WWU
MÜNSTER

A new Certified Hierarchical and Adaptive RB-ML-ROM Surrogate Model for Parametrized PDEs

International Conference on Adaptive Modeling and Simulation – ADMOS

Bernard Haasdonk, Hendrik Kleikamp, Mario Ohlberger,
Felix Schindler, Tizian Wenzel

June 20, 2023

living.knowledge

MM
Mathematics
Münster
Cluster of Excellence

Outline

Problem setting and surrogate models

Adaptive model hierarchy

Numerical experiment

Setting

Parametrized parabolic PDE in weak form: Find $u(\mu) \in L^2(0,T;V)$ s.t.

$$\begin{aligned} \langle \partial_t u(\mu), v\rangle + a(u(\mu), v; \mu) &= l(v;\mu), && \text{for all } v \in V, \\ u(0;\mu) &= u_0(\mu), && \text{for } u_0(\mu) \in V, \end{aligned}$$

with coercive bilinear form a, linear functional l and initial condition u_0.

Setting

Parametrized parabolic PDE in weak form: Find $u(\mu) \in L^2(0,T;V)$ s.t.

$$\begin{aligned} \langle \partial_t u(\mu), v\rangle + a(u(\mu), v; \mu) &= l(v;\mu), && \text{for all } v \in V, \\ u(0;\mu) &= u_0(\mu), && \text{for } u_0(\mu) \in V, \end{aligned}$$

with coercive bilinear form a, linear functional l and initial condition u_0.

Example later on:

- Monte Carlo estimation of a quantity of interest $\bar{f}(\mu)$ depending on $u(\mu)$.

Setting

Parametrized parabolic PDE in weak form: Find $u(\mu) \in L^2(0,T;V)$ s.t.

$$\begin{aligned} \langle \partial_t u(\mu), v\rangle + a(u(\mu), v; \mu) &= l(v;\mu), && \text{for all } v \in V, \\ u(0;\mu) &= u_0(\mu), && \text{for } u_0(\mu) \in V, \end{aligned}$$

with coercive bilinear form a, linear functional l and initial condition u_0.

Example later on:

- Monte Carlo estimation of a quantity of interest $\bar{f}(\mu)$ depending on $u(\mu)$.

Need for Model Order Reduction!

Reduced basis (RB) reduced order models

- Build reduced space $V_N \subset V_h$ with $N := \dim V_N \ll \dim V_h$ (e.g. using Greedy algorithm, POD, HaPOD,...).
- Perform Galerkin projection onto V_N instead of V_h.
- Full offline-online-decomposition under certain assumptions on parameter dependence of a and l.
- Solve small linear system in each time step t_k for the coefficients $\underline{u_{\mathrm{RB}}(\mu; t_k)} \in \mathbb{R}^N$ with respect to the basis $\varphi_1, \dots, \varphi_N$ of V_N.
- Reconstruct approximate solution by linear combination of $\varphi_1, \dots, \varphi_N$.

Reduced basis (RB) reduced order models

- Build reduced space $V_N \subset V_h$ with $N := \dim V_N \ll \dim V_h$ (e.g. using Greedy algorithm, POD, HaPOD,...).
- Perform Galerkin projection onto V_N instead of V_h.
- Full offline-online-decomposition under certain assumptions on parameter dependence of a and l.
- Solve small linear system in each time step t_k for the coefficients $\underline{u_{\text{RB}}(\mu; t_k)} \in \mathbb{R}^N$ with respect to the basis $\varphi_1, \dots, \varphi_N$ of V_N.
- Reconstruct approximate solution by linear combination of $\varphi_1, \dots, \varphi_N$.

Reduced basis (RB) reduced order models

- Build reduced space $V_N \subset V_h$ with $N := \dim V_N \ll \dim V_h$ (e.g. using Greedy algorithm, POD, HaPOD,...).
- Perform Galerkin projection onto V_N instead of V_h.
- Full offline-online-decomposition under certain assumptions on parameter dependence of a and l.
- Solve small linear system in each time step t_k for the coefficients $u_{\text{RB}}(\mu; t_k) \in \mathbb{R}^N$ with respect to the basis $\varphi_1, \dots, \varphi_N$ of V_N.
- Reconstruct approximate solution by linear combination of $\varphi_1, \dots, \varphi_N$.

Reduced basis (RB) reduced order models

- Build reduced space $V_N \subset V_h$ with $N := \dim V_N \ll \dim V_h$ (e.g. using Greedy algorithm, POD, HaPOD,...).
- Perform Galerkin projection onto V_N instead of V_h.
- Full offline-online-decomposition under certain assumptions on parameter dependence of a and l.
- Solve small linear system in each time step t_k for the coefficients $\underline{u_{\mathsf{RB}}(\mu; t_k)} \in \mathbb{R}^N$ with respect to the basis $\varphi_1, \dots, \varphi_N$ of V_N.
- Reconstruct approximate solution by linear combination of $\varphi_1, \dots, \varphi_N$.

Reduced basis (RB) reduced order models

- Build reduced space $V_N \subset V_h$ with $N := \dim V_N \ll \dim V_h$ (e.g. using Greedy algorithm, POD, HaPOD,...).
- Perform Galerkin projection onto V_N instead of V_h.
- Full offline-online-decomposition under certain assumptions on parameter dependence of a and l.
- Solve small linear system in each time step t_k for the coefficients $\underline{u_{\mathsf{RB}}(\mu; t_k)} \in \mathbb{R}^N$ with respect to the basis $\varphi_1, \dots, \varphi_N$ of V_N.
- Reconstruct approximate solution by linear combination of $\varphi_1, \dots, \varphi_N$.

Machine learning surrogate models

- Learn mapping

$$\mathcal{P} \ni \mu \mapsto \underline{u_{\mathsf{RB}}(\mu; t_k)} \in \mathbb{R}^N$$

using machine learning, e.g. kernel models or neural networks.

Machine learning surrogate models

- Learn mapping

$$\mathcal{P} \ni \mu \mapsto \underline{u_{\mathrm{RB}}(\mu; t_k)} \in \mathbb{R}^N$$

  using machine learning, e.g. kernel models or neural networks.
- Two different approaches:
  - *Time-"vectorized":* Train $\Phi\colon \mathbb{R}^p \to \mathbb{R}^{K \times N}$, with $p := \dim \mathcal{P}$ and K the number of time steps, to predict the ROM coefficients for all time steps at once.
  - *"Random-access"-in-time:* Train $\Phi\colon \mathbb{R}^{p+1} \to \mathbb{R}^N$ to predict the ROM coefficients at any point in time. (similar to [Wang/Hesthaven/Ray'19])

Machine learning surrogate models

- Learn mapping
$$\mathcal{P} \ni \mu \mapsto \underline{u_{\mathsf{RB}}(\mu; t_k)} \in \mathbb{R}^N$$
using machine learning, e.g. kernel models or neural networks.
- Two different approaches:
  - *Time-"vectorized":* Train $\Phi\colon \mathbb{R}^p \to \mathbb{R}^{K\times N}$, with $p := \dim \mathcal{P}$ and K the number of time steps, to predict the ROM coefficients for all time steps at once.
  - *"Random-access"-in-time:* Train $\Phi\colon \mathbb{R}^{p+1} \to \mathbb{R}^N$ to predict the ROM coefficients at any point in time. (similar to [Wang/Hesthaven/Ray'19])
- Evaluate $\Phi(\mu)$ to obtain DoF-vector $\underline{u_{\mathsf{ML}}(\mu; t_k)} \in \mathbb{R}^N$ and corresponding solution $u_{\mathsf{ML}}(\mu; t_k) \in V_N$.

Summary and comparison of the models

Full-order model (FOM)

Pros:

- arbitrarily accurate solutions (reference)

Cons:

- very slow when considering fine discretizations and many time steps

Summary and comparison of the models

Full-order model (FOM)

Pros:

- arbitrarily accurate solutions (reference)

Cons:

- very slow when considering fine discretizations and many time steps

Reduced model (ROM)

Pros:

- faster than FOM
- error estimator

Cons:

- slow iterative time stepping
- requires solving dense linear system in each time step

Summary and comparison of the models

Full-order model (FOM)

Pros:

- arbitrarily accurate solutions (reference)

Cons:

- very slow when considering fine discretizations and many time steps

Reduced model (ROM)

Pros:

- faster than FOM
- error estimator

Cons:

- slow iterative time stepping
- requires solving dense linear system in each time step

Machine learning (MLM)

Pros:

- faster than ROM
- parallel evaluation for different time instances possible

Cons:

- no certification available (so far)

Parabolic RB a posteriori error estimator [Grepl/Patera'05]

Offline-online-decomposable error estimate for difference between FOM-solution $u_h(\mu) \in V_h$ and ROM-solution $u_{\mathsf{RB}}(\mu) \in V_N$:

$$\|u_h(\mu) - u_{\mathsf{RB}}(\mu)\|_{L^2(0,T;V_h)} \leq E_{\mathsf{RB}}(u_{\mathsf{RB}}(\mu);\mu)$$

$$:= \underbrace{\alpha(\mu)}_{\substack{\text{coercivity}\\ \text{constant}}}^{-1} \Bigg(\sum_{k=1}^{K-1} \Delta t \| \underbrace{R_k(u_{\mathsf{RB}}(t_k;\mu);\mu)}_{\substack{\text{residuum in the}\\ k\text{-th time step}}} \|_{V_h'}^2 \Bigg)^{1/2}$$

Parabolic RB a posteriori error estimator [Grepl/Patera'05]

Offline-online-decomposable error estimate for difference between FOM-solution $u_h(\mu) \in V_h$ and ROM-solution $u_{\mathsf{RB}}(\mu) \in V_N$:

$$\begin{aligned}\|u_h(\mu) - u_{\mathsf{RB}}(\mu)\|_{L^2(0,T;V_h)} &\leq E_{\mathsf{RB}}(u_{\mathsf{RB}}(\mu);\mu) \\ &:= \underbrace{\alpha(\mu)}_{\substack{\text{coercivity} \\ \text{constant}}}^{-1} \Bigg(\sum_{k=1}^{K-1} \Delta t \| \underbrace{R_k(u_{\mathsf{RB}}(t_k;\mu);\mu)}_{\substack{\text{residuum in the} \\ k\text{-th time step}}} \|_{V_h'}^2 \Bigg)^{1/2}\end{aligned}$$

Idea: Apply error estimator E_{RB} to machine learning prediction $u_{\mathsf{ML}}(\mu)$!

$\Longrightarrow$ A posteriori certification of machine learning results!

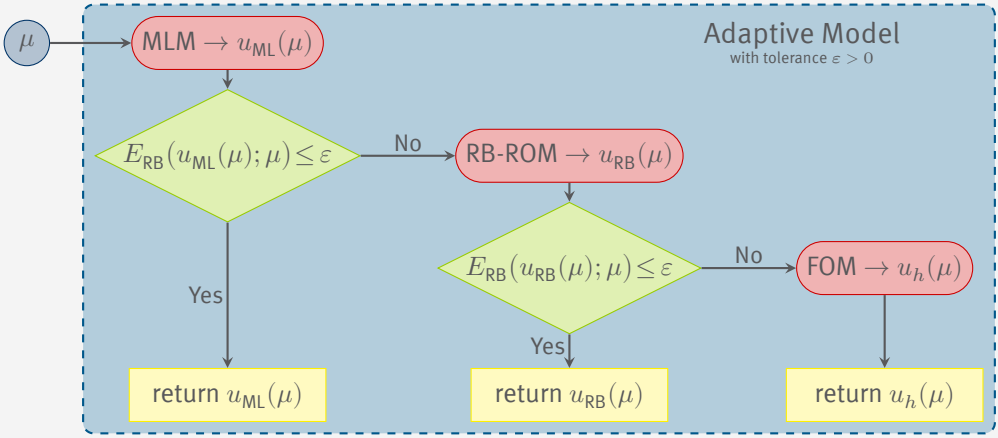
Outline

Problem setting and surrogate models

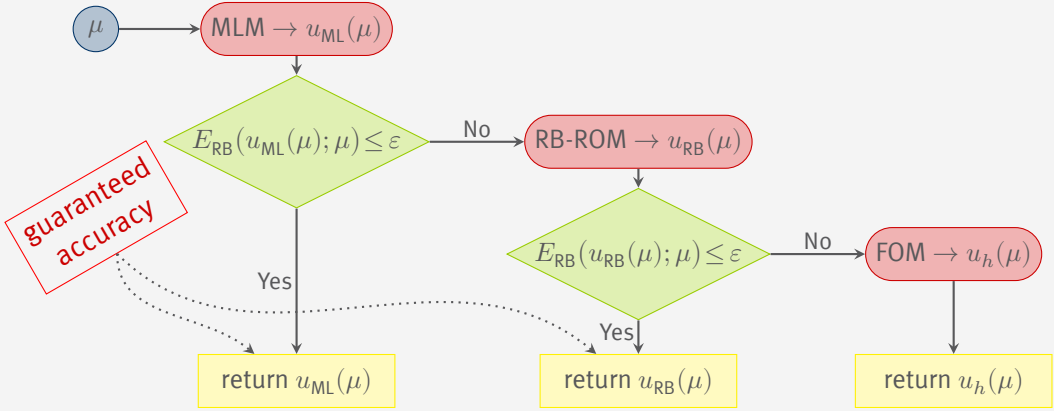
Adaptive model hierarchy

Numerical experiment

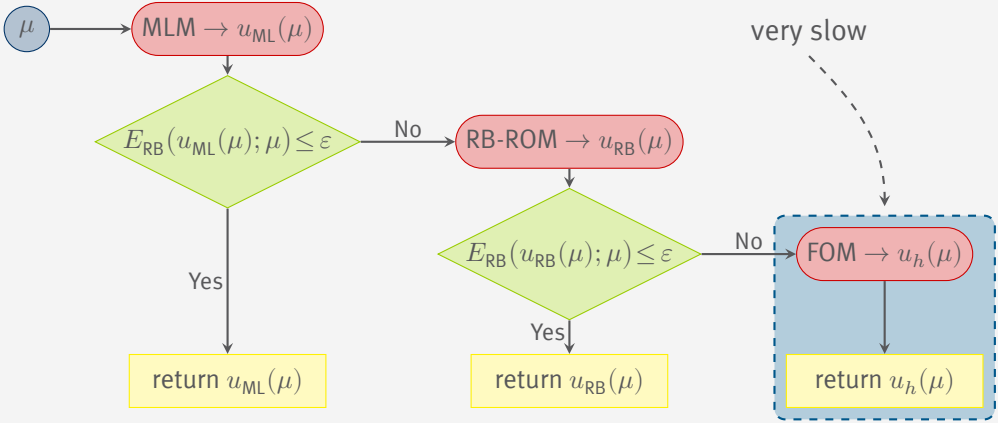
Adaptive model hierarchy [Haasdonk et al'22]



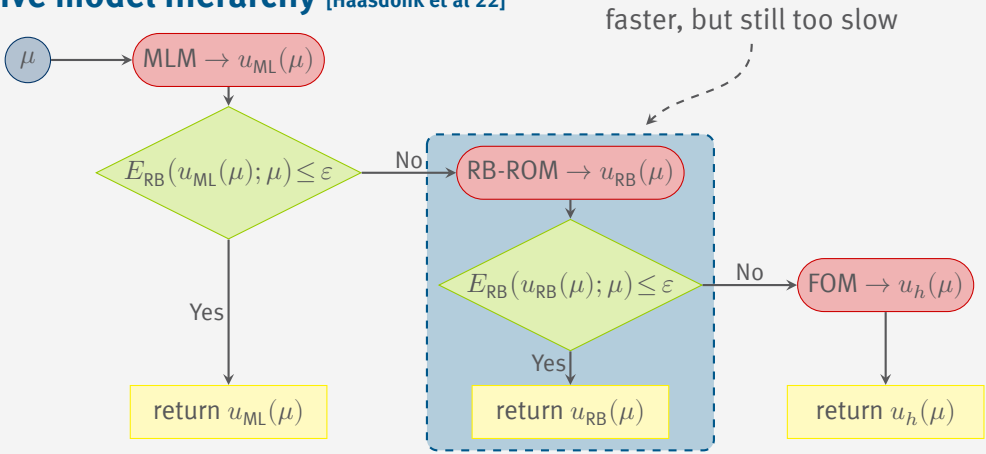
Adaptive model hierarchy [Haasdonk et al'22]



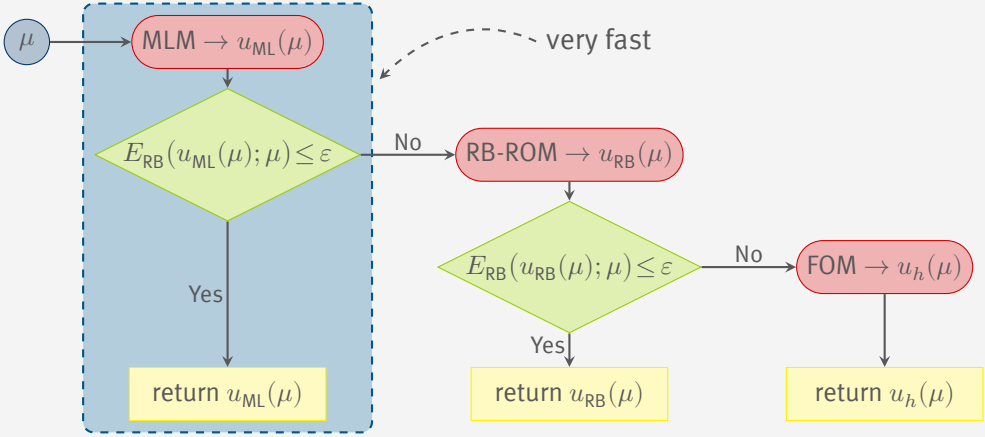
Adaptive model hierarchy [Haasdonk et al'22]



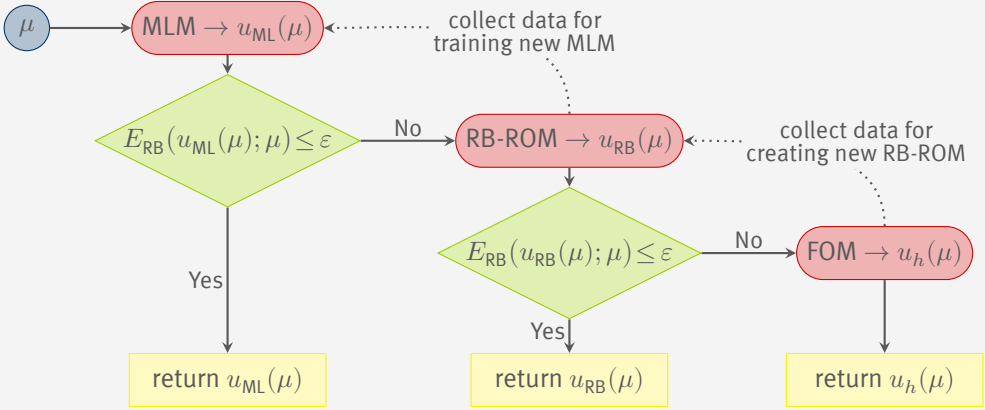
Adaptive model hierarchy [Haasdonk et al'22]



Adaptive model hierarchy [Haasdonk et al'22]



Adaptive model hierarchy [Haasdonk et al'22]



Outline

Problem setting and surrogate models

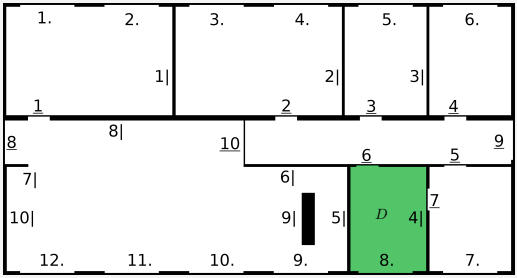
Adaptive model hierarchy

Numerical experiment

Numerical experiment – Monte Carlo estimation

Parameter-dependent bilinear form a and source term l:

$$a(u, v; \mu) = \int_\Omega \kappa(\mu)\, \nabla u(x) \cdot \nabla v(x)\, \mathrm{d}x, \qquad l(v; \mu; t) = \int_\Omega v\, \ell(\mu; t)\, \mathrm{d}x.$$



Numerical experiment – Monte Carlo estimation

Parameter-dependent bilinear form a and source term l:

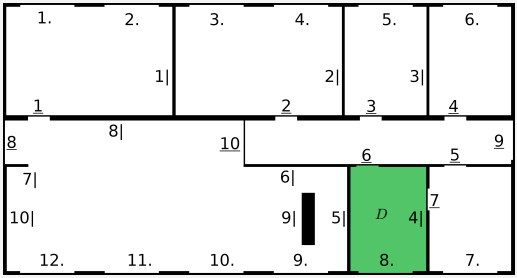
$$a(u, v; \mu) = \int_\Omega \kappa(\mu)\, \nabla u(x) \cdot \nabla v(x)\, \mathrm{d}x, \qquad l(v; \mu; t) = \int_\Omega v\, \ell(\mu; t)\, \mathrm{d}x.$$

Quantity of interest at time $t \in [0, T]$:

$$f(\mu; t) := \frac{1}{|D|} \int_D u(t, x; \mu)\, \mathrm{d}x$$

Average over time interval $\tau \subset [0, T]$:

$$\bar{f}(\mu) := \frac{1}{|\tau|} \int_\tau f(\mu; t)\, \mathrm{d}t$$



Numerical experiment – Monte Carlo estimation

Expectation and variance of $\bar{f}$ with respect to density $\rho\colon \mathcal{P} \to [0, \infty)$:

$$\mathrm{E}[\bar{f}] := \int_{\mathcal{P}} \rho(\mu)\bar{f}(\mu)\,\mathrm{d}\mu, \qquad \mathrm{Var}[\bar{f}] := \int_{\mathcal{P}} \rho(\mu)\left(\bar{f}(\mu) - \mathrm{E}[\bar{f}]\right)^2\,\mathrm{d}\mu$$

Numerical experiment – Monte Carlo estimation

Expectation and variance of $\bar{f}$ with respect to density $\rho\colon \mathcal{P} \to [0, \infty)$:

$$\mathrm{E}[\bar{f}] := \int_{\mathcal{P}} \rho(\mu)\bar{f}(\mu)\,\mathrm{d}\mu, \qquad \mathrm{Var}[\bar{f}] := \int_{\mathcal{P}} \rho(\mu)\left(\bar{f}(\mu) - \mathrm{E}[\bar{f}]\right)^2\,\mathrm{d}\mu$$

Monte Carlo estimation of $\mathrm{E}[\bar{f}]$ and $\mathrm{Var}[\bar{f}]$:

$$\mathrm{E}[\bar{f}] \approx \hat{\theta}_{\bar{f}} := \frac{1}{N_{\mathrm{mc}}} \sum_{\mu \in \mathcal{P}_{\mathrm{mc}}} \bar{f}(\mu),$$

$$\mathrm{Var}[\bar{f}] \approx \frac{1}{N_{\mathrm{mc}} - 1} \sum_{\mu \in \mathcal{P}_{\mathrm{mc}}} \left(\bar{f}(\mu) - \hat{\theta}_{\bar{f}}\right)^2,$$

where $\mathcal{P}_{\mathrm{mc}} \subset \mathcal{P}$ is randomly sampled according to ρ.

Numerical experiment – Surrogate models

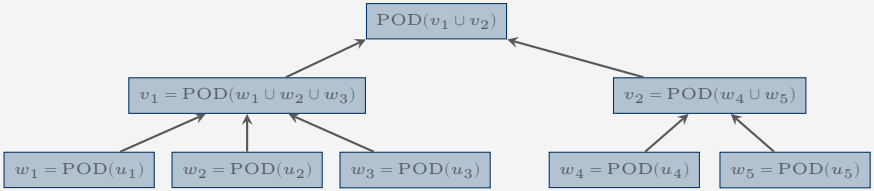
Components of the adaptive model:

- FOM: Finite Element grid with $N_h = 321{,}206$ DoFs, $K = 1000$ time steps

Numerical experiment – Surrogate models

Components of the adaptive model:

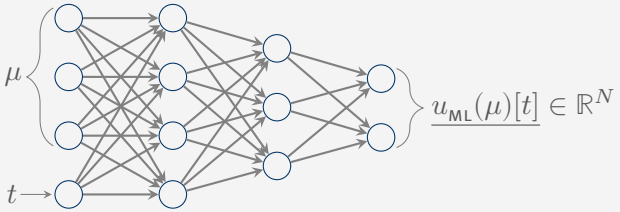
- FOM: Finite Element grid with $N_h = 321{,}206$ DoFs, $K = 1000$ time steps
- RB-ROM: **H**ierarchical **a**pproximate **P**roper **O**rthogonal **D**ecomposition (HaPOD)
  [Himpe/Leibner/Rave'18]



Numerical experiment – Surrogate models

Components of the adaptive model:

- MLM (option 1): “Random-access”-in-time approach with Deep Neural Network (DNN)
  [Wang/Hesthaven/Ray’19]

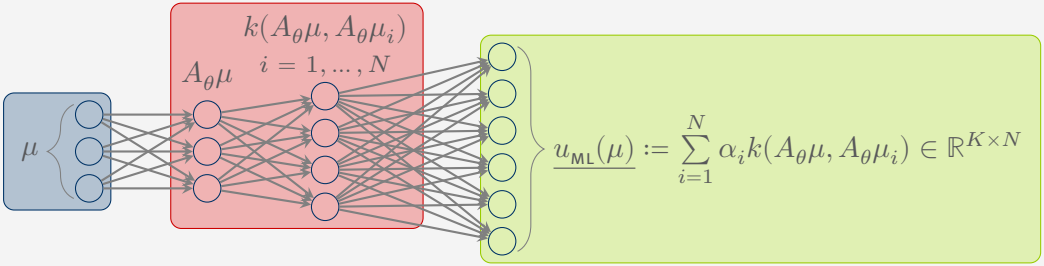


(Training is only performed after 200 new ROM solutions are available.)

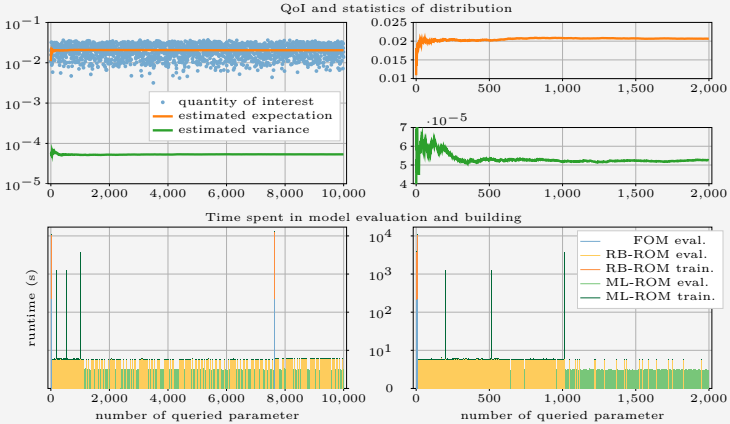
Numerical experiment – Surrogate models

Components of the adaptive model:

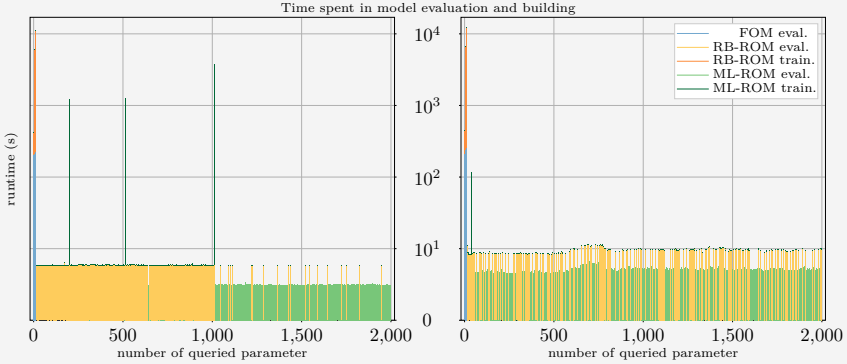
- MLM (option 2): Time-“vectorized” approach with two-layered **V**ectorial **K**ernel **O**rthogonal **G**reedy **A**lgorithm (2L-VKOGA) [Wenzel/Marchetti/Perracchione’23], [Santin/Haasdonk’21]



Numerical experiment – Monte Carlo using DNN (option 1)



Numerical experiment – Comparison of DNN and 2L-VKOGA



Comparison of DNN (left) and 2L-VKOGA (right) used for the RB-ML-ROM.

Thank you for your attention!

For more details, see:

B. Haasdonk, H. Kleikamp, M. Ohlberger, F. Schindler, T. Wenzel,
A New Certified Hierarchical and Adaptive RB-ML-ROM Surrogate Model for Parametrized PDEs,
SIAM Journal on Scientific Computing, Vol. 45, 3 (2023), DOI: 10.1137/22M1493318

T. Wenzel, B. Haasdonk, H. Kleikamp, M. Ohlberger, F. Schindler,
Application of Deep Kernel Models for Certified and Adaptive RB-ML-ROM Surrogate Modeling,
to appear in: Proceedings of the 14th International Conference on Large-Scale Scientific Computations (2023), https://arxiv.org/abs/2302.14526

References I

M. Grepl and A. Patera, *A posteriori error bounds for reduced-basis approximations of parametrized parabolic partial differential equations*, ESAIM: Mathematical Modelling and Numerical Analysis, 39 (2005), pp. 157–181, DOI: 10.1051/m2an:2005006.

Q. Wang, J. S. Hesthaven, and D. Ray, *Non-intrusive reduced order modeling of unsteady flows using artificial neural networks with application to a combustion problem*, J. Comput. Phys., 384 (2019), pp. 289–307, DOI: 10.1016/j.jcp.2019.01.031.

T. Wenzel, Francesco Marchetti, and Emma Perracchione, *Data-driven kernel designs for optimized greedy schemes: A machine learning perspective* arXiv preprint arxiv:2301.08047, 2023. Submitted.

References II

G. Santin and B. Haasdonk, *Kernel methods for surrogate modeling*, in Model Order Reduction, P. Benner, S. Grivet-Talocia, A. Quarteroni, G. Rozza, W. Schilders, and L. M. Silveira, eds., vol. 2, De Gruyter, 2021, pp. 311–353.

C. Himpe, L. Leibner and S. Rave, *Hierarchical approximate proper orthogonal decomposition*, in SIAM Journal on Scientific Computing, vol. 40, 2018, pp. A3267–A3292.