Active Learning for a Recursive Non-Additive Emulator for Multi-Fidelity Computer Experiments

Junoh Heo and Chih-Li Sung

Department of Statistics and Probability
Michigan State University

Joint Research Conference, 2024

This work is supported by NSF DMS 2113407 and 2338018.

Outline

Multi-fidelity simulation

- Computer simulations have been widely used in engineering and scientific research as valuable tools to understand complex systems.
- The simulation can be either
  - High-fidelity simulation: computationally expensive but accurate
  - Low-fidelity simulation: computationally cheaper but less accurate
  - (intermediate-fidelity simulation)

Motivated Example: Jet Engine Turbine Blade

- Thermal stress in a jet turbine engine blade under steady-state operating conditions (Carter, 2005; Wright and Han, 2006).
- The problem can be treated as a static structural computer model and can be solved *numerically* via finite element methods.
- **Input**: $\mathbf{x} = (x_1, x_2) = (\text{pressure}, \text{suction})$
- **Output**: $f(\mathbf{x})$: **maximum** of the thermal stress profile

Motivated Example: Jet Engine Turbine Blade

less accurate but cheaper   accurate but expensive

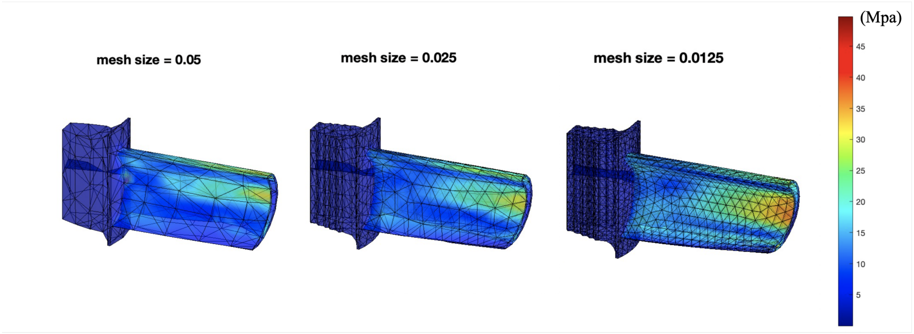


Illustration of finite element simulations at the input setting $\mathbf{x} = (0.5, 0.45)$.

Multi-fidelity simulation

- **Multi-fidelity emulation** is a computational technique by integrating simulations at multiple levels of fidelity
  - to yield an efficient predictive model
  - that maximizes the accurcy of model predictions,
  - while minimizing the simulation costs.
- Often called **emulator** or **surrogate model**.
- By strategically integrating these simulations, we can potentially improve accuracy without sacrificing excessive computational resources.

Problem setup

- Let $f_l(\mathbf{x})$ represent the simulation output of the computer code with input parameters $\mathbf{x} \in \Omega \subseteq \mathbb{R}^d$ at fidelity level $l = 1, \ldots, L$.
- 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 $l = 1, \ldots, L$

Problem setup

- Assume nested design, i.e.,

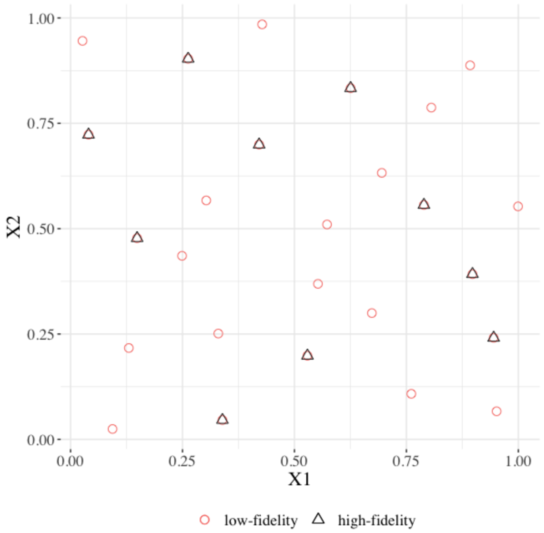
$$\mathcal{X}_L \subseteq \mathcal{X}_{L-1} \subseteq \cdots \subseteq \mathcal{X}_1 \subseteq \Omega,$$

  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, 2009; Qian et al., 2009; Haaland and Qian, 2010).

- C_l denote the simulation cost (e.g., in CPU hours) at fidelity level l with $0 < C_1 < C_2 < \ldots < C_L$.

Problem setup



Auto-regressive model

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

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

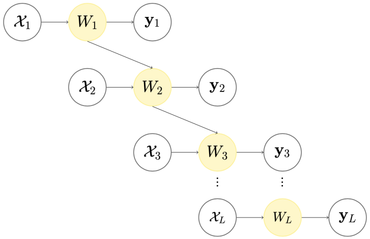
- Several extensions including Qian et al. (2006), Qian and Wu (2008), Le Gratiet (2013), Le Gratiet and Garnier (2014), and Perdikaris et al. (2017).

RNA emulator

- AR model may not adequately capture the nonlinear relationships between low- and high-fidelity data.
- We propose a Recursive Non-Additive emulator (called RNA emulator) to overcome this limitation in a recursive fashion:

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

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



RNA emulator

RNA emulator

- Model the relationship W_l using a GP prior, that is,

$$W_1(\mathbf{x}) \sim GP(\alpha_1(\mathbf{x}), \tau_1^2 \Phi_1(\mathbf{x}, \mathbf{x}')),$$

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

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

- Consider a constant mean, i.e., $\alpha_1(\mathbf{x}) = \alpha_1$ and $\alpha_l(\mathbf{z}) = \alpha_l$ for $l \geq 2$.

Nonlinear auto-regressive GP

- Nonlinear auto-regressive GP(NARGP) proposed by Perdikaris et al. (2017) also adopts the recursive scheme, but they rely on

1. Additive form of the kernel:

$$K_l(\mathbf{z},\mathbf{z}') = \Phi_{l1}(\mathbf{x},\mathbf{x}')\Phi_{l2}(f_{l-1}(\mathbf{x}), f_{l-1}(\mathbf{x}')) + \Phi_{l3}(\mathbf{x},\mathbf{x}'),$$

2. Monte Carlo integration for the intractable posterior distribution:

$$\begin{aligned} & p(f_l(\mathbf{x})|\mathbf{y}_1,\ldots,\mathbf{y}_l) \\ = & \int\cdots\int p(f_l(\mathbf{x})|\mathbf{y}_l, f_{l-1}(\mathbf{x}))p(f_{l-1}(\mathbf{x})|\mathbf{y}_{l-1}, f_{l-2}(\mathbf{x}))\cdots p(f_1(\mathbf{x})|\mathbf{y}_1)\mathrm{d}(f_{l-1}(\mathbf{x}))\ldots\mathrm{d}(f_1(\mathbf{x})). \end{aligned}$$

RNA emulator

- RNA emulator adopts the natural form of popular kernel choices:

$$K_1(\mathbf{z},\mathbf{z}') = \Phi(\mathbf{x},\mathbf{x}';\theta_1),$$
$$K_l(\mathbf{z},\mathbf{z}') = \phi(y,y';\theta_{ly})\prod_{j=1}^{d}\phi(x_j,x_j';\theta_{lj}), \quad l = 2,\cdots,L,$$

- With these kernel choices, RNA emulator has the closed form posterior mean and variance of $f_l(\mathbf{x})$ in a recursive fashion.

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):

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

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

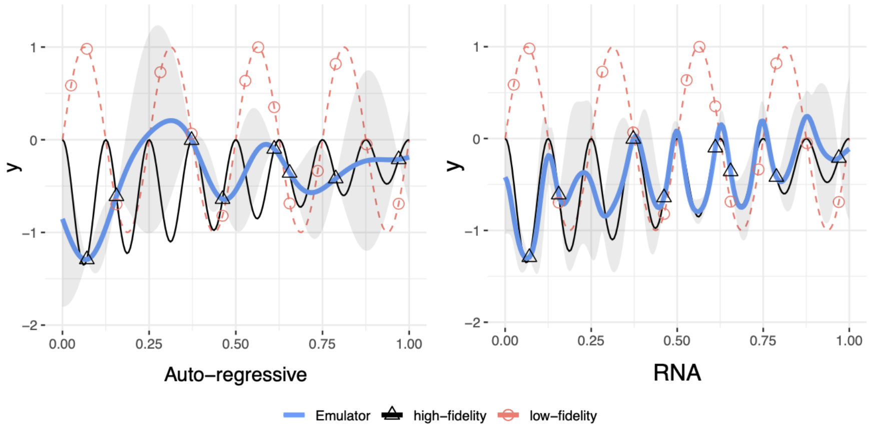
The closed form expression of RNA emulator

Proposition 2: Interpolation property

The RNA emulator exhibits interpolation property.

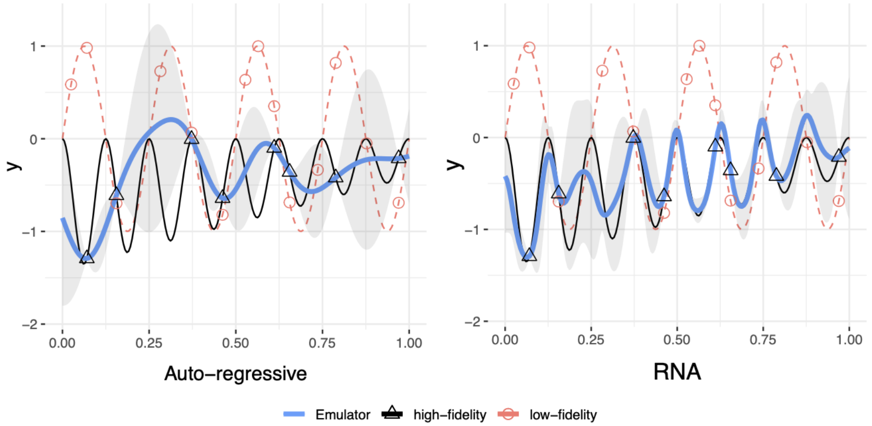
- The closed form expressions can be derived under a Matérn kernel with the smoothness parameter $\nu = 1.5$ and $\nu = 2.5$ as well.
- Adopt the moment matching method to approximate the posterior distribution.
- R package called RNAmf is available.

RNA emulator



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

After emulating...



However, the emulator still holds the uncertainty in some region!

Active Learning

- Active learning
  - is also known as sequential design,
  - sequentially searches for and acquires new data points at optimal location by a given criterion,
  - aims to achieve enhanced accuracy while managing the limited resources.
- Well-established for single-fidelity GP emulators, but research for multi-fidelity computer simulations is scarce and more challenging.

Active Learning for RNA emulator

- In multi-fidelity simulation, active learning requires
  - identifying optimal input locations,
  - identifying fidelity levels,
  - accounting for the respective simulation costs simultaneously.
- Three active learning strategies for RNA emulator will be introduced.
- The nested structure assumption implies that we need to run $f_l(\mathbf{x}^{[l]}_{n_l+1})$ with $\mathbf{x}^{[l]}_{n_l+1} = \mathbf{x}^{[l^*]}_{n_{l^*}+1}$ for all $1 \leq l \leq l^*$ to run the simulation $f_{l^*}(\mathbf{x}^{[l^*]}_{n_{l^*}+1})$.

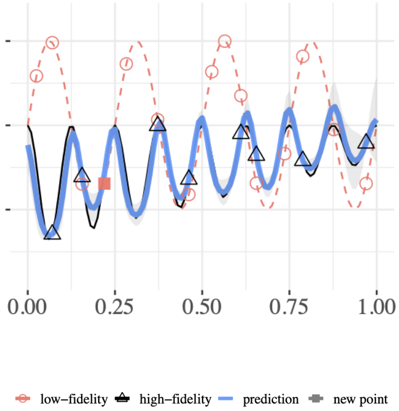
Active Learning MacKay

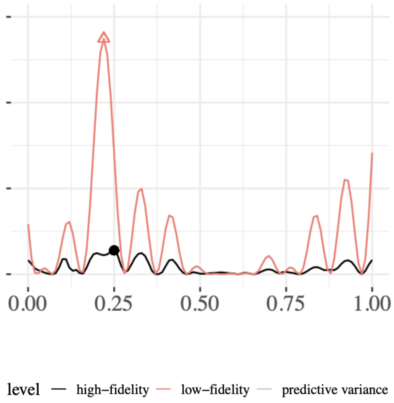
- Select the next point that maximizes the posterior predictive variance (MacKay, 1992).
- To account for the simulation cost C_l, choose the next point $\mathbf{x}^{[l]}_{n_l+1}$ at level l by maximizing ALM criterion:

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

- The closed-form expression facilitates the computation of ALM criterion.

Active Learning MacKay





Active Learning Cohn

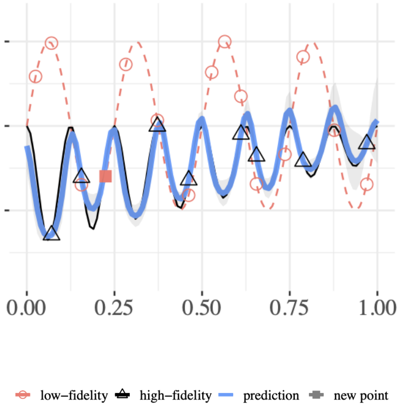
- Select an input location that maximizes the variance reduction **across the entire input space** after running this selected simulation (Cohn, 1993).

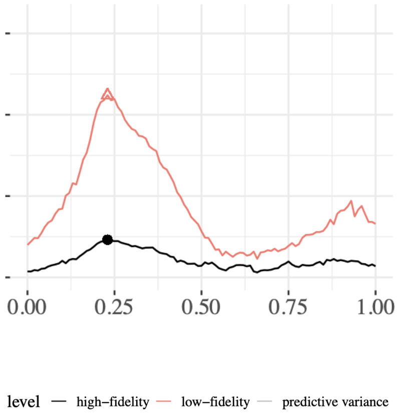
- Choose the next point $\mathbf{x}_{n_l+1}$ at fidelity level l by maximizing the ALC criterion:

$$(l^*, \mathbf{x}^{[l^*]}_{n_{l^*}+1}) = \underset{l\in\{1,\ldots,L\};\mathbf{x}\in\Omega}{\operatorname{argmax}} \frac{\Delta\sigma^2_L(l,\mathbf{x})}{\sum_{j=1}^{l} C_j},$$

where $\Delta\sigma^2_L(l,\mathbf{x}) = \int_\Omega \left\{\sigma^{*2}_L(\boldsymbol{\xi}) - \tilde{\sigma}^{*2}_L(\boldsymbol{\xi}; l, \mathbf{x})\right\} \mathrm{d}\boldsymbol{\xi}$ is the **average reduction in variance** (of the highest-fidelity emulator) with a choice of the fidelity level l and the input location $\mathbf{x}$.

Active Learning Cohn





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:

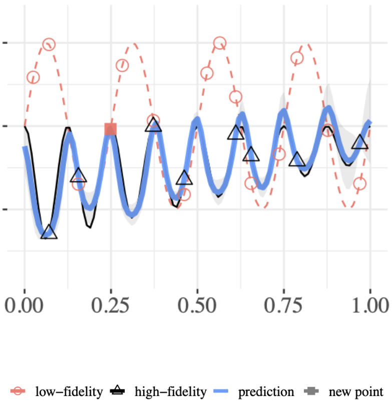
$$\mathbf{x}^* = \operatorname*{argmax}_{\mathbf{x}\in\Omega} \sigma_L^{*2}(\mathbf{x}).$$

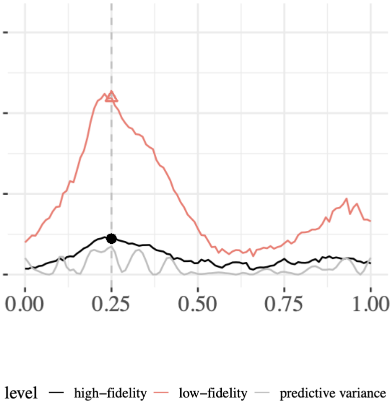
- Then, the ALC criterion determines the fidelity level with the identified input location:

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

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

Two-step approach: ALMC





Numerical studies: Emulation performance

- 6 different functions with 2 or 3 levels of fidelity.
- Compare proposed emulator `RNAmf` with `Cokriging` (Le Gratiet and Garnier, 2014) and `NARGP` (Perdikaris et al., 2017).
- 100 repetitions with $n_{\text{test}} = 1000$ random test input locations generated by space-filling designs.
- Evaluate the prediction performance based on two criteria: the root-mean-square error (RMSE) and continuous rank probability score (CRPS) (Gneiting and Raftery, 2007)

Numerical studies: Emulation performance

- Two-level Perdikaris function (Perdikaris et al., 2017),

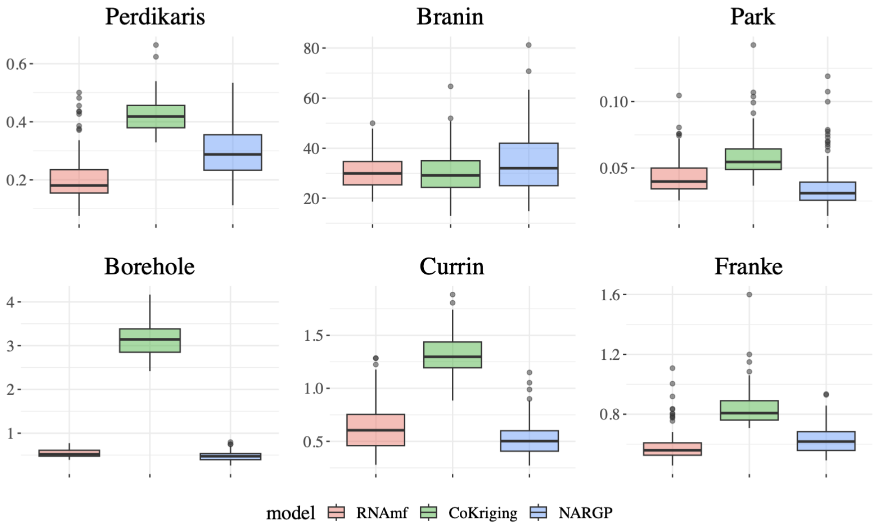
$$\begin{cases} f_1(x) = \sin(8\pi x) \\ f_2(x) = (x - \sqrt{2}) f_1^2(x) \end{cases} \text{for } x \in [0,1],$$

- Two-level Park function (Park, 1991; Xiong et al., 2013),

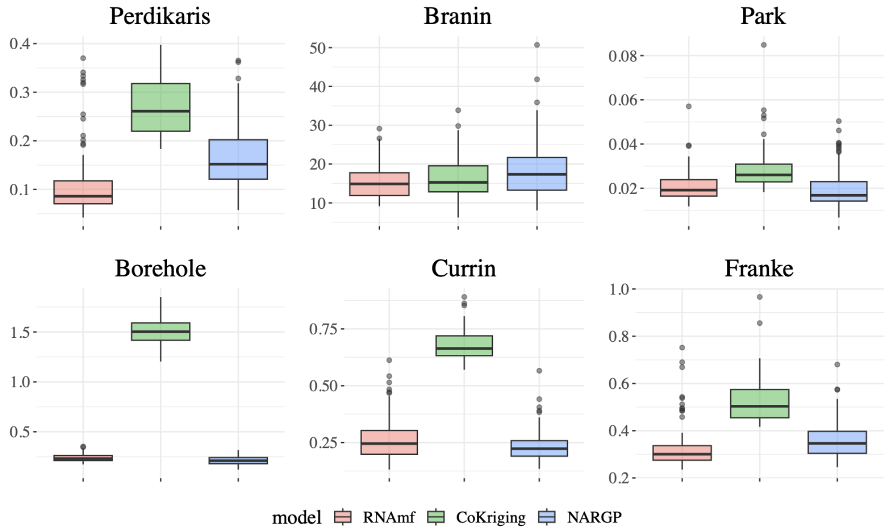
$$\begin{cases} f_1(\mathbf{x}) = f_2(\mathbf{x}) + \frac{\sin(x_1)}{10} f_2(\mathbf{x}) - 2x_1 + x_2^2 + x_3^2 + 0.5 \\ f_2(\mathbf{x}) = \frac{x_1}{2}\left[\sqrt{1 + (x_2 + x_3^2)\frac{x_4}{x_1^2}} - 1\right] + (x_1 + 3x_4)\exp\left(1 + \sin(x_3)\right) \end{cases} \text{for } \mathbf{x} \in [0,1]^4.$$

	Perdikaris	Branin	Park	Borehole	Currin	Franke
d	1	2	4	8	2	2
n_1	13	20	40	60	20	20
n_2	8	15	20	30	10	15
n_3		10				10

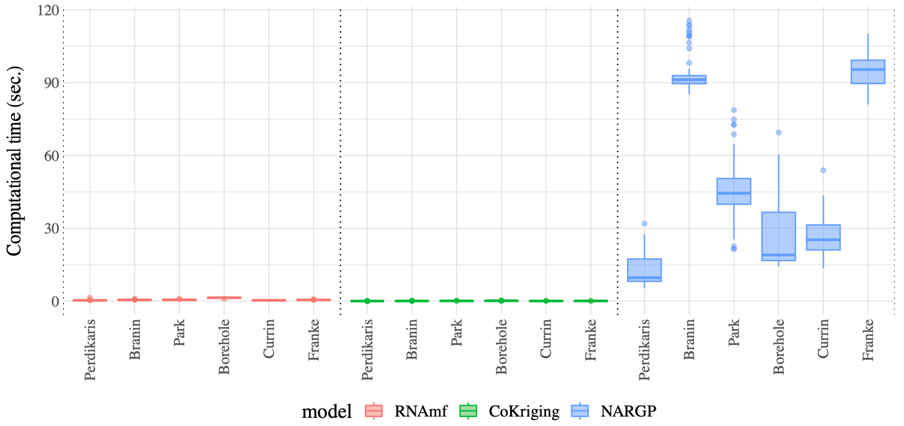
Numerical studies: RMSE



Numerical studies: CRPS



Numerical studies: Computational time

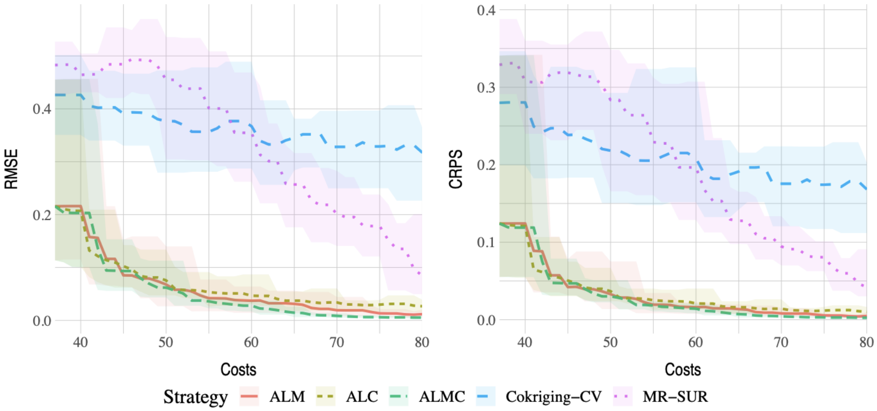


Computational time of six synthetic examples across 100 repetitions.

Numerical studies: Active learning performance

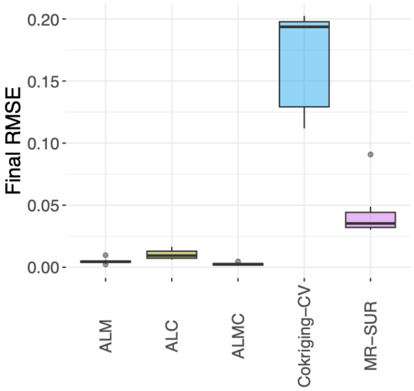
- Perdikaris function (1-dim, nonlinear).
- Compare three proposed strategies `ALM`, `ALC`, and `ALMC`, with a cokriging-based sequential design (`CoKriging-CV`) (Le Gratiet and Cannamela, 2015) and a sequential design maximizing the rate of stepwise uncertainty reduction using the AR model (`MR-SUR`) (Stroh et al., 2022).
- Simulation costs of low- and high-fidelity simulators are $C_1 = 1$ and $C_2 = 3$
- Total simulation budget of $C_{\text{total}} = 80$.
- 10 repetitions with different seeds.

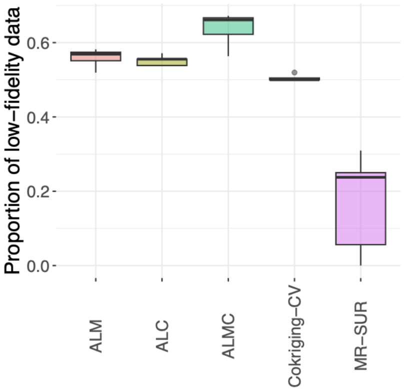
Numerical studies: Active learning performance



RMSE and CRPS for the Perdikaris function with respect to the simulation cost.

Numerical studies: Active learning performance





Final RMSE (left) and proportion of AL acquisitions choosing low-fidelity data (right).

Revisit motivated example

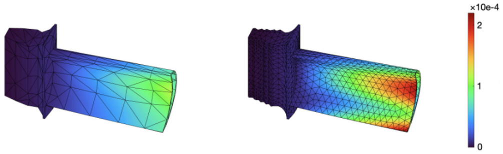


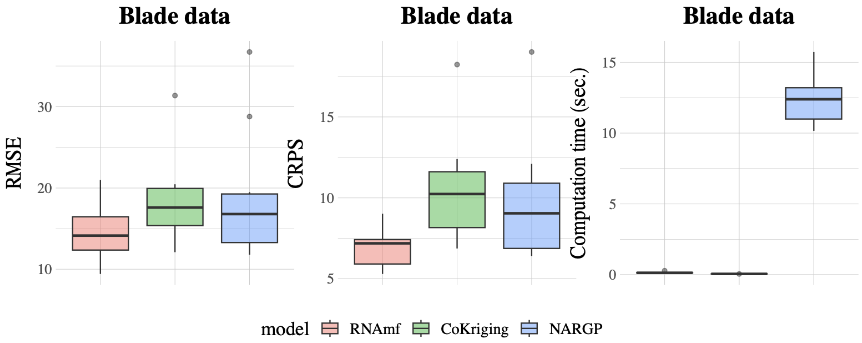
Illustration of low-fidelity (left, mesh size=0.05) and high-fidelity (right, mesh size=0.0125) finite element simulations at the input setting $\mathbf{x} = (0.5, 0.45)$.

- **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

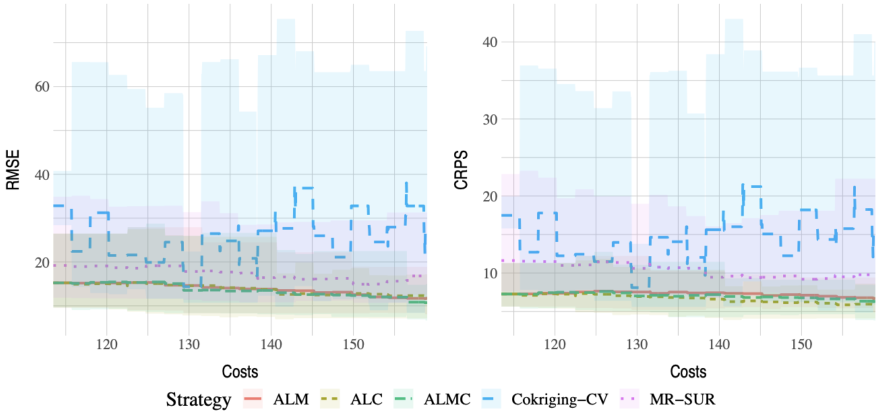
- Perform the finite element simulations with $n_1 = 20$ and $n_2 = 10$.
- The simulation time of the finite element simulations, which are respectively $C_1 = 2.25$ and $C_2 = 6.85$ (seconds) will be used for active learning.
- 10 repetitions with $n_{\text{test}} = 100$ random test input locations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

Blade: Emulation performance



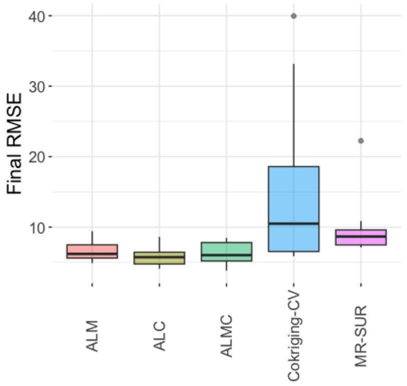
RMSE, CRPS, and computation time across 10 repetitions in the turbine blade application.

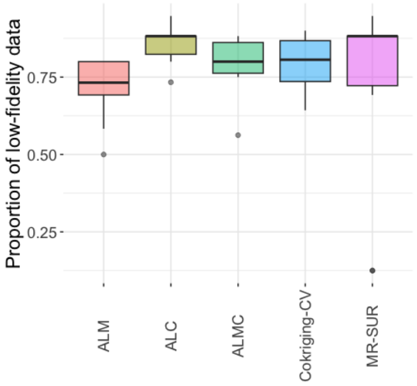
Blade: Active learning performance



RMSE and CRPS for the Park function with respect to the cost.

Blade: Active learning performance



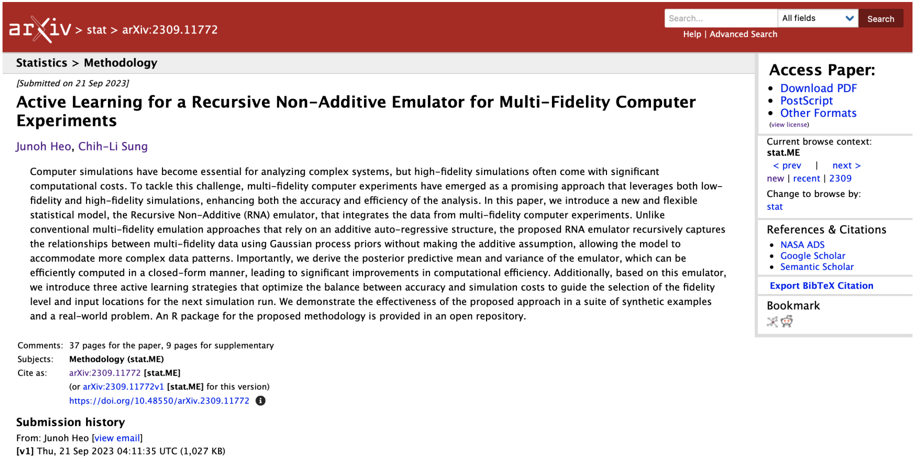


Final RMSE (left) and proportion of AL acquisitions choosing low-fidelity data (right).

Conclusion

- We propose a new model (RNA emulator) and three corresponding active learning strategies (ALM, ALC, and ALMC).
- RNA emulator provides the closed-form expressions for both the posterior mean and variance under the common kernel choices.
- Active learnings are facilitated by these closed form expressions.
- Numerical studies and real application show the effectiveness of our approach.
- R package RNAmf is available on CRAN.

Arxiv



Statistics > Methodology

[Submitted on 21 Sep 2023]

Active Learning for a Recursive Non-Additive Emulator for Multi-Fidelity Computer Experiments

Junoh Heo, Chih-Li Sung

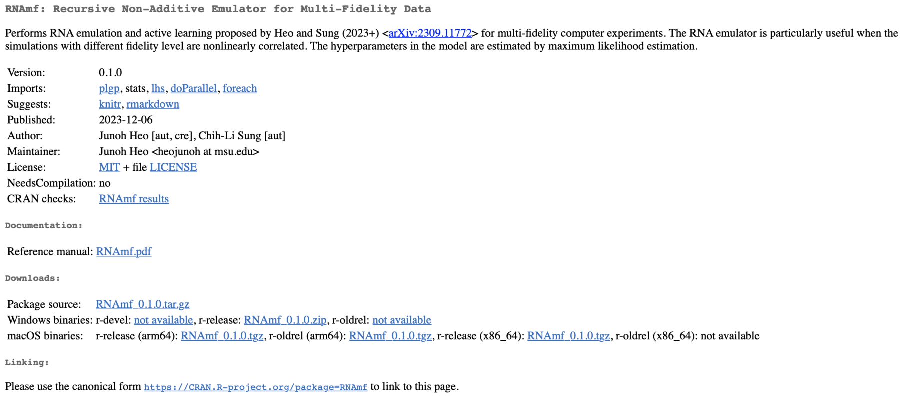
Computer simulations have become essential for analyzing complex systems, but high-fidelity simulations often come with significant computational costs. To tackle this challenge, multi-fidelity computer experiments have emerged as a promising approach that leverages both low-fidelity and high-fidelity simulations, enhancing both the accuracy and efficiency of the analysis. In this paper, we introduce a new and flexible statistical model, the Recursive Non-Additive (RNA) emulator, that integrates the data from multi-fidelity computer experiments. Unlike conventional multi-fidelity emulation approaches that rely on an additive auto-regressive structure, the proposed RNA emulator recursively captures the relationships between multi-fidelity data using Gaussian process priors without making the additive assumption, allowing the model to accommodate more complex data patterns. Importantly, we derive the posterior predictive mean and variance of the emulator, which can be efficiently computed in a closed-form manner, leading to significant improvements in computational efficiency. Additionally, based on this emulator, we introduce three active learning strategies that optimize the balance between accuracy and simulation costs to guide the selection of the fidelity level and input locations for the next simulation run. We demonstrate the effectiveness of the proposed approach in a suite of synthetic examples and a real-world problem. An R package for the proposed methodology is provided in an open repository.

Comments:	37 pages for the paper, 9 pages for supplementary
Subjects:	**Methodology (stat.ME)**
Cite as:	arXiv:2309.11772 **[stat.ME]**
	(or arXiv:2309.11772v1 **[stat.ME]** for this version)
	https://doi.org/10.48550/arXiv.2309.11772

Submission history

From: Junoh Heo [view email]
[v1] Thu, 21 Sep 2023 04:11:35 UTC (1,027 KB)

Code (CRAN)



RNAmf: Recursive Non-Additive Emulator for Multi-Fidelity Data

Performs RNA emulation and active learning proposed by Heo and Sung (2023+) <arXiv:2309.11772> for multi-fidelity computer experiments. The RNA emulator is particularly useful when the simulations with different fidelity level are nonlinearly correlated. The hyperparameters in the model are estimated by maximum likelihood estimation.

Version:	0.1.0
Imports:	plgp, stats, lhs, doParallel, foreach
Suggests:	knitr, rmarkdown
Published:	2023-12-06
Author:	Junoh Heo [aut, cre], Chih-Li Sung [aut]
Maintainer:	Junoh Heo <heojunoh at msu.edu>
License:	MIT + file LICENSE
NeedsCompilation:	no
CRAN checks:	RNAmf results

Documentation:

Reference manual:	RNAmf.pdf

Downloads:

Package source:	RNAmf_0.1.0.tar.gz
Windows binaries:	r-devel: not available, r-release: RNAmf_0.1.0.zip, r-oldrel: not available
macOS binaries:	r-release (arm64): RNAmf_0.1.0.tgz, r-oldrel (arm64): RNAmf_0.1.0.tgz, r-release (x86_64): RNAmf_0.1.0.tgz, r-oldrel (x86_64): not available

Linking:

Please use the canonical form `https://CRAN.R-project.org/package=RNAmf` to link to this page.

Thank You!

MICHIGAN STATE
UNIVERSITY