Quantification of Model Form Uncertainty for run-time Constrained Simulation based Predictions

M. Drohmann, K. Sargsyan, J. Wilke, B. Debuscherre and H. Najm

Sandia National Laboratories, Livermore, CA, (mdrohma@sandia.gov)

SIAM UQ 2014, Savannah, GA

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000

Problem setting and goals

- Parametrization $\mu \in \mathcal{P} \subset \mathbb{R}^p$ of
- a (very!) expensive, trust-worthy model $M(\mu)$,
- with approximative models $m_\alpha(\mu) \approx M(\mu)$ of varying accuracy α

Find a model and a quantification of the model-form uncertainty that can be computed in reasonable time

- model reduction
- UQ Validation/verification

Motivation: Run-time of exascale simulations

- SST/macro is a simulator of computer programmes' run–times on machines with arbitrary network configurations
- Goal: Estimate the costs/benefits of exascale computing

Motivation: Run-time of exascale simulations

- SST/macro is a simulator of computer programmes' run–times on machines with arbitrary network configurations
- Goal: Estimate the costs/benefits of exascale computing

Details

- Network (problem) size is the parameter μ,
- full model $M(\mu)$ is trust–worthy, but slow,
- approximative models $m_\alpha(\mu)$ simulate network traffic with artificial packet granularity α

Motivation: Run-time of exascale simulations

- SST/macro is a simulator of computer programmes' run–times on machines with arbitrary network configurations
- Goal: Estimate the costs/benefits of exascale computing

Details

- Computation time grows exponentially with network size,
+ Simulator is exponentially faster for smaller/unphysical packet sizes (granularity), but
- no error control for unphysical packet granularity

Surrogate models

In many query-contexts, (like design optimization, sensitivity analysis, decision analysis, inverse problems, . . .)surrogate models

+ can speed up computations, but
- usually do not capture physics anymore.
- Error quantification difficult for non-linear problems or extrapolation.

ROMES and ApproxES model

Idea: ROMES, ApproxES

- Reduced order model (ROM) or approximative model $m_\alpha(\boldsymbol{\mu})$ for cheaper computations of full model.
- Build surrogate for error $\delta_\alpha(\boldsymbol{\mu}) := m_\alpha(\boldsymbol{\mu}) - M(\boldsymbol{\mu})$.

ROMES and ApproxES model

Idea: ROMES, ApproxES

- Reduced order model (ROM) or approximative model $m_\alpha(\boldsymbol{\mu})$ for cheaper computations of full model.
- Build surrogate for error $\delta_\alpha(\boldsymbol{\mu}) := m_\alpha(\boldsymbol{\mu}) - M(\boldsymbol{\mu})$.
- ROM/Approx-error surrogates $\widetilde{\delta}_\alpha(\boldsymbol{\mu})$
- for predictions with model-form UQ $\widetilde{M}(\boldsymbol{\mu}) := m_\alpha(\boldsymbol{\mu}) - \widetilde{\delta}_\alpha(\boldsymbol{\mu})$

ROMES and ApproxES model

Idea: ROMES, ApproxES

- Reduced order model (ROM) or approximative model $m_\alpha(\boldsymbol{\mu})$ for cheaper computations of full model.
- Build surrogate for error $\delta_\alpha(\boldsymbol{\mu}) := m_\alpha(\boldsymbol{\mu}) - M(\boldsymbol{\mu})$.
- ROM/Approx-error surrogates $\widetilde{\delta}_\alpha(\boldsymbol{\mu})$
- for predictions with model-form UQ $\widetilde{M}(\boldsymbol{\mu}) := m_\alpha(\boldsymbol{\mu}) - \widetilde{\delta}_\alpha(\boldsymbol{\mu})$
- Consider error indicators $\rho_\alpha(\boldsymbol{\mu}) \approx \delta_\alpha(\boldsymbol{\mu})$ as training features.

ROMES and ApproxES model

Idea: ROMES, ApproxES

- Reduced order model (ROM) or approximative model $m_\alpha(\boldsymbol{\mu})$ for cheaper computations of full model.
- Build surrogate for error $\delta_\alpha(\boldsymbol{\mu}) := m_\alpha(\boldsymbol{\mu}) - M(\boldsymbol{\mu})$.
- ROM/Approx-error surrogates $\widetilde{\delta}_\alpha(\boldsymbol{\mu})$
- for predictions with model-form UQ $\widetilde{M}(\boldsymbol{\mu}) := m_\alpha(\boldsymbol{\mu}) - \widetilde{\delta}_\alpha(\boldsymbol{\mu})$
- Consider error indicators $\rho_\alpha(\boldsymbol{\mu}) \approx \delta_\alpha(\boldsymbol{\mu})$ as training features.

\+ Physical behavior captured by approximate model

\+ quantification of uncertainty through error surrogates

We will use Gaussian processes for the ROM/ApproxES $\widetilde{\delta}_\alpha(\boldsymbol{\mu})$

Gaussian process regression [Rasmussen & Williams, 2006]

1. Generate GP $\mathcal{G}_{\mathcal{T}}$ to approximate error $\mathcal{G}_{\mathcal{T}}(\alpha, \boldsymbol{\mu}) \approx \delta_\alpha(\boldsymbol{\mu})$ from
2. Training data: $\mathcal{T} := \{(\rho_\alpha(\boldsymbol{\mu}), \delta_\alpha(\boldsymbol{\mu}))|(\alpha, \boldsymbol{\mu}) \in P_{\text{train}}\}$.

Gaussian process regression [Rasmussen & Williams, 2006]

1. Generate GP $\mathcal{G}_\mathcal{T}$ to approximate error $\mathcal{G}_\mathcal{T}(\alpha, \boldsymbol{\mu}) \approx \delta_\alpha(\boldsymbol{\mu})$ from
2. Training data: $\mathcal{T} := \{(\rho_\alpha(\boldsymbol{\mu}), \delta_\alpha(\boldsymbol{\mu}))|(\alpha, \boldsymbol{\mu}) \in P_{\text{train}}\}$.

Algorithm

- We assume that a finite instance $\mathcal{G}(\boldsymbol{x}) := (\mathcal{G}(x_1), \ldots, \mathcal{G}(x_n))$ has a
- Gaussian distribution $\mathcal{G}(\boldsymbol{x}) \sim \mathcal{N}(m(\boldsymbol{x}), K(\boldsymbol{x}, \boldsymbol{x}))$ with
- fixed covariance $K(\boldsymbol{x}, \boldsymbol{x}) = (k(x_i, x_j))_{1 \leq i,j \leq n}$.
- Squared exponential: $k(x_i, x_j) = \sigma^2 \left(\exp\left(\frac{x_i - x_j}{2r^2} \right) + \delta_{ij} \right)$
- Inference of most likely hyperparameters r and σ^2 gives marginal predictor $\mathcal{G}_\mathcal{T}(\boldsymbol{x}^*)$.

Gaussian process regression [Rasmussen & Williams, 2006]

1. Generate GP $\mathcal{G}_\mathcal{T}$ to approximate error $\mathcal{G}_\mathcal{T}(\alpha, \boldsymbol{\mu}) \approx \delta_\alpha(\boldsymbol{\mu})$ from
2. Training data: $\mathcal{T} := \{(\rho_\alpha(\boldsymbol{\mu}), \delta_\alpha(\boldsymbol{\mu}))|(\alpha, \boldsymbol{\mu}) \in P_{\text{train}}\}$.

Algorithm

- We assume that a finite instance $\mathcal{G}(\boldsymbol{x}) := (\mathcal{G}(x_1), \ldots, \mathcal{G}(x_n))$ has a
- Gaussian distribution $\mathcal{G}(\boldsymbol{x}) \sim \mathcal{N}(m(\boldsymbol{x}), K(\boldsymbol{x}, \boldsymbol{x}))$ with
- fixed covariance $K(\boldsymbol{x}, \boldsymbol{x}) = (k(x_i, x_j))_{1 \leq i,j \leq n}$.
- Squared exponential: $k(x_i, x_j) = \sigma^2 \left(\exp\left(\frac{x_i - x_j}{2r^2} \right) + \delta_{ij} \right)$
- Inference of most likely hyperparameters r and σ^2 gives marginal predictor $\mathcal{G}_\mathcal{T}(\boldsymbol{x}^*)$.

Inferred $\mathcal{N}(0, \sigma^2)$ quantifies model-form uncertainty.

Numerical experiments: SST/macro simulator

- SST/macro is simulator of computer programmes' run-times on arbitrary network configurations.
- Approximate model $m_\alpha(\boldsymbol{\mu})$ based on artificial packet granularity is available, but has no uncertainty quantification
- "True" packet granularity is α^*: $M(\boldsymbol{\mu}) = m_{\alpha^*}(\boldsymbol{\mu})$.

Numerical experiments: SST/macro simulator

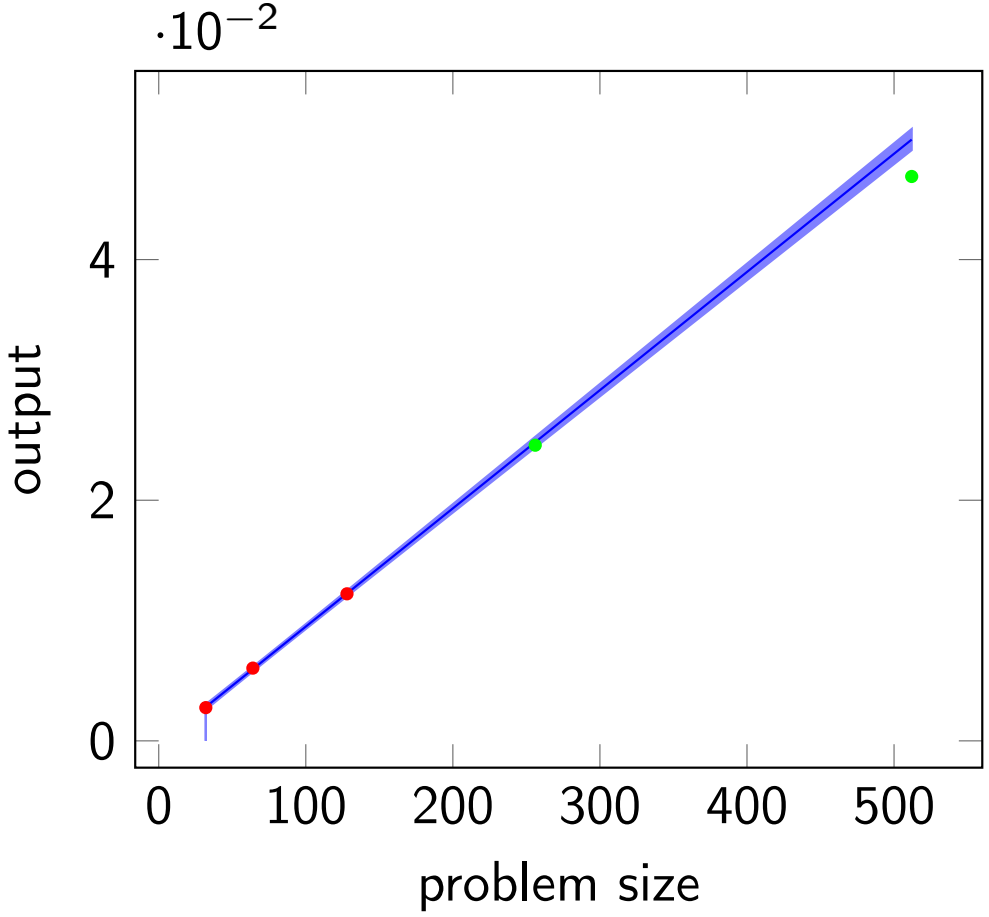
- SST/macro is simulator of computer programmes' run-times on arbitrary network configurations.
- Approximate model $m_\alpha(\boldsymbol{\mu})$ based on artificial packet granularity is available, but has no uncertainty quantification
- "True" packet granularity is α^*: $M(\boldsymbol{\mu}) = m_{\alpha^*}(\boldsymbol{\mu})$.

Experiment: allgather

Single allgather collective on network with $\boldsymbol{\mu}$ computing nodes.

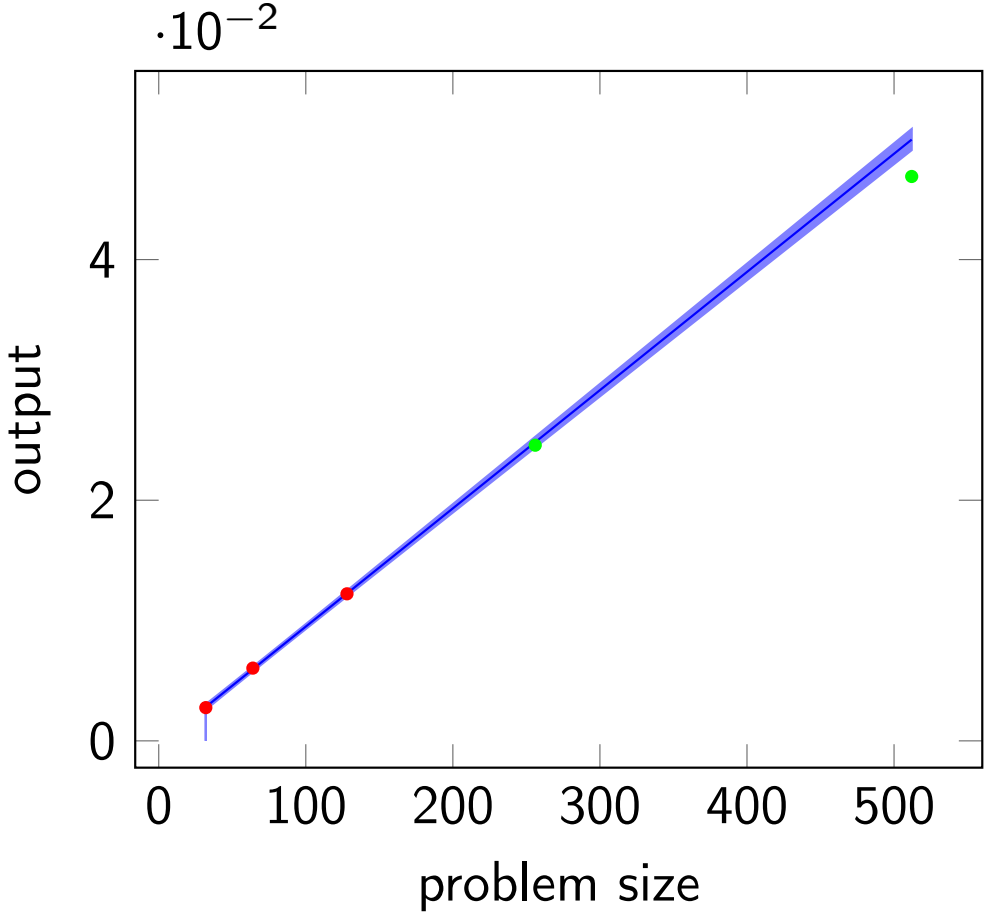
Ansatz A: 1D Extrapolation (problem size)

Training points, "Truth", Prediction (mean + 95% confidence)



Ansatz A: 1D Extrapolation (problem size)

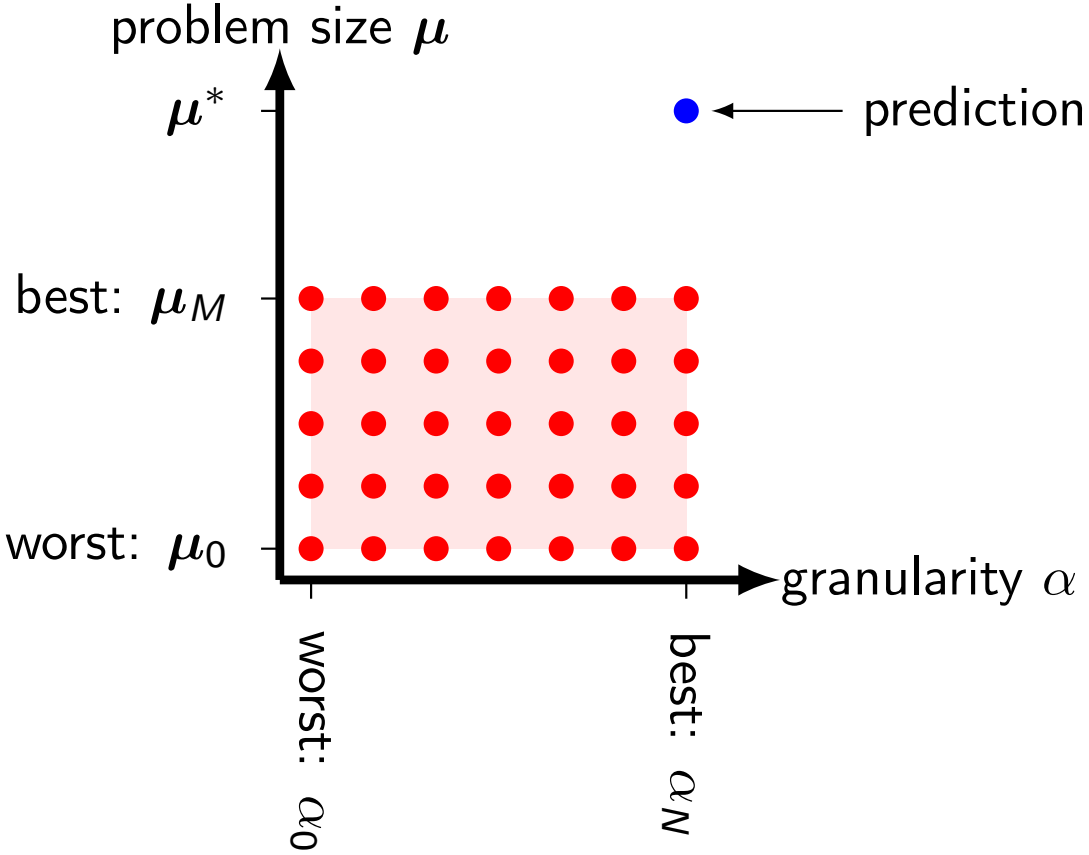
Training points, "Truth", Prediction (mean + 95% confidence)



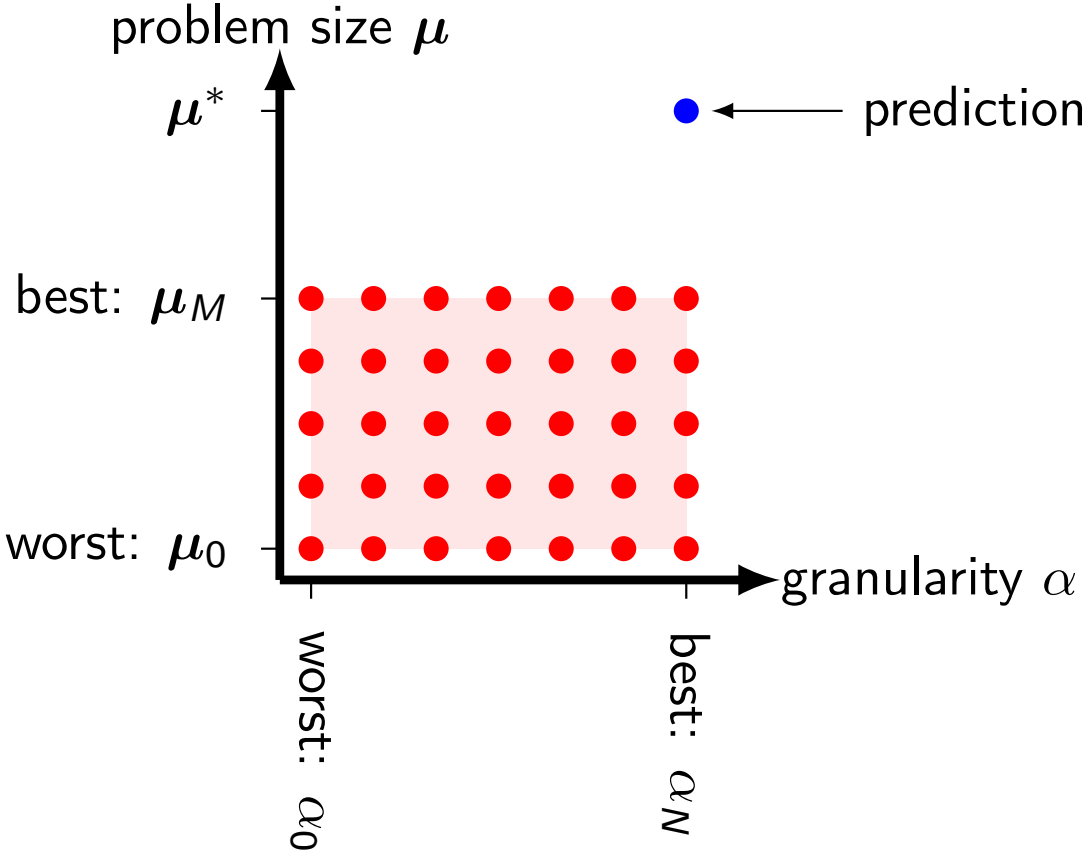
+ Relative error of mean prediction: 6.54%
+ Run-time gain: ≈ 18
- "Truth" in 95% confidence interval: No

- Cannot capture physical phenomenons

Method B: Approx-ES (Version I)

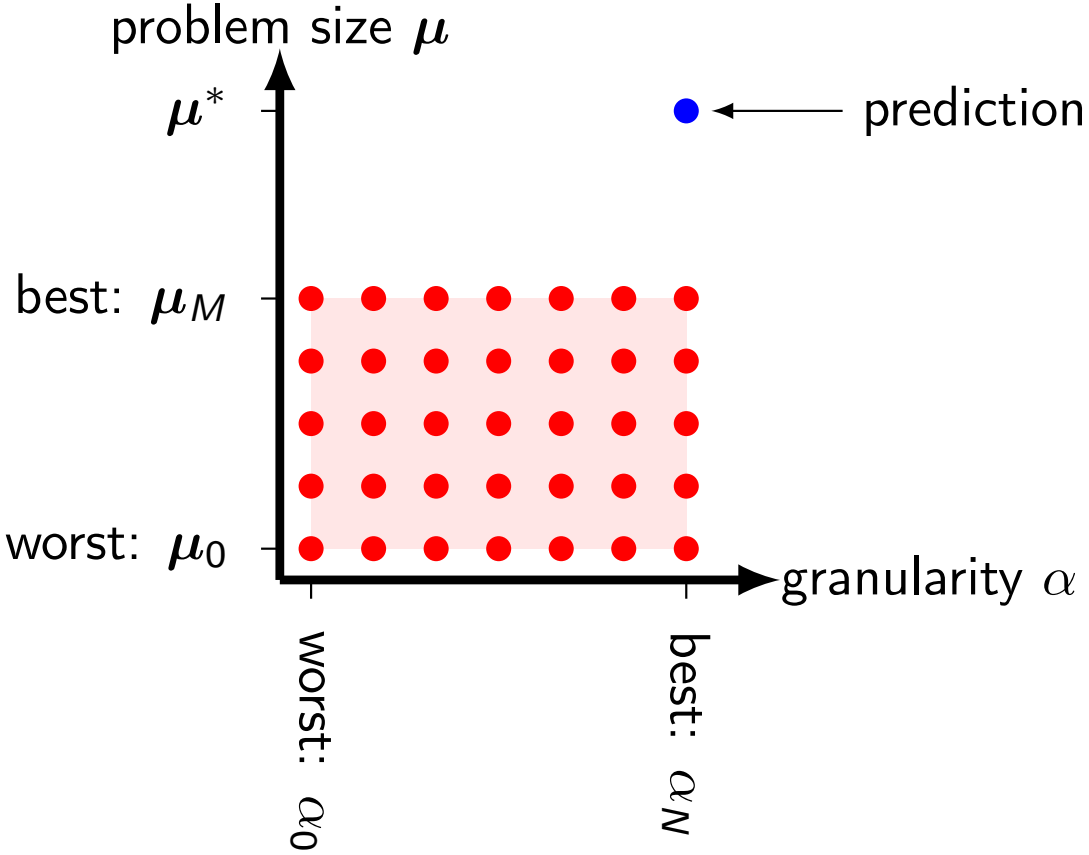


Method B: Approx-ES (Version I)



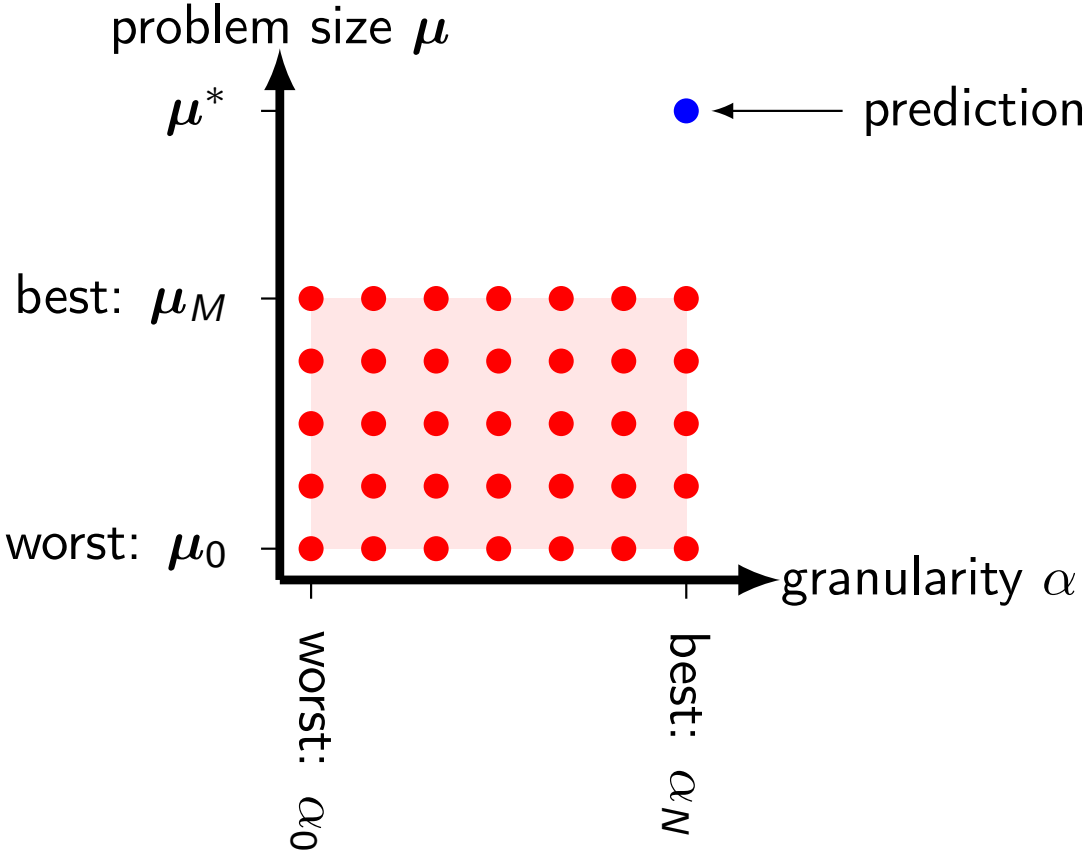
- Based on approximation errors: $\delta(\alpha, \boldsymbol{\mu}) := m_\alpha(\boldsymbol{\mu}) - M(\boldsymbol{\mu})$

Method B: Approx-ES (Version I)



- Based on approximation errors: $\delta(\alpha, \boldsymbol{\mu}) := m_\alpha(\boldsymbol{\mu}) - M(\boldsymbol{\mu})$
- Training domain: $\mathcal{D} := \{\alpha_0, \ldots, \alpha_N\} \times \{\boldsymbol{\mu}_0, \ldots, \boldsymbol{\mu}_M\}$
- Training set: $\mathcal{T} := \{((\alpha, \boldsymbol{\mu}), \delta(\alpha, \boldsymbol{\mu})) \,|\, (\alpha, \boldsymbol{\mu}) \in \mathcal{D}\}$

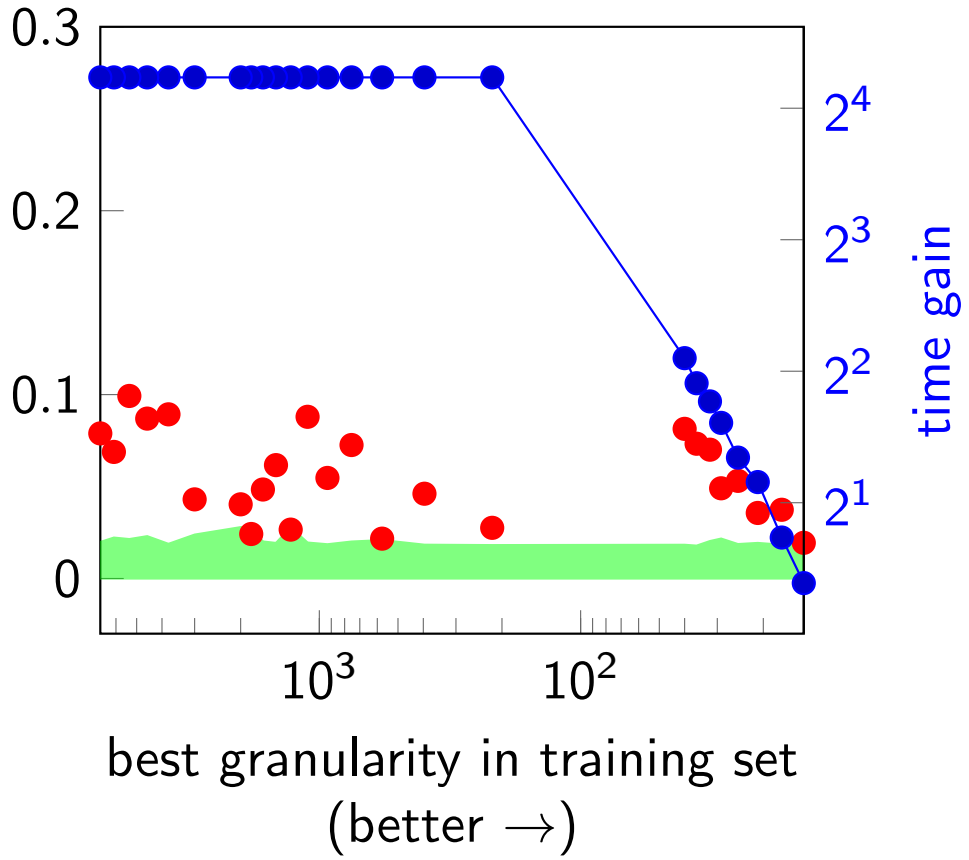
Method B: Approx-ES (Version I)



- Based on approximation errors: $\delta(\alpha, \boldsymbol{\mu}) := m_\alpha(\boldsymbol{\mu}) - M(\boldsymbol{\mu})$
- Training domain: $\mathcal{D} := \{\alpha_0, \ldots, \alpha_N\} \times \{\boldsymbol{\mu}_0, \ldots, \boldsymbol{\mu}_M\}$
- Training set: $\mathcal{T} := \{((\alpha, \boldsymbol{\mu}), \delta(\alpha, \boldsymbol{\mu})) \,|\, (\alpha, \boldsymbol{\mu}) \in \mathcal{D}\}$
- Prediction: $\widetilde{M}(\boldsymbol{\mu}^*) = m_{\alpha_N}(\boldsymbol{\mu}^*) - \mathcal{G}_{\mathcal{T}}(\alpha_N, \boldsymbol{\mu}^*)$

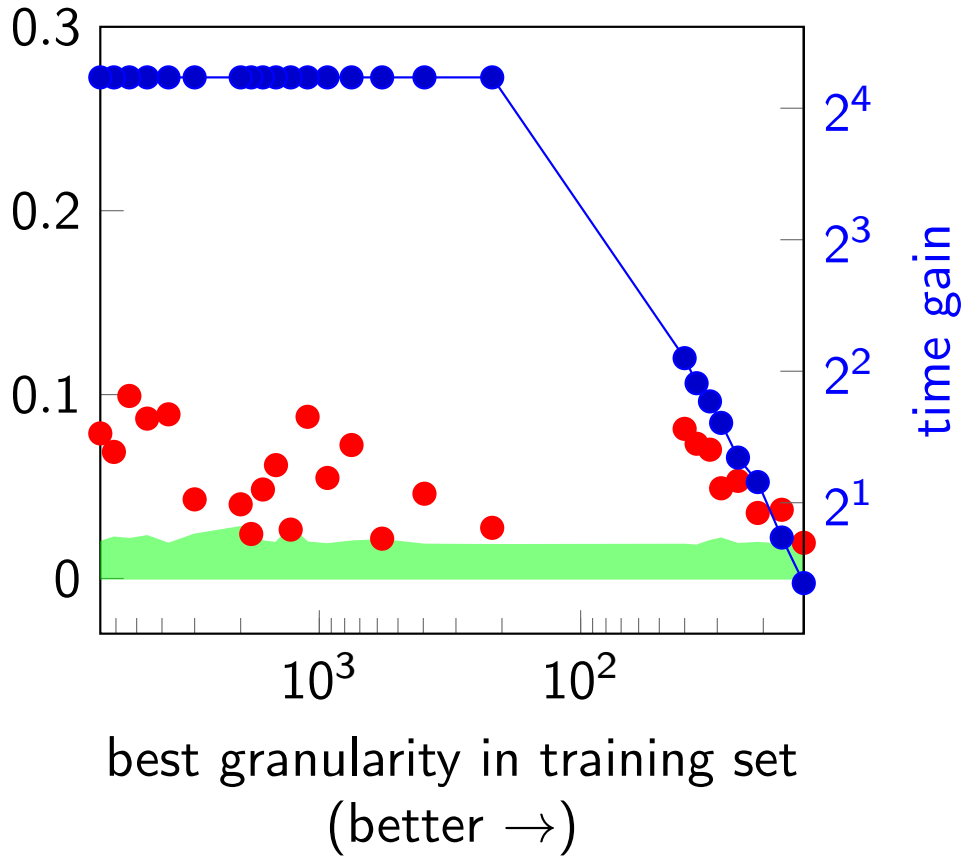
Method B: Approx-ES (Version I)

Mean prediction error, Confidence, Time gain



Method B: Approx-ES (Version I)

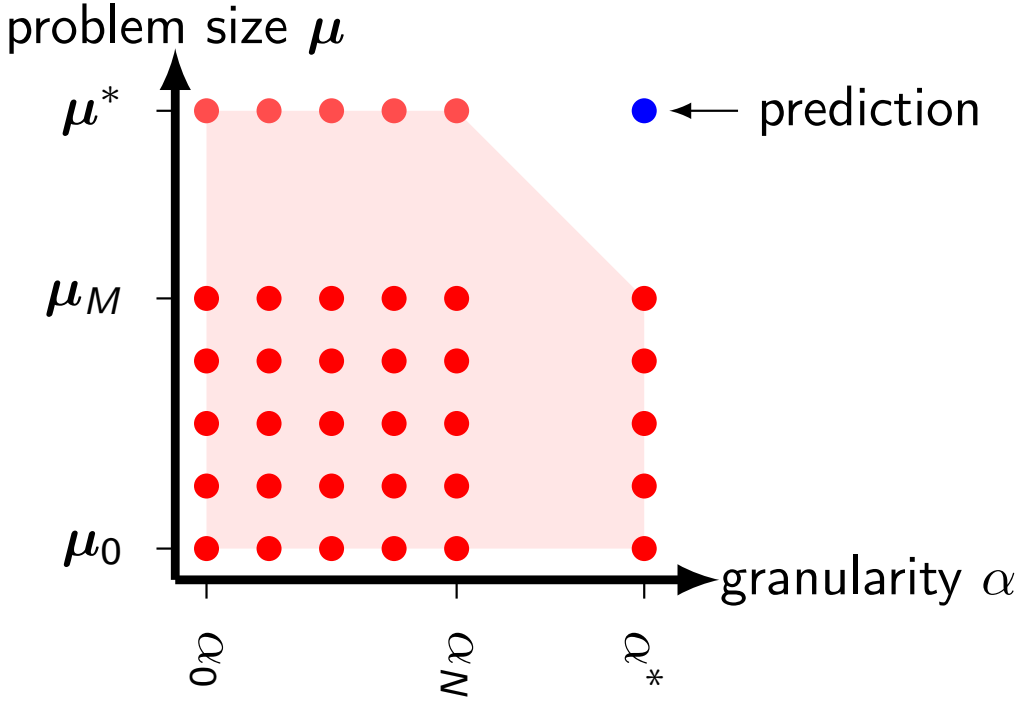
Mean prediction error, Confidence, Time gain



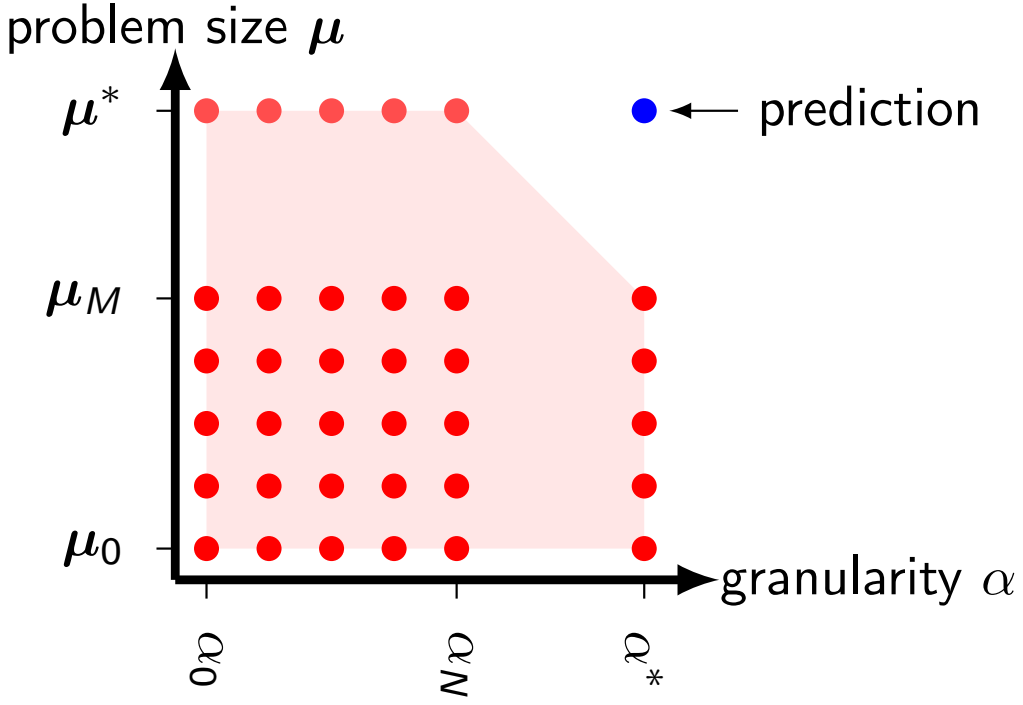
+ Relative error of mean prediction: $2.1 - 8.2\%$
+ Run-time gain: ≈ 18
- "Truth" in 95% confidence interval: 6%

- No useful uncertainty quantification

Method C: Approx-ES (Version II)

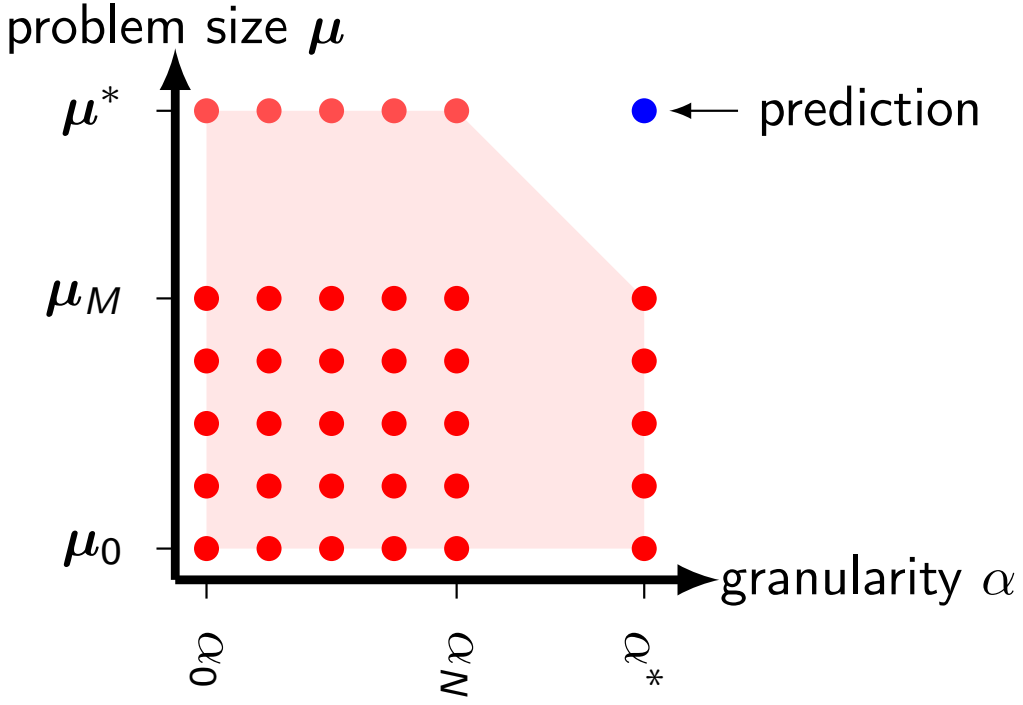


Method C: Approx-ES (Version II)



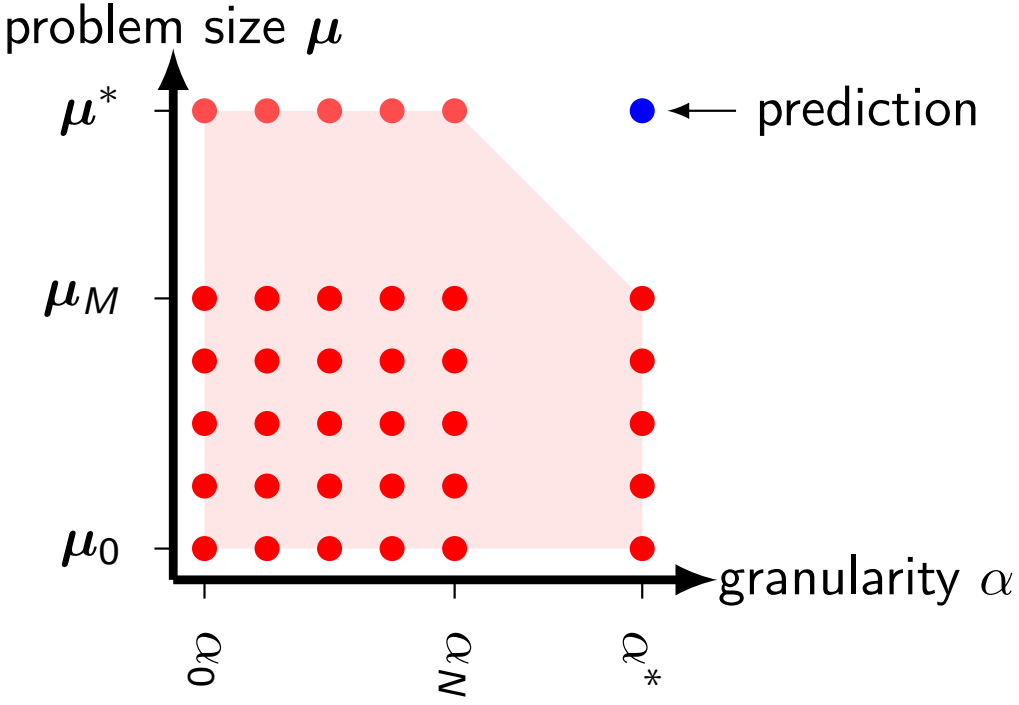
- Based on pivot error
  $\Delta(\alpha, \boldsymbol{\mu}) := m_{\alpha_N}(\boldsymbol{\mu}) - m_\alpha(\boldsymbol{\mu})$
- NB: $\Delta(\alpha^*, \boldsymbol{\mu}) = \delta(\alpha_N, \boldsymbol{\mu})$

Method C: Approx-ES (Version II)



- Based on pivot error $\Delta(\alpha, \boldsymbol{\mu}) := m_{\alpha_N}(\boldsymbol{\mu}) - m_\alpha(\boldsymbol{\mu})$
- NB: $\Delta(\alpha^*, \boldsymbol{\mu}) = \delta(\alpha_N, \boldsymbol{\mu})$
- Training domain: $\mathcal{D} := \{\alpha_0, \ldots, \alpha_N\} \times \{\boldsymbol{\mu}_0, \ldots, \boldsymbol{\mu}_M, \boldsymbol{\mu}^*\} \cup \{\alpha^*\} \times \{\boldsymbol{\mu}_0, \ldots, \boldsymbol{\mu}_M\}$
- Training set: $\mathcal{T} := \{((\alpha, \boldsymbol{\mu}), \Delta(\alpha, \boldsymbol{\mu})) \,|\, (\alpha, \boldsymbol{\mu}) \in \mathcal{D}\}$

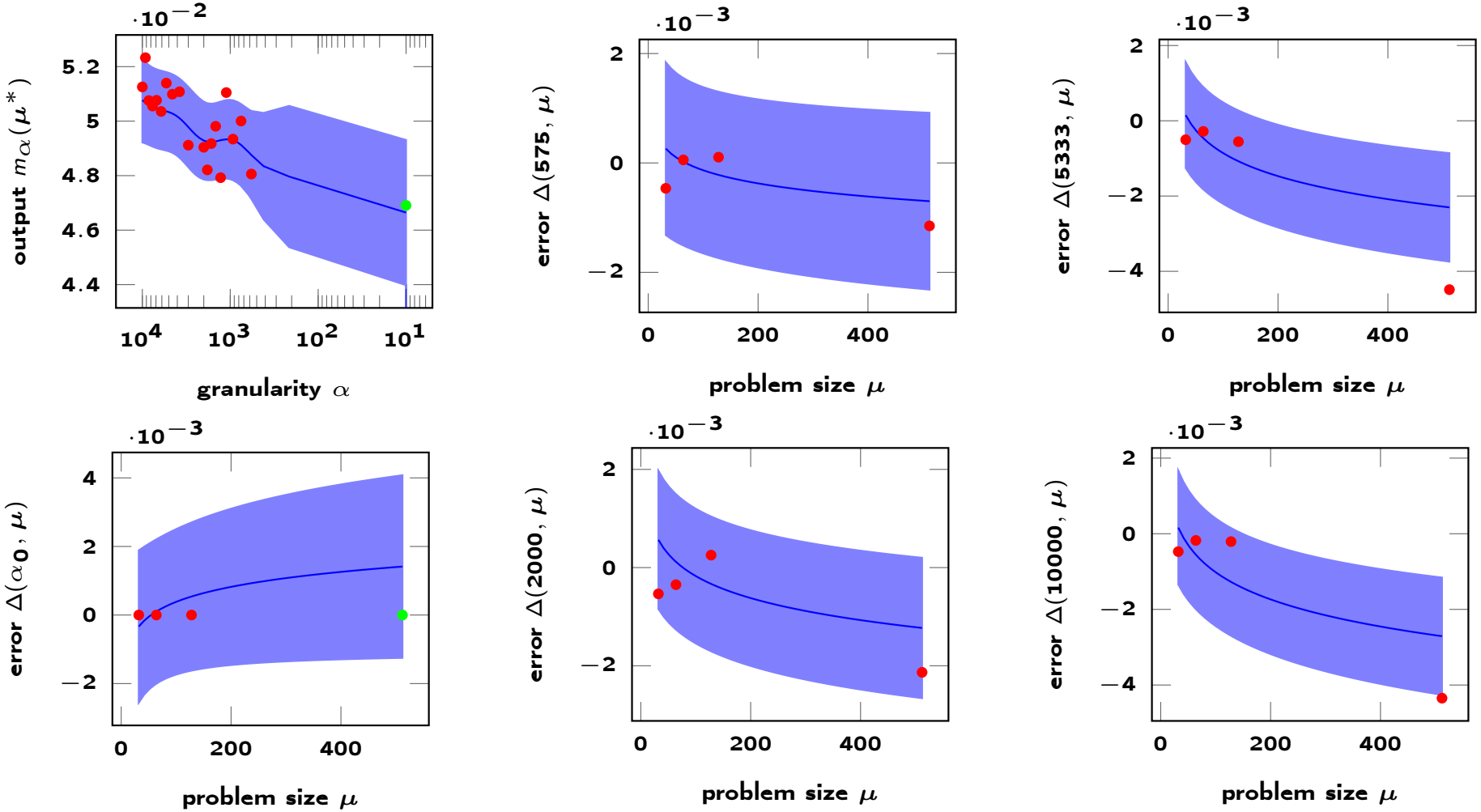
Method C: Approx-ES (Version II)



- Based on pivot error $\Delta(\alpha, \boldsymbol{\mu}) := m_{\alpha_N}(\boldsymbol{\mu}) - m_\alpha(\boldsymbol{\mu})$
- NB: $\Delta(\alpha^*, \boldsymbol{\mu}) = \delta(\alpha_N, \boldsymbol{\mu})$
- Training domain: $\mathcal{D} := \{\alpha_0, \ldots, \alpha_N\} \times \{\boldsymbol{\mu}_0, \ldots, \boldsymbol{\mu}_M, \boldsymbol{\mu}^*\} \cup \{\alpha^*\} \times \{\boldsymbol{\mu}_0, \ldots, \boldsymbol{\mu}_M\}$
- Training set: $\mathcal{T} := \{((\alpha, \boldsymbol{\mu}), \Delta(\alpha, \boldsymbol{\mu})) \,|\, (\alpha, \boldsymbol{\mu}) \in \mathcal{D}\}$
- Prediction: $\widetilde{M}(\boldsymbol{\mu}^*) = m_{\alpha_N}(\boldsymbol{\mu}) - \mathcal{G}_{\mathcal{T}}(\alpha^*, \boldsymbol{\mu}^*)$

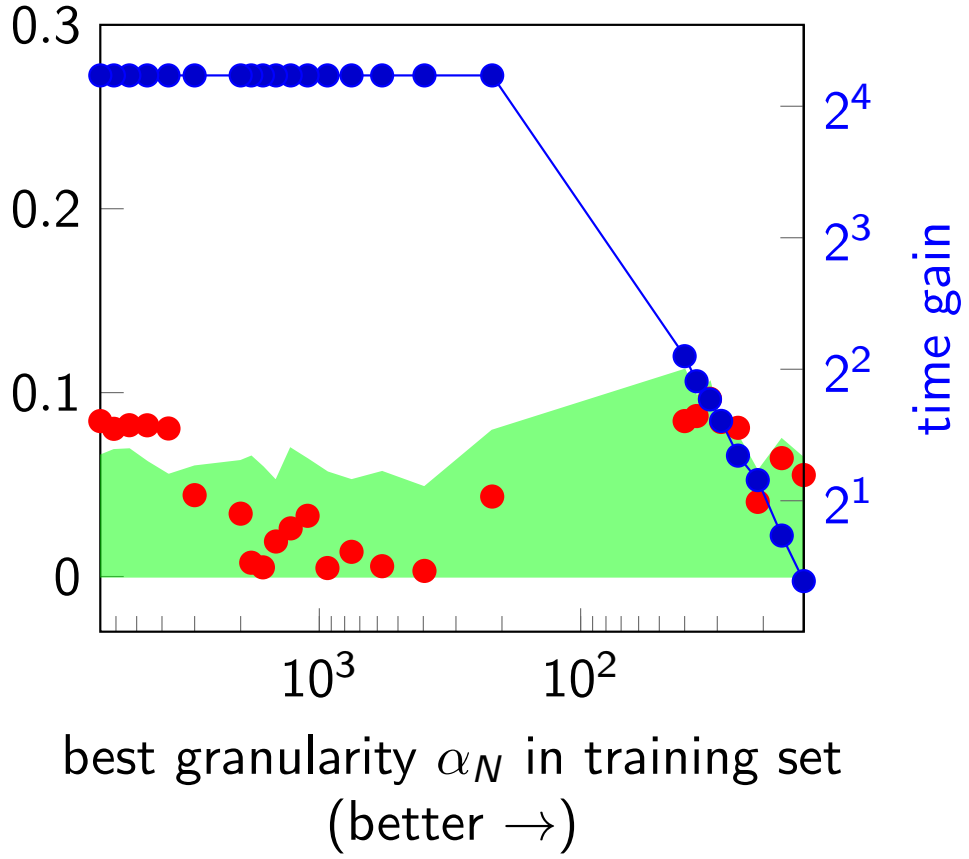
Method C: Approx-ES (Version II)

GP for best training granularity $\alpha_N = 575$:
Training points, "Truth", Prediction



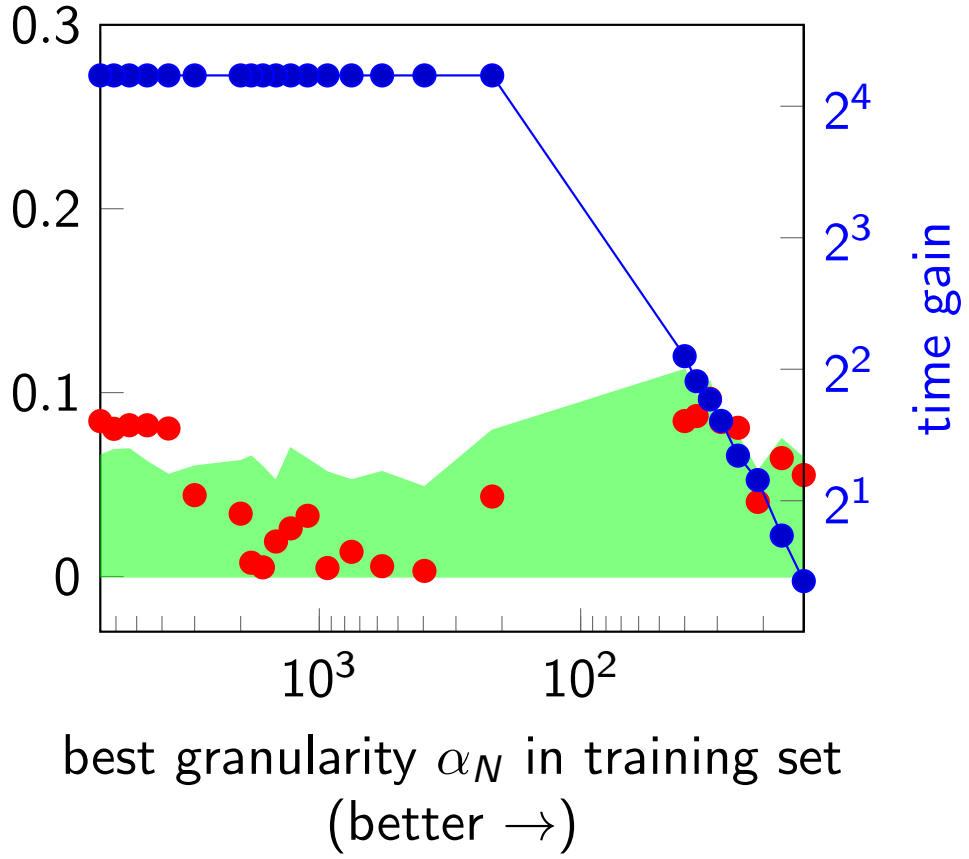
Method D: Approx-ES (Version II)

Mean prediction error, Confidence, Time gain



Method D: Approx-ES (Version II)

Mean prediction error, Confidence, Time gain



+ Relative error of mean prediction: $0.3 - 8.9\%$
+ Run-time gain: ≈ 18
+ "Truth" in 95% confidence interval: 85%

+ Good error correction, and useful uncertainty quantification

SST/macro: summary

	model-form	quantification of model-form uncertainty Δ	time for 10%	
			error	Δ
	full model $M(\boldsymbol{\mu})$	yes	1	1
	approximative model $m_\alpha(\boldsymbol{\mu})$	-	2^{-6}	-
A	extrapolation to problem size	useless	2^{-4}	-
B	Approx-ES I	useless	2^{-4}	2^{-4}
C	Approx-ES II	maybe	2^{-4}	2^{-4}

SST/macro: summary

	model-form	quantification of model-form uncertainty Δ	time for 10%	
			error	Δ
	full model $M(\boldsymbol{\mu})$	yes	1	1
	approximative model $m_\alpha(\boldsymbol{\mu})$	-	2^{-6}	-
A	extrapolation to problem size	useless	2^{-4}	-
B	Approx-ES I	useless	2^{-4}	2^{-4}
C	Approx-ES II	maybe	2^{-4}	2^{-4}
D	TODO: error estimation with good indicators	yes	?	?

Challenge: Find better and more physical indicators for Approx-ES

ROMES method [D & Carlberg, 2014]

ROMES method: Error surrogates for ROMs

Key observation

- Reduced order models often come with error bounds η, with

$$\delta(\alpha, \boldsymbol{\mu}) \leq \eta(\alpha, \boldsymbol{\mu}) := C \, \|r(\alpha, \boldsymbol{\mu})\|$$

- based on cheap residual $\|r(\alpha, \boldsymbol{\mu})\|$ and
- some expensive and ineffective factor C.

ROMES method [D & Carlberg, 2014]

ROMES method: Error surrogates for ROMs

Key observation

- Reduced order models often come with error bounds η, with

$$\delta(\alpha, \boldsymbol{\mu}) \leq \eta(\alpha, \boldsymbol{\mu}) := C \, \|r(\alpha, \boldsymbol{\mu})\|$$

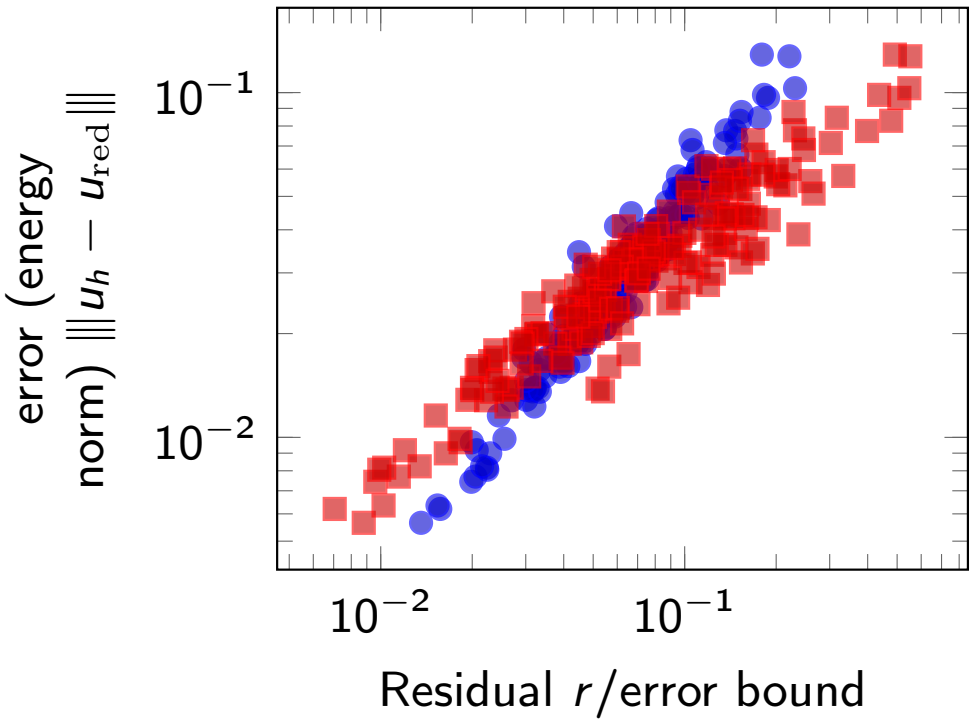
- based on cheap residual $\|r(\alpha, \boldsymbol{\mu})\|$ and
- some expensive and ineffective factor C.

- So, residuals are good error indicators?

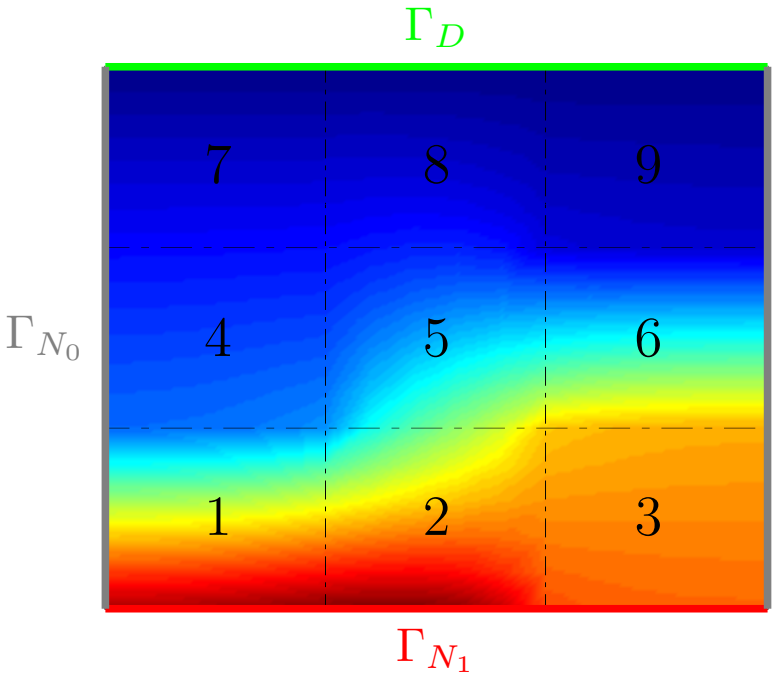
ROMES method [D & Carlberg, 2014]

ROMES method: Error surrogates for ROMs

So, residuals are good error indicators? Yes!!



ROMES method: Numerical example



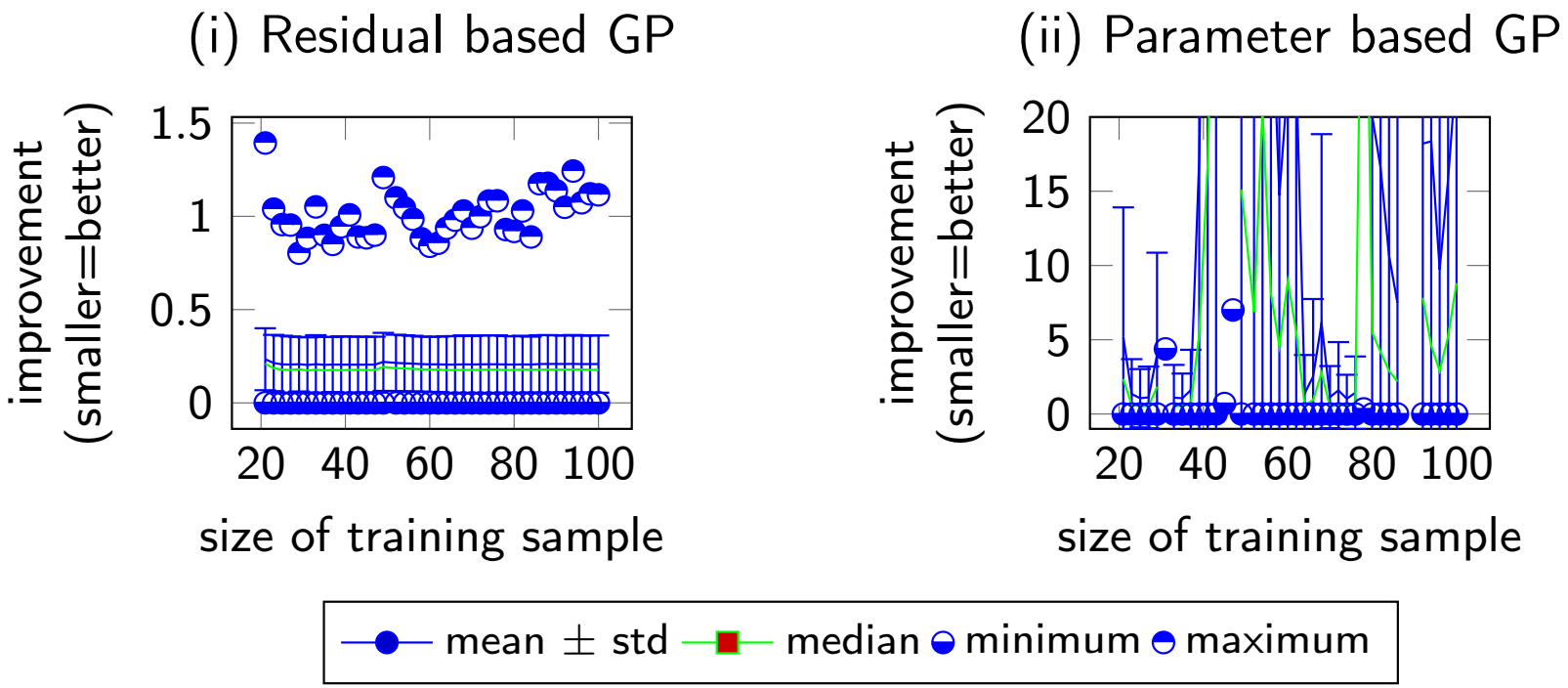
$$\triangle c(x;\boldsymbol{\mu})u(x;\boldsymbol{\mu}) = 0 \text{ in } \Omega \qquad u(\boldsymbol{\mu}) = 0 \text{ on } \Gamma_D$$

$$\nabla c(\boldsymbol{\mu})u(\boldsymbol{\mu}) \cdot n = 0 \text{ on } \Gamma_{N_0} \qquad \nabla c(\boldsymbol{\mu})u(\boldsymbol{\mu}) \cdot n = 1 \text{ on } \Gamma_{N_1}$$

- Inputs $\boldsymbol{\mu} \in [0.1, 10]^9$ define diffusivity c in subdomains
- Output $\boldsymbol{y}(\boldsymbol{\mu}) = \int_{\Gamma_{N_1}} u(\boldsymbol{\mu})\mathrm{d}x$ is compliant
- ROM constructed via RB–Greedy [Patera&Rozza, 2007]
- Consider two errors: 1) energy norm of state-space error $|||u(\boldsymbol{\mu}) - u_{\mathrm{red}}(\boldsymbol{\mu})|||$, 2) output bias $\boldsymbol{y}(\boldsymbol{\mu}) - \boldsymbol{y}_{\mathrm{red}}(\boldsymbol{\mu})$

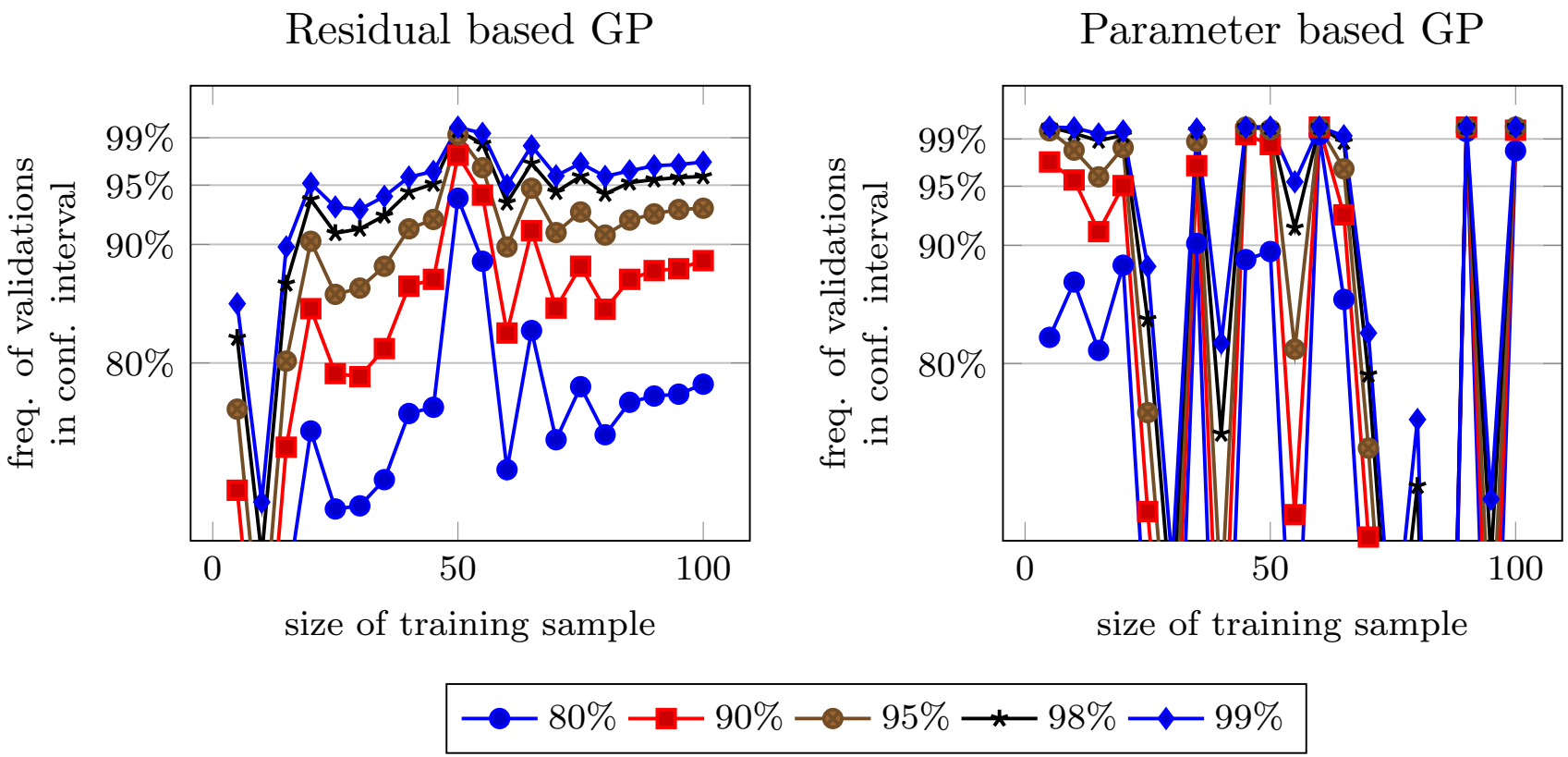
Bias improvement

$$\text{bias reduction} = \frac{E\left(\boldsymbol{y}_{\mathrm{red}}(\boldsymbol{\mu}) + \widetilde{\delta}_{\boldsymbol{y}}(\boldsymbol{\mu}) - \boldsymbol{y}(\boldsymbol{\mu})\right)}{\boldsymbol{y}_{\mathrm{red}}(\boldsymbol{\mu}) - \boldsymbol{y}(\boldsymbol{\mu})}$$



+ ROMES reduces bias by roughly an order of magnitude
- Multifidelity correction often *increases* the bias

Gaussian-process assumption verification



+ ROMES (Kernel GP, residual indicator) confidence intervals converge.
- Multifidelity correction (Kernel GP, input indicator) confidence intervals do not converge.

Conclusion

Error surrogates

- can be used to quantify and correct the model-form error of approximative or reduced order models.
- are
  - cheap,
  - trust-worthy and
  - allow for calibration
- rely on good error indicators.

Conclusion

Error surrogates

- can be used to quantify and correct the model-form error of approximative or reduced order models.
- are
  - cheap,
  - trust-worthy and
  - allow for calibration
- rely on good error indicators.

Future work

- Find error indicators for SST/macro simulator
- Apply ApproxES/ROMES to more and more complex / non-linear systems
- Integrate into UQ framework

Thanks!

SST/macro network model