Coarse-grained models for PDEs with random coefficients

C. Grigo and P.-S. Koutsourelakis

Continuum Mechanics Group
Department of Mechanical Engineering
Technical University of Munich
SIAM CSE Atlanta

1 Mar 2017

SPDE with random coefficients

Stochastic PDE:

$$\mathcal{K}\left(\boldsymbol{x},\lambda(\boldsymbol{x},\xi)\right)u(\boldsymbol{x},\lambda(\boldsymbol{x},\xi)) = f(\boldsymbol{x}), \qquad +\text{B.C.}$$

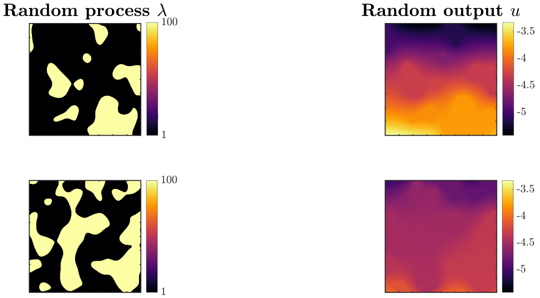


Figure: Random process $\lambda(\boldsymbol{x},\xi)$ leads to random solutions $u(\boldsymbol{x},\xi)$.

Outline

The Full-Order Model (FOM)

- Discretize

$$\mathcal{K}\left(\boldsymbol{x}, \lambda(\boldsymbol{x}, \xi)\right) u(\boldsymbol{x}, \lambda(\boldsymbol{x}, \xi)) = f(\boldsymbol{x}), \qquad +\text{B.C.}$$

  to a set of algebraic equations

$$\boldsymbol{r}_f(\boldsymbol{U}_f, \boldsymbol{\lambda}_f(\xi)) = \mathbf{0}$$

- Usually large ($N_{\text{equations}} \sim$ millions)
- Expensive, repeated evaluations for UQ (and various deterministic tasks, e.g. optimization/control, inverse problems. . .)

Surrogate models

Idea: Replace FOM $\boldsymbol{U}_f = \boldsymbol{U}_f(\boldsymbol{\lambda}_f)$ by cheaper, but less accurate input-output map $\boldsymbol{U}_f = \boldsymbol{f}(\boldsymbol{\lambda}_f; \boldsymbol{\theta})$ based on training data $\mathcal{D} = \left\{\boldsymbol{U}_f^{(i)}, \boldsymbol{\lambda}_f^{(i)}\right\}_{i=1}^{N}$

Surrogate models

Idea: Replace FOM $\boldsymbol{U}_f = \boldsymbol{U}_f(\boldsymbol{\lambda}_f)$ by cheaper, but less accurate input-output map $\boldsymbol{U}_f = \boldsymbol{f}(\boldsymbol{\lambda}_f; \boldsymbol{\theta})$ based on training data $\mathcal{D} = \left\{\boldsymbol{U}_f^{(i)}, \boldsymbol{\lambda}_f^{(i)}\right\}_{i=1}^{N}$

Problem: High dimensional uncertainties $\boldsymbol{\lambda}_f$ - learning direct functional mapping (e.g. PCE [Ghanem, Spanos 1991] , GP [Rasmussen 2006], neural nets [Bishop 1995]) will fail

Surrogate models

Idea: Replace FOM $\boldsymbol{U}_f = \boldsymbol{U}_f(\boldsymbol{\lambda}_f)$ by cheaper, but less accurate input-output map $\boldsymbol{U}_f = \boldsymbol{f}(\boldsymbol{\lambda}_f; \boldsymbol{\theta})$ based on training data $\mathcal{D} = \left\{\boldsymbol{U}_f^{(i)}, \boldsymbol{\lambda}_f^{(i)}\right\}_{i=1}^{N}$

Problem: High dimensional uncertainties $\boldsymbol{\lambda}_f$ - learning direct functional mapping (e.g. PCE [Ghanem, Spanos 1991] , GP [Rasmussen 2006], neural nets [Bishop 1995]) will fail

Solution: Coarse-grained model: Use models based on coarser discretization of PDE, $\boldsymbol{U}_c = \boldsymbol{U}_c(\boldsymbol{\lambda}_c)$

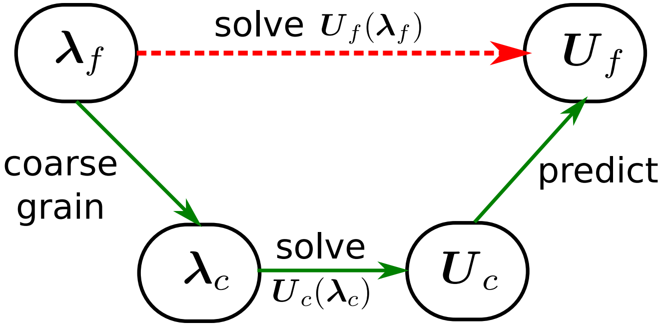
Surrogate models

Idea: Replace FOM $\boldsymbol{U}_f = \boldsymbol{U}_f(\boldsymbol{\lambda}_f)$ by cheaper, but less accurate input-output map $\boldsymbol{U}_f = \boldsymbol{f}(\boldsymbol{\lambda}_f; \boldsymbol{\theta})$ based on training data $\mathcal{D} = \left\{\boldsymbol{U}_f^{(i)}, \boldsymbol{\lambda}_f^{(i)}\right\}_{i=1}^{N}$

Problem: High dimensional uncertainties $\boldsymbol{\lambda}_f$ - learning direct functional mapping (e.g. PCE [Ghanem, Spanos 1991] , GP [Rasmussen 2006], neural nets [Bishop 1995]) will fail

Solution: Coarse-grained model: Use models based on coarser discretization of PDE, $\boldsymbol{U}_c = \boldsymbol{U}_c(\boldsymbol{\lambda}_c)$

Question: Relation between $\boldsymbol{U}_f$ and coarse output $\boldsymbol{U}_c$, but also relation between fine/coarse inputs $\boldsymbol{\lambda}_f, \boldsymbol{\lambda}_c$



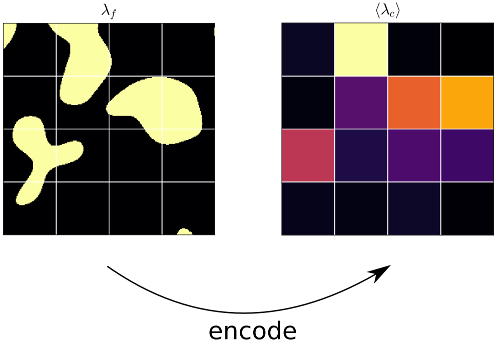
- Retain as much as possible information on $\boldsymbol{U}_f$ during coarse-graining, i.e.

Information bottleneck [Tishby, Pereira, Bialek, 1999]

$$\max_{\boldsymbol{\theta}} \; I(\boldsymbol{\lambda}_c, \boldsymbol{U}_f; \boldsymbol{\theta}) \quad \text{s.t.} \quad I(\boldsymbol{\lambda}_f, \boldsymbol{\lambda}_c; \boldsymbol{\theta}) \leq I_0$$

Concept: Coarse grain random field λ,...

- Probabilistic mapping $\boldsymbol{\lambda}_f \to \boldsymbol{\lambda}_c$: $p_c(\boldsymbol{\lambda}_c|\boldsymbol{\lambda}_f, \boldsymbol{\theta}_c)$



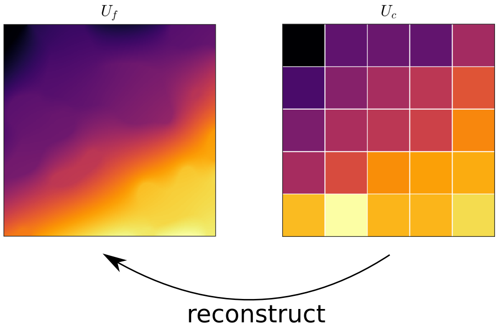
- **Goal:** Prediction of $\boldsymbol{U}_f$, not reconstruction of $\boldsymbol{\lambda}_f$!

$\ldots$ solve ROM and reconstruct $\boldsymbol{U}_f$ from $\boldsymbol{U}_c$

- $\boldsymbol{\lambda}_c \to \boldsymbol{U}_c$: solve

$$\boldsymbol{r}_c(\boldsymbol{U}_c, \boldsymbol{\lambda}_c) = \mathbf{0}$$

- Decode via coarse-to-fine map $\boldsymbol{U}_c \to \boldsymbol{U}_f$: $p_{cf}(\boldsymbol{U}_f | \boldsymbol{U}_c, \boldsymbol{\theta}_{cf})$



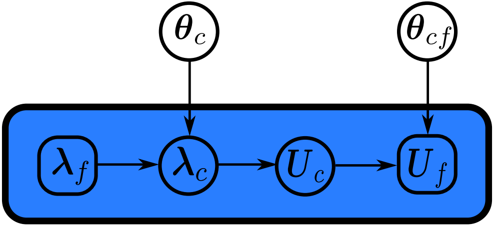


Figure: Bayesian network defining $\bar{p}(\boldsymbol{U}_f|\boldsymbol{\lambda}_f, \boldsymbol{\theta}_c, \boldsymbol{\theta}_{cf})$.

$$\begin{aligned}\bar{p}(\boldsymbol{U}_f|\boldsymbol{\lambda}_f, \boldsymbol{\theta}_c, \boldsymbol{\theta}_{cf}) &= \int p_{cf}(\boldsymbol{U}_f|\boldsymbol{U}_c, \boldsymbol{\theta}_{cf}) p(\boldsymbol{U}_c|\boldsymbol{\lambda}_c) p_c(\boldsymbol{\lambda}_c|\boldsymbol{\lambda}_f, \boldsymbol{\theta}_c) d\boldsymbol{U}_c d\boldsymbol{\lambda}_c \\ &= \int p_{cf}(\boldsymbol{U}_f|\boldsymbol{U}_c(\boldsymbol{\lambda}_c), \boldsymbol{\theta}_{cf}) p_c(\boldsymbol{\lambda}_c|\boldsymbol{\lambda}_f, \boldsymbol{\theta}_c) d\boldsymbol{\lambda}_c.\end{aligned}$$

Model training

- Maximum likelihood:

$$\begin{pmatrix} \boldsymbol{\theta}_c^* \\ \boldsymbol{\theta}_{cf}^* \end{pmatrix} = \arg\max_{\boldsymbol{\theta}_c, \boldsymbol{\theta}_{cf}} \sum_{i=1}^{N} \log \bar{p}(\boldsymbol{U}_f^{(i)} | \boldsymbol{\lambda}_f^{(i)}, \boldsymbol{\theta}_c, \boldsymbol{\theta}_{cf})$$

- Maximum posterior:

$$\begin{pmatrix} \boldsymbol{\theta}_c^* \\ \boldsymbol{\theta}_{cf}^* \end{pmatrix} = \arg\max_{\boldsymbol{\theta}_c, \boldsymbol{\theta}_{cf}} \sum_{i=1}^{N} \log \bar{p}(\boldsymbol{U}_f^{(i)} | \boldsymbol{\lambda}_f^{(i)}, \boldsymbol{\theta}_c, \boldsymbol{\theta}_{cf}) + \log p(\boldsymbol{\theta}_c, \boldsymbol{\theta}_{cf})$$

- Data:

$$\boldsymbol{\lambda}_f^{(i)} \sim p(\boldsymbol{\lambda}_f^{(i)}), \qquad \boldsymbol{U}_f^{(i)} = \boldsymbol{U}_f(\boldsymbol{\lambda}_f^{(i)}).$$

Expectation-Maximization algorithm

$$\bar{p}(\boldsymbol{U}_f^{(i)}|\boldsymbol{\lambda}_f^{(i)}, \boldsymbol{\theta}_c, \boldsymbol{\theta}_{cf}) = \int p_{cf}(\boldsymbol{U}_f^{(i)}|\boldsymbol{U}_c(\boldsymbol{\lambda}_c^{(i)}), \boldsymbol{\theta}_{cf}) p_c(\boldsymbol{\lambda}_c^{(i)}|\boldsymbol{\lambda}_f^{(i)}, \boldsymbol{\theta}_c) d\boldsymbol{\lambda}_c^{(i)}$$

→ Likelihood contains N integrals over N latent variables $\boldsymbol{\lambda}_c^{(i)}$

→ Use Expectation-Maximization algorithm [Dempster, Laird, Rubin 1977] : find lower bound

$$\begin{aligned}
&\log(\bar{p}(\boldsymbol{U}_f^{(i)}|\boldsymbol{\lambda}_f^{(i)}, \boldsymbol{\theta}_c, \boldsymbol{\theta}_{cf})) \\
&\geq \int q^{(i)}(\boldsymbol{\lambda}_c^{(i)}) \log\left(\frac{p_{cf}(\boldsymbol{U}_f^{(i)}|\boldsymbol{U}_c(\boldsymbol{\lambda}_c^{(i)}), \boldsymbol{\theta}_{cf}) p_c(\boldsymbol{\lambda}_c^{(i)}|\boldsymbol{\lambda}_f^{(i)}, \boldsymbol{\theta}_c)}{q^{(i)}(\boldsymbol{\lambda}_c^{(i)})}\right) d\boldsymbol{\lambda}_c^{(i)} \\
&= \mathcal{F}^{(i)}(\boldsymbol{\theta}; q_t^{(i)}(\boldsymbol{\lambda}_c^{(i)})), \qquad \text{where} \quad \boldsymbol{\theta} = [\boldsymbol{\theta}_c, \boldsymbol{\theta}_{cf}].
\end{aligned}$$

Expectation-Maximization algorithm

- Maximize iteratively:

E-step: Find optimal $q_t^{(i)}(\boldsymbol{\lambda}_c^{(i)}) \propto p_{cf}(\boldsymbol{U}_f^{(i)}|\boldsymbol{U}_c(\boldsymbol{\lambda}_c^{(i)}), \boldsymbol{\theta}_{cf}) p_c(\boldsymbol{\lambda}_c^{(i)}|\boldsymbol{\lambda}_f^{(i)}, \boldsymbol{\theta}_c)$ given current estimate $\boldsymbol{\theta}_t$ of optimal $\boldsymbol{\theta}$ and compute expectation values (MCMC, VI, EP)

M-step: Maximize lower bound $\mathcal{F}_t(\boldsymbol{\theta}) = \sum_i \mathcal{F}_t^{(i)}(\boldsymbol{\theta}; q_t^{(i)}(\boldsymbol{\lambda}_c^{(i)}))$ w.r.t. $\boldsymbol{\theta}$ to find $\boldsymbol{\theta}_{t+1}$

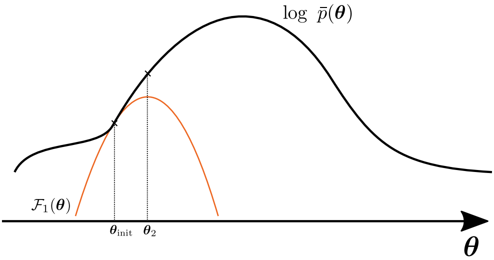


Figure: Expectation-Maximization algorithm illustration

Expectation-Maximization algorithm

- Maximize iteratively:

E-step: Find optimal $q_t^{(i)}(\boldsymbol{\lambda}_c^{(i)}) \propto p_{cf}(\boldsymbol{U}_f^{(i)}|\boldsymbol{U}_c(\boldsymbol{\lambda}_c^{(i)}), \boldsymbol{\theta}_{cf}) p_c(\boldsymbol{\lambda}_c^{(i)}|\boldsymbol{\lambda}_f^{(i)}, \boldsymbol{\theta}_c)$ given current estimate $\boldsymbol{\theta}_t$ of optimal $\boldsymbol{\theta}$ and compute expectation values (MCMC, VI, EP)

M-step: Maximize lower bound $\mathcal{F}_t(\boldsymbol{\theta}) = \sum_i \mathcal{F}_t^{(i)}(\boldsymbol{\theta}; q_t^{(i)}(\boldsymbol{\lambda}_c^{(i)}))$ w.r.t. $\boldsymbol{\theta}$ to find $\boldsymbol{\theta}_{t+1}$

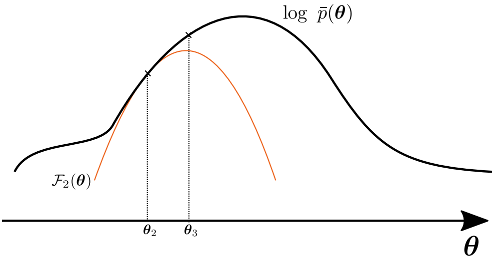


Figure: Expectation-Maximization algorithm illustration

Expectation-Maximization algorithm

- Maximize iteratively:

E-step: Find optimal $q_t^{(i)}(\boldsymbol{\lambda}_c^{(i)}) \propto p_{cf}(\boldsymbol{U}_f^{(i)}|\boldsymbol{U}_c(\boldsymbol{\lambda}_c^{(i)}), \boldsymbol{\theta}_{cf}) p_c(\boldsymbol{\lambda}_c^{(i)}|\boldsymbol{\lambda}_f^{(i)}, \boldsymbol{\theta}_c)$ given current estimate $\boldsymbol{\theta}_t$ of optimal $\boldsymbol{\theta}$ and compute expectation values (MCMC, VI, EP)

M-step: Maximize lower bound $\mathcal{F}_t(\boldsymbol{\theta}) = \sum_i \mathcal{F}_t^{(i)}(\boldsymbol{\theta}; q_t^{(i)}(\boldsymbol{\lambda}_c^{(i)}))$ w.r.t. $\boldsymbol{\theta}$ to find $\boldsymbol{\theta}_{t+1}$

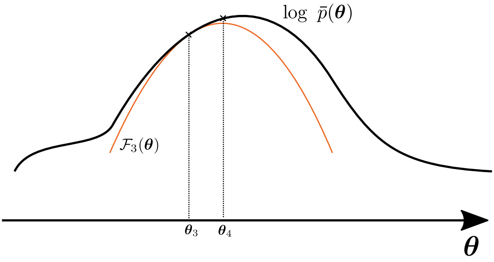


Figure: Expectation-Maximization algorithm illustration

Expectation-Maximization algorithm

- Maximize iteratively:

E-step: Find optimal $q_t^{(i)}(\boldsymbol{\lambda}_c^{(i)}) \propto p_{cf}(\boldsymbol{U}_f^{(i)}|\boldsymbol{U}_c(\boldsymbol{\lambda}_c^{(i)}), \boldsymbol{\theta}_{cf})p_c(\boldsymbol{\lambda}_c^{(i)}|\boldsymbol{\lambda}_f^{(i)}, \boldsymbol{\theta}_c)$ given current estimate $\boldsymbol{\theta}_t$ of optimal $\boldsymbol{\theta}$ and compute expectation values (MCMC, VI, EP)

M-step: Maximize lower bound $\mathcal{F}_t(\boldsymbol{\theta}) = \sum_i \mathcal{F}_t^{(i)}(\boldsymbol{\theta}; q_t^{(i)}(\boldsymbol{\lambda}_c^{(i)}))$ w.r.t. $\boldsymbol{\theta}$ to find $\boldsymbol{\theta}_{t+1}$

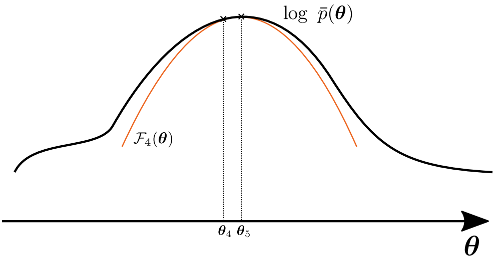


Figure: Expectation-Maximization algorithm illustration

Expectation-Maximization algorithm

- Maximize iteratively:

E-step: Find optimal $q_t^{(i)}(\boldsymbol{\lambda}_c^{(i)}) \propto p_{cf}(\boldsymbol{U}_f^{(i)}|\boldsymbol{U}_c(\boldsymbol{\lambda}_c^{(i)}), \boldsymbol{\theta}_{cf})p_c(\boldsymbol{\lambda}_c^{(i)}|\boldsymbol{\lambda}_f^{(i)}, \boldsymbol{\theta}_c)$ given current estimate $\boldsymbol{\theta}_t$ of optimal $\boldsymbol{\theta}$ and compute expectation values (MCMC, VI, EP)

M-step: Maximize lower bound $\mathcal{F}_t(\boldsymbol{\theta}) = \sum_i \mathcal{F}_t^{(i)}(\boldsymbol{\theta}; q_t^{(i)}(\boldsymbol{\lambda}_c^{(i)}))$ w.r.t. $\boldsymbol{\theta}$ to find $\boldsymbol{\theta}_{t+1}$

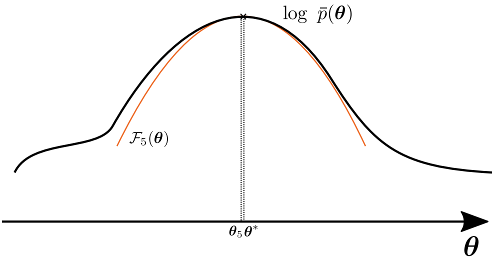


Figure: Expectation-Maximization algorithm illustration

Model predictions

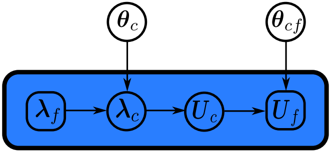


Figure: Bayesian network defining $\bar{p}(\boldsymbol{U}_f|\boldsymbol{\lambda}_f, \boldsymbol{\theta}_c, \boldsymbol{\theta}_{cf})$.

$$\bar{p}(\boldsymbol{U}_f|\boldsymbol{\lambda}_f, \boldsymbol{\theta}_c^*, \boldsymbol{\theta}_{cf}^*) = \int p_{cf}(\boldsymbol{U}_f|\boldsymbol{U}_c(\boldsymbol{\lambda}_c), \boldsymbol{\theta}_{cf}^*) p_c(\boldsymbol{\lambda}_c|\boldsymbol{\lambda}_f, \boldsymbol{\theta}_c^*) d\boldsymbol{\lambda}_c$$

Given $\boldsymbol{\lambda}_f$ and $\boldsymbol{\theta}_c^*, \boldsymbol{\theta}_{cf}^*$,

- sample $\boldsymbol{\lambda}_c$ from $p_c(\boldsymbol{\lambda}_c|\boldsymbol{\lambda}_f, \boldsymbol{\theta}_c^*)$,
- solve coarse model $\boldsymbol{U}_c = \boldsymbol{U}_c(\boldsymbol{\lambda}_c)$,
- sample $\boldsymbol{U}_f$ from $p(\boldsymbol{U}_f|\boldsymbol{U}_c, \boldsymbol{\theta}_{cf}^*)$

Sample problem: 2D heat equation

$$\nabla_{\boldsymbol{x}}(-\lambda(\boldsymbol{x},\xi(\boldsymbol{x}))\nabla_{\boldsymbol{x}}T(\boldsymbol{x},\xi(\boldsymbol{x}))) = 0, \qquad +\text{B.C.}$$

where $\xi(\boldsymbol{x}) \sim GP(0, \text{cov}(\boldsymbol{x}_i, \boldsymbol{x}_j))$ with

$$\text{cov}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left\{-\frac{|\boldsymbol{x}_i - \boldsymbol{x}_j|^2}{l^2}\right\},$$

and

$$\lambda(\boldsymbol{x},\xi(\boldsymbol{x})) = \begin{cases} \lambda_{\text{hi}}, & \text{if} \qquad \xi(\boldsymbol{x}) > k, \\ \lambda_{\text{lo}}, & \text{otherwise} \end{cases}$$

FOM data samples

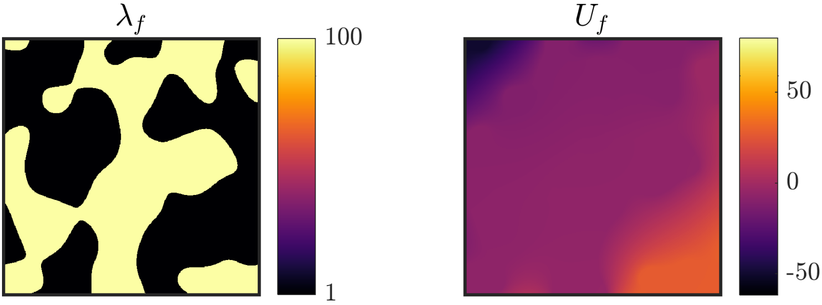


Figure: Data samples for $\phi_{\text{hi}} = 0.35$, $l = 0.098$, $c = 100$

FOM data samples

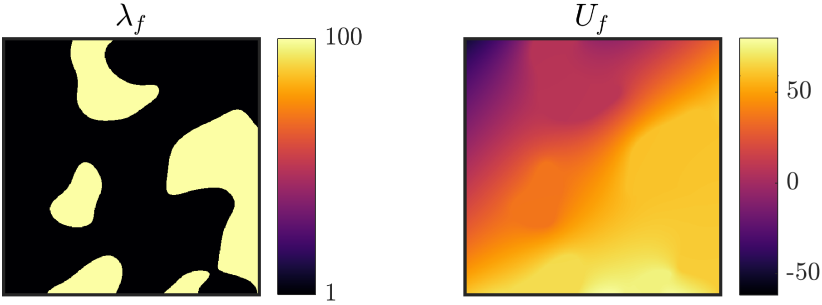


Figure: Data samples for $\phi_{\mathrm{hi}} = 0.35$, $l = 0.098$, $c = 100$

FOM data samples

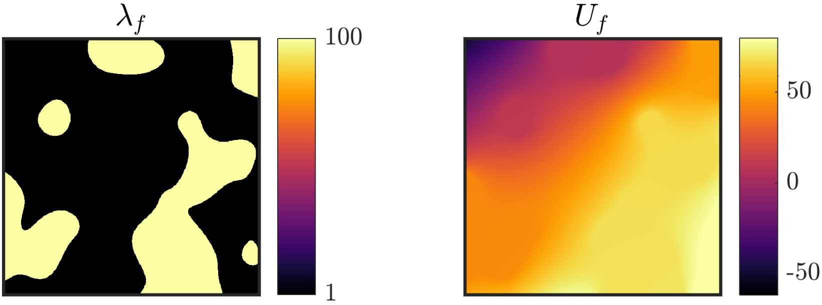


Figure: Data samples for $\phi_{\text{hi}} = 0.35$, $l = 0.098$, $c = 100$

FOM data samples

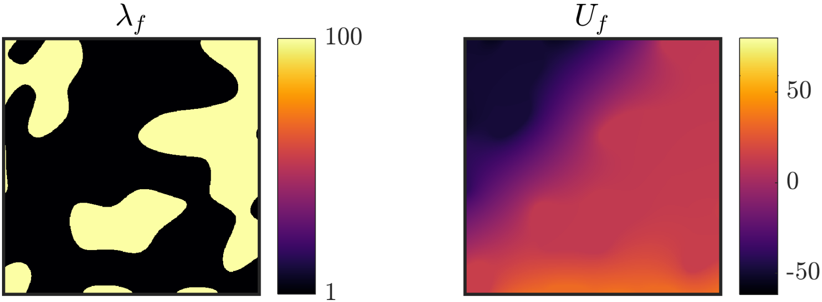
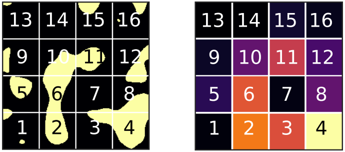


Figure: Data samples for $\phi_{\text{hi}} = 0.35$, $l = 0.098$, $c = 100$

Model specifications

- $\boldsymbol{\lambda}_f \to \boldsymbol{\lambda}_c$:



Element numbering with index k

$$\log \lambda_{c,k} = \sum_{j=1}^{N_{\text{features}}} \theta_{c,j} \varphi_j(\boldsymbol{\lambda}_{f,k}) + \sigma_k Z_k, \;\; Z_k \sim \mathcal{N}(0,1),$$

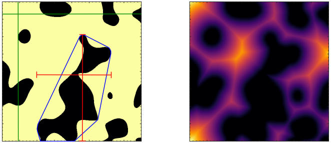
- $\boldsymbol{U}_c \to \boldsymbol{U}_f$:

$$p_{cf}(\boldsymbol{U}_f | \boldsymbol{U}_c(\boldsymbol{z}_c), \boldsymbol{\theta}_{cf}) = \mathcal{N}(\boldsymbol{U}_f | \boldsymbol{W} \boldsymbol{U}_c(\boldsymbol{z}_c), \boldsymbol{S})$$

with feature functions φ_i, coarse-to-fine projection $\boldsymbol{W}$, diagonal covariance $\boldsymbol{S} = \text{diag}(\boldsymbol{s})$.

Feature functions $\varphi_i(\boldsymbol{\lambda}_{f,k})$

- Any function $\varphi_i : (\mathbb{R}^+)^{\dim(\boldsymbol{\lambda}_{f,k})} \mapsto \mathbb{R}$ admissible
- Could/should be guided by physical insight:
  - **Effective-medium approximations**
    - Self-consistent approximation (SCA)[Bruggeman 1935],
    - Maxwell-Garnett approximation (MGA)[Maxwell 1873],
    - Differential effective medium approximation (DEM) [Bruggeman 1935]...
  - **Morphology-describing features:**
    - Lineal path function[Lu, Torquato 1992],
    - (Convex) Blob area,
    - Blob extent,
    - Distance transformations...



Left: Convex area (blue), max. extent (red), pixel-cross (green).
Right: distance transform

Sparsity priors

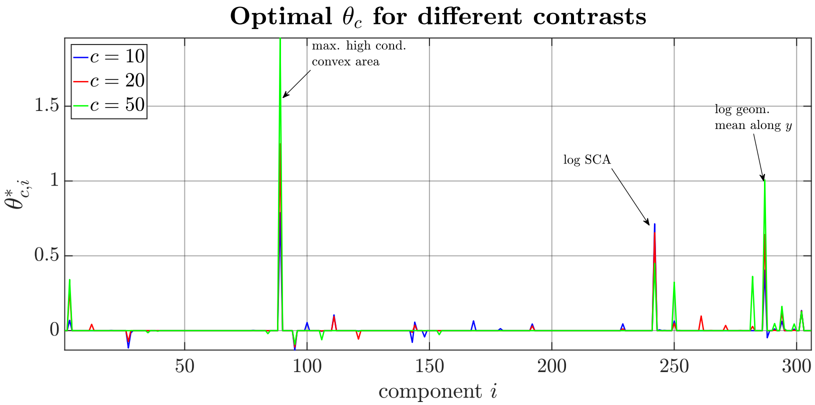
- **Strategy:** Include as many features φ_j as possible, employ sparsity prior for feature selection
- **Laplace prior (LASSO):**

$$p(\theta_{c,i}) \propto \exp\left\{-\sqrt{\gamma}\left|\theta_{c,i}\right|\right\},$$

- **ARD prior:**

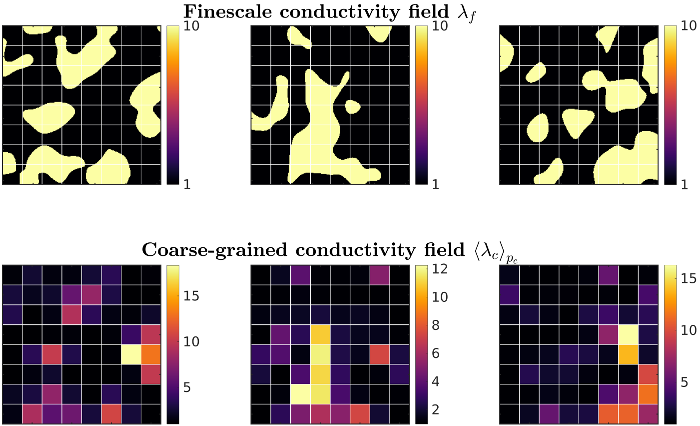
$$p(\theta_{c,i}) \propto \int_0^\infty \frac{1}{\tau_i}\mathcal{N}(\theta_{c,i}|0,\tau_i)d\tau_i = \frac{1}{|\theta_{c,i}|}$$

Which features are activated?



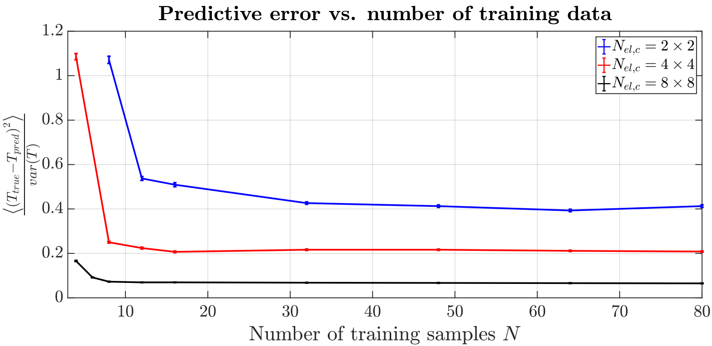
- The higher the contrast, the more geometry matters

Learned effective property $\boldsymbol{\lambda}_c$



- Note that $p_c(\boldsymbol{\lambda}_c|\boldsymbol{\lambda}_f, \boldsymbol{\theta}_c) = \mathcal{N}(\log \boldsymbol{\lambda}_c|\boldsymbol{\Phi}\boldsymbol{\theta}_c, \boldsymbol{\Sigma} = \mathrm{diag}(\boldsymbol{\sigma}^2))$, and we plot $\langle \boldsymbol{\lambda}_c \rangle_{p_c} = \boldsymbol{\Phi}\boldsymbol{\theta}_c + \frac{1}{2}\boldsymbol{\sigma}^2$

How many training samples do we need?



- Few data is needed, errors converge quickly
- The finer the coarse mesh, the better the predictions
- The finer the coarse mesh, the less data is needed
- **But:** the finer the coarse mesh, the more expensive the training/predictions

Predictions

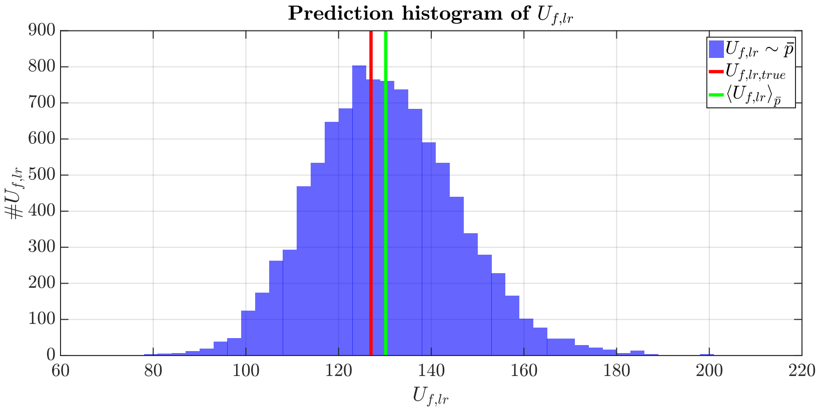


Figure: Histogrammatic predictive distribution of temperature at lower right corner, $\bar{p}(U_{f,lr}|\boldsymbol{\lambda}_f, \boldsymbol{\theta})$

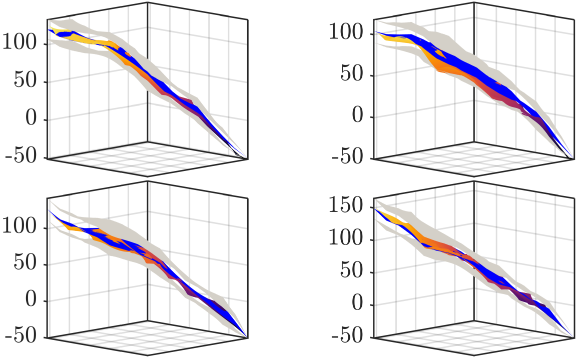


Figure: Predictions on different test data samples for $N_{\mathrm{el},c} = 8 \times 8$, $\phi_{\mathrm{hi}} = 0.2$, $l = 0.078$ and $c = \frac{\lambda_{\mathrm{hi}}}{\lambda_{\mathrm{lo}}} = 10$. Colored: $\boldsymbol{U}_f$, blue: $\langle \boldsymbol{U}_f \rangle_{\bar{p}}$, grey: $\pm\sigma$.

Predictions

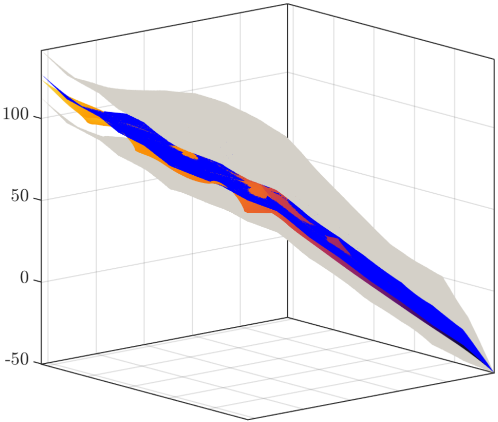


Figure: Test sample 3 from different angles

Predictions

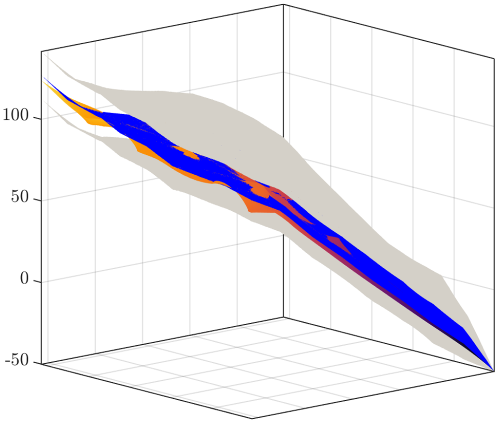


Figure: Test sample 3 from different angles

Predictions

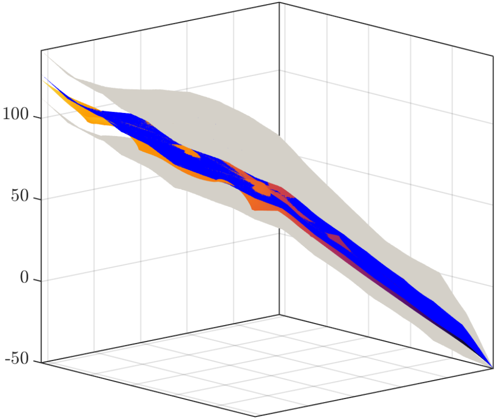


Figure: Test sample 3 from different angles

Predictions

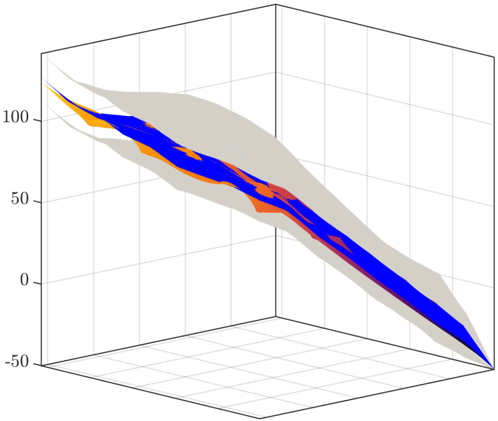


Figure: Test sample 3 from different angles

Predictions

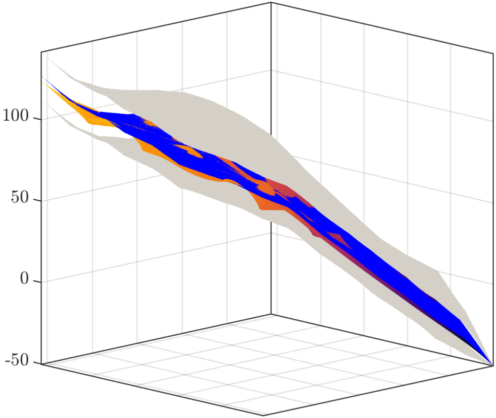


Figure: Test sample 3 from different angles

Predictions

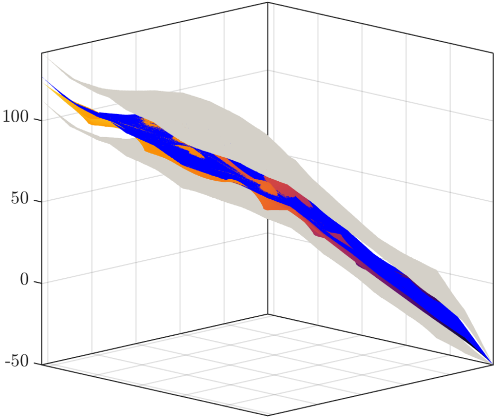


Figure: Test sample 3 from different angles

Predictions

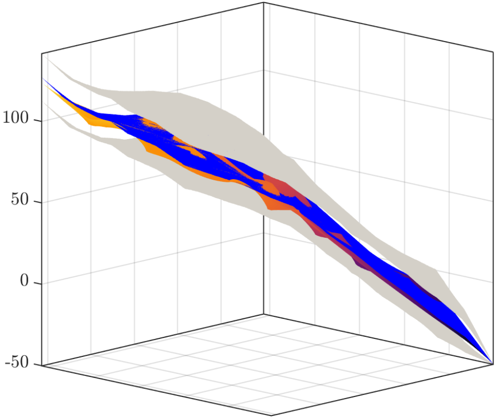


Figure: Test sample 3 from different angles

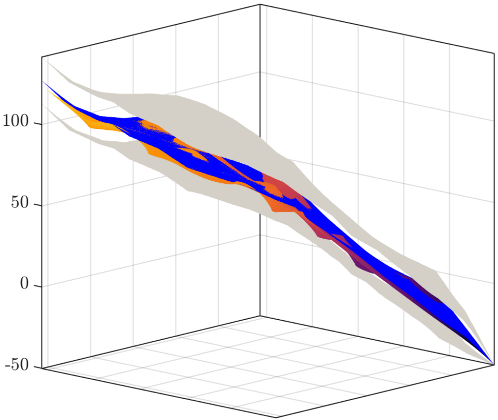


Figure: Test sample 3 from different angles

Predictions

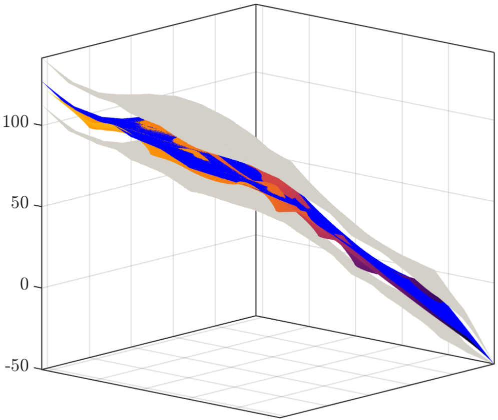


Figure: Test sample 3 from different angles

Predictions

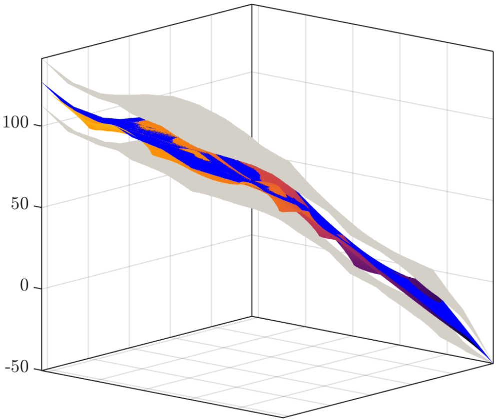


Figure: Test sample 3 from different angles

Predictions

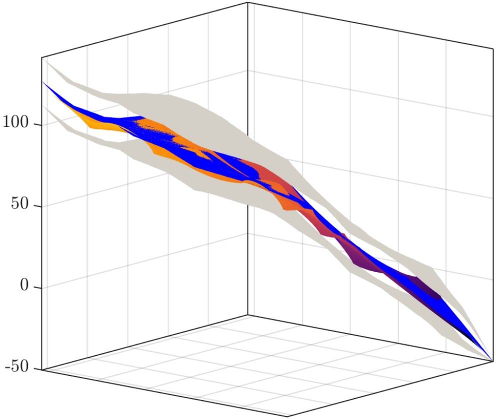


Figure: Test sample 3 from different angles

Predictions

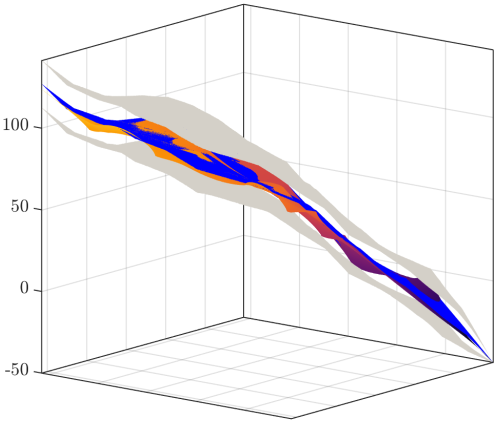


Figure: Test sample 3 from different angles

Predictions

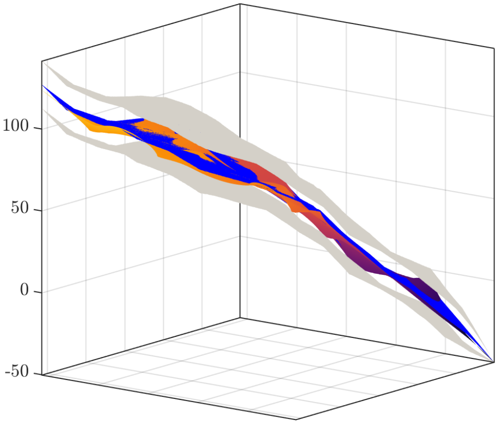


Figure: Test sample 3 from different angles

Predictions

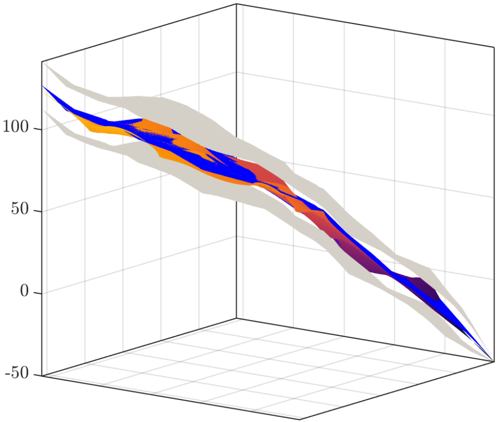


Figure: Test sample 3 from different angles

Predictions

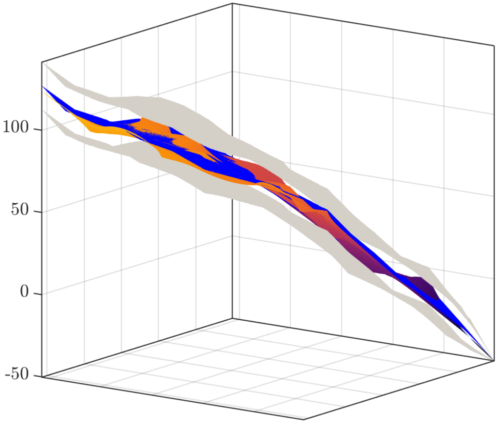


Figure: Test sample 3 from different angles

Predictions

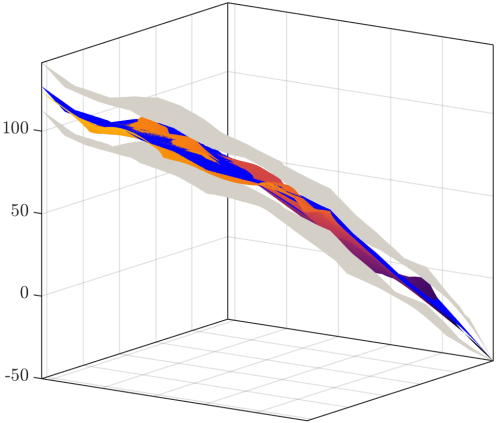


Figure: Test sample 3 from different angles

Predictions

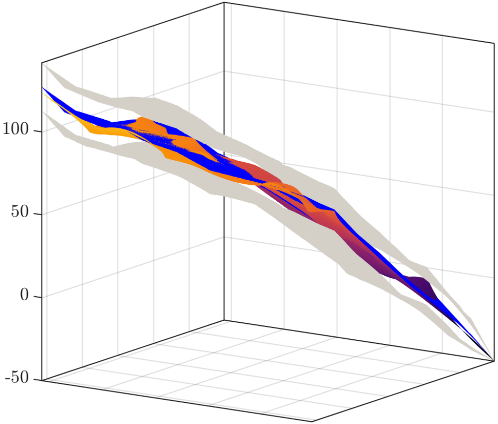


Figure: Test sample 3 from different angles

Predictions

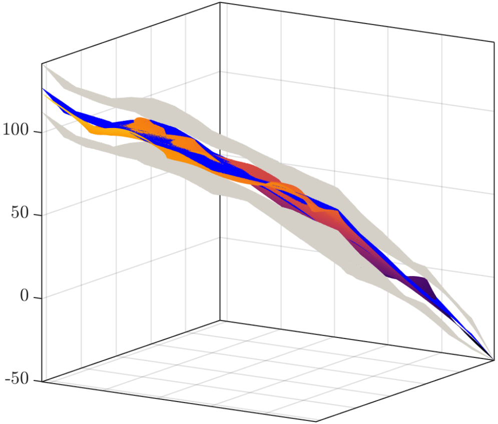


Figure: Test sample 3 from different angles

Predictions

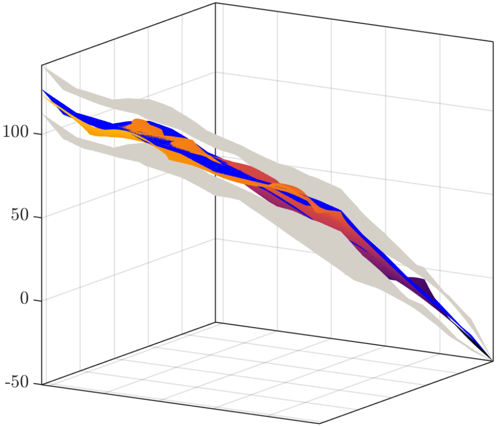


Figure: Test sample 3 from different angles

Predictions

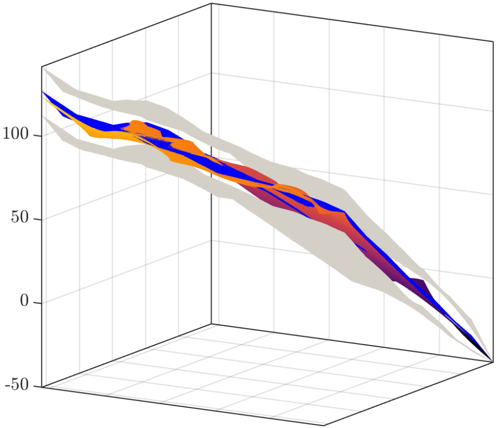


Figure: Test sample 3 from different angles

Predictions

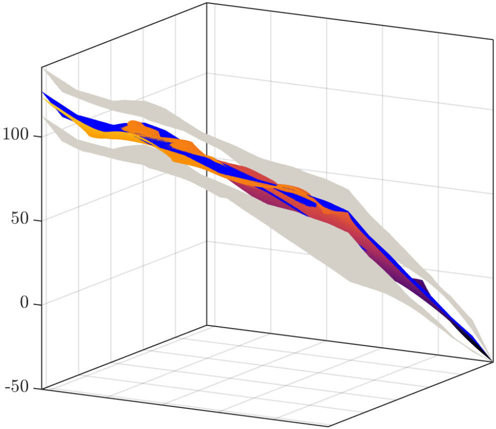


Figure: Test sample 3 from different angles

Predictions

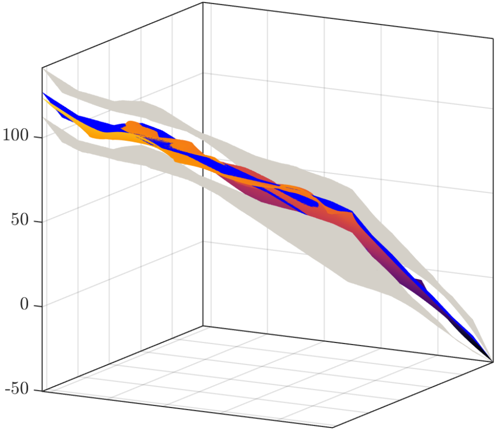


Figure: Test sample 3 from different angles

Predictions

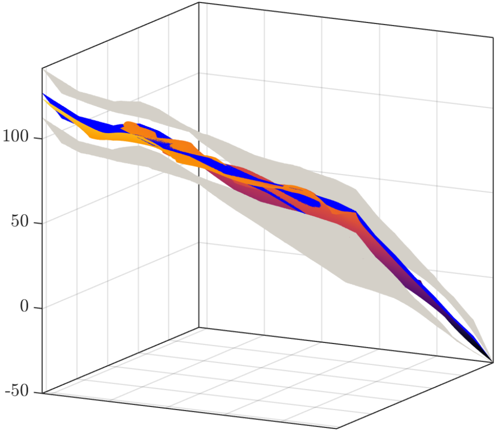


Figure: Test sample 3 from different angles

Predictions

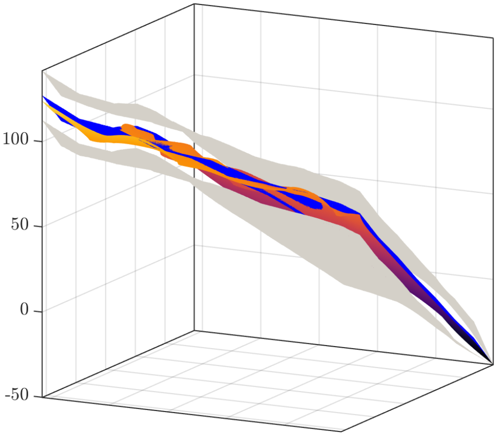


Figure: Test sample 3 from different angles

Predictions

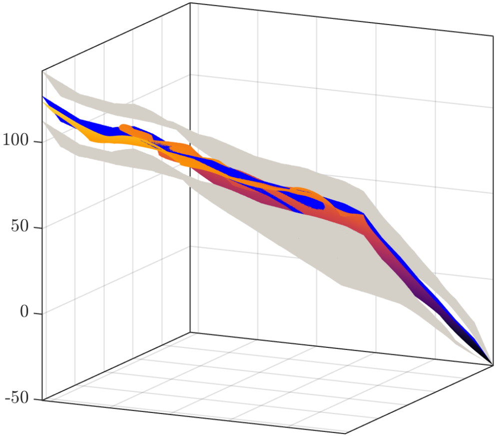


Figure: Test sample 3 from different angles

Predictions

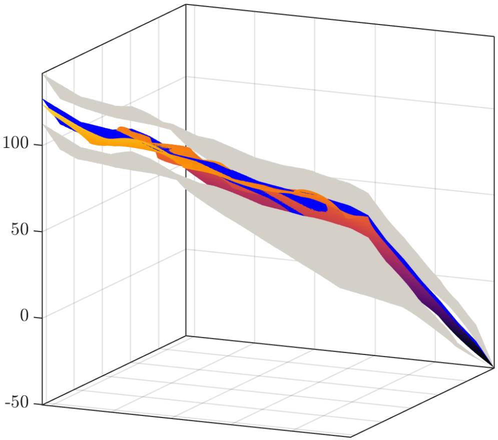


Figure: Test sample 3 from different angles

Predictions

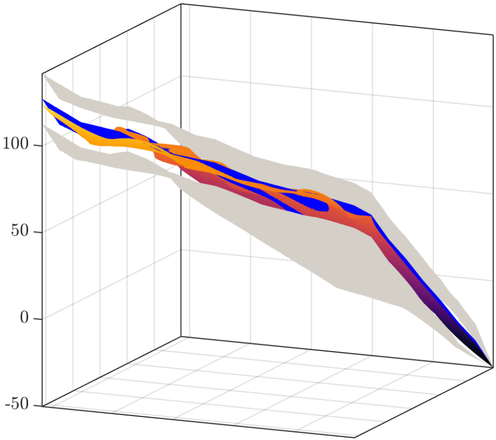


Figure: Test sample 3 from different angles

Predictions

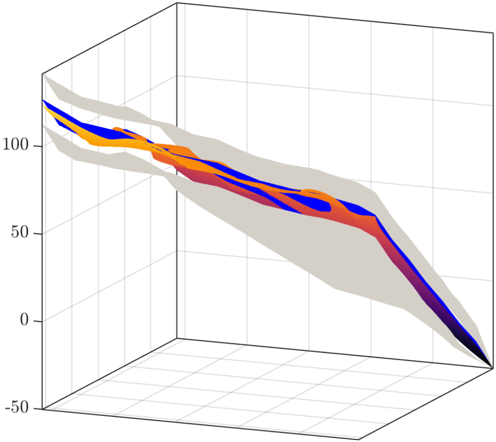


Figure: Test sample 3 from different angles

Predictions

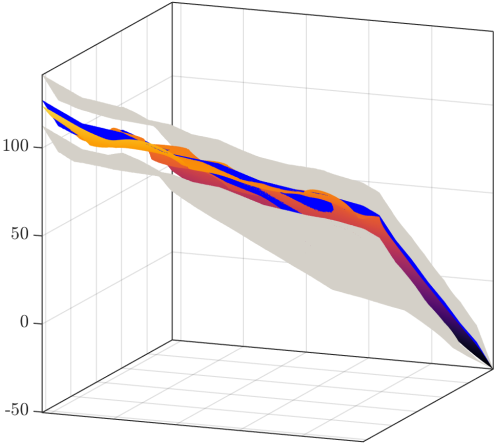


Figure: Test sample 3 from different angles

Predictions

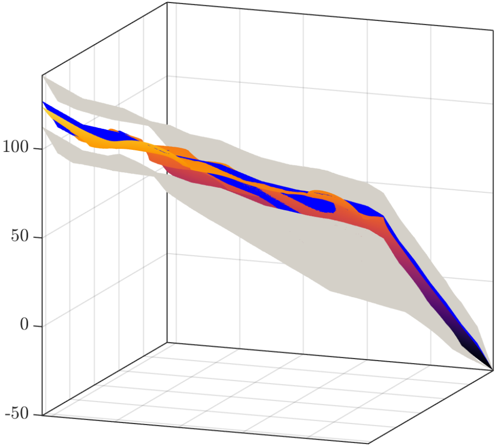


Figure: Test sample 3 from different angles

Predictions

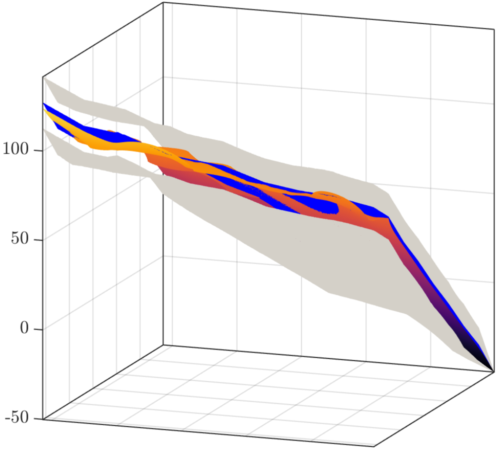


Figure: Test sample 3 from different angles

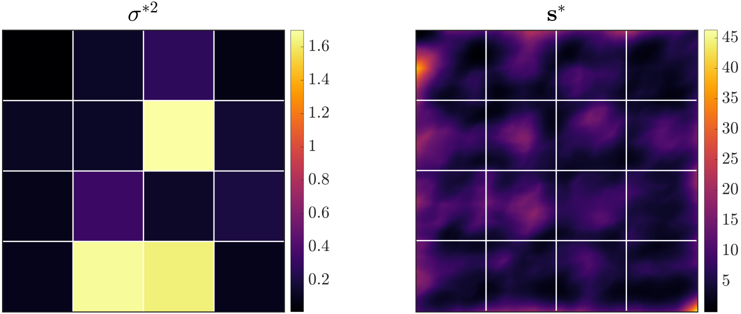


Figure: Optimal variances $\boldsymbol{\sigma}^{*2}$ of p_c (l.) and optimal variances $\boldsymbol{s}$ of p_{cf}.

Scaling of the algorithm

Training:

Quantity N	Scaling
#Data	$\mathcal{O}(N)$
$\dim(\boldsymbol{\lambda}_f)$	?
$\dim(\boldsymbol{U}_f)$	$\mathcal{O}(N)$
$\dim(\boldsymbol{\lambda}_c), \dim(\boldsymbol{U}c)$	$\mathcal{O}(N^3)$
$\dim(\boldsymbol{\theta}_c)$	$\mathcal{O}(N^3)$

Predictions:

Quantity N	Scaling
#Data	$\mathcal{O}(1)$
$\dim(\boldsymbol{\lambda}_f)$	?
$\dim(\boldsymbol{U}_f)$	$\mathcal{O}(N)$
$\dim(\boldsymbol{\lambda}_c), \dim(\boldsymbol{U}c)$	$\mathcal{O}(N^3)$
$\dim(\boldsymbol{\theta}_c)$	$\mathcal{O}(N)$

Is the model applicable for all kinds of microstructures?

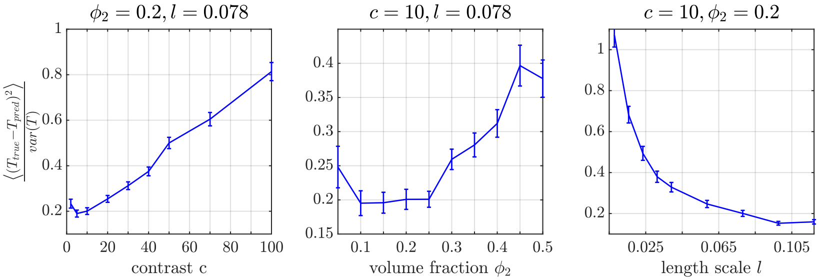


Figure: Predictive error for different microstructural parameters and a 4×4 coarse grid.

- There are regimes where the model works optimally/ will fail

Summary & Outlook

Summary

- Replace FOM by cheaper, but less accurate ROM
- Learn probabilistic output-output, but also input/input mappings between fine and coarse solver
- Predict by sampling $\boldsymbol{\lambda}_c$, solving coarse model, sampling $\boldsymbol{U}_f$
- Potentially find interpretable features for effective material properties

Outlook

- Anisotropic $\boldsymbol{\lambda}_c$
- Account for correlations among $\lambda_{c,k}$'s
- Adaptive coarse mesh refinement