires
  - identifying optimal input locations,
  - identifying fidelity levels,
  - accounting for the respective simulation costs simultaneously.

Active Learning for RNA emulator

- Active learning (also known as Sequential Design)
  - sequentially searches for and acquires new data points at optimal location by a given criterion.
- In multi-fidelity simulation, active learning requires
  - identifying optimal input locations,
  - identifying fidelity levels,
  - accounting for the respective simulation costs simultaneously.
- Four active learning strategies (ALD, ALM, ALC, and ALMC) for RNA emulator are introduced.

Active Learning for RNA emulator

- Active learning (also known as Sequential Design)
  - sequentially searches for and acquires new data points at optimal location by a given criterion.
- In multi-fidelity simulation, active learning requires
  - identifying optimal input locations,
  - identifying fidelity levels,
  - accounting for the respective simulation costs simultaneously.
- Four active learning strategies (ALD, ALM, ALC, and ALMC) for RNA emulator are introduced.
- The nested structure assumption implies that, in order to run the simulation $f_l(\mathbf{x}^{[l]}_{n_l+1})$, we need to run $f_k(\mathbf{x}^{[k]}_{n_k+1})$ with $\mathbf{x}^{[k]}_{n_k+1} = \mathbf{x}^{[l]}_{n_l+1}$ for all $1 \leq k \leq l$.

Active Learning Decomposition (ALD)

- Select the next point that maximizes the posterior predictive variance $\sigma_L^{*2}(\mathbf{x})$.

Active Learning Decomposition (ALD)

- Select the next point that maximizes the posterior predictive variance $\sigma_L^{*2}(\mathbf{x})$.

- Suppose $L = 2$. We have

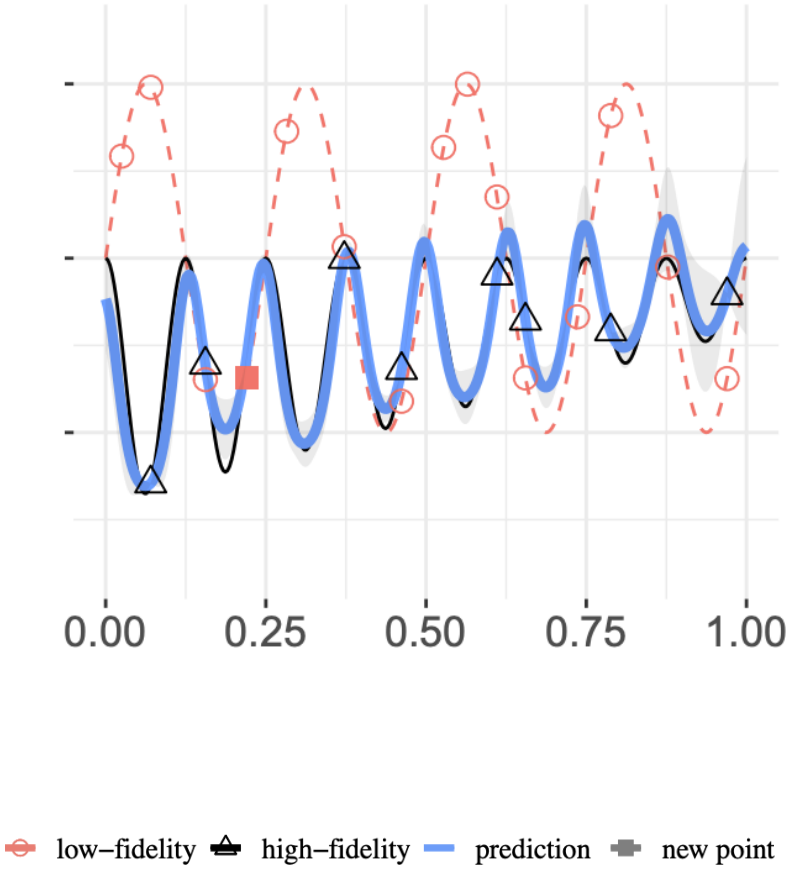
$$\begin{aligned}\sigma_2^{*2}(\mathbf{x}) =& \mathbb{V}\left[\mathbb{E}[f_2(\mathbf{x})|f_1(\mathbf{x}), \mathbf{y}_1, \mathbf{y}_2]\right] + \mathbb{E}\left[\mathbb{V}[f_2(\mathbf{x})|f_1(\mathbf{x}), \mathbf{y}_1, \mathbf{y}_2]\right] \\ :=& V_1(\mathbf{x}) + V_2(\mathbf{x})\end{aligned}$$

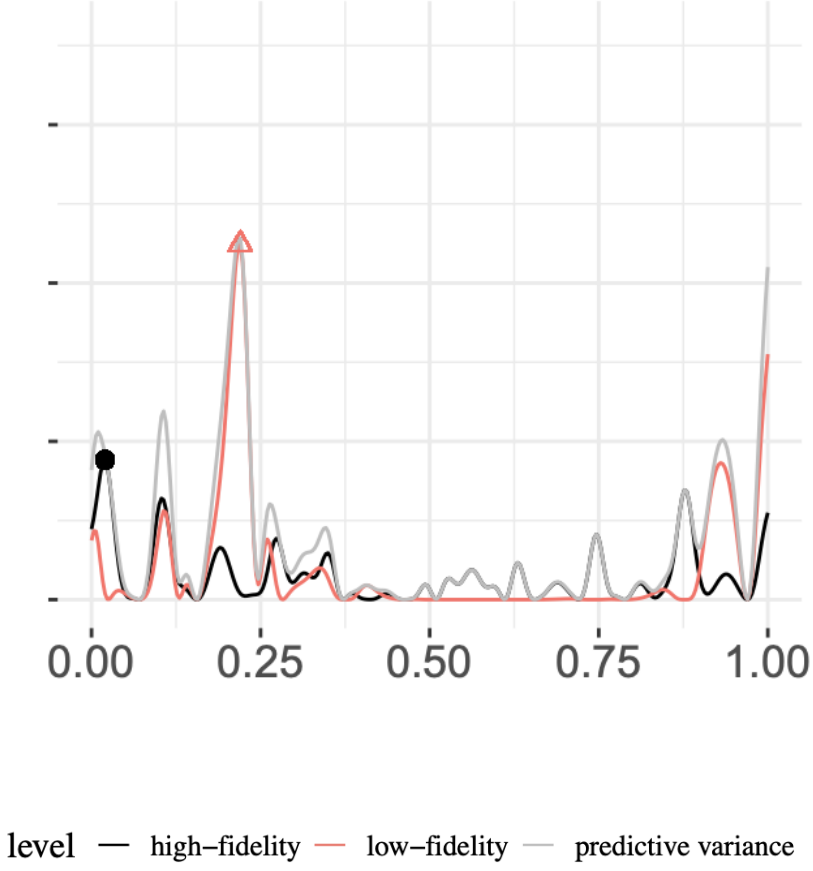
- To account for the simulation cost C_l, choose the next point $\mathbf{x}_{n_l+1}^{[l]}$ at level l by maximizing ALD criterion:

$$(l, \mathbf{x}_{n_l+1}^{[l]}) = \underset{k\in\{1,2\};\mathbf{x}\in\Omega}{\operatorname{argmax}} \frac{V_k(\mathbf{x})}{\sum_{j=1}^{k} C_j}.$$

- The closed-form expression facilitates the computation of ALD.

Active Learning Decomposition (ALD)





Active Learning MacKay (ALM)

- Select the next point that maximizes the posterior predictive variance $\sigma_l^{*2}(\mathbf{x})$ (MacKay, 1992).

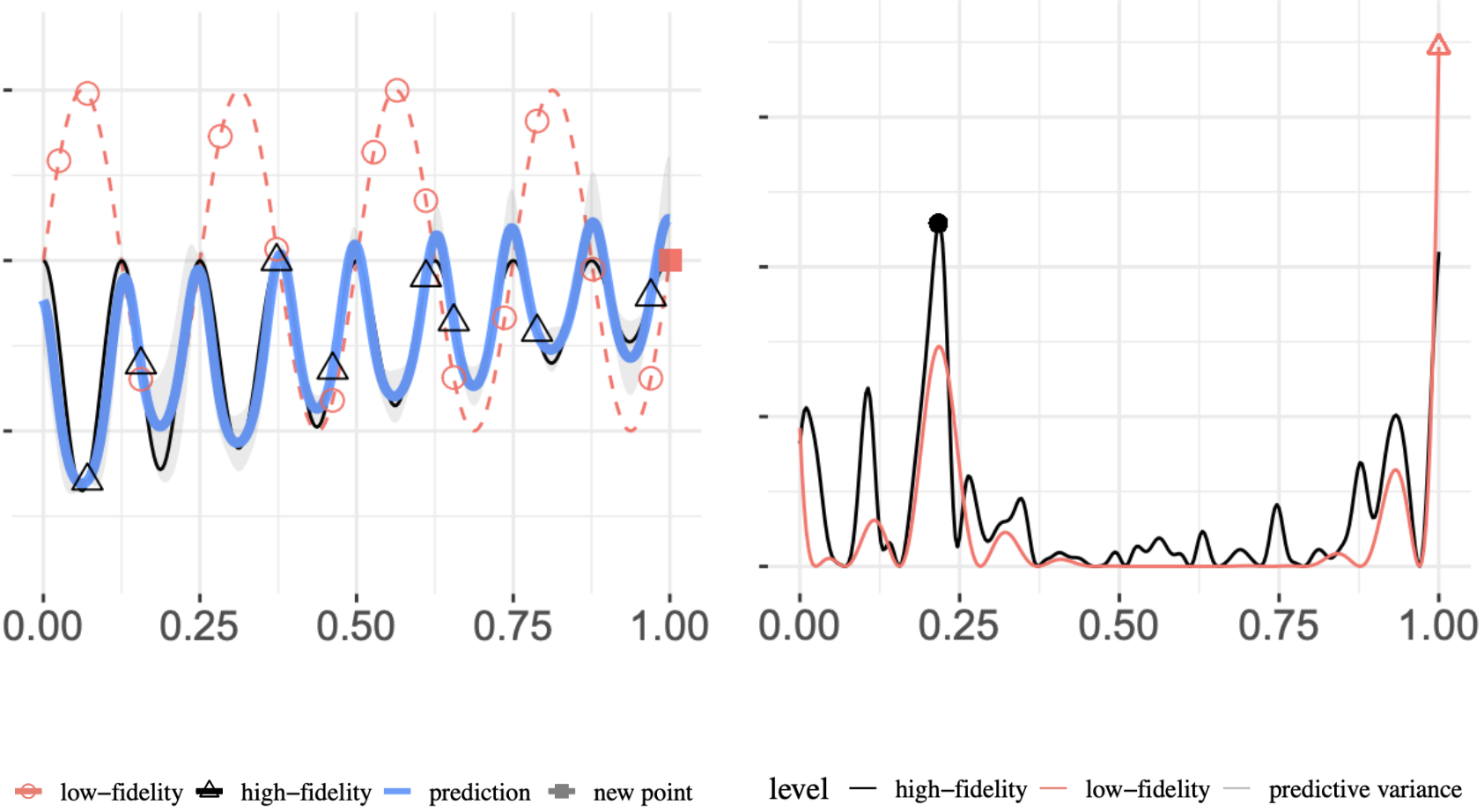
Active Learning MacKay (ALM)

- Select the next point that maximizes the posterior predictive variance $\sigma_l^{*2}(\mathbf{x})$ (MacKay, 1992).
- To account for the simulation cost C_l, choose the next point $\mathbf{x}_{n_l+1}^{[l]}$ at level l by maximizing ALM criterion:

$$(l, \mathbf{x}_{n_l+1}^{[l]}) = \underset{k \in \{1,\ldots,L\}; \mathbf{x} \in \Omega}{\operatorname{argmax}} \frac{\sigma_k^{*2}(\mathbf{x})}{\sum_{j=1}^{k} C_j}.$$

- The closed-form expression of $\sigma_k^{*2}(\mathbf{x})$ facilitates the computation of ALM criterion.

Active Learning MacKay (ALM)



Active Learning Cohn (ALC)

- Select an input location that maximizes the variance reduction **across the entire input space** after running this selected simulation (Cohn, 1993).

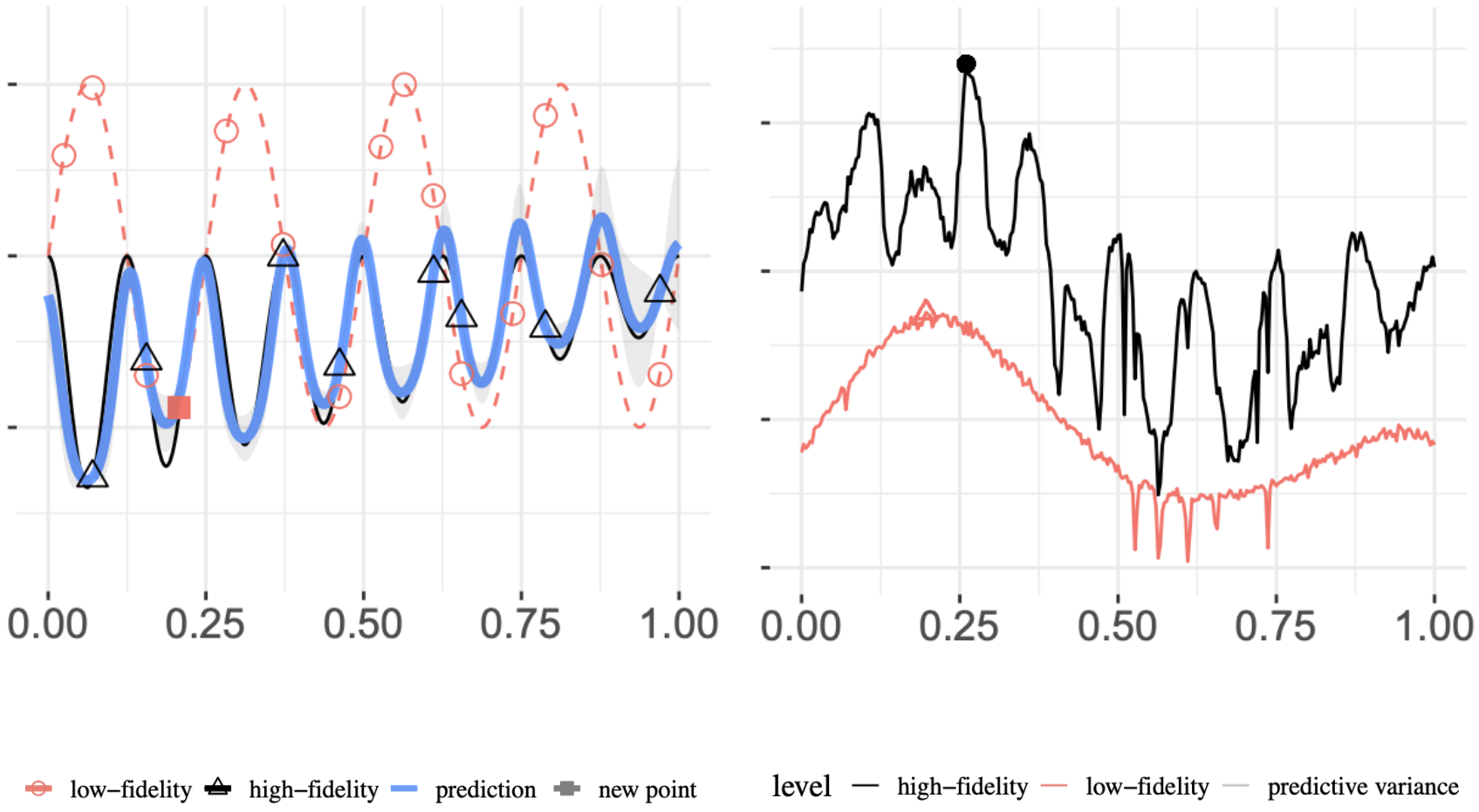
Active Learning Cohn (ALC)

- Select an input location that maximizes the variance reduction **across the entire input space** after running this selected simulation (Cohn, 1993).
- Choose the next point $\mathbf{x}_{n_l+1}$ at fidelity level l by minimize the ALC criterion:

$$(l, \mathbf{x}^{[l]}_{n_l+1}) = \arg_{k\in\{1,\ldots,L\}} \min_{\mathbf{x}\in\Omega} \frac{\int_\Omega \tilde{\sigma}^{*2}_L(\boldsymbol{\xi}; k, \mathbf{x})\mathrm{d}\boldsymbol{\xi}}{\sum_{j=1}^k C_j},$$

where the numerator is the **average in variance** (of the highest-fidelity emulator) with a choice of the fidelity level k and the input location $\mathbf{x}$.

Active Learning Cohn (ALC)



Two-step approach: ALMC

- Inspired by Le Gratiet and Cannamela (2015), consider the combination of ALM and ALC.

Two-step approach: ALMC

- Inspired by Le Gratiet and Cannamela (2015), consider the combination of ALM and ALC.
- First, the optimal input location is selected by maximizing the posterior predictive variance of the highest fidelity emulator:

$$\mathbf{x}^* = \underset{\mathbf{x}\in\Omega}{\operatorname{argmax}}\, \sigma_L^{*2}(\mathbf{x}).$$

Two-step approach: ALMC

- Inspired by Le Gratiet and Cannamela (2015), consider the combination of ALM and ALC.
- First, the optimal input location is selected by maximizing the posterior predictive variance of the highest fidelity emulator:

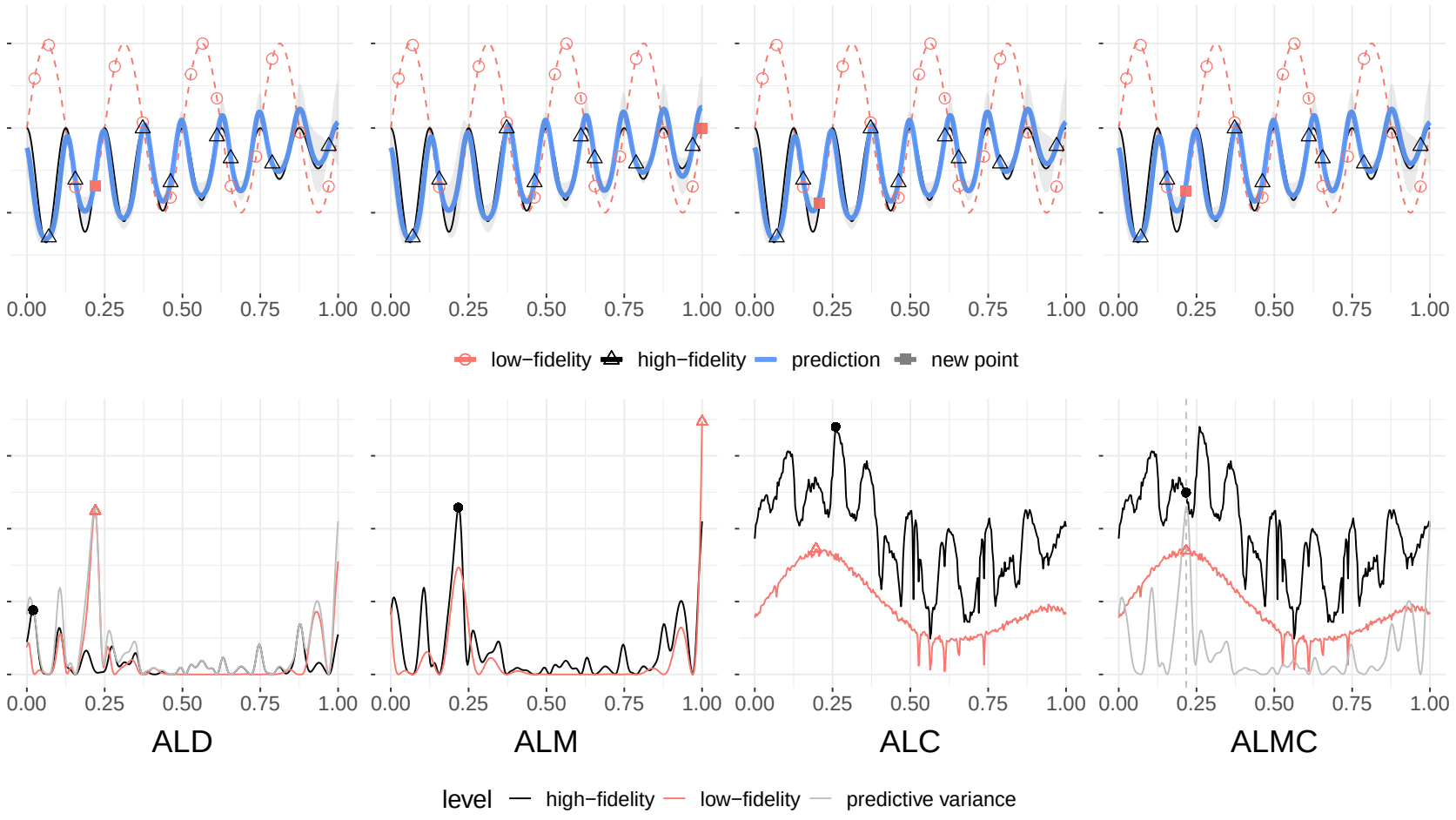
$$\mathbf{x}^* = \underset{\mathbf{x}\in\Omega}{\operatorname{argmax}}\, \sigma_L^{*2}(\mathbf{x}).$$

- Then, the ALC criterion determines the fidelity level with the identified input location:

$$l^* = \underset{l\in\{1,\ldots,L\}}{\operatorname{argmax}} \frac{\Delta\sigma_L^2(l,\mathbf{x}^*)}{\sum_{j=1}^{l} C_j},$$

which aims to maximize the ratio between the variance reduction and the associated simulation cost.

Demonstration



Summary of the Four Active Learning Strategies

- **ALM**: the influence of design augmentation on the variance of the highest-fidelity emulator (of f_L) is unclear, but evaluating the criterion is easy!

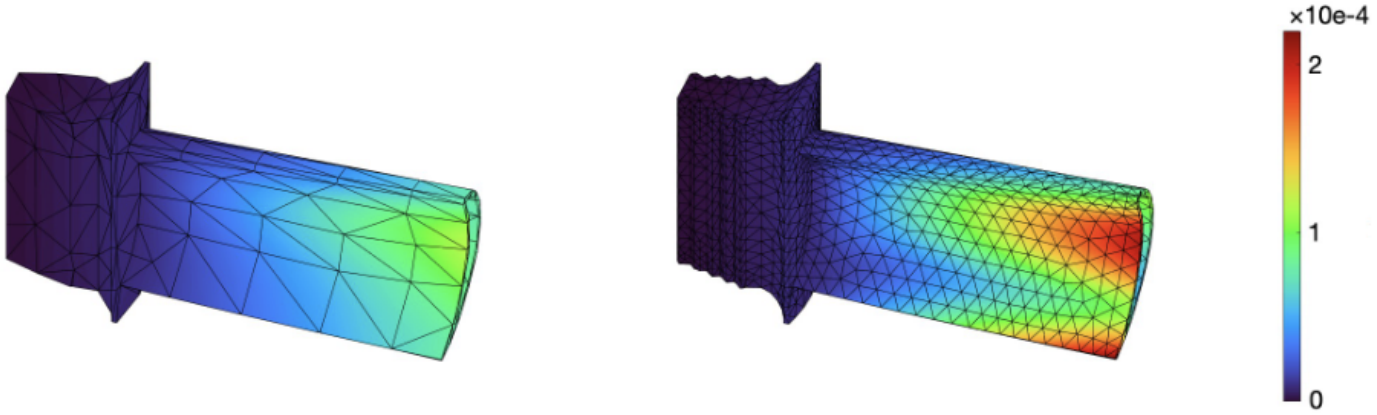
Summary of the Four Active Learning Strategies

- **ALM**: the influence of design augmentation on the variance of the highest-fidelity emulator (of f_L) is unclear, but evaluating the criterion is easy!
- **ALC**: maximize the reduction in variance of the highest-fidelity emulator (of f_L) but evaluating the criterion is quite computationally intensive.

Summary of the Four Active Learning Strategies

- **ALM**: the influence of design augmentation on the variance of the highest-fidelity emulator (of f_L) is unclear, but evaluating the criterion is easy!
- **ALC**: maximize the reduction in variance of the highest-fidelity emulator (of f_L) but evaluating the criterion is quite computationally intensive.
- Both **ALD** and **ALMC** generally emerge as favorable choices, offering accurate RNA emulators along with computational efficiency.

Revisit Motivated Example



Low-fidelity (left) and high-fidelity (right) simulations at $\mathbf{x} = (0.5, 0.45)$.

- **Input**: $\mathbf{x} = (x_1, x_2) = (\text{pressure}, \text{suction}) \in \Omega = [0.25, 0.75]^2$
- **Output**: $f(\mathbf{x})$: **maximum** of the thermal stress profile

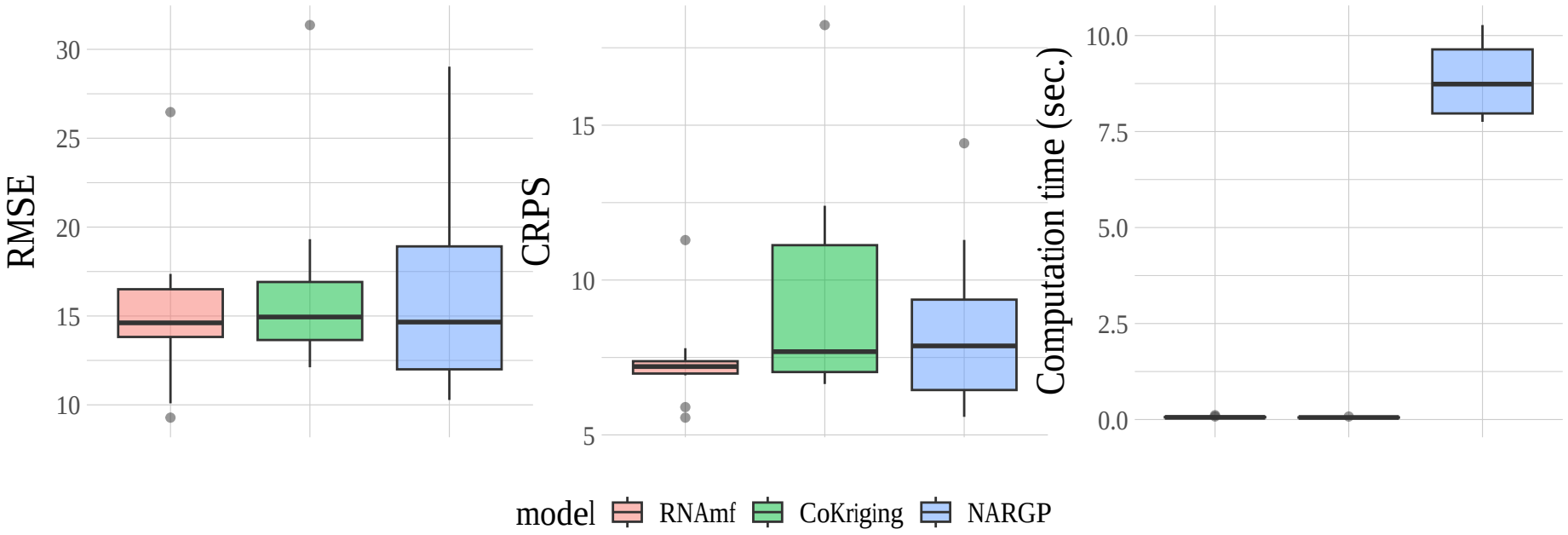
Revisit Motivated Example

- Perform the finite element simulations with $n_1 = 20$ and $n_2 = 10$.
- The simulation time of the finite element simulations, which are respectively $C_1 = 2.25$ and $C_2 = 6.85$ (seconds) will be used for active learning.

Revisit Motivated Example

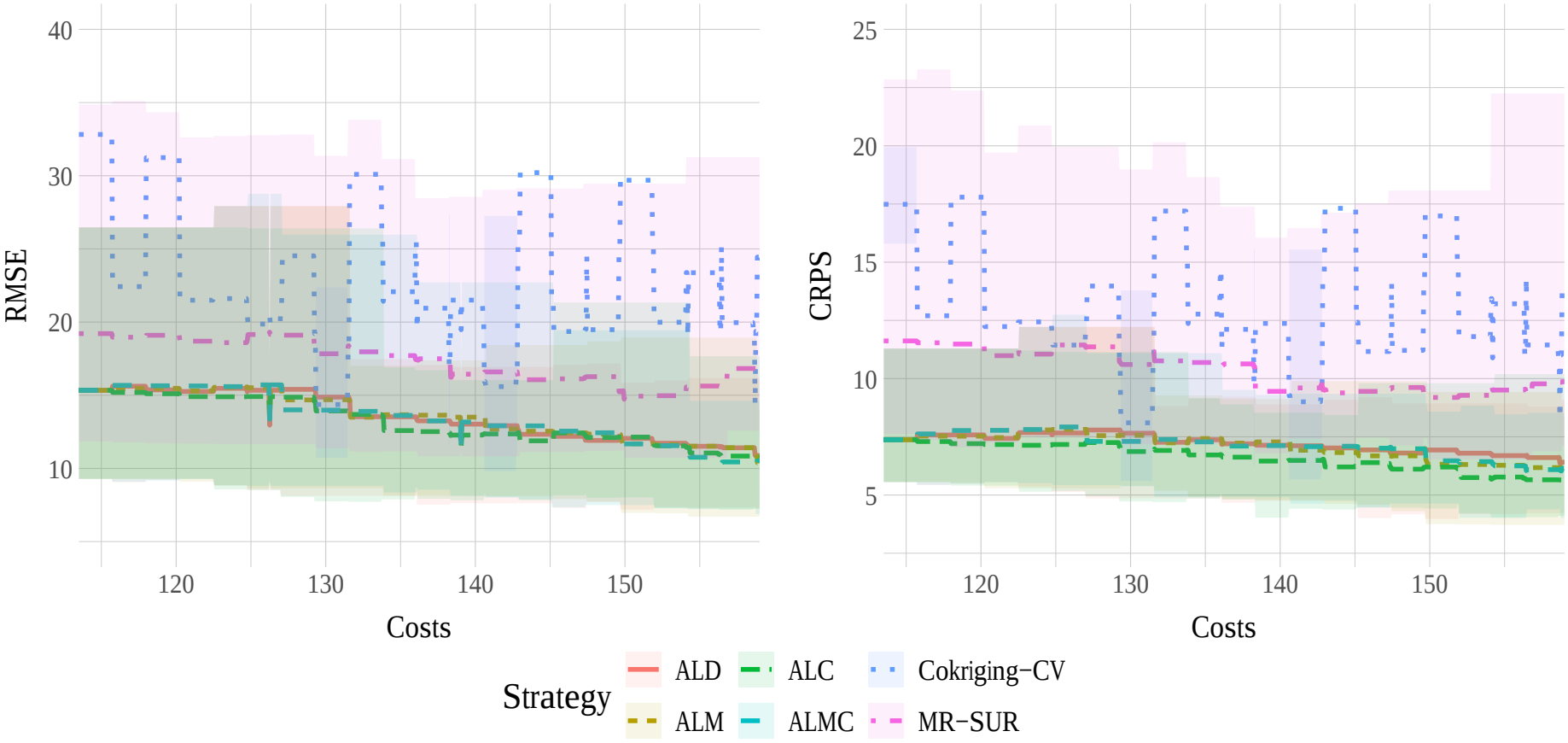
- Perform the finite element simulations with $n_1 = 20$ and $n_2 = 10$.
- The simulation time of the finite element simulations, which are respectively $C_1 = 2.25$ and $C_2 = 6.85$ (seconds) will be used for active learning.
- 10 repetitions with $n_{\text{test}} = 100$ random test input locations generated by a space-filling design.
- We perform finite element simulations using the Partial Differential Equation Toolbox in MATLAB.

Blade: Emulation performance



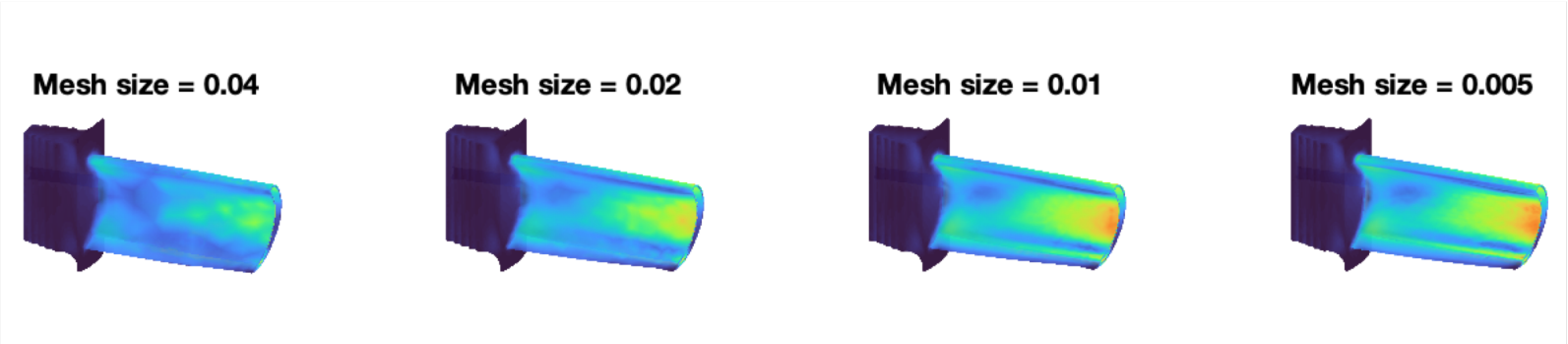
RMSE, CRPS, and computation time across 10 repetitions in the turbine blade application.

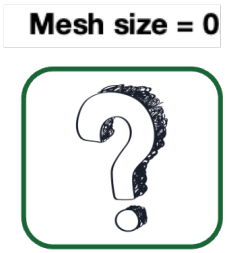
Blade: Active learning performance



RMSE and CRPS for the blade data with respect to the cost.

Discussion on Jet Blade Simulations

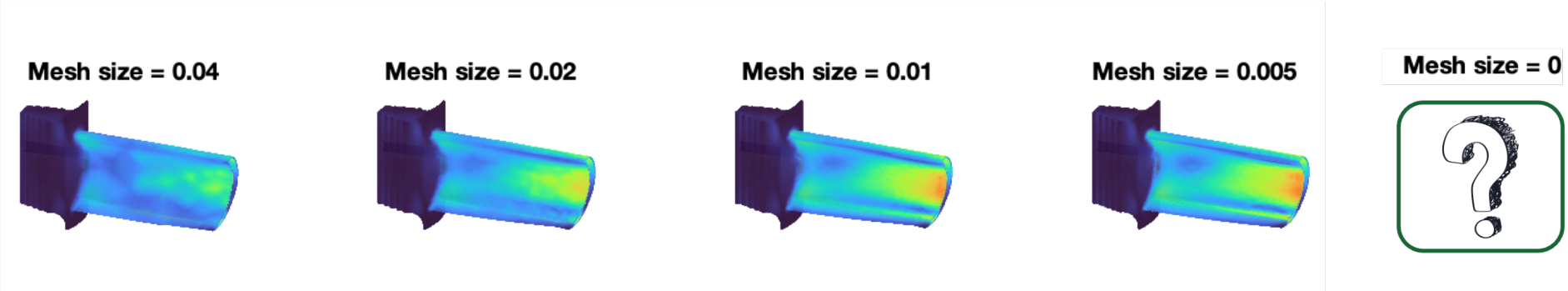




Jet blade simulations with different mesh configurations

- The fidelity level is often controlled by a tuning parameter (e.g., mesh size).

Discussion on Jet Blade Simulations



Jet blade simulations with different mesh configurations

- The fidelity level is often controlled by a tuning parameter (e.g., mesh size).
- Q: Can we account for the tuning parameter and extrapolate the **exact solution** of finite element simulations (i.e., mesh size $= 0$)?

Boutelet, R. and **Sung, C.-L.** (2025)

Active learning for finite element simulations with adaptive non-stationary kernel function, *arXiv:2503.23158*.

MICHIGAN STATE
UNIVERSITY

Non-Stationary Model

Non-Stationary Model (Tuo et al., 2014)

The response variable y, at input location $\mathbf{x} \in \mathcal{D}$ with mesh size $t \in \mathcal{T}$, is assumed to be:

$$y(\mathbf{x}, t) = \underbrace{\varphi(\mathbf{x})}_{\text{exact solution}} + \underbrace{\delta(\mathbf{x}, t)}_{\text{error}},$$

where $\varphi(\mathbf{x}) := y(\mathbf{x}, 0)$ and $\delta(\mathbf{x}, t)$ are realizations of two mutually independent GPs, respectively.

<u>Remark:</u> Since $y(\mathbf{x}, t)$ must equal to the exact solution $\varphi(\mathbf{x})$ as $t \to 0$, we need δ to satisfy $\delta(\mathbf{x}, t) \xrightarrow[t \to 0]{} 0$, for all $\mathbf{x}$.

Non-Stationary Model (Tuo et al., 2014)

- The **mean function** is assumed to have a separable form, such that

$$\mathbb{E}[\varphi(\mathbf{x})] = f_1^T(\mathbf{x})\beta_1, \quad \mathbb{E}[\delta(\mathbf{x}, t)] = f_2^T(\mathbf{x}, t)\beta_2$$

- The **covariance function** of our response variable is

$$K(\mathbf{x}, \mathbf{x}', t, t') = K_1(\mathbf{x}, \mathbf{x}') + K_2(\mathbf{x}, \mathbf{x}', t, t'),$$

where

Non-Stationary Model (Tuo et al., 2014)

- The **mean function** is assumed to have a separable form, such that

$$\mathbb{E}[\varphi(\mathbf{x})] = f_1^T(\mathbf{x})\beta_1, \quad \mathbb{E}[\delta(\mathbf{x}, t)] = f_2^T(\mathbf{x}, t)\beta_2$$

- The **covariance function** of our response variable is

$$K(\mathbf{x}, \mathbf{x}', t, t') = K_1(\mathbf{x}, \mathbf{x}') + K_2(\mathbf{x}, \mathbf{x}', t, t'),$$

where

  - $\varphi(\mathbf{x})$ has a stationary covariance function of the form

    $K_1(\mathbf{x}, \mathbf{x}') = \sigma_1^2 \prod_{i=1}^d e^{-\phi_1^2(x_i - x_i')^2}$

  - $\delta(\mathbf{x}, t)$ has a **non-stationary covariance function** of the form

    $K_2(\mathbf{x}, \mathbf{x}', t, t') = \sigma_2^2 K_H(t, t') \prod_{i=1}^d e^{-\phi_2^2(x_i - x_i')^2}$

Adaptive Non-Stationary Kernel

- Tuo et al. (2014) proposed a Brownian Motion (BM) kernel, $K_{\mathrm{BM}}(t, t') = \min(t, t')^l$.

Adaptive Non-Stationary Kernel

- Tuo et al. (2014) proposed a Brownian Motion (BM) kernel, $K_{\mathrm{BM}}(t, t') = \min(t, t')^l$.
- We introduce a new covariance function on the mesh size t, adapted from the Fractional Brownian Motion (FBM):

$$K_H(t, t') = \left\{ \frac{1}{2}(t^{2H} + (t')^{2H} + |t - t'|^{2H}) \right\}^{\frac{l}{2H}}, \quad 0 \leq H \leq 1.$$

The parameter H can be estimated using MLE.

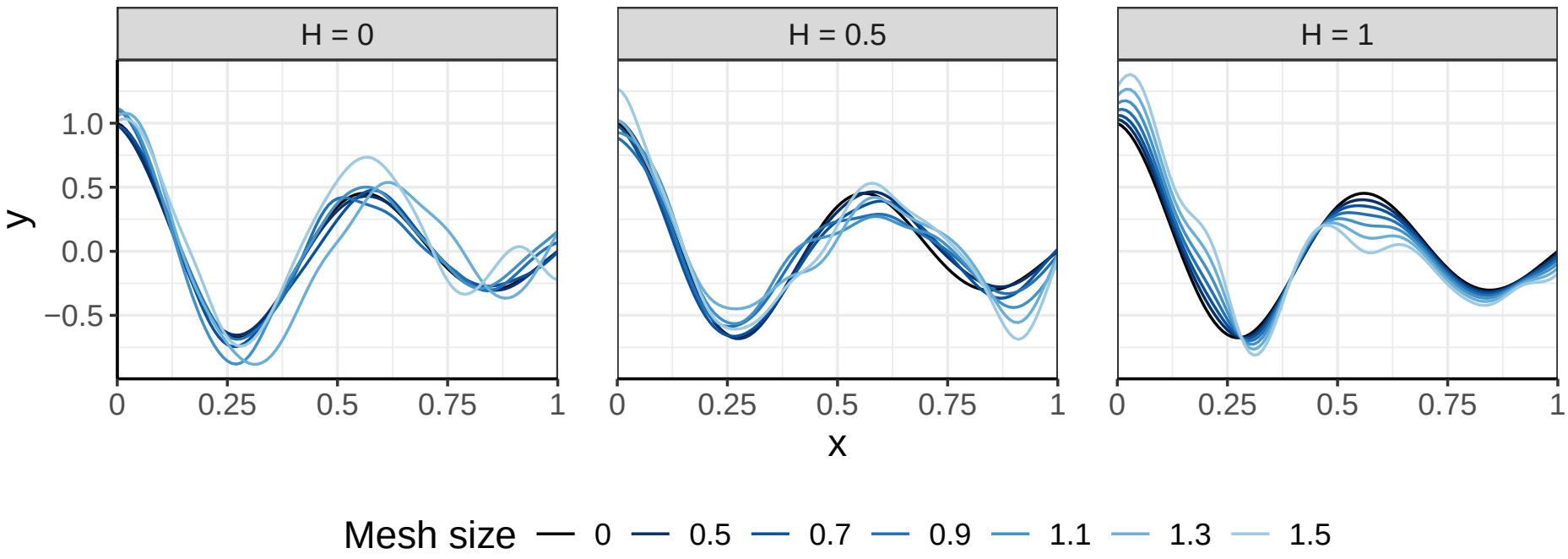
Adaptive Non-Stationary Kernel

- Tuo et al. (2014) proposed a Brownian Motion (BM) kernel, $K_{\mathrm{BM}}(t, t') = \min(t, t')^l$.
- We introduce a new covariance function on the mesh size t, adapted from the Fractional Brownian Motion (FBM):

$$K_H(t, t') = \left\{ \frac{1}{2}(t^{2H} + (t')^{2H} + |t - t'|^{2H}) \right\}^{\frac{l}{2H}}, \quad 0 \leq H \leq 1.$$

  The parameter H can be estimated using MLE.
- This BM kernel becomes a special case of FBM kernel ($H = 0.5$).

Adaptive Non-Stationary Kernel



Sample paths of the non-stationary model using the FBM kernel with three distinct values of H.

Active Learning for Finite Element Simulations

- We employ the Integrated Mean Squared Prediction Error (IMSPE) as the foundation of our active learning criterion.
- Specifically, the IMSPE from the n design points, I_n, can be written as

$$I_n := \mathrm{IMSPE}(\mathbf{X}_n, \mathbf{t}_n) = \int_{\mathbf{x} \in \mathcal{D}} \sigma_n^2(\mathbf{x}, 0) d\mathbf{x}.$$

Active Learning for Finite Element Simulations

- We employ the Integrated Mean Squared Prediction Error (IMSPE) as the foundation of our active learning criterion.
- Specifically, the IMSPE from the n design points, I_n, can be written as

$$I_n := \mathrm{IMSPE}(\mathbf{X}_n, \mathbf{t}_n) = \int_{\mathbf{x}\in\mathcal{D}} \sigma_n^2(\mathbf{x}, 0) d\mathbf{x}.$$

- Recall that $\sigma_n^2(\mathbf{x}, 0)$ is the **predictive variance** for the exact solution (i.e., $y(\mathbf{x}, t)$ at $t = 0$).

Active Learning for Finite Element Simulations

- Our active learning objective is to find the next best design location $(\mathbf{x}_{n+1}, t_{n+1})$ by minimizing $I_{n+1}(\mathbf{x}_{n+1}, t_{n+1}) := \mathrm{IMSPE}(\mathbf{X}_{n+1}, \mathbf{t}_{n+1})$.

Active Learning for Finite Element Simulations

- Our active learning objective is to find the next best design location $(\mathbf{x}_{n+1}, t_{n+1})$ by minimizing $I_{n+1}(\mathbf{x}_{n+1}, t_{n+1}) := \mathrm{IMSPE}(\mathbf{X}_{n+1}, \mathbf{t}_{n+1})$.

Theorem: IMSPE Reduction

The IMSPE associated with an additional design point $(\tilde{\mathbf{x}}, \tilde{t})$ given the current design $(\mathbf{X}_n, \mathbf{t}_n)$ can be written in an iterative form as (Binois et al., 2019)

$$I_{n+1}(\tilde{\mathbf{x}}, \tilde{t}) = I_n - R_{n+1}(\tilde{\mathbf{x}}, \tilde{t})$$

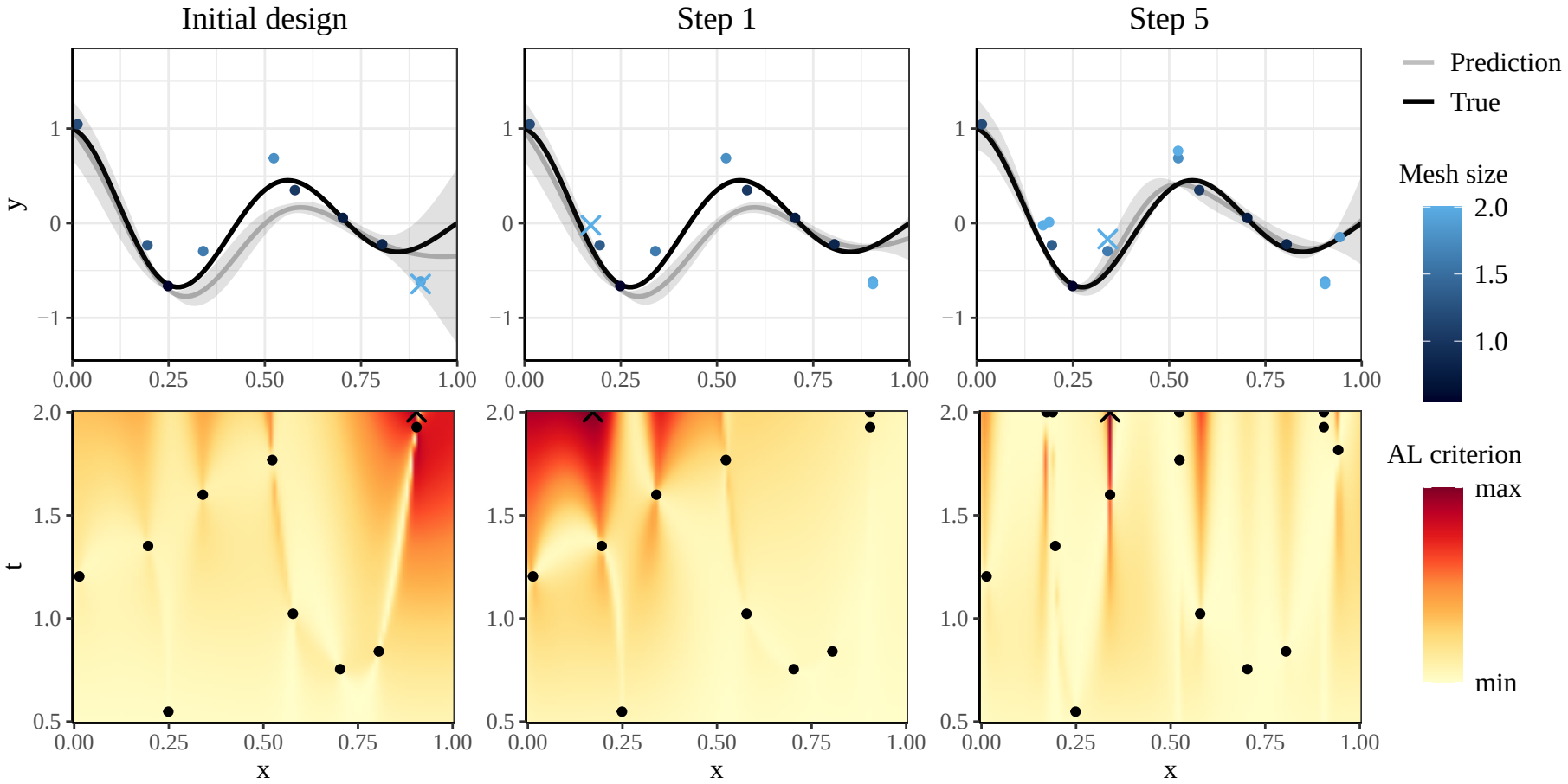
where $R_{n+1}(\tilde{\mathbf{x}}, \tilde{t})$, the **IMSPE reduction**, has a closed-form expression and can be computed with an $\mathcal{O}(n^2)$ cost complexity.

Cost Adjusted IMSPE Reduction

- To take the **computational cost** into account for our criterion, we choose the next point $(\mathbf{x}_{n+1}, t_{n+1})$ by maximizing the ratio between the IMSPE reduction and the cost:

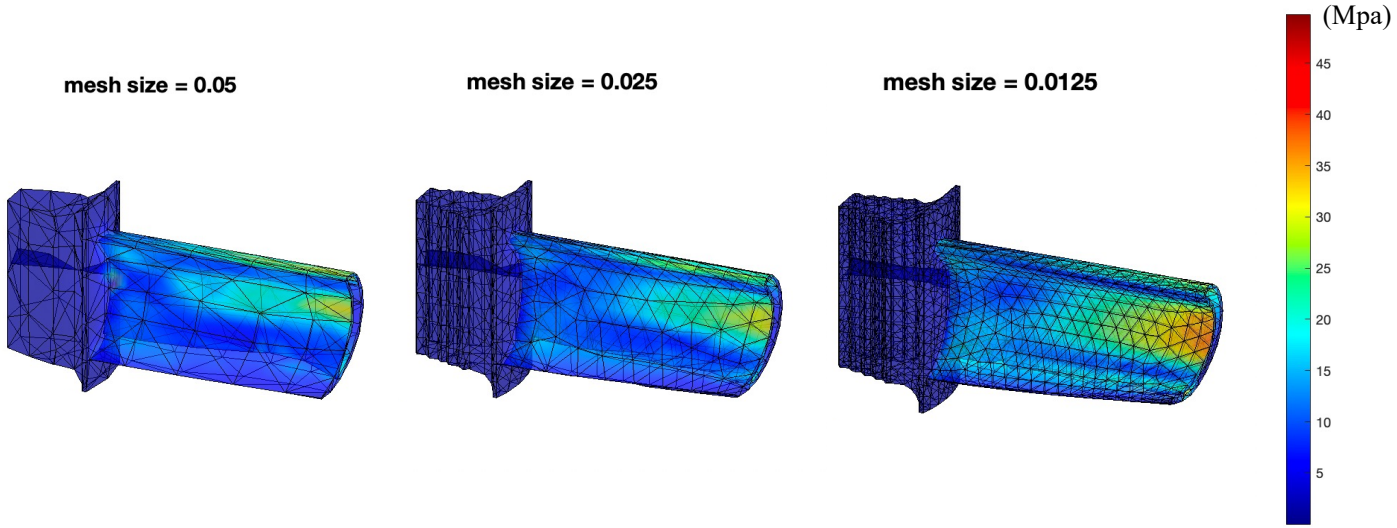
$$(\mathbf{x}_{n+1}, t_{n+1}) = \arg \max_{(\tilde{\mathbf{x}}, \tilde{t}) \in \mathcal{X} \times \mathcal{T}} \frac{R_{n+1}(\tilde{\mathbf{x}}, \tilde{t})}{C(\tilde{t})}.$$

Demonstration



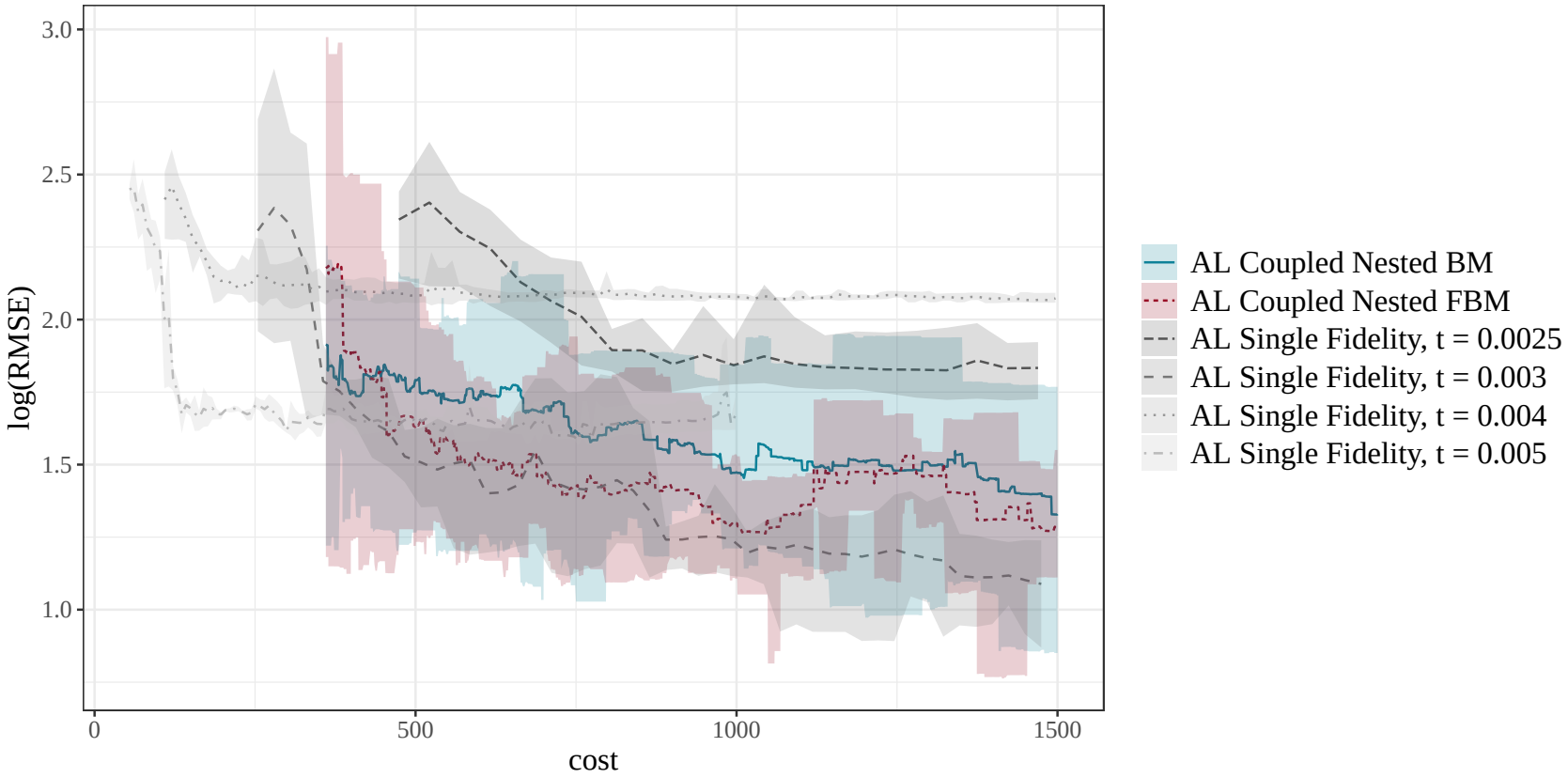
Prediction of our model (top), and active learning criterion surface (bottom). The points represent the current design locations (•), and the best next design point according to the criterion (×).

Revisit Motivated Example



- **Input**: $\mathbf{x} = (x_1, x_2) = (\text{pressure}, \text{suction}) \in \Omega = [0.25, 0.75]^2$
- **Output**: $f(\mathbf{x})$: **maximum** of the thermal stress profile
- **Test data**: Simulations with mesh size $t = 0.001$ at 50 uniform test input locations are conducted to examine the performance

Revisit Motivated Example



RMSE (in logarithmic scale) for the jet engine turbine blade case study with respect to the simulation cost. Solid lines indicate the average over 5 repetitions, while shaded regions represent the range.

Conclusion

- We propose a new, flexible model (RNA emulator) and the corresponding active learning strategies for multi-fidelity simulations with **discrete** fidelity levels.
- We introduce a new, adaptive non-stationary kernel function (FBM kernel) and the IMSPE-based active learning for multi-fidelity simulations with **continuous** fidelity levels.
- `R` packages are available for both works.

Thank You!

Thank NSF DMS 2113407 and 2338018 for supporting this work.

MICHIGAN STATE
UNIVERSITY

Binois, M., Huang, J., Gramacy, R. B., and Ludkovski, M. (2019). Replication or exploration? sequential design for stochastic simulation experiments. *Technometrics*, 61(1):7–23.

Cohn, D. (1993). Neural network exploration using optimal experiment design. *Advances in Neural Information Processing Systems*, 6:1071–1083.

Kennedy, M. C. and O'Hagan, A. (2000). Predicting the output from a complex computer code when fast approximations are available. *Biometrika*, 87(1):1–13.

Kyzyurova, K. N., Berger, J. O., and Wolpert, R. L. (2018). Coupling computer models through linking their statistical emulators. *SIAM/ASA Journal on Uncertainty Quantification*, 6(3):1151–1171.

Le Gratiet, L. (2013). Bayesian analysis of hierarchical multifidelity codes. *SIAM/ASA Journal on Uncertainty Quantification*, 1(1):244–269.

Le Gratiet, L. and Cannamela, C. (2015). Cokriging-based sequential design strategies using fast cross-validation techniques for multi-fidelity computer codes. *Technometrics*, 57(3):418–427.

Le Gratiet, L. and Garnier, J. (2014). Recursive co-kriging model for design of computer experiments with multiple levels of fidelity. *International Journal for Uncertainty Quantification*, 4(5):365–386.

MacKay, D. J. C. (1992). Information-based objective functions for active data selection. *Neural Computation*, 4(4):590–604.

Ming, D. and Guillas, S. (2021). Linked Gaussian process emulation for systems of computer models using Matérn kernels and adaptive design. *SIAM/ASA Journal on Uncertainty Quantification*, 9(4):1615–1642.

Perdikaris, P., Raissi, M., Damianou, A., Lawrence, N. D., and Karniadakis, G. E. (2017). Nonlinear information fusion algorithms for data-efficient multi-fidelity modelling. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473(2198):20160751.

Qian, P. Z. G. (2009). Nested latin hypercube designs. *Biometrika*, 96(4):957–970.

Qian, P. Z. G., Ai, M., and Wu, C. F. J. (2009). Construction of nested space-filling designs. *Annals of Statistics*, 37(6A):3616–3643.

Qian, P. Z. G. and Wu, C. F. J. (2008). Bayesian hierarchical modeling for integrating low-accuracy and high-accuracy experiments. *Technometrics*, 50(2):192–204.

Qian, Z., Seepersad, C. C., Joseph, V. R., Allen, J. K., and Wu, C. F. J. (2006). Building surrogate models based on detailed and approximate simulations. *Journal of Mechanical Design*, 128(4):668–677.

Tuo, R., Wu, C. F. J., and Yu, D. (2014). Surrogate modeling of computer experiments with different mesh densities. *Technometrics*, 56(3):372–380.