Reduced basis methods for PDEs

P.-J. Bénard[1] L. Duguet[1] S. Treillou[1]

[1]GMM - MMN
INSA Toulouse

May 2020

Introduction

The Reduced Basis Method (RBM) provides a tool allowing to solve faster Parametrized PDEs with a certified and controled error.

Interests

- real-time computation
- parameters optimization

Introduction

The Reduced Basis Method (RBM) provides a tool allowing to solve faster Parametrized PDEs with a certified and controled error.

Interests

- real-time computation
- parameters optimization

Contents

Reduced basis method explanation

Let consider a linear PDE, solved with following system $Ax = b$ in discretized solutions space $\mathbf{V}_\delta$, with A a dense square matrix of sixe $n \times n$. The best possible solver has complexity $O(n^2)$.
If we halve the linear system, we divide time by 4. This is the RBM principle.

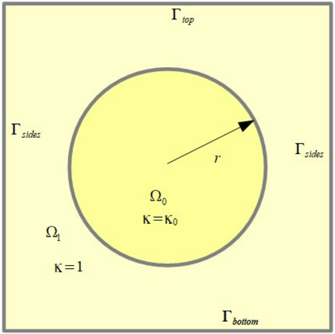
RBM principle

The RBM is find the best compromise between reduction for faster computation and error certification.

The toy problem

on which we will demonstrate the RBM's efficiency

Heat propagation equation with parameters (μ_1, μ_2)



$$\begin{cases} \nabla \cdot \mu_1 \nabla u(\mu) & = & 0 & \text{in } \Omega_0, \\ \Delta u(\mu) & = & 0 & \text{in } \Omega_1, \\ u(\mu) & = & 0 & \text{on } \Gamma_{top}, \\ \nabla u(\mu) \cdot n & = & 0 & \text{on } \Gamma_{side}, \\ \nabla u(\mu) \cdot n & = & \mu_2 & \text{on } \Gamma_{base}. \end{cases}$$

Reduced basis method explanation

with the POD method

Proper Orthogonal Decomposition (POD)

- Compute all solutions for a chosen discretized parameter space

Reduced basis method explanation

with the POD method

Proper Orthogonal Decomposition (POD)

- Compute all solutions for a chosen discretized parameter space
- Perform an analysis to retain relevant information using Singular Value Decomposition (SVD)

Proper Orthogonal Decomposition (POD)

- Compute all solutions for a chosen discretized parameter space
- Perform an analysis to retain relevant information using Singular Value Decomposition (SVD)

For a given reduced space dimension, this is the best possible basis in terms of space approximation.

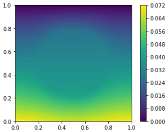


Figure: Reduced basis solution

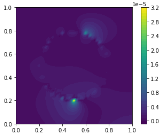


Figure: Difference with truth solution

Gain of time : approx. 20 000 times faster for very small error.

Greedy Algorithm

- Iterative method based on local choices

Reduced basis method explanation

with the Greedy algorithm

Greedy Algorithm

- Iterative method based on local choices
- Pseudo-code:
  - Initialize the reduced basis $\mathbb{V}_{rb}$ with a random chosen vector.
  - For each iteration, adds to $\mathbb{V}_{rb}$ the vector with the biggest error
  - Stops iteration once biggest error is smaller than the tolerance

Reduced basis method explanation

with the Greedy algorithm

Greedy Algorithm

- Iterative method based on local choices
- Pseudo-code:
  - Initialize the reduced basis $\mathbb{V}_{rb}$ with a random chosen vector.
  - For each iteration, adds to $\mathbb{V}_{rb}$ the vector with the biggest error
  - Stops iteration once biggest error is smaller than the tolerance

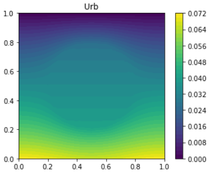


Figure: Reduced basis solution

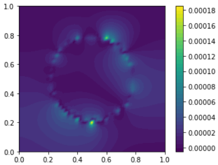


Figure: Difference with truth solution

POD vs. Greedy algorithm : comparison

Convergence rate

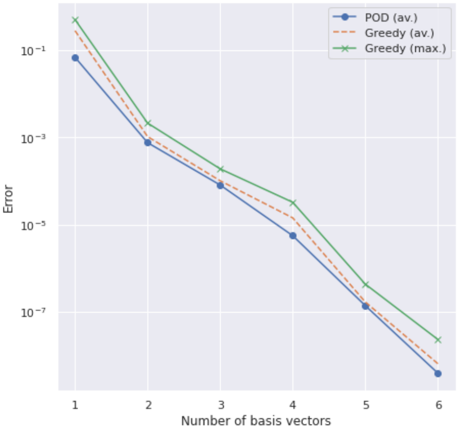


Figure: Errors comparison between Greedy algorithm and POD

For a given number of basis vector, $err^{avg}_{Greedy} \geq err^{avg}_{POD}$.
The average error with the Greedy algorithm is higher for a given number of vector in the reduced basis than the one with the POD.

POD vs. Greedy algorithm : comparison

Computation time

	POD	Greedy algorithm
Accuracy [1]	Highest possible accuracy	Less accurate than POD
Computation time [2]	Slower than POD	Faster than POD because of its iterative form

Note that we cannot illustrate the computation time assertion due to difference in implementations.

[1] For a given basis dimension

[2] For a given tolerance

Implementation of RBM for Burgers equation

Burgers equation, a nonlinear PDE

$$\partial_t u(t,x) + u(t,x)\partial_x u(t,x) - \mu \partial_{xx} u(t,x) = 0 \ , \ \forall t \in]0;T], x \in [0;1]$$

Implementation of RBM for Burgers equation

Burgers equation, a nonlinear PDE

$$\partial_t u(t,x) + u(t,x)\partial_x u(t,x) - \mu\partial_{xx} u(t,x) = 0 \ , \ \forall t \in]0; T], x \in [0; 1]$$

Cole Hopf transformation

$$u(t,x) = -2\mu \frac{\partial_x v(t,x)}{v(t,x)} \ , \ \forall t \in [0; T], x \in [0; 1]$$

Offline phase:

- Discretize parameter space;
- Generate reduced basis for the heat equation with POD;
- Precompute quantities from the affine assumption.

Offline phase:

- Discretize parameter space;
- Generate reduced basis for the heat equation with POD;
- Precompute quantities from the affine assumption.

Online phase:

- Assemble operators from precomputed quantities for a given parameter;
- Compute $V_i^0 = e^{\frac{-1}{2\mu} \int_0^{x_i} u_0(s) ds}$;
- while $t_j < T$ solve the linear system at current time $A^j V^j = V^{j-1}$;
- Back to Burgers solution: $U_i^j = -\mu \frac{V_{i+1}^j - V_{i-1}^j}{\delta_x V_i^j}$;

Implementation of RBM for Burgers

Results for a 20 vectors basis

Error calculation

$$mean\ error(Approx, Ref) = \frac{1}{N \times T_{max}} \sum_{i=0}^{N} \sum_{j=0}^{Tmax} |Approx_i^j - Ref_i^j|$$

$$ratio\ error\ max(Approx, Ref) = \frac{\max_{i=0}^{N} \max_{j=0}^{Tmax} |Approx_i^j - Ref_i^j|}{\max_{i=0}^{N} \max_{j=0}^{Tmax} |Ref_i^j|}$$

	mean error	ratio error max
reduced heat solution	2.307×10^{-6}	2.809×10^{-5}
reduced Burgers solution	5.302×10^{-5}	1.175×10^{-3}

Time:

Reduced Solver Time: 3.83 ms True Solver Time: 9.85 ms $\Rightarrow$ 2.5x faster

Main results and discussion

Our results

- The implemented RBM works
- POD provides a lower error than the Greedy algorithm for a given reduced basis dimension
- These algorithms are robust
- Non-linear PDEs can be resolved in the same way

Main results and discussion

Our results

- The implemented RBM works
- POD provides a lower error than the Greedy algorithm for a given reduced basis dimension
- These algorithms are robust
- Non-linear PDEs can be resolved in the same way

To go further

- discuss about Cole-Hopf reduced basis stability
- implement parallel computing or data training
- implement the POD-Greedy algorithm

Any questions ?