Construction of Reduced Basis Approximations of the Solutions to a Non-linear Eco-evolutionary Model

Peter Rashkov

Institute of Mathematics and Informatics, Sofia, Bulgaria

Supported by the Fund for Scientific Research via NSP *Peter Beron i NIE* of the Bulgarian Ministry of Education and Science [Grant KP-06-DB-5]

10^{th} Conference on Numerical Methods and Applications
Borovec, 22-26.8.2022

Setting

parametric parabolic problem with input $\mu \in \mathcal{M}$ – compact
reaction-diffusion system on $[0, t_{max}] \times \Omega$
competition of 2 populations

$$
\begin{aligned}
\frac{\partial u_1}{\partial t} &= d_1 \Delta u_1 + u_1(a_1 - \mu - u_1 - c_1 u_2), \\
\frac{\partial u_2}{\partial t} &= d_2 \Delta u_2 + u_2(a_2 - u_2 - c_2 u_1) \\
\text{b.c. } u_1(t,x) &= u_2(t,x) = 0, \quad x \in \partial\Omega, t \in [0, t_{max}] \\
\text{i.c. } u_1(0,\cdot)&, u_2(0,\cdot) \in H_0^1(\Omega) - \text{given}
\end{aligned}
\tag{1}
$$

Cantrell and Cosner 1989; Carrère 2017

Numerical scheme for (1)

$f(u;\mu)$ – nonlinearity in the right-hand side of (1)
time step $\tau > 0$
backward Euler scheme: $\partial_t u(\cdot, k\tau) \approx \frac{u^k - u^{k-1}}{\tau}$ where $u^k = u(\cdot, k\tau)$,
weak formulation $\forall \phi \in H_0^1 \times H_0^1$

$$\frac{1}{\tau}\langle u^k, \phi\rangle_2 + \alpha(u^k, \phi) - \langle f(u^k;\mu), \phi\rangle_2 = \frac{1}{\tau}\langle u^{k-1}, \phi\rangle_2 \qquad (2)$$

with

$$\langle u, v\rangle_2 \stackrel{\text{def}}{=} \int_\Omega (u_1 v_1 + u_2 v_2)\, dx,$$

$$\alpha(u, v) \stackrel{\text{def}}{=} \int_\Omega d_1 \nabla u_1 \cdot \nabla v_1 + d_2 \nabla u_2 \cdot \nabla v_2 \, dx, \quad u, v \in H_0^1 \times H_0^1 .$$

Energy norm: $\|w\|_\alpha \stackrel{\text{def}}{=} \sqrt{\alpha(w, w)}$

- $\mathcal{T}_h$ – a triangulation of Ω
- Galerkin approximation with a finite element space $\mathbb{V}_h \stackrel{\text{def}}{=} \mathbb{W}_h \times \mathbb{W}_h$: $\mathbb{W}_h \subset H_0^1(\Omega)$ consists of finite element functions whose restriction on each element of $\mathcal{T}_h$ is piecewise polynomial of a fixed degree
- $(\mathcal{T}_h, \mathbb{W}_h)$ is assumed to satisfy classical assumptions on regularity, affine equivalence and compact support of the finite element functions (Ciarlet 2002, p. 132)
- assume $\mathcal{T}_h$ and $\mathbb{W}_h$ approximate the solution to (2) with sufficient accuracy

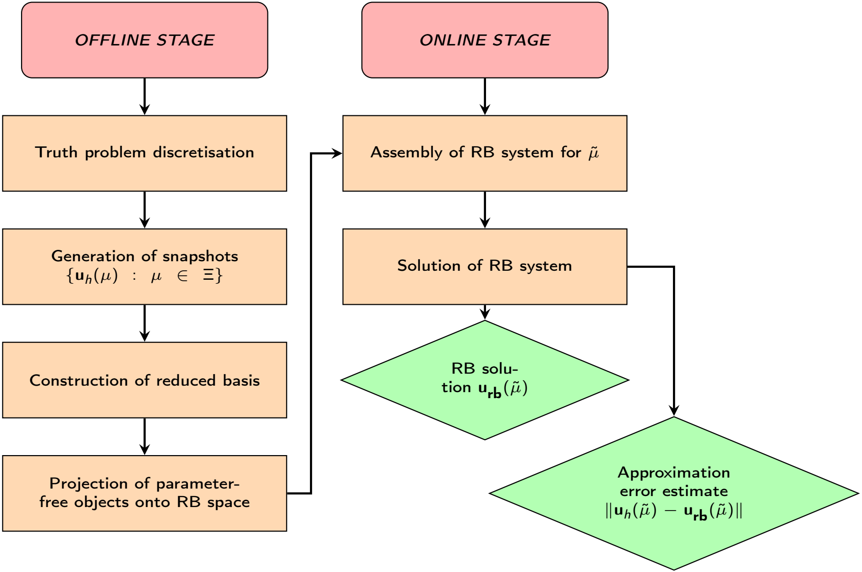
Truth space and snapshot

- FE space $\mathbb{V}_h$: *truth space* (Hesthaven, Rozza, and Stamm 2016; Quarteroni, Manzoni, and Negri 2016).
- *solution snapshot* of (2) for a given $\mu \in \mathcal{M}$ – the sequence of functions $U_h(\mu) = \{u_h^k(\mu) \in \mathbb{V}_h : k = 0, 1, \dots k_{max}\}$ that presents the numerical solutions to (2) in the truth space
- Aim: reduce computational cost for multi-query solutions of (1) for different $\mu \in \mathcal{M}$

Reduced basis method: advantages

- the manifold containing the solutions in the truth space for different parameter values could sometimes be approximated by linear combinations of basis elements of a subspace of lower dimension (*reduced basis space*)
- conditional upon
  - the affine dependence of the considered problem on the free parameter μ
  - choice of training sample Ξ for the collection of snapshots $U_h(\mu), \mu \in \Xi$

Reduced basis method



Quarteroni, Manzoni, and Negri 2016, p. 9

Approach for construction of RB

- POD-greedy reduced basis:
  - POD in time,
  - greedy in μ – requires an a posteriori error estimate
  - reason: to avoid stalling of the algorithm (Haasdonk and Ohlberger 2008; Hesthaven, Rozza, and Stamm 2016)
- simple, straighforward POD with sequential sampling for $\mu \in \Xi$

Assume that the reduced basis space

$$\mathbb{V}_{\mathrm{rb}}^{N} = \mathrm{span}\ \{\xi_i\}_{i=1}^{N} \subset \mathbb{V}_h$$

with $N \ll \dim \mathbb{V}_h$ has already been found.
Let

$$u_{\mathrm{rb}}^{k}(\mu) = \sum_{i=1}^{N} \mathbf{u}_{k,i}^{\mu} \xi_i, \quad \mathbf{x}^s = \sum_{i=1}^{N} \mathbf{x}_i^s \xi_i \tag{3}$$

- Arrive from (2) to

$$\mathbf{G}(x, \phi, \mu; u_{\mathrm{rb}}^{k-1}(\mu)) = 0.$$

- Solve using Newton's method.

Algorithm 1 Newton iteration

Require: $\mathbf{x}^0, \mathbf{u}^{\mu}_{k-1}, \varepsilon_{Newton}$

while $\|\mathbf{G}(\sum_{i=1}^{N} \mathbf{x}_i^s \xi_i, \xi_j, \mu; \mathbf{u}^{\mu}_{k-1,i})\|_\alpha > \varepsilon_{Newton}$ **do**

solve

$$\mathbf{DG}(\xi_i, \xi_j, \mu; \mathbf{x}^s)\delta = -\mathbf{G}(\sum_{i=1}^{N} \mathbf{x}_i^s \xi_i, \xi_j, \mu; \mathbf{u}^{\mu}_{k-1,i}) \quad (4)$$

set $\mathbf{x}^{s+1} = \mathbf{x}^s + \delta, s = s + 1$

end while

Return: $\mathbf{u}^{\mu}_{k} = \mathbf{x}^s$

Rewrite the matrix **DG** and the functional **G** in terms of the reduced basis elements ξ_i.
Denote

$$\mathcal{A}_N = \begin{pmatrix} a_1 \mathbb{I}_N & 0 \\ 0 & a_2 \mathbb{I}_N \end{pmatrix}, \quad \mathcal{C}_N = \begin{pmatrix} 0 & c_1 \mathbb{I}_N \\ c_2 \mathbb{I}_N & 0 \end{pmatrix}, \quad \mathcal{I}_N = \begin{pmatrix} \mathbb{I}_N & 0 \\ 0 & 0 \end{pmatrix}$$

where $\mathbb{I}_N$ is $N \times N$ identity matrix.

Define matrices $\mathbb{M}, \mathbb{A}, \mathbb{B}_1, \mathbb{B}_2$:

$$(\mathbb{M})_{ij} \stackrel{\text{def}}{=} \langle \xi_i, \xi_j \rangle_2, \ (\mathbb{A})_{ij} \stackrel{\text{def}}{=} \alpha(\xi_i, \xi_j),$$
$$(\mathbb{B}_1)_{ij} \stackrel{\text{def}}{=} \langle \mathcal{A}_N \xi_i, \xi_j \rangle_2, \ (\mathbb{B}_2)_{ij} \stackrel{\text{def}}{=} \langle \mathcal{I}_N \xi_i, \xi_j \rangle_2,$$
$$i, j = 1, \dots N.$$

matrix $\mathbb{L}(\mathbf{y})$:

Matrix formulation for (4), slide II

$$\mathbb{L}(\mathbf{y}) : (\mathbb{L})_{ij}(\mathbf{y}) \stackrel{\text{def}}{=} -\sum_{l=1}^{N} \mathbf{y}_l \sum_{m=0}^{2} \beta_m(\xi_l, \xi_i, \xi_j), \quad \mathbf{y} \in \mathbb{R}^N$$

where

$$\beta_0(\xi_i, \xi_j, \xi_l) \stackrel{\text{def}}{=} \int_\Omega \xi_i \xi_j \xi_l, \quad \beta_1(\xi_i, \xi_j, \xi_l) \stackrel{\text{def}}{=} \int_\Omega (\mathcal{C}_N \xi_i) \xi_j \xi_l,$$
$$\beta_2(\xi_i, \xi_j, \xi_l) \stackrel{\text{def}}{=} \int_\Omega \xi_i (\mathcal{C}_N \xi_j) \xi_l \ ,$$

and arrays of matrices

$$(\mathbb{P}^j)_{i_1 i_2} \stackrel{\text{def}}{=} \beta_0(\xi_{i_1}, \xi_{i_2}, \xi_j) \ , (\mathbb{Q}^j)_{i_1 i_2} \stackrel{\text{def}}{=} \beta_2(\xi_{i_1}, \xi_{i_2}, \xi_j)$$

with $i_1, i_2 = 1, \ldots N$.

Matrix formulation for (4), slide III

ФОНД НАУЧНИ ИЗСЛЕДВАНИЯ
МИНИСТЕРСТВО НА ОБРАЗОВАНИЕТО И НАУКАТА

To evaluate the nonlinear terms inside (4) in the reduced basis setting, we compute vectors in $\mathbb{R}^N$:

$$\mathbb{P}(\mathbf{y}) \stackrel{\text{def}}{=} \{\mathbf{y}^T \mathbb{P}^j \mathbf{y}\}_{j=1}^N, \ \mathbb{Q}(\mathbf{y}) \stackrel{\text{def}}{=} \{\mathbf{y}^T \mathbb{Q}^j \mathbf{y}\}_{j=1}^N$$

for appropriate $\mathbf{y} \in \mathbb{R}^N$.

In this notation we rewrite (4) as:

$$\begin{aligned} \mathbb{DG}(\mu, \mathbf{x}^s)\delta &= -\mathbb{G}(\mu, \mathbf{x}^s, \mathbf{u}^\mu_{k-1}) \quad \text{where} \\ \mathbb{DG}(\mu, \mathbf{x}^s) &= \frac{1}{\tau}\mathbb{M} + \mathbb{A} - \mathbb{B}_1 + \mu\mathbb{B}_2 + \mathbb{L}(\mathbf{x}^s) \\ \mathbb{G}(\mu, \mathbf{x}^s, \mathbf{u}^\mu_{k-1}) &= \frac{1}{\tau}\mathbb{M}(\mathbf{x}^s - \mathbf{u}^\mu_{k-1}) + \mathbb{A}\mathbf{y}^s - \mathbb{B}_1\mathbf{x}^s + \mu\mathbb{B}_2\mathbf{x}^s \\ &\quad + \mathbb{P}(\mathbf{x}^s) + \mathbb{Q}(\mathbf{x}^s) \end{aligned} \tag{5}$$

Matrix formulation for (4), slide IV

- The objects $\mathbb{M}, \mathbb{A}, \mathbb{B}_1, \mathbb{B}_2, \mathbb{L}, \mathbb{P}^i, \mathbb{Q}^i, i = 1, \ldots N$ are matrices or arrays of matrices that are *independent* of μ. They can be stored once and for all after the offline stage.
- To solve (4) for any given value $\mu \in \mathcal{M}$, $\mathbb{DG}, \mathbb{G}$ can be assembled during the online stage.
- Compute $\mathbf{u}_k^\mu$ via the Newton iteration (algorithm 1).
- Recover the solution in the reduced basis approximation (3) from $\mathbf{u}_k^\mu$.
- Estimate the approximation error between the solutions in the truth space and in the reduced basis space.

- In order to find the reduced basis via a POD-greedy algorithm, we need an error estimator.
- Introduce a residual:

$$r^k(\phi;\mu) \stackrel{\text{def}}{=} \langle f(u_{\text{rb}}^k;\mu),\phi\rangle_2 - \frac{1}{\tau}\langle u_{\text{rb}}^k - u_{\text{rb}}^{k-1},\phi\rangle_2 - \alpha(u_{\text{rb}}^k,\phi), \quad \forall\phi\in\mathbb{V}_h \tag{6}$$

and a norm in the dual space

$$\|r^k(\cdot;\mu)\|_{\alpha'} \stackrel{\text{def}}{=} \sup_{\phi\in\mathbb{V}_h,\phi\neq 0}\frac{|r^k(\phi;\mu)|}{\|\phi\|_\alpha} \tag{7}$$

- f is (locally) Lipschitz continuous with constant ℓ_{sup}

Approximation error estimate

ФОНД НАУЧНИ ИЗСЛЕДВАНИЯ
МИНИСТЕРСТВО НА ОБРАЗОВАНИЕТО И НАУКАТА

Proposition (Rashkov 2022)

The approximation error at k-th layer,

$$e_k(\mu) = u_h^k(\mu) - u_{rb}^k(\mu),$$

under scheme (2) *satisfies*

$$\|e_k\|_2^2 \leq \frac{\|e_0\|_2^2}{(1 - 2\tau\ell_{sup})^k} + \tau \sum_{j=1}^{k} \frac{\|r^j(\mu)\|_{\alpha'}^2}{(1 - 2\tau\ell_{sup})^{k+1-j}} \qquad (8)$$

for time step $\tau < \frac{1}{2\ell_{sup}}$.

Computing the *a posteriori* error estimator

It suffices to set as the *a posteriori* error estimator

$$\Delta_{\mathrm{rb}}^{N}(\mu) \stackrel{\mathrm{def}}{=} \Delta_{k_{max}}(\mu)$$

because for a given solution trajectory $U_h(\mu)$,
the quantity $\Delta_k(\mu)$ attains its maximum at $k = k_{max}$ (Haasdonk and Ohlberger 2008)

ФОНД НАУЧНИ ИЗСЛЕДВАНИЯ
МИНИСТЕРСТВО НА ОБРАЗОВАНИЕТО И НАУКАТА

Algorithm 2 POD-greedy algorithm

Require: $\varepsilon_{\mathrm{tol}}, \Xi, n_1, n_2 \in \mathbb{N}, n_2 < n_1, \mu_1$
Ensure: $N = 0, \ell = 1, \Delta_{\mathrm{rb}}(\mu_1) = 2\varepsilon_{\mathrm{tol}}, \mathcal{Z} = \varnothing$
  while $\Delta_{\mathrm{rb}}(\mu_\ell) > \varepsilon_{\mathrm{tol}}$ **do**
    compute snapshot $U_h(\mu_\ell)$
    compress $U_h(\mu_\ell)$ using POD, retain n_1 principal nodes $\{\zeta_j\}_{j=1}^{n_1}$
    set $\mathcal{Z} = \mathcal{Z} \cup \{\zeta_j\}_{j=1}^{n_1}$
    if $\ell = 1$ **then**
      $N = n_1$
    else
      $N = N + n_2$
      compress $\mathcal{Z}$ using POD, retain N principal nodes $\{\xi_j\}_{j=1}^{N}$
      $\mathcal{Z} = \{\xi_j\}_{j=1}^{N}$
    end if
    compute the error estimator $\Delta_{\mathrm{rb}}(\mu), \forall \mu \in \Xi$ using $\mathcal{Z}$ as basis
    set $\mu_{\ell+1} = \arg\max_{\mu \in \Xi} \Delta_{\mathrm{rb}}(\mu), \Xi = \Xi \setminus \mu_{\ell+1}, \ell = \ell + 1$
  end while
  Return: $\mathbb{V}_{\mathrm{rb}}^N = \mathcal{Z}, N$

Numerical experiment

finite element library `FreeFem++`

- $\Omega = [0, 10]^2$
- $\mathbb{W}_h$ = Lagrange finite elements of degree 2 on Ω with 6561 dof
- $T_{\text{end}} = 3.99$
- tolerances: $\varepsilon_{\text{Newton}} = 10^{-6}, \varepsilon_{\text{tol}} = 1$
- initial conditions: $u_1(0, x, y) = u_2(0, x, y) = \sin \pi x \sin \pi y$

Table: Parameter values for the numerical experiment.

Experiment No.	a_1	a_2	c_1	c_2	τ
I	1.5	1.0	0.05	0.03	0.03
II	1.5	1.0	0.07	0.15	0.03

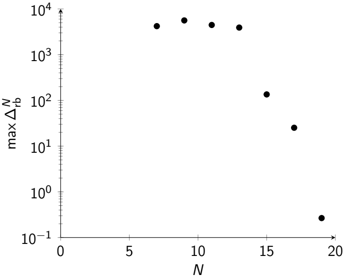
- diffusion parameters: $d_1 = d_2 = 1$
- parameter range: $\mu \in \mathcal{M} = [0, 0.16]$

Numerical experiment

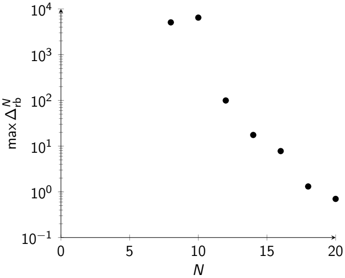
Table: Parameters used for constructing the reduced basis at the offline stage

Experiment No.	n_1	n_2	resulting dim. of RB N
I	7	2	19
II	8	2	20

Offline stage: POD-greedy algorithm



(a) Experiment I



(b) Experiment II

Figure: Value of the *a posteriori* error estimator computed by the POD-greedy algorithm vs. incremental dimension of RB.

Online stage: measures of effectivity

- CPU time gain factor $= \dfrac{\text{CPU time truth}}{\text{CPU time RB}}$, averaged over 10 trials
- L^2-approximation error

$$\|e_{T_{end}}(\mu)\|_2 = \|u_{\mathrm{rb}}(\mu, T_{end}) - u_h(\mu, T_{end})\|_2$$

Table: Comparison of the effectivity when the reduced basis is computed in two ways.

μ	CPU time gain factor	L^2-error	*a posteriori* error
POD-greedy algorithm ($N = 19$)			
0.04	15.99	4.02×10^{-4}	5.45×10^{-1}
0.07	17.13	3.91×10^{-4}	3.03×10^{-1}
0.11	16.96	3.78×10^{-4}	1.83×10^{-1}
POD with sequential sampling ($N = 24$)			
0.04	8.80	8.84×10^{-5}	1.92×10^{-1}
0.07	8.87	8.35×10^{-5}	3.81×10^{-1}
0.11	8.73	7.73×10^{-5}	5.99×10^{-1}

Online stage, Experiment II

ФОНД НАУЧНИ ИЗСЛЕДВАНИЯ
МИНИСТЕРСТВО НА ОБРАЗОВАНИЕТО И НАУКАТА

Table: Comparison of the effectivity when the reduced basis is computed in 2 ways.

μ	CPU time gain factor	L^2-error	*a posteriori* error
POD-greedy algorithm ($N = 20$)			
0.04	12.3880	3.03×10^{-4}	5.81×10^{-2}
0.07	12.3402	2.69×10^{-4}	4.81×10^{-2}
0.11	13.5057	2.47×10^{-4}	3.76×10^{-2}
POD with sequential sampling ($N = 24$)			
0.04	7.6595	2.25×10^{-4}	3.13×10^{-2}
0.07	7.0407	2.18×10^{-4}	2.89×10^{-2}
0.11	7.6818	2.09×10^{-4}	2.99×10^{-2}

Conclusions

- Due to the offline/online decomposition, and the low dimension of the constructed reduced basis space, significant computational savings are obtained.
- Development of a posteriori error estimators for the reduced basis approximation is closely linked to the problem at hand, as noted elsewhere (Hesthaven, Rozza, and Stamm 2016; Quarteroni, Manzoni, and Negri 2016).
- For nonlinear reaction-diffusion equations the estimator reveals a trade-off of sharpness of the estimate and the time integration step.

Thank you for your attention!

Supported by the Fund for Scientific Research via NSP *Peter Beron i NIE* of the Bulgarian Ministry of Education and Science [Grant KP-06-DB-5].

References:

Cantrell, R.S., and C. Cosner. 1989. "On the uniqueness and stability of positive solutions in the Lotka-Volterra competition model with diffusion". *Houston J. Math.* 15 (3): 341–361.

Carrère, C. 2017. "Optimization of an in vitro chemotherapy to avoid resistant tumours". *J. Theor. Biol.* 413:24–33.

Ciarlet, P. G. 2002. *The Finite Element Method for Elliptic Problems*. SIAM.

Haasdonk, B., and M. Ohlberger. 2008. "Reduced Basis Method for Finite Volume Approximations of Parametrized Linear Evolution Equations". *ESAIM Math. Model. Num.* 42:277–302.

Hesthaven, J. S., G. Rozza, and B. Stamm. 2016. *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*. Springer.

Quarteroni, A., A. Manzoni, and F. Negri. 2016. *Reduced Basis Methods for Partial Differential Equations*. Springer.

Rashkov, P. 2022. "Reduced basis approximation for a spatial Lotka-Volterra model". *Mathematics* 10 (12): 1983.