Model Order Reduction in support of the Virtual Element Method

Fabio Credali
Joint work with Silvia Bertoluzza

Pavia - March 17$^{\text{th}}$, 2022

UNIVERSITÀ DI PAVIA
UNIPV-USI
International PhD Program in Computational Mathematics and Decision Sciences
Università della Svizzera italiana
جامعة الملك عبدالله للعلوم والتقنية
King Abdullah University of Science and Technology

Outline

- The Virtual Element Method
- Parametric PDEs and Reduced Basis Method
- RB in support of VEM
- Examples

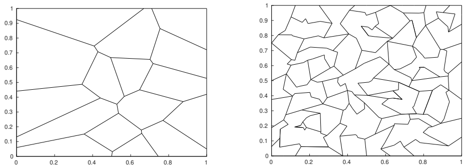
Model problem

Poisson equation

Find $u \in V = H^1_0(\Omega)$ such that

$$a(u, v) = \int_\Omega \nabla u \cdot \nabla v \, dx = \int_\Omega fv \, dx \quad \forall v \in V \tag{1}$$

- $\Omega \subset R^2$ is a polygonal domain
- Ω is decomposed with a mesh $\mathcal{T}_h$ made up of polygonal elements K



VEM space, degree 1

- $r = 1$, polynomial degree (accuracy)
- As in Finite Elements, we define the space in each element

Local VEM space

$$V_1^K = \{v \in H^1(K) : v_{|e} \in P_1(e)\ \forall e \in \partial K \text{ and } \Delta v_{|K} = 0\}$$

VEM space, degree 1

- $r = 1$, polynomial degree (accuracy)
- As in Finite Elements, we define the space in each element

Local VEM space

$$V_1^K = \{v \in H^1(K) : v_{|e} \in P_1(e)\ \forall e \in \partial K \text{ and } \Delta v_{|K} = 0\}$$

- **Remark:** $P_1(K) \subset V_1^K$
- DOFs: the values at the vertices of K

VEM space, degree 1

- $r = 1$, polynomial degree (accuracy)
- As in Finite Elements, we define the space in each element

Local VEM space

$$V_1^K = \{v \in H^1(K) : v_{|e} \in P_1(e) \; \forall e \in \partial K \text{ and } \Delta v_{|K} = 0\}$$

- **Remark:** $P_1(K) \subset V_1^K$
- DOFs: the values at the vertices of K

We *glue* them by continuity

Global VEM space

$$V_h = \{v \in H_0^1(\Omega) : v_{|K} \in V_1^K \; \forall K \in \mathcal{T}_h\}$$

Discretization

Galerkin method

Find $u_h \in V_h$ such that

$$a(u_h, v_h) = \int_\Omega f\, v_h \quad \forall v_h \in V_h$$

In particular,

$$a(u_h, v_h) = \int_\Omega \nabla u_h \cdot \nabla v_h \, dx = \sum_{K \in \mathcal{T}_h} \int_K \nabla u_h \cdot \nabla v_h \, dx = \sum_{K \in \mathcal{T}_h} a^K(u_h, v_h)$$

Discretization

Galerkin method

Find $u_h \in V_h$ such that

$$a(u_h, v_h) = \int_\Omega f\, v_h \quad \forall v_h \in V_h$$

In particular,

$$a(u_h, v_h) = \int_\Omega \nabla u_h \cdot \nabla v_h \, dx = \sum_{K \in \mathcal{T}_h} \int_K \nabla u_h \cdot \nabla v_h \, dx = \sum_{K \in \mathcal{T}_h} a^K(u_h, v_h)$$

Warning!

We cannot assemble a^K on each element directly using the basis functions of V_1^K (like in FEM) because they are themselves solutions of PDEs in K

A computable bilinear form

- Decomposition of $V_1^K = P_1(K) \oplus V_\perp^K$
- $V_\perp^K \subset V_1^K$ H^1-orthogonal to $P_1(K)$
- $u_h = p + p^\perp$ and $v_h = q + q^\perp$

A computable bilinear form

- Decomposition of $V_1^K = P_1(K) \oplus V_\perp^K$
- $V_\perp^K \subset V_1^K$ H^1-orthogonal to $P_1(K)$
- $u_h = p + p^\perp$ and $v_h = q + q^\perp$
- p, q computable, $p^\perp, q^\perp$ **not** computable

$$a^K(u_h, v_h) = a^K(p, q) + \cancel{a^K(p, q^\perp)} + \cancel{a^K(p^\perp, q)} + a^K(p^\perp, q^\perp)$$

A computable bilinear form

- Decomposition of $V_1^K = P_1(K) \oplus V_\perp^K$
- $V_\perp^K \subset V_1^K$ H^1-orthogonal to $P_1(K)$
- $u_h = p + p^\perp$ and $v_h = q + q^\perp$
- p, q computable, $p^\perp, q^\perp$ **not** computable

$$a^K(u_h, v_h) = a^K(p, q) + \cancel{a^K(p, q^\perp)} + \cancel{a^K(p^\perp, q)} + a^K(p^\perp, q^\perp)$$

Idea: stabilization term $S^K(p^\perp, q^\perp)$ symmetric and bilinear s. t.

$$c_\star a^K(p^\perp, p^\perp) \leq S^K(p^\perp, p^\perp) \leq C^\star a^K(p^\perp, p^\perp)$$

Scaled diagonal: $S^K(p^\perp, q^\perp) = \sum_i dof_i(p^\perp) dof_i(q^\perp) |\Pi_k^\nabla e_i|_{1,K}$

A look at the Reduced Basis method

Parametric PDEs and Reduced Basis Method

Model problem

Find $u(\mu) \in V$ such that

$$a(u(\mu), v; \mu) = f(v; \mu) \quad \forall v \in V \tag{2}$$

where μ is a parameter.

- **Galerkin.** We can compute an approximation

$$u_{\mathcal{N}} \in V_{\mathcal{N}} = span\{\varphi_1, \dots, \varphi_{\mathcal{N}}\} \subset V$$

- if we need to solve this for many values of μ, this will be extremely expensive

Reduced Basis solution

- Solution manifold $\mathcal{M} = \{u(\mu) : \mu\} \subset V$
- **Idea:** $\mathcal{M}$ can be hopefully approximated by a lower dim. space

Reduced Basis solution

- Solution manifold $\mathcal{M} = \{u(\mu) : \mu\} \subset V$
- **Idea:** $\mathcal{M}$ can be hopefully approximated by a lower dim. space
- We want to introduce a new space $W_M \subset V_\mathcal{N}$

$$W_M = span\{\xi_1, \ldots, \xi_M\} \quad \text{with } M \ll \mathcal{N}$$

- $u_M(\mu) = \sum_{i=1}^{M} u_i \xi_i$

Reduced Basis solution

- Solution manifold $\mathcal{M} = \{u(\mu) : \mu\} \subset V$
- **Idea:** $\mathcal{M}$ can be hopefully approximated by a lower dim. space
- We want to introduce a new space $W_M \subset V_\mathcal{N}$

$$W_M = span\{\xi_1, \ldots, \xi_M\} \quad \text{with } M \ll \mathcal{N}$$

- $u_M(\mu) = \sum_{i=1}^{M} u_i \xi_i$

$$a(u_M, \xi_j; \mu) = \sum_{i=1}^{M} a(\xi_i, \xi_j; \mu) u_i(\mu) = f(\xi_j; \mu) \text{ for } i \leq j \leq M \quad (3)$$

Reduced Basis solution

- Solution manifold $\mathcal{M} = \{u(\mu) : \mu\} \subset V$
- **Idea:** $\mathcal{M}$ can be hopefully approximated by a lower dim. space
- We want to introduce a new space $W_M \subset V_{\mathcal{N}}$

$$W_M = span\{\xi_1, \ldots, \xi_M\} \quad \text{with } M \ll \mathcal{N}$$

- $u_M(\mu) = \sum_{i=1}^{M} u_i \xi_i$

$$a(u_M, \xi_j; \mu) = \sum_{i=1}^{M} a(\xi_i, \xi_j; \mu) u_i(\mu) = f(\xi_j; \mu) \text{ for } i \leq j \leq M \quad (3)$$

How do we construct W_M?

It is generated by combinations of snapshots $u(\mu_i)$ for a small set of parameters $\mu_1, \ldots, \mu_{N_s}$.

Computation of $a(\xi_i, \xi_j; \mu)$

We assume that

- $a(u, v; \mu) = \sum_{q=1}^{Q_a} \theta_q^a(\mu) a^q(u, v)$
- $f(v; \mu) = \sum_{q=1}^{Q_f} \theta_q^f(\mu) f^q(v)$

Hence, (3) becomes

$$\sum_{i=1}^{M} \left[\theta_q^a(\mu) a^q(\xi_i, \xi_j)\right] u_i(\mu) = \sum_{q=1}^{Q_f} \theta_q^f(\mu) f^q(\xi_j) \qquad (4)$$

Offline-online strategy

Offline - Sample

Build sample $S = \{\mu_1, \ldots, \mu_{N_s}\}$

Comment

S set of the parameters to build the reduced basis.

- Random
- Equidistributed/log equidistributed
- From error estimator

Offline-online strategy

Offline - Sample

Build sample $S = \{\mu_1, \ldots, \mu_{N_s}\}$

Offline - Build basis

$\forall \mu_i \in S$

- $A^{\mathcal{N}}_{\ell,k} = a(\varphi_\ell, \varphi_k; \mu_i)$
- $F^{\mathcal{N}}_k = f(\varphi_k; \mu_i)$
- Solve $A^{\mathcal{N}} u^{\mathcal{N}}(\mu_i) = F^{\mathcal{N}}$
- Choose the r.b. functions (POD)

Comment

Compute the snapshots

- Compute $u^{\mathcal{N}}(\mu_i) \quad \forall i$
- Cost depends on $\mathcal{N}$
- ξ_i stored

Only ONCE!

Offline-online strategy

Offline - Sample

Build sample $S = \{\mu_1, \ldots, \mu_{N_s}\}$

Offline - Build basis

$\forall \mu_i \in S$

- $A^{\mathcal{N}}_{\ell,k} = a(\varphi_\ell, \varphi_k; \mu_i)$
- $F^{\mathcal{N}}_k = f(\varphi_k; \mu_i)$
- Solve $A^{\mathcal{N}} u^{\mathcal{N}}(\mu_i) = F^{\mathcal{N}}$
- Choose the r.b. functions (POD)

Offline - Precomputations

- $A^{M,q}_{i,j} = a^q(\xi_i, \xi_j)$
- $F^{M,q}_j = f^q(\xi_j)$

Comment

ξ_i, ξ_j known functions in $V_{\mathcal{N}}$
Building block for affine decomposition $a(\cdot, \cdot; \mu)$ and $f(\cdot, \mu)$

- $A^{M,q}_{i,j} = a^q(\xi_i, \xi_j)$
- $F^{M,q}_j = f^q(\xi_j)$

Precomputed and stored ONCE!

Offline-online strategy

Offline - Sample

Build sample $S = \{\mu_1, \dots, \mu_{N_s}\}$

Offline - Build basis

$\forall \mu_i \in S$

- $A^{\mathcal{N}}_{\ell,k} = a(\varphi_\ell, \varphi_k; \mu_i)$
- $F^{\mathcal{N}}_k = f(\varphi_k; \mu_i)$
- Solve $A^{\mathcal{N}} u^{\mathcal{N}}(\mu_i) = F^{\mathcal{N}}$
- Choose the r.b. functions (POD)

Offline - Precomputations

- $A^{M,q}_{i,j} = a^q(\xi_i, \xi_j)$
- $F^{M,q}_j = f^q(\xi_j)$

Online:

For each new parameter μ:

- $A^M_{i,j} = \sum_{q=1}^{Q_a} \theta_a^q(\mu) A^q_{i,j}$
- $F^M_j = \sum_{q=1}^{Q_f} \theta_f^q(\mu) F^q_j$
- Solve $A^M x = F^M$
- $u_M(\mu) = \sum_{i=1}^{M} x_i \xi_i$

Proper Orthogonal Decomposition

1. Build the correlation matrix of all the snapshots of the sample

$$C = \frac{1}{N_s} U^T U \text{ where } U = \left[u^{\mathcal{N}}(\mu_1) \;\middle|\; \dots \;\middle|\; u^{\mathcal{N}}(\mu_{N_s}) \right]$$

2. Solve the eigenvalues problem $Cz_n = \lambda_n z_n$
3. Choose the eigenvectors corresponding to the first M greatest eigenvalues
4. The POD basis can be computed as

$$\xi_i(x) = \frac{1}{\sqrt{N_s}} \sum_{m=1}^{N_s} (z_n)_m u^{\mathcal{N}}(\mu_m)(x) \quad i = 1, \dots, M$$

Back to VEM

Order 1 VEM

- Consider a polygon K
- $\mathrm{v}_1, \ldots, \mathrm{v}_N$ vertices of K

Order 1 VEM

- Consider a polygon K
- $\mathrm{v}_1, \ldots, \mathrm{v}_N$ vertices of K
- **Goal:** we want to compute a possibly very rough approximation of the basis functions for V_1^K
- **Why?** Post-processing of solutions and (maybe) stabilization term

Order 1 VEM

- Consider a polygon K
- $\mathrm{v}_1, \ldots, \mathrm{v}_N$ vertices of K
- **Goal:** we want to compute a possibly very rough approximation of the basis functions for V_1^K
- **Why?** Post-processing of solutions and (maybe) stabilization term

Basis of V_1^K

For $n = 1, \ldots, N$

$$\begin{aligned} -\Delta e^n &= 0 \quad \text{in } K \\ e^n &\text{ p.w lin. on } \partial K \\ e^n(\mathrm{v}_m) &= \delta_{n,m} \end{aligned} \tag{5}$$

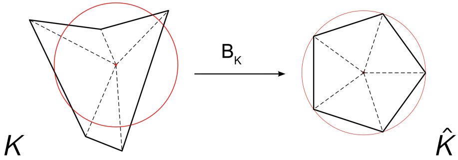
Geometric parametrization

- $\hat{K}$ reference (regular) polygon
- $\mathcal{B}_K : K \to \hat{K}$ piecewise affine transformation such that

$$\mathcal{B}_K(v_n) = \hat{v}_n \text{ and } \mathcal{B}_K(x_K) = \hat{x}_K$$

for $x_K \in K$ and $\hat{x}_K \in \hat{K}$.

- In particular, B_K is a piecewise constant matrix



Affine decomposition

- We can partition K and $\hat{K}$ in as many triangles as there are edges
$$\hat{K} = \cup_{n=1}^{N} \hat{T}_n \text{ and } K = \cup_{n=1}^{N} T_n$$
where $\hat{T}_n = \mathcal{B}_K T_n$
- $\mathcal{B}_K$ is affine on T_n, we have

$$a(u, v; K) = \sum_{n=1}^{N} \int_{\hat{T}_n} B_K B_K^T \nabla \hat{u} \cdot \nabla \hat{v}$$

Affine decomposition

- We can partition K and $\hat{K}$ in as many triangles as there are edges
$$\hat{K} = \cup_{n=1}^{N} \hat{T}_n \text{ and } K = \cup_{n=1}^{N} T_n$$
where $\hat{T}_n = \mathcal{B}_K T_n$
- $\mathcal{B}_K$ is affine on T_n, we have
$$a(u, v; K) = \sum_{n=1}^{N} \int_{\hat{T}_n} B_K B_K^T \nabla \hat{u} \cdot \nabla \hat{v}$$
- $B_K B_{K\,|T_n}^T = \sum_{\nu=1}^{3} a_\nu^n A^\nu = a_1^n \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + a_2^n \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} + a_3^n \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
- Hence, $q = (n, \nu) \Rightarrow a_\nu^q(u, v) := \int_{\hat{T}_n} A^\nu \nabla u \cdot \nabla v$

Offline phase

- N is fixed
- Generate a set of trial polygons K^ℓ
- Compute the affine mapping $\mathcal{B}_\ell : K^\ell \to \hat{K}$
- Compute $e_\ell^1, \ldots, e_\ell^N$ by solving their equations in K^ℓ (FEM)
- Map on $\hat{K}$ the VEM basis just computed $\rightsquigarrow \hat{e}_\ell^1, \ldots, \hat{e}_\ell^N$
- Compute and store $a_\nu^q(\hat{e}_\ell^n, \hat{e}_\ell^{n'})$

Online phase

After building the reduced basis $\{\hat{\xi}_\ell^n \in H^1(\hat{K}) : \ell = 1, \dots, M\}$ using the POD

- Generate a set of test polygons
- $\forall K$, $\mathcal{B} : K \to \hat{K}$ s.t. $BB^T_{|T_i} = \sum_{\nu=1}^3 a_i^\nu A^\nu$
- We look for $\hat{e}^n = \sum_{\ell=1}^M x_\ell^n \hat{\xi}_\ell^n$ s.t.

$$-\nabla \cdot (BB^T)\nabla \hat{e}^n = 0 \text{ in } \hat{K}$$
$$\hat{e}^n \text{ p.w lin. on } \partial\hat{K}$$
$$\hat{e}^n(\hat{\mathrm{v}}_m) = \delta_{n,m}$$

- $\hat{e}^n$ are the VEM basis for K mapped on $\hat{K}$, hence we go back to obtain e^n for $n = 1, \dots, N$.

Different uses of our basis

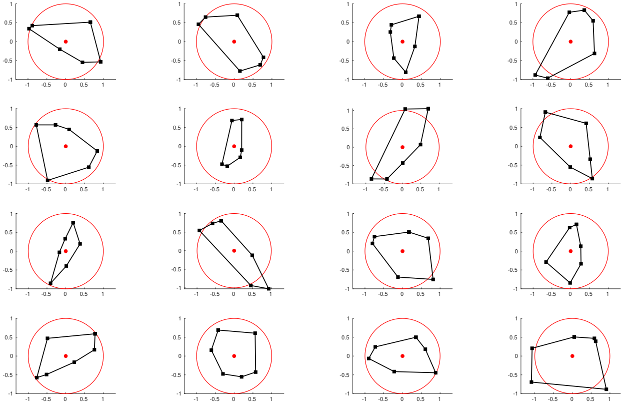
- Build the stabilization matrix
- Post-processing of VEM solutions and reconstruction in subdomains
- Evaluate the error with respect to the true solution (academic purpose)

Reduced Basis generation for VEM stabilization

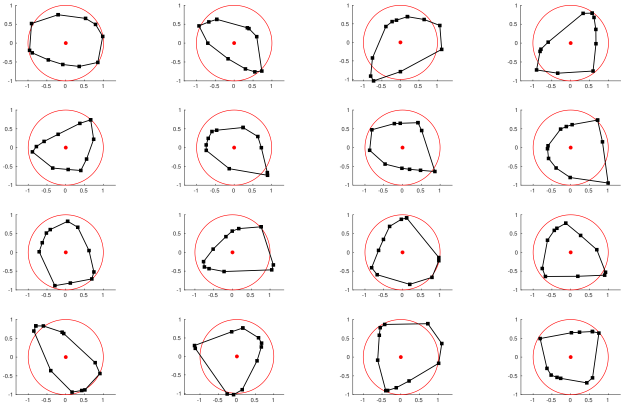
- In order to perform a numerical test on a VEM solution, we generated Reduced Basis (with several choices of M) on sets of convex random polygons with $N = 4, 5, \ldots, 14$
- For each N, we generated 300 trial polygons and 500 test polygons
- We studied the ratio $C^\star / c_\star$, where

$$c_\star a^K(p^\perp, p^\perp) \leq S_{RB}^K(p^\perp, p^\perp) \leq C^\star a^K(p^\perp, p^\perp)$$

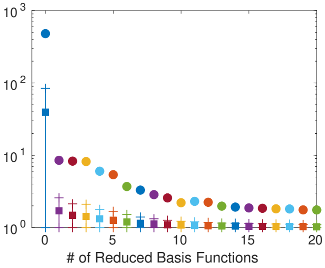
$N = 6$ - some polygons



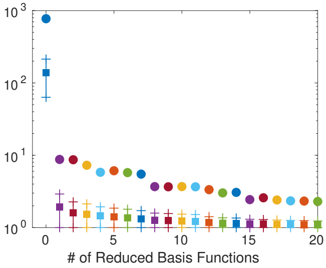
N = 11 - some polygons



Ratio $C^\star / c_\star$



(a) N=6



(b) N=11

RB for post-processing

- We want to solve Poisson for $u(x, y) = \frac{\sin(4x\pi)\sin(4y\pi)}{2(4\pi)^2}$ in $\Omega = [0, 1]^2$
- We work on a sequence of Voronoi meshes with $16, 64, 100, 256, 1024, 4096$ elements

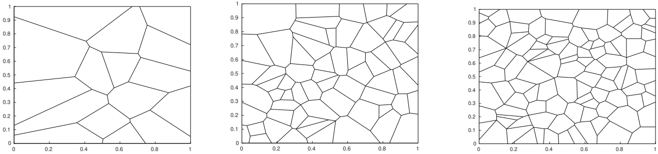


Figure: The first three meshes of the sequence

RB for post-processing

- We want to solve Poisson for $u(x, y) = \frac{\sin(4x\pi)\sin(4y\pi)}{2(4\pi)^2}$ in $\Omega = [0, 1]^2$
- We work on a sequence of Voronoi meshes with $16, 64, 100, 256, 1024, 4096$ elements

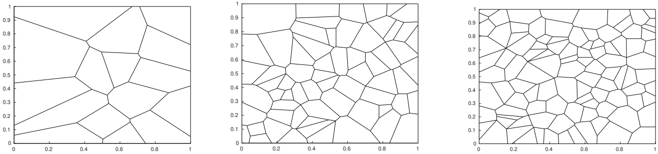
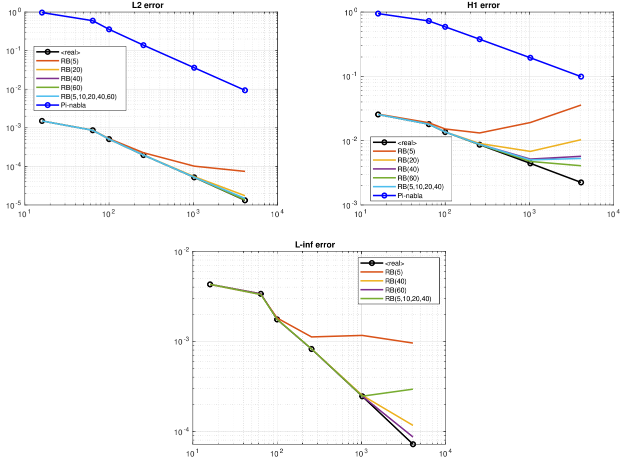


Figure: The first three meshes of the sequence

- We study the L^2, H^1 and L^∞ error with different approaches

Convergence plots



Fixed vs adapted number of basis for polygon.
Errors in dependence of the number of polygons of the mesh

Limits and perspectives

- The adaptive choice of the number of RBs in dependence on the geometry of the polygons has been done in a rough way computing the ratio between the radius of the inscribe circle with the radius of the circumscribed one

Limits and perspectives

- The adaptive choice of the number of RBs in dependence on the geometry of the polygons has been done in a rough way computing the ratio between the radius of the inscribe circle with the radius of the circumscribed one
- We need to find a robust criterion to understand how many RBs we need to get a good approximation on each K and improve the convergence (Artificial Intelligence?)

Limits and perspectives

- The adaptive choice of the number of RBs in dependence on the geometry of the polygons has been done in a rough way computing the ratio between the radius of the inscribe circle with the radius of the circumscribed one
- We need to find a robust criterion to understand how many RBs we need to get a good approximation on each K and improve the convergence (Artificial Intelligence?)
- The idea is to obtain a cheap method for the post-processing of VEM solutions (for instance, reconstruction in subdomains)

Some references

- Beirão da Veiga, L., Brezzi, F., Cangiani, A., Manzini, G., Marini, L. D., Russo, A. (2013). *Basic principles of virtual element methods.* Mathematical Models and Methods in Applied Sciences, 23(01), 199-214.
- Hesthaven, J. S., Rozza, G., Stamm, B. (2016). *Certified reduced basis methods for parametrized partial differential equations* (Vol. 590). Berlin: Springer.
- Sorgente T., Prada D., Cabiddu D., Biasotti S., Patane G., Pennacchio M., Bertoluzza S., Manzini G., Spagnuolo M. (2021). *VEM and the Mesh.* arXiv preprint arXiv:2103.01614.