A non intrusive reduced basis method for heat transfer problem

Rachida CHAKIR

in collaboration with Yvon Maday, Pascal Joly and Philippe Parnaudeau
Motivations

Eco-city

Data center hots spots

ENERGY SAVING

ENERGY EFFICIENTY

Needs
- 2D/3D Numerical Modeling
- Fast and reliable methods
- Control of quantity of interest
- Uncertainties quantifications

Challenges
- Multiphysics Modeling
- Non-Linearities and Coupling
- Complex geometries
- Optimization

Reduced basis methods
**Context**: optimization process or characterization in real-time of systems governed by a parameters dependent PDEs.

→ Classicals discretization techniques such as finite element methods are generally too expensives

Given $\mu$ in $\mathcal{D} \subset \mathbb{R}^d$,

$\Rightarrow$ Find $u^N(\mu)$ in $X_N$ such that

$$a(u^N(\mu), v^N; \mu) = f(v^N; \mu)$$

$\Rightarrow$ The reduced basis (R.B.) methods exploits the parametric structure of the governing PDE to construct rapidly convergent and computationally efficient approximations.

$\Rightarrow$ Assume that $\mathcal{M}^N(\mathcal{D}) = \{u^N(\mu), \mu \in \mathcal{D}\}$ has a small (kolmogorov) dimension ...
we can select a set of parameters \((\mu^1, \cdots, \mu^N)\) in such way that \(\mathcal{M}^N(D)\) can be approximated by \(W_N^{N} = \text{span}\{u^N(\mu^n), 1 \leq n \leq N\}\).

Evaluation of the dimension of \(\mathcal{M}^N(D)\)?

Principal Analysis Component in appropriate norms:

\[
S^M_{k, \ell} = \langle u^N(\mu_k), u^N(\mu_\ell) \rangle_X, \; 1 \leq k, \ell \leq M, \; M : \text{number of snapshots}
\]
The R.B. method is based on the fact that for any $\varepsilon_N > 0$, there exist a set of parameters $(\mu_1, \cdots, \mu_N) \in \mathcal{D}^N$ such that:

$$\forall \mu \in \mathcal{D}, \exists (\alpha_i(\mu)) \in \mathbb{R}^N, \quad \|u(\mu) - \sum_{i=1}^{N} \alpha_i(\mu) u(\mu^i)\|_{H^1(\Omega)} \leq \varepsilon.$$ 

The R.B. method is a Galerkin approach within the space $W^N_N$.

<table>
<thead>
<tr>
<th>THE REDUCED BASIS METHOD</th>
<th>VS</th>
<th>A CLASSICAL DISCRETIZATION METHOD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Find $u^h_N(\mu)$ in $W^N_N$ s.t.:</td>
<td>vs</td>
<td>Find $u^N_N(\mu)$ in $X^N$ s.t.:</td>
</tr>
<tr>
<td>$a(u^h_N(\mu), v^h_N; \mu) = (f, v^h_N)$,</td>
<td></td>
<td>$a(u^N_N(\mu), v^h; \mu) = (f, v^h)$.</td>
</tr>
<tr>
<td>$\forall v^h_N \in W^N_N$</td>
<td></td>
<td>$\forall v^h \in X^N$</td>
</tr>
<tr>
<td>$\Rightarrow \mathcal{O}(N)$</td>
<td></td>
<td>$\Rightarrow \mathcal{O}(N')$</td>
</tr>
</tbody>
</table>

→The reduced basis is promising if $N$ is small! ($N \ll N$)
Requirements of the reduced basis method:

- How to select the good sampling set \((\mu^1, \cdots, \mu^N)\)?
  - Random
  - P.O.D
  - Greedy’s algorithm

Algorithm 1 Example of a Greedy’s algorithm

Given \(\Xi_{\text{train}} = (\mu_1, \cdots, \mu_{n_{\text{train}}}) \in \mathcal{D}^{n_{\text{train}}}, n_{\text{train}} >> 1\)

Choose randomly \(\mu_1\), \(S_1 = \{\mu_1\}\) and \(W_1^N = \{u^N(\mu_1)\}\)

for \(N = 2\) to \(N_{\text{max}}\) do

\(\mu_N = \arg\max_{\mu \in \Xi_{\text{train}}} \|u^N(\mu) - u_h^{N-1}(\mu)\|_X\)

\(S_N = S_{N-1} \cup \mu_N\) and \(W_N^N = W_{N-1}^N + \text{span}\{u^N(\mu_N)\}\)

end for

→ This version of the Greedy’s algorithm is quite expensive!
Requirements of the reduced basis method:

- How to select the good sampling set \((\mu^1, \cdots, \mu^N)\)?
  - Random
  - P.O.D
  - Greedy’s algorithm

**Algorithm 2 Example of a Greedy’s algorithm**

Given \(\Xi_{train} = (\mu_1, \cdots, \mu_{n_{train}}) \in \mathcal{D}^{n_{train}}, n_{train} \gg 1\)

Choose randomly \(\mu_1\), \(S_1 = \{\mu_1\}\) and \(W_{1}^{\mathcal{N}} = \{u^{\mathcal{N}}(\mu_1)\}\)

for \(N = 2\) to \(N_{\text{max}}\) do
  
  \(\mu_N = \arg \max_{\mu \in \Xi_{train}} \Delta_{N-1}(\mu)\)

  \(S_N = S_{N-1} \cup \mu_N\) and \(W_N^{\mathcal{N}} = W_{N-1}^{\mathcal{N}} + \text{span}\{u^{\mathcal{N}}(\mu_N)\}\)

end for

\(\Delta_N(\mu)\): sharp, inexpensive *a posteriori* error bound of \(\|u^N(\mu) - u^h_N(\mu)\|_X\)

→ Only the actual \(u^N(\mu_N)\) are computed by the Greedy’s algorithm.
Requirements of the reduced basis method:

- How to select the good set of $(\mu^1, \cdots, \mu^N)$? (OFFLINE)
  - Random
  - “P.O.D”
  - Greedy algorithm

- How to actually computes the reduced solution $u_N^h(\mu)$ for a given $\mu$?
  - Get the classical solution $(u_N^N(\mu^n))_{1 \leq n \leq N}$ (for example using a FEM code), from which the orthogonal basis function $(\xi_{1}^{RB}, \cdots, \xi_{N}^{RB})$ of $W_N$ will be computed. (offline)
  - For each new value of $\mu$:
    - build the matrix $[A^N(\mu)]_{k,\ell} = a(\xi_k^{RB}, \xi_\ell^{RB}; \mu)_{1 \leq k, \ell \leq N}$ and the vector $[F^N(\mu)]_{\ell} = f(\xi_\ell^{RB}; \mu)_{1 \leq \ell \leq N}$ (offline + online)
    - solve the system $A^N(\mu)\alpha^N,h(\mu) = F^N(\mu)$ and build output:
      $$s(u_N^N(\mu)) = \sum_{\ell=1}^{N} \alpha_{\ell}^{N,h}(\mu) s(\xi_{\ell}^{BR})$$ (Online)

- One of the keys of the R.B method is the decomposition of the computational work into an OFFLINE and an ONLINE stage.
Introduction to reduced basis methods

Certified reduced basis method

Requirements of the reduced basis method:

- How to select the good set of $\left( \mu^1, \cdots, \mu^N \right)$? \textbf{(OFFLINE)}
  - Random
  - “P.O.D”
  - Greedy algorithm

- How to actually compute the reduced solution $u_h^N(\mu)$ for a given $\mu$?
  - Get the classical solution $\left( u^N(\mu^n) \right)_{1 \leq n \leq N}$ (for example using a FEM code), from which the orthogonal basis function $\left( \xi_1^{RB}, \cdots, \xi_N^{RB} \right)$ of $W_N^N$ will be computed. \textbf{(OFFLINE)}

For each new value of $\mu$:
  - build the matrix $[A^N(\mu)]_{k,\ell} = a(\xi_k^{RB}, \xi_\ell^{RB}; \mu)_{1 \leq k, \ell \leq N}$ and the vector $[F^N(\mu)]_{\ell} = f(\xi_\ell^{RB}; \mu)_{1 \leq \ell \leq N}$ \textbf{(OFFLINE + ONLINE)}
  - solve the system $A^N(\mu) \alpha^{N,h}(\mu) = F^N(\mu)$ and build output:

$$s(u_h^N(\mu)) = \sum_{\ell=1}^{N} \alpha_\ell^{N,h}(\mu) s(\xi_\ell^{BR}) \textbf{ (ONLINE)}$$

- One of the keys of the R.B method is the decomposition of the computational work into an \textbf{OFFLINE} and an \textbf{ONLINE} stage.
Requirements of the reduced basis method:

- How to select the good set of $(\mu^1, \cdots, \mu^N)$? \textbf{(OFFLINE)}
  - Random
  - “P.O.D”
  - Greedy algorithm

- How to actually computes the reduced solution $u_h^N(\mu)$ for a given $\mu$?
  - Get the classical solution $(u^N(\mu^n))_{1 \leq n \leq N}$ (for example using a FEM code), from which the orthogonal basis function $(\xi_1^{RB}, \cdots, \xi_N^{RB})$ of $W_N^{\mathcal{N}}$ will be computed. \textbf{(OFFLINE)}

  For each new value of $\mu$:
  - build the matrix $[A^N(\mu)]_{k,\ell} = a(\xi_k^{RB}, \xi_\ell^{RB}; \mu)_{1 \leq k, \ell \leq N}$ and the vector $[F^N(\mu)]_{\ell} = f(\xi_\ell^{RB}; \mu)_{1 \leq \ell \leq N}$ \textbf{(OFFLINE + ONLINE)}
  - solve the system $A^N(\mu) \alpha^{N,h}(\mu) = F^N(\mu)$ and build output:
    \[ s(u_h^N(\mu)) = \sum_{\ell=1}^N \alpha_{\ell}^{N,h}(\mu) s(\xi_{\ell}^{BR}) \textbf{(ONLINE)} \]

  → One of the keys of the R.B method is the decomposition of the computational work into an \textbf{OFFLINE} and an \textbf{ONLINE} stage
Requirements of the reduced basis method:

- How to select the good set of \((\mu^1, \cdots, \mu^N)\)? *(OFFLINE)*
  
  → Random
  → “P.O.D”
  → Greedy algorithm

- How to actually computes the reduced solution \(u^h_N(\mu)\) for a given \(\mu\)?
  
  → Get the classical solution \((u^N(\mu^n))_{1 \leq n \leq N}\) (for example using a FEM code), from which the orthogonal basis function \((\xi^{RB}_1, \cdots, \xi^{RB}_N)\) of \(W_N^\infty\) will be computed. *(OFFLINE)*

  For each new value of \(\mu\):
  
  → build the matrix \([A^N(\mu)]_{k, \ell} = a(\xi^{RB}_k, \xi^{RB}_\ell; \mu)_{1 \leq k, \ell \leq N}\) and the vector \([F^N(\mu)]_{\ell} = f(\xi^{RB}_\ell; \mu)_{1 \leq \ell \leq N}\) *(OFFLINE + ONLINE)*
  
  → solve the system \(A^N(\mu) \alpha^{N,h}(\mu) = F^N(\mu)\) and build output:

\[
s(u^N_h(\mu)) = \sum_{\ell=1}^{N} \alpha^{N,h}_\ell(\mu) s(\xi^{BR}_\ell) \quad (ONLINE)
\]

→ One of the keys of the R.B method is the decomposition of the computational work into an OFFLINE and an ONLINE stage.
Requirements of the reduced basis method:

How \( A^N(\mu) \) is generated?

**Direct affine’s decomposition**

\[
a(\xi_i^{RB}, \xi_j^{RB}; \mu) = \sum_{k=1}^{K} \theta_k(\mu) a_k(\xi_i^{RB}, \xi_j^{RB})
\]

**Empirical interpolation method**

\[
a(\xi_i^{RB}, \xi_j^{RB}; \mu) = \sum_{k=1}^{K} \Phi_k(\mu) a(\xi_i^{RB}, \xi_j^{RB}; q_k)
\]

**Offline:** \( a_k(\xi_i^{RB}, \xi_j^{RB}) \) (or \( a(\xi_i^{RB}, \xi_j^{RB}; q_k) \)) are precomputed

**Online:**
- \( A^N(\mu) \) generation’s requires only \( K \times N^2 \) operations instead of \( N^2 \).
- \( A^N(\mu) \) inversion’s is done in \( N^3 \) operations instead of \( N^3 \). (direct inversion)

Generation of the output \( s(u_h^N(\mu)) \Rightarrow s(\xi_i^{BR}) \) also precomputed **Offline**.

→ All expensive computations are done in the **Offline** stage

⇒ Then the **Online** stage computations are in scale with \( N \)
Requirements of the reduced basis method:

**How \( A^N(\mu) \) is generated?**

**Direct affine’s decomposition**

\[
a(\xi_i^{RB}, \xi_j^{RB}; \mu) = \sum_{k=1}^{K} \theta_k(\mu) \ a_k(\xi_i^{RB}, \xi_j^{RB})
\]

**Empirical interpolation method**

\[
a(\xi_i^{RB}, \xi_j^{RB}; \mu) = \sum_{k=1}^{K} \Phi_k(\mu) \ a(\xi_i^{RB}, \xi_j^{RB}; q_k)
\]

**OFFLINE**: \( a_k(\xi_i^{RB}, \xi_j^{RB}) \) (or \( a(\xi_i^{RB}, \xi_j^{RB}; q_k) \)) are precomputed

**ONLINE**: • \( A^N(\mu) \) generation’s requires only \( K \times N^2 \) operations instead of \( N^2 \).
  • \( A^N(\mu) \) inversion’s is done in \( N^3 \) operations instead of \( N^3 \). (direct inversion)

Generation of the output \( s(u_h^N(\mu)) \) ⇒ \( s(\xi_i^{BR}) \) also precomputed **OFFLINE**.

→ All expensive computations are done in the **OFFLINE** stage

⇒ Then the **ONLINE** stage computations are in scale with \( N \)
Requirements of the reduced basis method:

How \( A^N(\mu) \) is generated?

\[
\begin{align*}
\text{Direct affine’s decomposition} & & \text{Empirical interpolation method} \\
\sum_{k=1}^{\mathcal{K}} \theta_k(\mu) a_k(\xi_i^{RB}, \xi_j^{RB}) & = \sum_{k=1}^{\mathcal{K}} \Phi_k(\mu) a(\xi_i^{RB}, \xi_j^{RB}; q_k)
\end{align*}
\]

**OFFLINE:** \( a_k(\xi_i^{RB}, \xi_j^{RB}) \) (or \( a(\xi_i^{RB}, \xi_j^{RB}; q_k) \)) are precomputed

**ONLINE:**
- \( A^N(\mu) \) generation’s requires only \( \mathcal{K} \times N^2 \) operations instead of \( N^2 \).
- \( A^N(\mu) \) inversion’s is done in \( N^3 \) operations instead of \( N^3 \). (direct inversion)

Generation of the output \( s(u_h^N(\mu)) \rightarrow s(\xi_i^{BR}) \) also precomputed **OFFLINE**.

→ All expensive computations are done in the **OFFLINE** stage

⇒ Then the **ONLINE** stage computations are in scale with \( N \)
Requirements of the reduced basis method:

How $A^N(\mu)$ is generated?

Direct affine’s decomposition

$$a(\xi_i^{RB}, \xi_j^{RB}; \mu) = \sum_{k=1}^{\mathcal{K}} \theta_k(\mu) a_k(\xi_i^{RB}, \xi_j^{RB})$$

Empirical interpolation method

$$a(\xi_i^{RB}, \xi_j^{RB}; \mu) = \sum_{k=1}^{\mathcal{K}} \Phi_k(\mu) a(\xi_i^{RB}, \xi_j^{RB}; q_k)$$

**OFFLINE:** $a_k(\xi_i^{RB}, \xi_j^{RB})$ (or $a(\xi_i^{RB}, \xi_j^{RB}; q_k)$) are precomputed

**ONLINE:**
- $A^N(\mu)$ generation’s requires only $\mathcal{K} \times N^2$ operations instead of $N^2$.
- $A^N(\mu)$ inversion’s is done in $N^3$ operations instead of $N^3$. (direct inversion)

What happens when the FEM simulation code is used as black box?

→ It’s not possible to use this code to perform all the OFFLINE computations required for an efficient performance of the R.B method

(since we want the online computation to be done with a $N$ complexity and not with a complexity of the finite element method)

→ An alternative: a non intrusive reduced basis method
A non intrusive reduced basis method : How ?

Let \( \tilde{u}_h^N(\mu) \) be the \( L^2 \)-projection of \( u^N(\mu) \) in \( W_N^N \) defined by

\[
\tilde{u}_h^N(\mu) = \sum_{i=1}^{N} \beta_i^{N,h}(\mu) \xi_i^{RB}
\]

with

\[
\beta_i^{N,h}(\mu) = \int_{\Omega} u^N(\mu) \xi_i^{RB}
\]

\( \rightarrow \) The standard R.B. method aims at evaluating the coefficients \( \alpha_i^{N,h}(\mu) \) those can appear as a substitute to the optimal coefficients \( \beta_i^h(\mu) \).

Since, the computation of \( u^{N_H}(\mu) \), for \( H >> h \) and \( X_{N_H} \subset X_N \), is less expensive than the one of \( u^N(\mu) \).

\( \rightarrow \) Our alternative method [1,2] consists in proposing an another surrogate to \( \beta_i^{N,h}(\mu) \) defined by

\[
\beta_i^{N,H}(\mu) = \int_{\Omega} u^{N_H}(\mu) \xi_i^{RB}
\]


A non intrusive reduced basis method: How?

Let $\tilde{u}_h^N(\mu)$ be the $L^2$-projection of $u^N(\mu)$ in $W_h^N$ defined by

$$
\tilde{u}_h^N(\mu) = \sum_{i=1}^{N} \beta_i^{N,h}(\mu) \xi_i^{RB}
$$

with

$$
\beta_i^{N,h}(\mu) = \int_{\Omega} u^N(\mu) \xi_i^{RB}
$$

The standard R.B. method aims at evaluating the coefficients $\alpha_i^{N,h}(\mu)$ those can appear as a substitute to the optimal coefficients $\beta_i^h(\mu)$.

Since, the computation of $u^{N,H}(\mu)$, for $H >> h$ and $X_{N,h} \subset X_N$, is less expensive than the one of $u^N(\mu)$.

Our alternative method [1,2] consists in proposing an another surrogate to $\beta_i^{N,h}(\mu)$ defined by

$$
\beta_i^{N,H}(\mu) = \int_{\Omega} u^{N,H}(\mu) \xi_i^{RB}
$$

We can build a reduced solution \( u_{H,h}^N(\mu) \) and the output \( s(u_{H,h}^N(\mu)) \):

\[
 u_{H,h}^N(\mu) = \sum_{i=1}^{N} \beta_i^{N,H}(\mu) \xi_i^{RB} \\
 s(u_{H,h}^N(\mu)) = \sum_{i=1}^{N} \beta_i^{N,H}(\mu) s(\xi_i^{RB})
\]

This method is based on the fact that the error measured in the \( L^2 \)-norm converge faster than the one measured \( H^1 \)-norm.

Why this can still be a good approximation?

\( \rightarrow \) The basis functions \( \xi_i^{RB} \) have to be orthonormal in \( H^1 \) and \( L^2 \) norm.

\( X^N \) and \( X_i^N \): \( \mathbb{P}^k \)- F.E discretization space \( \rightarrow \) \( \| u(\mu) - u_N(\mu) \|_X \leq c(\mu) h^k \)

\( \rightarrow \) Using the orthogonality of \( \xi_i^{BR} \), we easily can prove that:

\[
 \| u(\mu) - u_{H,h}^N(\mu) \|_X \leq \varepsilon + C(\mu) (h^k + H^{2k})
\]

which is asymptotically similar to \( \| u(\mu) - u_h^N(\mu) \|_X \leq \varepsilon + C(\mu) h^k \) when we choose \( h \sim H^2 \).
We can build a reduced solution $u_{H,h}^N(\mu)$ and the output $s(u_{H,h}^N(\mu))$:

$$u_{H,h}^N(\mu) = \sum_{i=1}^{N} \beta_i^{N,H}(\mu) \xi_i^{RB} \quad \text{and} \quad s(u_{H,h}^N(\mu)) = \sum_{i=1}^{N} \beta_i^{N,H}(\mu) s(\xi_i^{RB})$$

→ This method is based on the fact that the error measured in the $L^2$-norm converge faster than the one measured $H^1$-norm.

Why this can still be a good approximation?

→ The basis functions $\xi_i^{RB}$ have to be orthonormal in $H^1$ and $L^2$ norm.

$X^N$ and $X_H^N$: $\mathbb{P}^k$- F.E discretization space \(\rightarrow\) $||u(\mu) - u^N(\mu)||_X \leq c(\mu) h^k$

→ Using the orthogonality of $\xi_i^{BR}$, we easily can prove that:

$$||u(\mu) - u_{H,h}^N(\mu)||_X \leq \varepsilon + C(\mu) (h^k + H^{2k})$$

which is asymptotically similar to $||u(\mu) - u_h^N(\mu)||_X \leq \varepsilon + C(\mu) h^k$ when we choose $h \sim H^2$. 
Post-process to improve the computation of the $\beta_{i}^{N,H}(\mu)$

We computes the matrix $T_{N} \in \mathbb{R}^{N \times N}$ solution of the following system:

$$
T_{N} \times \begin{pmatrix}
\beta_{1}^{N,H}(\mu_1) & \cdots & \beta_{1}^{N,H}(\mu_N) \\
\vdots & \ddots & \vdots \\
\beta_{N}^{N,H}(\mu_1) & \cdots & \beta_{N}^{N,H}(\mu_N)
\end{pmatrix} = \begin{pmatrix}
\beta_{1}^{N,h}(\mu_1) & \cdots & \beta_{1}^{N,h}(\mu_N) \\
\vdots & \ddots & \vdots \\
\beta_{N}^{N,h}(\mu_1) & \cdots & \beta_{N}^{N,h}(\mu_N)
\end{pmatrix}
$$

$\Rightarrow$ We replace $u_{H,h}^{N}(\mu)$ and $s(u_{H,h}^{N}(\mu))$:

\[
\tilde{u}_{H,h}^{N}(\mu) = \sum_{i=1}^{N} T_{ij}^{N} \beta_{i}^{N,H}(\mu) \xi^{RB}_{i}
\]

and

\[
s(\tilde{u}_{H,h}^{N}(\mu)) = \sum_{i,j=1}^{N} T_{ij}^{N} \beta_{i}^{N,H}(\mu) s(\xi^{RB}_{i})
\]
What do we need?

- F.E. code used as **black box**
- **F.E. library** (Freefem++)

Compute snapshots $u_h(\mu_i)$
coarse solution $u_H(\mu)$

Return fine mesh $\mathcal{T}_h$
coarse mesh $\mathcal{T}_H$

To compute $L^2$ and $H^1$
scalar product

Interpolate from $\mathcal{T}_H$
to $\mathcal{T}_h$
Main characteristic of Freefem++\(^1\)

→ **Wide range of finite elements**: continuous P1, P2 elements, discontinuous P0, P1, RT0, RT1, BDM1 elements, vectorial elements, ...

→ **Automatic interpolation** of data from a mesh to an other one (with matrix construction if need), so a finite element function is view as a function of \((x; y; z)\) or as an array.

→ **Link with other soft**: paraview, gmsh, vtk, medit, gnuplot, ...

→ **Dynamic linking to add plugin**.

→ **Full MPI interface**

\(^1\) [http://www.freefem.org/ff++]
IMPLEMENTATION

OFFLINE stage

1. Construction of a reduced approximation’s space.
   ▶ computation of a sample of solutions (black box software)
   ▶ selection of N solutions to build the reduced basis (F.E. Library).

2. Orthonormalisation in $L^2$ and $H^1$-norm of the reduced basis functions (F.E. Library).

3. Preparation for the post-processing.
   ▶ computation of the N coarse solutions $u_{\mathcal{N}}^{H}(\mu_i)$ (black box software)
   ▶ construction of matrix $T^N$ (F.E. Library).

ONLINE stage

1. Computation of the coarse solution $u_{\mathcal{N}}^{H}(\mu)$ (black box software)

2. Compute the coefficient $\beta_i^{N,H}(\mu)$. (F.E. Library)

3. Apply the post-processing on the $\beta_i^{N,H}(\mu)$. (F.E. Library)

4. Build the output $s(u_{\mathcal{N}}^{H,h}(\mu))$. (F.E. Library)
An application of NIRB method to heat transfer problem

Model: Incompressible steady Navier Stokes + Heat equation (Boussinesq’s approximation)

Cooling air velocity
\( V_{in} \in [0.5; 2], \)

Cooling air temperature
\( \theta_{in} \in [288; 292], \)

Rack’s air velocity
\( V_{rack} \in [0.1; 0.4], \)

Rack’s air temperature
\( \theta_{rack} \in [295; 315]. \)
An application of NIRB method to heat transfer problem

Sampling to extract the reduced basis

<table>
<thead>
<tr>
<th>$\theta_{in}$</th>
<th>$V_{in}$</th>
<th>$V_{rack}$</th>
<th>$\theta_{rack}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>288</td>
<td>0.5</td>
<td>0.1</td>
<td>295</td>
</tr>
<tr>
<td>292</td>
<td>1</td>
<td>0.2</td>
<td>300</td>
</tr>
<tr>
<td>2</td>
<td>0.3</td>
<td>0.4</td>
<td>310</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>315</td>
</tr>
</tbody>
</table>

Computations of 120 snapshots using a $\mathbb{P}_2 - \mathbb{P}_1$ F.E steady Navier-Stokes solver within Freefem++ on a reference mesh.

**Figure:** Values of the $N$ largest eigenvalues of the matrix $S^{M}$
An application of NIRB method to heat transfer problem

Temperature

Velocity

Relative error in $H^1$-norm (log)

$N$ (log)

Reference mesh
Ndof = 20213
Case 1

Coarse embedded mesh
Ndof = 5148
Case 2

Coarse non embedded mesh
Ndof = 5088
Case 3

Very coarse non embedded mesh
Ndof = 1871
Case 4

Rachida Chakir

Journées Freefem++
An application of NIRB method to heat transfer problem

Relative error plot between the reference F.E. and the NIRB solutions
(cases $4 + p.p.$ with $N = 15$)

![Velocity magnitude](image1)

![Temperature](image2)

Mean value of the online's stage with post-processing execution's time - $N = 15$

<table>
<thead>
<tr>
<th>Reference FEM</th>
<th>NIRB - case 2</th>
<th>NIRB - case 3</th>
<th>NIRB - case 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 sec</td>
<td>52 sec</td>
<td>52 sec</td>
<td>17 sec</td>
</tr>
<tr>
<td></td>
<td>53 sec</td>
<td>53 sec</td>
<td>18 sec</td>
</tr>
<tr>
<td></td>
<td>54 sec</td>
<td>54 sec</td>
<td>19 sec</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Temperature</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Velocity</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Both</td>
</tr>
</tbody>
</table>

Rachida Chakir

Journées Freefem++
Conclusion

We note that the post-processing improved even more the approximation since it allows to recover the truth error even starting from the computations of the coarsest NIRB solution.

Perspectives

Apply to more complex application:

– time dependent problem
– take geometry as a parameter