A Trust Region RB-ML-ROM Approach for Parabolic PDE Constrained Optimization ⋆

Benedikt Klein * **Mario Ohlberger** *

* *Mathematics Münster, University of Münster, Münster, Germany, (e-mail: {benedikt.klein, mario.ohlberger}@uni-muenster.de).*

1. INTRODUCTION & PROBLEM SETUP

Optimization problems constrained by parabolic PDEs are common in science and engineering, involving challenges like optimal control and inverse problems. Traditional methods require numerous iterations to solve discretized PDEs, often creating a computational bottleneck. Surrogate models, especially reduced order models (ROM) obtained from reduced basis (RB) methods, offer increased efficiency by approximating these high-fidelity solutions. This contribution explores how machine learning (ML) can enhance surrogate model construction in the framework of error aware trust region (TR) optimization methods.

Consider a bounded domain $\Omega \subset \mathbb{R}^d$ and let V be a Hilbert space, with $H_0^1(\Omega) \subset V \subset H^1(\Omega)$. Our goal is to find a parameter μ within a bounded parameter domain $\mathcal{P} \subset \mathbb{R}^P$, minimizing the least-squares objective functional, relative to a desired state $g_{\text{ref}} \in V^K$:

$$J(u(\mu);\mu) := \Delta t \sum_{k=1}^{K} \|u^k(\mu) - g_{\text{ref}}^k\|_V^2 + \lambda \mathcal{R}(\mu),$$

where $u(\mu) := (u^k(\mu))_{k\in\{0,\dots,K\}} \in V^{K+1}$ represents the solution trajectory to a time-discretized parametrized parabolic PDE (primal problem), i.e, $u(\mu)$ solves

$$\frac{(u^k(\mu) - u^{k-1}(\mu), v)_{L^2(\Omega)}}{\Delta t} + a(u^k(\mu), v; \mu) = b(t^k) f(v;\mu)$$
$$u^0(\mu) = 0$$

for all $v \in V$ and $k \in \{1,\dots,K\}$, where $a(\cdot,\cdot;\mu)$ and $f(\cdot\,;\mu)$ are parameter-dependent (bi)linear forms, and b a time-dependent forcing input. Regularization is provided by a smooth $\mathcal{R}(\mu)$ for $\lambda > 0$, cf. Qian et al. (2017).

The gradient of $\mathcal{J}(\mu) := J(u(\mu);\mu)$ will be computed via an adjoint approach, by $\nabla_\mu \mathcal{J}(\mu) = \nabla_\mu \mathcal{L}(u(\mu), p(\mu);\mu)$, with the Lagrangian $\mathcal{L}$ and the adjoint solution $p(\mu) \in V^{K+1}$ w.r.t. $u(\mu)$, cf. Qian et al. (2017) for details.

2. MODEL REDUCTION AND MACHINE LEARNING

To numerically solve the primal and adjoint problems, spatial discretization is employed, projecting the problems into a high-dimensional space $V_h \subset V$. RB methods, project these problems further into a space $V_{\text{RB}} \subset V_h$ with significant lower dimension, thus reducing the computational burdens. This approach yields RB approximations $u_{\text{RB}}(\mu)$ and $p_{\text{RB}}(\mu)$ to the high-fidelity solutions and thereby for the objective functional $\mathcal{J}_{\text{RB}}(\mu) := J(u_{\text{RB}}(\mu);\mu)$ and its gradient $\nabla_\mu \mathcal{J}_{\text{RB}}(\mu)$. Additionally, an a posteriori error estimator $\Delta^J_{\text{RB}}(\mu)$ measuring the error $|\mathcal{J}(\mu) - \mathcal{J}_{\text{RB}}(\mu)|$ is provided, cf. Qian et al. (2017); Keil et al. (2021).

A way to further reduce computational costs is to replace these RB-ROMs with a machine learning surrogate, cf. Fresca and Manzoni (2022); Haasdonk et al. (2023). These models allow, by design, faster evaluations, providing an approximation to $u_{\text{ML}}(\mu)$. Suitable ML models are, for example, kernel methods. Here, the Degree of Freedom vector for the primal RB trajectory $\underline{u_{\text{RB}}(\mu)}$ is approximated by a linear combination of smooth kernel functions. The associated coefficients are learned using previously collected RB solutions as training data, $(\mu_1, \underline{u_{\text{RB}}(\mu_1)}), \dots, (\mu_N, \underline{u_{\text{RB}}(\mu_N)})$. The surrogate to the objective functional, utilizing an ML-ROM, is defined similarly to that of RB-ROMs. However, the gradient will be calculated directly by applying the chain rule.

3. TRUST REGION OPTIMIZATION

A trust region method iteratively replaces the global optimization problem by solving a sequence of sub-problems restricted to a local trust region $T^{(i)} \subset \mathcal{P}$, i.e.,

$$\mu^{(i+1)} := \operatorname*{argmin}_{\mu\in\mathcal{P}} J^{(i)}(\mu) \text{ s.t. } \mu \in T^{(i)}, \tag{1}$$

employing a local surrogate $J^{(i)}$ to $\mathcal{J}$, for $i \in \mathbb{I} := \{0,\dots,I\}$, with $I \in \mathbb{N}_0 \cup \{\infty\}$.

Problem (1) is addressed using a projected BFGS method with an Armijo-type backtracking line search, starting at $\mu^{(i)}$ and stopping if a suitable termination criterion is reached, generating a sequence $(\mu^{(i,l)})_{l\in\{0,\dots,L^{(i)}\}}$, cf. Keil et al. (2021). The guess $\mu^{(i,L^{(i)})}$ will be rejected and (1) solved again with a shrunken trust region, if a sufficient decay condition is not satisfied and accepted otherwise.

For each $i \in \mathbb{I}$, let $V_{\text{RB}}^{(i)} \subset V_h$ be fixed RB spaces, iteratively constructed by basis enrichment, starting with $V_{\text{RB}}^{(0)} := \langle\emptyset\rangle$. This enrichment is performed by computing high-fidelity solutions at $\mu^{(i)}$ and applying proper orthogonal decomposition (POD) to them. After orthogonalization, the so selected singular vectors are added to the basis of $V_{\text{RB}}^{(i)}$. This process uniquely defines for all $i \in \mathbb{I}$ the RB surrogate

⋆ Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy EXC 2044–390685587, Mathematics Münster: Dynamics–Geometry–Structure.

DOI: 10.34726/9000

$J_{\mathrm{RB}}^{(i)}(\mu)$, its gradient $\nabla_\mu J_{\mathrm{RB}}^{(i)}(\mu)$, and the error estimator $\Delta_{\mathrm{RB}}^{J,(i)}(\mu)$.

For the ML models, in contrast, it is essential to re-adapt (retrain) them to the RB solution manifold at a relatively high frequency compared to the RB space enrichment. This adaptation is necessary for ensuring that the error remains acceptably small.

Let $J_{\mathrm{ML}}^{(i,m)}(\mu)$, $\nabla_\mu J_{\mathrm{ML}}^{(i,m)}(\mu)$ for $m \in \mathbb{N}$, denote the approximative cost functional and its gradient given by trajectories in $V_{\mathrm{RB}}^{(i)}$, yielded from the m-th ML-ROM. When the underlying ML model is (re)trained, the functionals are updated to $J_{\mathrm{ML}}^{(i,m+1)}(\mu)$ and $\nabla_\mu J_{\mathrm{ML}}^{(i,m+1)}(\mu)$.

In general, ML surrogate models lack an *efficent* a posteriori error estimator. For RB-ML-ROMs, however, it is possible to extend the error estimators of the RB-ROMs to the ML setting. Nevertheless, the evaluation of such estimators is usually very costly relative to the faster parameter inference of ML models Haasdonk et al. (2023). This could offset the reduction in computational time gained by the ML-ROM, when training and less accurate approximations of the ML model are counted in.

This raises the problem that a solely error-based trust region, such as

$$\tilde{T}^{(i)} := \left\{ \mu \in \mathcal{P} \;\middle|\; \Delta_{\mathrm{RB}}^{J,(i)}(\mu)/J_{\mathrm{RB}}^{(i)}(\mu) \leq \epsilon^{(i)} \right\}$$

for some $\epsilon^{(i)} > 0$, as outlined in Qian et al. (2017), cannot be used due to the lack of efficient error estimations for the ML surrogate. To address this, we choose

$$T^{(i)} := \left(\tilde{T}^{(i)} \cup \bigcup_{\mu \in \partial \tilde{T}^{(i)}} \left\{ \tilde{\mu} \in \mathbb{R}^P \,\middle|\, \|\tilde{\mu} - \mu\|_{\mathbb{R}^{\mathbb{P}}} \leq \kappa^{(i)} \right\} \right) \cap \mathcal{P}$$

with $\kappa^{(i)} := \alpha_0^{(i)} l_{\mathrm{check}}$, where $l_{\mathrm{check}} \in \mathbb{N}$ and $\alpha_0^{(i)}$ is the step size from the Armijo-type line search. By verifying $\mu^{(i,l)} \in \tilde{T}^{(i)}$ every l_{check}-th step during the BFGS optimization, we ensure that all queried parameters are within $T^{(i)}$. If this condition fails, the last accepted parameter $\mu^{(i,l)}$ is returned as an approximate solution to (1). For shrinking the trust region, we set $\epsilon^{(i+1)} := \beta_1 \epsilon^{(i)}$ and $\alpha_0^{(i+1)} := \beta_2 \alpha_0^{(i)}$, where $\beta_1, \beta_2 \in (0,1)$, if $\mu^{(i+1)}$ is rejected.

Utilizing ML-ROM poses another challenge: the line search may result in excessively small updates of the parameter, due to the generally larger error of the ML models compared to RB-ROMs. We therefore propose terminating the line search for ML surrogates if it does not succeeds within a reasonable number of steps. In this case, the line search is rerun using $J_{\mathrm{RB}}^{(i)}(\mu)$ and $\nabla_\mu J_{\mathrm{RB}}^{(i)}(\mu)$ and the so collected RB trajectories will be used to retrain the ML-ROM. If the second backtracking also fails, the last accepted step $\mu^{(i,l)}$ is returned. Initially, the first $l_{\mathrm{warmup}} \in \mathbb{N}$ line searches use RB-ROMs only to gather training data. Details are given in Algorithm 1.

4. CONCLUSION

This contribution shows how machine learning can be integrated in a trust region method for parameter optimizations constrained by parabolic PDEs, by combining reduced basis models and kernel-based ML surrogates. However, maintaining accuracy requires frequent retraining and error management due to inherent errors of the ML-ROM. This highlights the potential for efficiently solving high-dimensional problems, but emphasizing the need to carefully balance between efficiency and accuracy.

REFERENCES

Fresca, S. and Manzoni, A. (2022). POD-DL-ROM: Enhancing deep learning-based reduced order models for nonlinear parametrized PDEs by proper orthogonal decomposition. *Comput. Methods Appl. Mech. Eng.*, 388, 114181.

Haasdonk, B., Kleikamp, H., Ohlberger, M., Schindler, F., and Wenzel, T. (2023). A new certified hierarchical and adaptive RB-ML-ROM surrogate model for parametrized PDEs. *SIAM SISC*, 45(3), A1039–A1065.

Keil, T., Mechelli, L., Ohlberger, M., Schindler, F., and Volkwein, S. (2021). A non-conforming dual approach for adaptive trust-region reduced basis approximation of PDE-constrained parameter optimization. *ESAIM: M2AN*, 55(3), 1239–1269. doi:10.1051/m2an/2021019.

Qian, E., Grepl, M., Veroy, K., and Willcox, K. (2017). A certified trust region reduced basis approach to PDE-constrained optimization. *SIAM SISC*, 39(5), S434–S460.

Algorithm 1 Inner loop

1: Set $l := 0$, $m := 0$, $\mu^{(i,0)} := \mu^{(i)}$, `no_progress` := `false`, `use_ML` := `true` and choose $l_{\mathrm{warmup}}, k_{\max} \in \mathbb{N}$.
2: **while** a termination criteria is not met or $l = 0$ **do**
3: **if** $l \bmod l_{\mathrm{check}} = 0$ or `no_progress` **then**
4: **if** $\mu^{(i,l)} \notin \tilde{T}^{(i)}$ **then** Go to line 33.
5: **end if**
6: Set $k := 0$ and `no_progress` $\leftarrow$ `false`.
7: **while** a line search stopping criteria is not met **do**
8: Get $\mu^{(i,l)}(k)$ by k-th line search iteration.
9: **if** $l \leq l_{\mathrm{warmup}}$ or not `use_ML` **then**
10: Get $J_{\mathrm{RB}}^{(i)}(\mu^{(i,l)}(k)), \nabla_\mu J_{\mathrm{RB}}^{(i)}(\mu^{(i,l)}(k))$
11: **else**
12: Get $J_{\mathrm{ML}}^{(i,m)}(\mu^{(i,l)}(k)), \nabla_\mu J_{\mathrm{ML}}^{(i,m)}(\mu^{(i,l)}(k))$
13: **end if**
14: **if** $k = k_{\max}$ and $l > 0$ **then**
15: `no_progress` $\leftarrow$ `true`
16: Go to line 20.
17: **end if**
18: $k \leftarrow k + 1$
19: **end while**
20: **if** `no_progress` and not `use_ML` **then**
21: Go to line 33.
22: **else if** `no_progress` **then**
23: `use_ML` $\leftarrow$ `false`
24: Update the search direction, using RB-ROMs.
25: **else**
26: `use_ML` $\leftarrow$ `true` and `no_progress` $\leftarrow$ `false`
27: $\mu^{(i,l+1)} := \mu^{(i,l)}(k)$
28: Update the search direction.
29: Train ML-ROM.
30: $l \leftarrow l + 1$ and $m \leftarrow m + 1$
31: **end if**
32: **end while**
33: **return** $\mu^{(i+1)} := \mu^{(i,l)}$