Batch Multi-Fidelity Active Learning with Budget Constraints

Shibo Li*, Jeff M. Phillips*, Xin Yu, Robert M. Kirby *and* Shandian Zhe

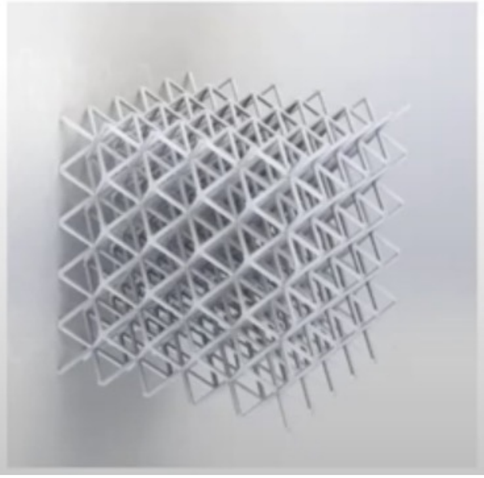
School of Computing, University of Utah

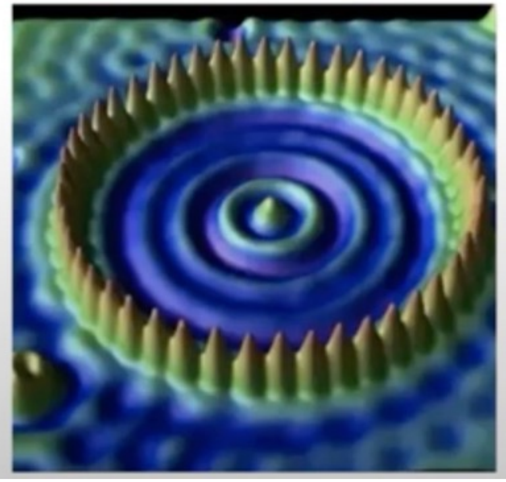
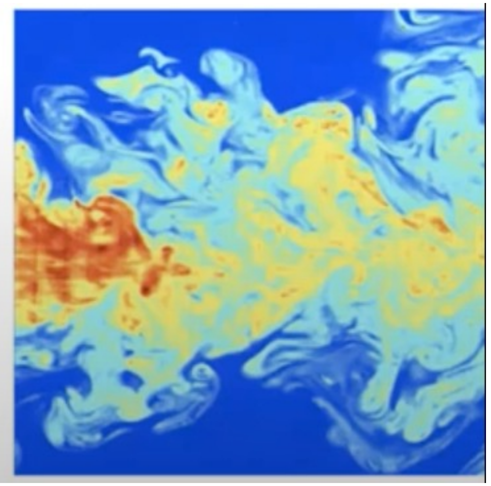
Presenter: Shibo Li

shibo@cs.utah.edu

Partial Differential Equations

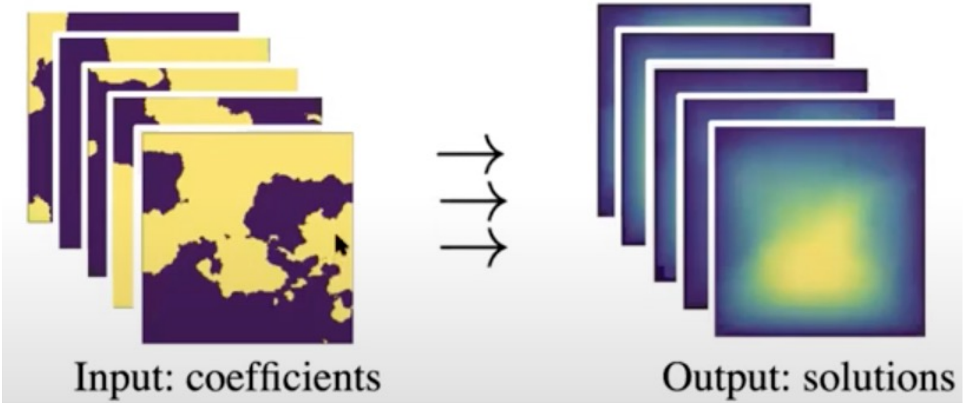
- PDE as the foundation for scientific computing
- Challenges:
  - Identify the governing model for complex systems
  - Efficiently solving large-scale non-linear systems of equations



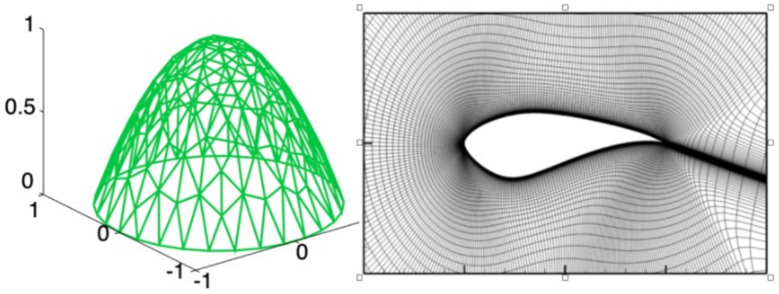



PDE Solutions: *Solve* vs. *Learn*

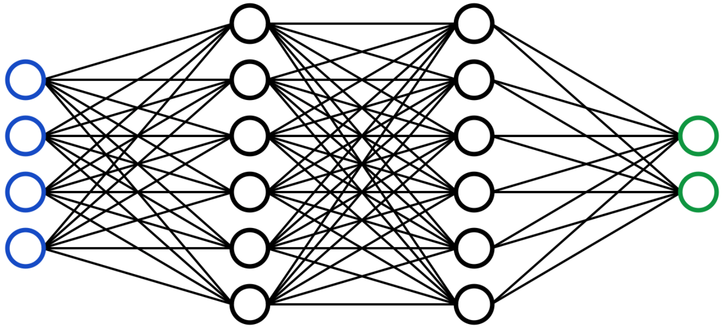
- Solving complicated PDEs:
  - Numerical solvers
    - Solve one instance
  - Data-driven solvers
    - Learn a family of solutions
    - Slow to train; fast to evaluate



Zongyi Li et al. 2020

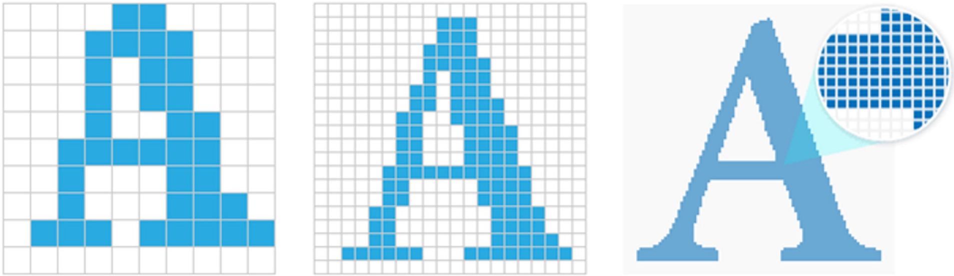


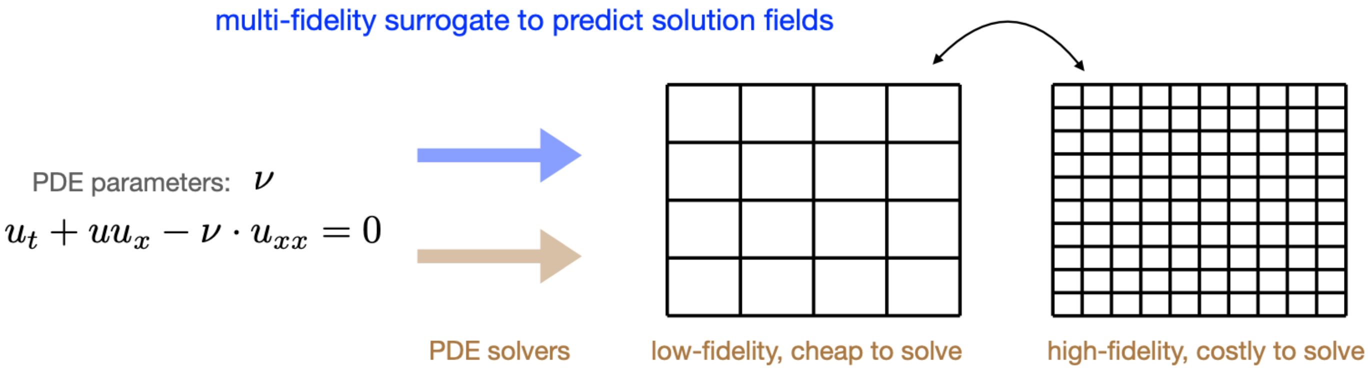
vs.



Multi-fidelity Active Learning

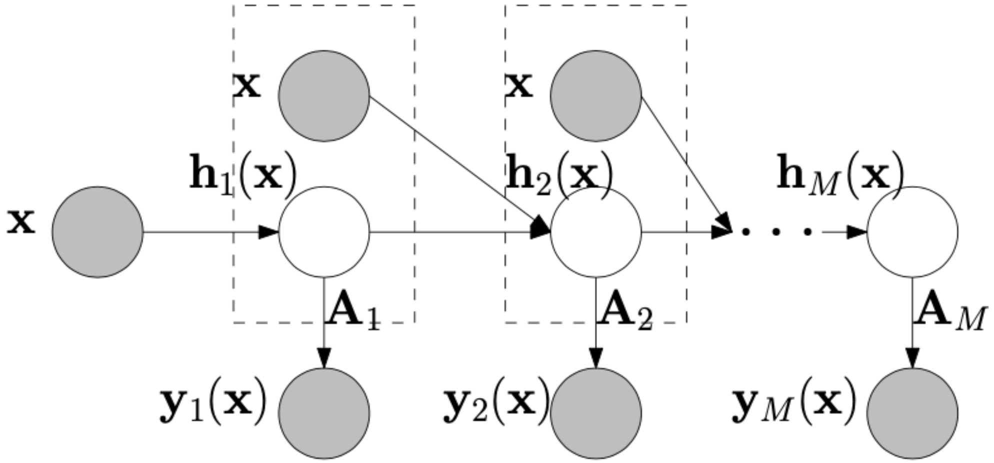
- Multi-fidelity Evaluations
  - *Low-fidelity solutions:* cheap to acquire but inaccurate
  - *High-fidelity solutions:* accurate but expensive to acquire
- Active Learning: Training examples are expensive





Deep Multi-Fidelity Active Learning

- DMFAL (Li et al. 2022)



Concatenate inputs

Low-rank outputs

$$\boldsymbol{\xi}_m = [\mathbf{x}; \mathbf{h}_{m-1}(\mathbf{x})], \quad \mathbf{h}_m(\mathbf{x}) = \mathbf{W}_m \boldsymbol{\rho}_{\boldsymbol{\theta}_m}(\boldsymbol{\xi}_m)$$

$$\mathbf{y}_m(\mathbf{x}) = \mathbf{A}_m \mathbf{h}_m(\mathbf{x}) + \boldsymbol{\epsilon}_m,$$

Outputs in solution space

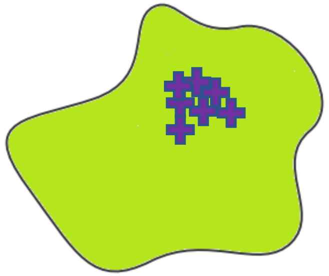
MF Modeling

Jointly determines the query point and fidelity level

$$a(\mathbf{x}, m) = \frac{1}{\lambda_m} \mathbb{I}\left(\mathbf{y}_m(\mathbf{x}), \mathbf{y}_M(\mathbf{x}) | \mathcal{D}\right)$$
$$= \frac{1}{\lambda_m}\left(\mathbb{H}(\mathbf{y}_m|\mathcal{D}) + \mathbb{H}(\mathbf{y}_M|\mathcal{D}) - \mathbb{H}(\mathbf{y}_m, \mathbf{y}_M|\mathcal{D})\right)$$

MF Acquisition

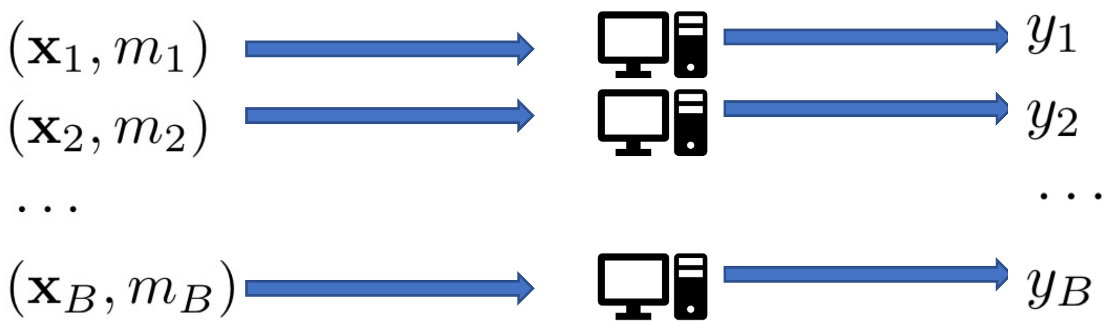
Ignoring the correlations between the successive queries

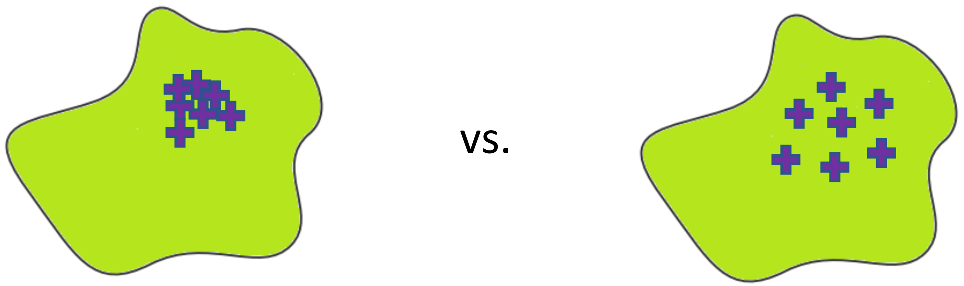


Our Contribution

- A ***batch*** multi-fidelity active learning approach for high-dimensional outputs
- Consider ***budget*** constraints in query samples
- A proved efficient ***greedy*** algorithms that <u>*nearly*</u>***1-1/e optimality***

Why batch matters?





Sequential strategy

Batch strategy

Novel Batch Acquisition

- Acquisition over the distribution over the input space:

$$a_s(m, \mathbf{x}) = \underline{\mathbb{E}_{p(\mathbf{x}')}}\left[\mathbb{I}\left(\mathbf{y}_m(\mathbf{x}), \mathbf{y}_{\underline{M}}(\mathbf{x}')|\mathcal{D}\right)\right]$$

- Consider the batch budget

$$a_{\text{batch}}(\mathcal{M}, \mathcal{X}) = \mathbb{E}_{p(\mathbf{x}')}\left[\mathbb{I}\left(\{\mathbf{y}_{m_j}(\mathbf{x}_j)\}_{j=1}^n, \mathbf{y}_M(\mathbf{x}')|\mathcal{D}\right)\right], \quad \text{s.t.} \sum_{j=1}^n \lambda_{m_j} \le B$$

$$\mathcal{M} = \{m_1, \ldots, m_n\}, \mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$$

- Monte-Carlo approximation of target acquisition

$$\boxed{\widehat{a}_{\text{batch}}(\mathcal{M}, \mathcal{X}) = \frac{1}{A}\sum_{l=1}^{A} \mathbb{I}\left(\{\mathbf{y}_{m_j}(\mathbf{x}_j)\}_{j=1}^n, \mathbf{y}_M(\mathbf{x}'_l)|\mathcal{D}\right)}$$

Incurs combinatorial search over multiple fidelities

Weighted Greedy Optimization

- A weighed incremental version of acquisition

$$\hat{a}_{k+1}(\mathbf{x}, m) = \frac{1}{A}\sum_{l=1}^{A} \frac{\mathbb{I}\left(\mathcal{Y}_k \cup \{\mathbf{y}_m(\mathbf{x})\}, \mathbf{y}_M(\mathbf{x}'_l)|\mathcal{D}\right) - \mathbb{I}\left(\mathcal{Y}_k, \mathbf{y}_M(\mathbf{x}'_l)|\mathcal{D}\right)}{\lambda_m}$$

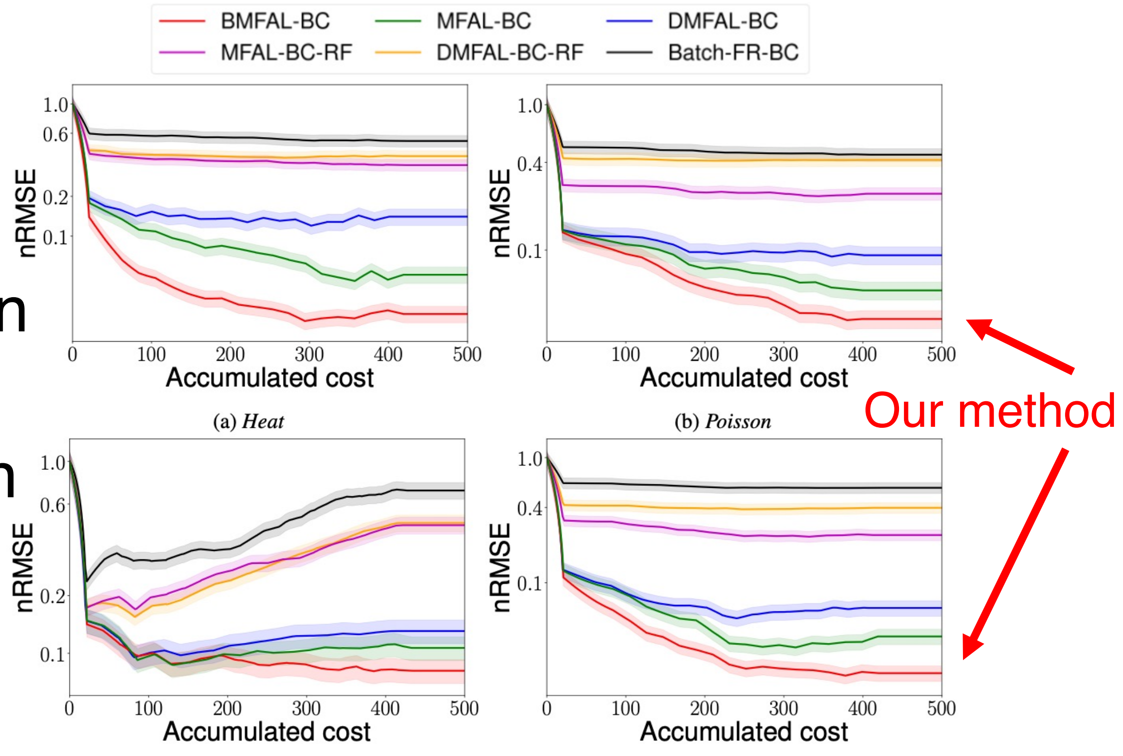
$$\text{s.t.} \quad \lambda_m + \sum_{\widehat{m}\in\mathcal{M}_k} \lambda_{\widehat{m}} \leq B,$$

$\boxed{(1-1/e)\text{-approximation of the optimal}}$

Proved

Experiment: Solving PDEs

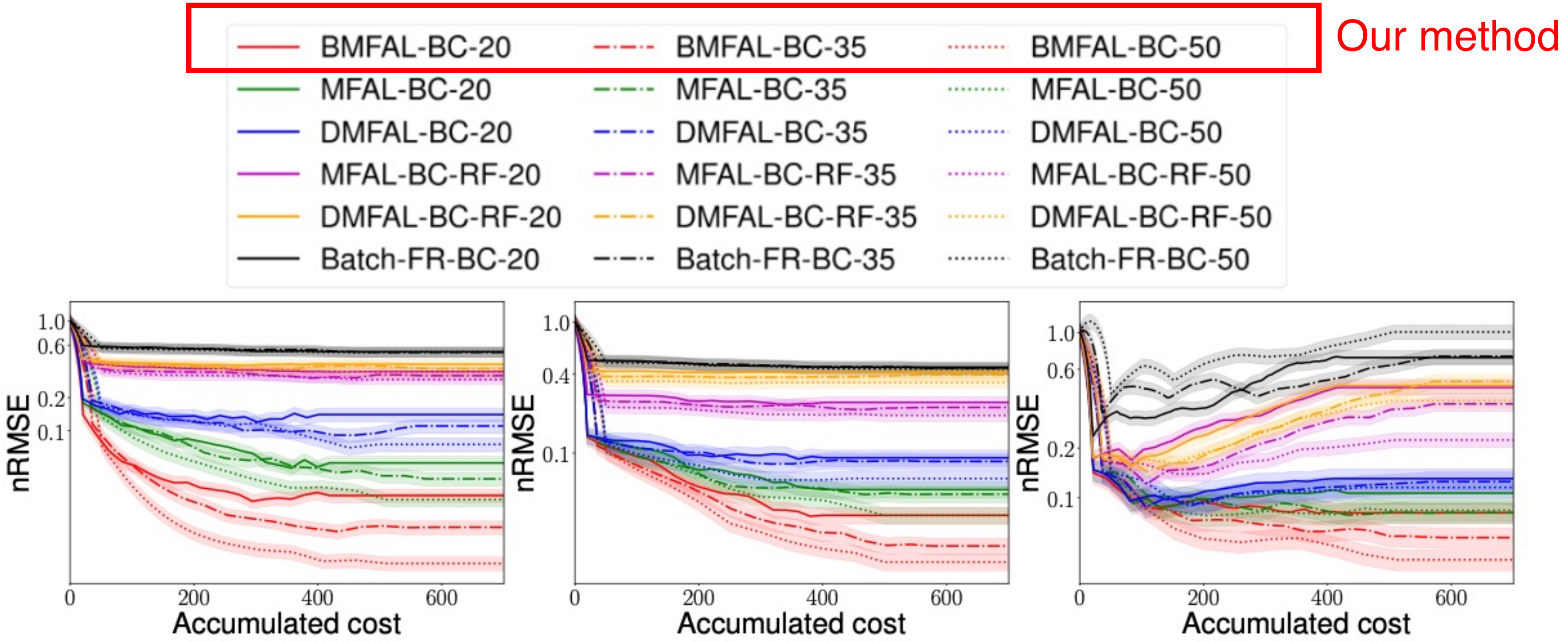
- BMFAL-BC: our weighted greedy strategy
- DMFAL-BC: DMFAL with original acquisition
- MFAL-BC: DMFAL with modified acquisition
- DMFAL-BC-RF: DMFAL with original acquisition but random fidelity selection
- MFAL-BC-RF: DMFAL with modified acquisition but random fidelity selection
- Batch-RF-BC: Pure random selections of fidelities and query inputs



(a) *Heat* (b) *Poisson* (c) *Burgers* (d) *Poisson-3*

For all compared methods, the surrogate is updated in a batch fashion until the budgets are exhausted.

Experiment: Varying Budgets



(a) *Heat* (b) *Poisson* (c) *Burgers*

Figure 4: nRMSE *vs.* the accumulated cost under different budgets per batch: $B \in \{20, 35, 50\}$.

Welcome to our poster!

Shibo Li (shibo@cs.utah.edu)