MOX–Report No. 03/2009

Reduced Basis Method for Parametrized Differential Algebraic Equations

Marta D'ELIA, Luca DEDÉ, Alfio QUARTERONI

Reduced Basis Method for Parametrized Differential Algebraic Equations

Marta D'ELIA[1], Luca DEDE'[2], Alfio QUARTERONI[2,3]

January 11st 2009

[1] MathCS – Mathematics & Computer Science Center
Emory University
400 Dowman Dr., Atlanta, GA 30322, USA

[2] MOX – Modeling & Scientific Computing
Dipartimento di Matematica "F. Brioschi"
Politecnico di Milano,
Piazza Leonardo da Vinci 32, 20133, Milano, Italy

[3] Chair of Modeling and Scientific Computing (CMCS)
Institute of Analysis and Scientific Computing (IACS)
École Politechnique Fédérale de Lausanne,
CH-1015 Lausanne, Switzerland

`mdelia2@emory.edu, luca.dede@polimi.it, alfio.quarteroni@epfl.ch`

Abstract

Parametrized systems of Differential Algebraic Equations (DAEs) stand at the base of several mathematical models in Microelectronics, Computational Fluid Dynamics and other Engineering fields. Since the dimension of these systems can be huge, high computational costs could occur, so efficient numerical methods are needed in order to contain the computational cost of the simulations. In this field, Model Order Reduction (MOR) methods represent a valid and efficient approach. In particular, in this work we propose to use Reduced Basis (RB) methods for the solution of parametrized systems of DAEs. Our starting point is the formulation of the RB method for parametrized Partial Differential Equations (PDEs) and the one for non-parametrized DAEs. We describe how to obtain a projection of the solution of the original problem onto a parameter dependent reduced subspace and we provide an a priori estimate for the approximation error. Numerical tests on problems of interest for electronic circuit design highlight the effectiveness of the proposed method. Comparison is made with the parametrized Proper Orthogonal Decomposition (POD) method, which is a typical MOR method.

Keywords: *Parametrized systems of Differential Algebraic Equations; Reduced Basis method; a priori error estimation; circuits modeling.*

Introduction

In this work we deal with the solution of parametrized systems of nonlinear Differential Algebraic Equations (DAEs). DAEs typically arise from Engineering problems which

are described by parametrized dynamical systems. However, DAEs are also obtained after spatial discretization (e.g. by means of the Finite Element, Finite Volume, Finite Difference methods) of parametrized, either parabolic or hyperbolic, Partial Differential Equations (PDEs).
More specifically, we consider microelectronics applications for which the system of DAEs describes circuits of resistors, conductors, diodes and transistors. Due to new technologies in the field of microelectronics, the design of modern Integrated Circuits (ICs) has become more complex and there is an increasing demand for new, effective and efficient optimization tools, in order to avoid costly and repeated re-designs [22]. In particular, the following requirements ought be fulfilled while approximating the original mathematical models: (i) feasible and accurate reproduction of the complete models; (ii) reproduction of the dynamics of the system according to specific choices of the design parameters (e.g. resistances, capacities and diodes' characteristic currents and voltages); (iii) saving of the computational costs. These challenges have prompted to the development of the so called reduced order models, by means of a projection of the continuous solution onto a parameter dependent reduced subspace.
Among reduced order models, Model Order Reduction (MOR) methods have proved to be efficient for the solution of linear non-parametrized systems of DAEs [8, 9, 10, 11, 26]. They have been extended to the nonlinear case in the late '80s in view of applications to several fields of Microelectronics and Computational Fluid Dynamics [1, 2, 3, 22, 24, 25], Chemical, Biological and Mechanical Engineering [22]. Some of these methods were lately extended to a parametrized version [4, 5], as in the case of the parametrized Proper Orthogonal Decomposition (POD$_\mu$) method [5]. Nevertheless, as pointed out in [4], developing effective and efficient methods for parametrized DAEs remains an open issue and constitutes an area of intense investigation, mainly because the computational gain obtained by using such reduction methods for parametrized problems not yet satisfactory.
In this work we propose a Reduced Basis (RB) approach, which, for the sake of simplicity, we indicate as RB$_\mu$-DAEs method, for the reduction of the dimension of nonlinear parametrized systems of DAEs in analogy with the RB method for parametrized PDEs (RB$_\mu$-PDEs) [16, 23] and with the RB method for non-parametrized DAEs (RB-DAEs) [14, 15, 17, 18]. A relevant feature that legitimate the use of the RB$_\mu$-DAEs method is that the solution of a parametrized problem lies in a lower-dimensional subspace induced by the parametric dependence; precisely, we consider the reduced solution as a combination of some "truth" approximated solutions corresponding to certain values of the parameter (the RB$_\mu$-PDEs contribute) which form a basis of the reduced subspace as time evolves (the RB-DAEs contribute). Moving from these assumptions, we propose the RB$_\mu$-DAEs method for the solution of both linear and nonlinear parametrized systems of DAEs; in its terms, the dependence of the parameters can be either linear or nonlinear.

The paper is organized as follows. In Sec.1 we introduce the mathematical problem and the application under consideration: an analogic circuit. In Sec.2 we present the state of the art of the MOR methods for the solution of parametrized systems of DAEs and we report the formulation of the POD$_\mu$ method which will be used in comparison with our RB$_\mu$-DAEs method. In Sec.3 we provide the mathematical formulation of the RB$_\mu$-DAEs method. Firstly, we briefly describe both the RB$_\mu$-PDEs and the RB-DAEs methods; then, we present the mathematical formulation of the RB$_\mu$-DAEs method and we propose a procedure for the choice of the basis. Moreover, we propose an a priori error estimate for the RB$_\mu$-DAEs approximation error. In Sec.4 we report some

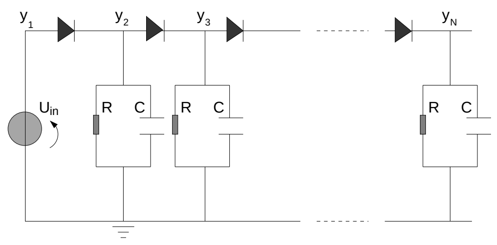


Figure 1: A simple diode chain.

numerical results showing the effectiveness of the proposed method in comparison with the POD$_\mu$ one.

1 The problem at hand

We consider the following N dimensional system of parametrized DAEs:

$$\begin{cases} C\dfrac{d\mathbf{y}(t,\boldsymbol{\mu})}{dt} = \mathbf{f}(t,\mathbf{y}(t,\boldsymbol{\mu});\boldsymbol{\mu}) \quad t \in (0,T], \\ \mathbf{y}(0,\boldsymbol{\mu}) = \mathbf{y}_0, \end{cases} \tag{1}$$

where $\boldsymbol{\mu} \in D^{\boldsymbol{\mu}} \subset \mathbb{R}^P$ is the parameter vector, with $D^{\boldsymbol{\mu}}$ the parameter set and $P \in \mathbb{N}$, $C \in \mathbb{R}^{N\times N}$ is a parameter independent matrix, $\mathbf{f}(t,\mathbf{y};\boldsymbol{\mu}) : (0,T] \times \mathbb{R}^N \to \mathbb{R}^N$ is a sufficiently regular function w.r.t. $t \in (0,T]$ and a Lipschitz-continuous function w.r.t. $\mathbf{y}$, with a parameter independent Lipschitz constant L, and $\mathbf{y}_0 \in \mathbb{R}^N$ is the (parameter independent) initial state condition. These assumptions ensure that Eq.(1) is well posed on the time interval $[0,T]$.
As application, we consider the diode chain reported in Fig.1 (introduced in [26]), which is composed by resistors, capacitors and diodes set in parallel. This device, depending on the input signal, $U_{in}(t)$, could highlight a strongly non linear dynamical response due to the presence of the diodes.
By using the standard Kirkhhoff's laws, the behavior of the diode chain can be modeled by using Eq.(1) [26]. In particular, we assume the following expressions for the source term $\mathbf{f}(t,\mathbf{y};\boldsymbol{\mu})$ of Eq.(1): $\mathbf{f}(t,\mathbf{y};\boldsymbol{\mu}) = -R(\boldsymbol{\mu})\ \mathbf{y}$ in the case of a *linear problem* and $\mathbf{f}(t,\mathbf{y};\boldsymbol{\mu}) = -R(\boldsymbol{\mu})\,\mathbf{y}+\mathbf{g}(t,\mathbf{y};\boldsymbol{\mu})$ for a *nonlinear problem*, being $\mathbf{g}(t,\mathbf{y};\boldsymbol{\mu}) : (0,T]\times\mathbb{R}^N \to \mathbb{R}^N$ a nonlinear function of $\mathbf{y}$, $\forall\boldsymbol{\mu} \in D^{\boldsymbol{\mu}}$; if $\mathbf{g}(t,\mathbf{y};\boldsymbol{\mu}) = \mathbf{g}(t,\mathbf{y})$ then, the dependece on the parameter is *linear*, otherwise, it is *nonlinear*. More precisely, we choose the parameter vector as $\boldsymbol{\mu} = [\boldsymbol{\mu}_R, \boldsymbol{\mu}_V] \in \mathcal{D}^{\boldsymbol{\mu}_R} \times \mathcal{D}^{\boldsymbol{\mu}_V} \equiv \mathcal{D}^{\boldsymbol{\mu}}$, where $\boldsymbol{\mu}_R \in \mathcal{D}^{\boldsymbol{\mu}_R} \subset \mathbb{R}^k$ represents the values of the resistances, while $\boldsymbol{\mu}_V \in \mathcal{D}^{\boldsymbol{\mu}_V} \subset \mathbb{R}^s$ those of the reference voltages of the diodes, being $k, s \in \mathbb{N}$ s.t. $k + s = P$. Also, $C \in \mathbb{R}^{N\times N}$ and $R(\boldsymbol{\mu}) \in \mathbb{R}^{N\times N}$ are the

diagonal singular matrices chosen as:

$$C = c\,\mathrm{diag}(0,1,1,...,1),$$

$$R(\boldsymbol{\mu}_R) = \begin{bmatrix} 1 & 0 & 0 & ... & 0 \\ 0 & -\dfrac{1}{(\boldsymbol{\mu}_R)_1} I_{z_1} & 0 & ... & 0 \\ 0 & 0 & -\dfrac{1}{(\boldsymbol{\mu}_R)_2} I_{z_2} & ... & 0 \\ 0 & 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & ... & -\dfrac{1}{(\boldsymbol{\mu}_R)_k} I_{z_k} \end{bmatrix}, \tag{2}$$

where $c \in \mathbb{R}$ ($[F]$) represents the capacity of the conductors and $I_{z_i} \in \mathbb{R}^{z_i \times z_i}$, for $i = 1,...,k$, is the identity matrix; clearly, $\sum_{i=1}^{k} z_i = N-1$. The function $\mathbf{g}(t,\mathbf{y};\boldsymbol{\mu})$ is defined as:

$$\mathbf{g}(t,\mathbf{y};\boldsymbol{\mu}) = \begin{bmatrix} -U_{in}(t) \\ d(\mathbf{y}_2 - \mathbf{y}_1, \boldsymbol{\mu}_V) - d(\mathbf{y}_3 - \mathbf{y}_2, \boldsymbol{\mu}_V) \\ ... \\ d(\mathbf{y}_j - \mathbf{y}_{j-1}, \boldsymbol{\mu}_V) - d(\mathbf{y}_{j+1} - \mathbf{y}_j, \boldsymbol{\mu}_V) \\ ... \\ d(\mathbf{y}_N - \mathbf{y}_{N-1}, \boldsymbol{\mu}_V) \end{bmatrix}, \tag{3}$$

where, the scalar function $d : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ represents the constitutive equation of the diode and it is defined as follows:

$$d(v, \mu_V) = \begin{cases} 0 & v < 0.5\ V \\ I_s(e^{v/\mu_V} - 1) & v \geq 0.5\ V, \end{cases} \tag{4}$$

being I_s the diode current ($[A]$) and μ_V the characteristic voltage ($[V]$); moreover, we choose $U_{in}(t)$ as:

$$U_{in}(t) = \begin{cases} 20 & t \in [0, 10]\ ns, \\ 170 - 15(10^9\ t) & t \in (10, 11]\ ns, \\ 5 & t \in (11\ ns, +\infty). \end{cases} \tag{5}$$

In Eq.(1) each solution component $\mathbf{y}_i$, $i = 1,...,N$, represents the electrical voltage in each node of the circuit.
Since the number N of the components in modern ICs might be huge (e.g. $N \cong 10^6$), high computational costs could occur while solving numerically the problem. In Tab.1 we report the computational costs required for the numerical solution of system (1) by the implicit Backward-Euler (BE) method for time discretization[1]. We choose a single parameter dependence for which $\boldsymbol{\mu} = \boldsymbol{\mu}_R = \mu \in \mathcal{D}^{\boldsymbol{\mu}_R} \subset \mathbb{R}$ represents the resistance of all the resistors of the circuit. In particular, for the example considered, we choose $\mu = 10^4\ \Omega$ and a discretization time-step $\Delta t = 0.1\ ns$. When addressing more relevant circuits, alternative approaches, such as those based on model or dimensional reduction, become therefore mandatory.

[1]The simulations are carried out on an Intel 2.20 GHz processor with 4 MB of cache memory and 2 GB of RAM.

N	200	500	1000
CPU [sec]	10.84	58.17	245.6

Table 1: Computational costs (CPU times) required for solving the system of DAEs induced by the electronic device described by Eq.s (2) and (3).

2 MOR methods for systems of parametrized DAEs: state of the art

In this Section we give a brief account on the existing MOR techniques applied to systems of parametrized DAEs; in particular, we focus on the so-called POD$_\mu$ method. MOR methods are the most widespread techniques used for the solution of (linear or nonlinear) systems of DAEs [2, 22, 26]. These methods are based on a suitable projection of the solution of the original system onto a lower dimensional state space in order to obtain a new model of reduced dimension. Different projection strategies yield different MOR methods.
The use of MOR techniques in the field of parametrized problems is less common. In this respect, a typical approach consists in combining a non-parametrized MOR technique with a supplementary procedure for parametrized problems (e.g. interpolation or moment matching). In particular, in this paper we consider the POD$_\mu$ method (introduced in [5]), which is described in details in Sec.2.1 in view of the comparison with the RB$_\mu$-DAE method that we are going to introduce in Sec.3.3. Among MOR methods for parametrized nonlinear systems we mention the Trajectory Piece-Wise Linear method [4], in which the original system is linearized around suitably selected states, then it is approximated by a weighted combination of such linearized models, each one reduced by means of the Krylov-subspace technique [8, 9, 10]. The dependence on the parameter is taken into account with a moment matching technique [4].

2.1 The parametrized Proper Orthogonal Decomposition method

In this Section we describe the POD$_\mu$ method, firstly proposed for Aerodynamics applications [5], highlighting the decomposition of the procedure in an *offline* step (actually the standard POD procedure) and in an *online* one (where the solution is evaluated for a given value of the parameter $\overline{\boldsymbol{\mu}} \in D^{\boldsymbol{\mu}}$). More precisely, this method consists in performing at the *offline* step the non-parametrized POD procedure on a set of problems corresponding to selected values of the parameter. In this way a set of projectors (matrices) onto a reduced subspace, one for each of the selected parameters, is obtained. Once this set of projectors is given, the POD$_\mu$ procedure combines the projection matrices by means of an interpolation technique in order to obtain a parameter dependent projector which is used at the *online* step to build the reduced model. Moreover, in order to furtherly reduce the computational costs, we propose in this work to combine the POD$_\mu$ method with the Missing Point Estimation (MPE) technique. We notice that this approach (POD-MPE) has been already introduced in [2, 24] even if in the non-parametrized case.
We summarize firstly the *offline* step of the POD$_\mu$ method.
Given a set of selected parameters $\overline{\mathcal{D}}^{\boldsymbol{\mu}} \subset D^{\boldsymbol{\mu}}$ defined as $\overline{\mathcal{D}}^{\boldsymbol{\mu}} := \{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, ..., \boldsymbol{\mu}_D\}$, with $D \in \mathbb{N}$:

A perform the following steps of the non-parametrized POD procedure: for $i =$

$1, ..., D$

(a) collect $J \in \mathbb{N}$ evaluations at different times of the unknowns of the system (e.g. by measurements of the voltages of the system at some selected time-steps), $\mathbf{y}(t_j, \boldsymbol{\mu}_i)$, $j = 1, ..., J$, and assemble the snapshots matrix, say $Y(\boldsymbol{\mu}_i) \in \mathbb{R}^{N \times J}$, s.t.

$$Y(\boldsymbol{\mu}_i) = [\mathbf{y}(t_1, \boldsymbol{\mu}_i) \ ... \ \mathbf{y}(t_J, \boldsymbol{\mu}_i)]; \tag{6}$$

(b) solve the following eigenvalue problems:

$$H(\boldsymbol{\mu}_i)\mathbf{u}_j(\boldsymbol{\mu}_i) = \lambda_j(\boldsymbol{\mu}_i)\mathbf{u}_j(\boldsymbol{\mu}_i) \qquad \forall\, j = 1, ..., J, \tag{7}$$

where $\mathbf{u}_j(\boldsymbol{\mu}_i) \in \mathbb{R}^N$ and $\lambda_j(\boldsymbol{\mu}_i) \in \mathbb{R}$, $j = 1, ..., J$, are respectively the eigenvectors (which are assumed to be sufficiently regular w.r.t. each $\boldsymbol{\mu}_i$) and the eigenvalues of the correlation matrix $H(\boldsymbol{\mu}_i) \in \mathbb{R}^{N \times N}$ defined as $H(\boldsymbol{\mu}_i) := \dfrac{1}{J} Y(\boldsymbol{\mu}_i) Y(\boldsymbol{\mu}_i)^T$;

(c) apply the total energy criterion (see [2]) taking into account for the 99% of the total energy, for which compute $E_j(\boldsymbol{\mu}_i) = \sum_{s=1}^{j} \lambda_s(\boldsymbol{\mu}_i) / \sum_{s=1}^{J} \lambda_s(\boldsymbol{\mu}_i)$, $j = 1, ..., J$, and set:

$$K(\boldsymbol{\mu}_i) := \underset{j=1,\dots,J}{\operatorname{argmin}} \ |E_j(\boldsymbol{\mu}) - 0.99|, \qquad K(\boldsymbol{\mu}_i) \in \mathbb{N}; \tag{8}$$

B set $K := \min_{i=1,\dots,D} K(\boldsymbol{\mu}_i)$;

C for $i = 1, ..., D$, set $\widetilde{U}(\boldsymbol{\mu}_i) := [\mathbf{u}_1(\boldsymbol{\mu}_i) \ ... \ \mathbf{u}_K(\boldsymbol{\mu}_i)]$ and perform the cubic spline interpolation for each column vector obtaining, as described in [6], the interpolant $\Pi_k(\boldsymbol{\mu}) : \mathbb{R}^P \to \mathbb{R}^N$, $k = 1, ..., K$ s.t. $\Pi_k(\boldsymbol{\mu}_i) = \mathbf{u}_k(\boldsymbol{\mu}_i)$, $\forall k = 1, ..., K$, $\forall i = 1, ..., D$.

These *offline* steps are performed just once and, hence, they are allowed to be computationally expensive.
At the *online* step we aim at rapidly computing the reduced solution for a given value of the parameter $\overline{\boldsymbol{\mu}} \in D^{\boldsymbol{\mu}}$. In particular, we assemble the matrix $V(\overline{\boldsymbol{\mu}}) := [\Pi_1(\overline{\boldsymbol{\mu}}) \ ... \ \Pi_K(\overline{\boldsymbol{\mu}})]$ and we perform the projection returning the reduced system. In practise, we project the solution of the original system (1) onto the reduced subspace and we replace $\mathbf{y}(t, \boldsymbol{\mu})$ in Eq.(1) by the reduced solution $\mathbf{y}_R(t, \overline{\boldsymbol{\mu}}) := V(\overline{\boldsymbol{\mu}})\mathbf{z}(t, \overline{\boldsymbol{\mu}})$, being $\mathbf{z}(t, \overline{\boldsymbol{\mu}}) \in \mathbb{R}^K$. Finally, we solve the reduced system obtained by multiplying both the r.h.s. and the l.h.s. of Eq.(1) by $V(\overline{\boldsymbol{\mu}})^T$:

$$\begin{cases} C_R(\overline{\boldsymbol{\mu}}) \dfrac{d\mathbf{z}(t, \overline{\boldsymbol{\mu}})}{dt} = \mathbf{f}_R(t, \mathbf{z}(t, \overline{\boldsymbol{\mu}}); \overline{\boldsymbol{\mu}}) \qquad t \in (0, T], \\ \mathbf{z}(0, \overline{\boldsymbol{\mu}}) = \mathbf{z}_0; \end{cases} \tag{9}$$

where $C_R(\overline{\boldsymbol{\mu}}) := V(\overline{\boldsymbol{\mu}})^T C \, V(\overline{\boldsymbol{\mu}})$, $\mathbf{f}_R(t, \mathbf{z}(t, \overline{\boldsymbol{\mu}}); \overline{\boldsymbol{\mu}}) := V(\overline{\boldsymbol{\mu}})^T \mathbf{f}(t, V(\overline{\boldsymbol{\mu}})\mathbf{z}(t, \overline{\boldsymbol{\mu}}); \overline{\boldsymbol{\mu}})$ and $\mathbf{z}_0 \in \mathbb{R}^K$ is chosen such that $V(\overline{\boldsymbol{\mu}})\mathbf{z}_0 = \mathbf{y}_0$.

3 The Reduced Basis method for systems of parametrized DAEs

In this Section we describe the mathematical formulation of the RB$_\mu$-DAEs method, which we propose for solving parametrized nonlinear systems of DAEs. In particular,

we specify the choices of approximation schemes and linear solvers and we propose a procedure for the generation of the basis; finally, we present an *a priori* error estimate for the approximation error associated with the method.
In order to make straightforward the presentation of the method, we briefly recall the RB-DAEs and the RB$_\mu$-PDEs methods from which the basic idea of the RB$_\mu$-DAEs method stems out.

3.1 The RB-DAEs method

We report the mathematical formulation of the RB-DAEs method introduced by Porsching and Lee [13, 17, 18]; for further details see also [14, 15].
The RB-DAEs method has been firstly introduced in the context of parameterized nonlinear systems of Algebraic (i.e. stationary) Equations, say RB$_\mu$-AEs method, where a projection of the solution of the original system onto a parameter dependent manifold is performed. In [18] Porsching and Lee proposed an approach adapting the RB$_\mu$-AEs method to non-parametrized nonlinear systems of DAEs, say the RB-DAEs method, where the time variable of the dynamical system is chosen as the parameter used to perform the projection. The basic idea of the method consists in partitioning the time interval and then performing a projection of the solution of the original system onto a lower-dimensional subspace in each subinterval. The complete reduced solution is obtained by assembling the reduced solutions computed into every subinterval.
In particular, we divide the time interval $[0,T]$ in $J \in \mathbb{N}$ subintervals $I_j = (t_{j-1}, t_j]$ s.t. $0 = t_0 < ... < t_J = T$ and $\bigcup_{j=1}^{J} I_j = (0,T]$. Then, the RB-DAEs method consists in finding a reduced solution, let say $\mathbf{y}_{R,j}(t)$, in each subinterval I_j, $j = 1, ..., J$; the complete reduced solution is given by $\mathbf{y}_R(t) = \mathbf{y}_{R,j}(t)$, $t_j < t \le t_{j+1}$, $j = 0, 1, ..., J-1$, provided that $\mathbf{y}_R(t_0) = \mathbf{y}_0$.
In the following steps we describe the procedure for the computation of $\mathbf{y}_{R,j}$, for $j = 1, ..., J$:

1. choose the reduced subspace $\mathcal{S}_j \subset \mathbb{R}^N$, $\forall\, j = 1, ..., J$, which the reduced solution belongs to, which is defined as
$$\mathcal{S}_j = \text{span}\{\mathbf{u}_j^1, \mathbf{u}_j^2, ..., \mathbf{u}_j^{M_j}\}, \qquad M_j \in \mathbb{N}, \tag{10}$$
for some $\{\mathbf{u}_j^1, ..., \mathbf{u}_j^{M_j}\} \subset \mathbb{R}^N$ which can be chosen in different ways (see [18] for more details) and depend on the exact solution or on the r.h.s. of Eq.(1);

2. assemble the projection matrix $Y_j \in \mathbb{R}^{N \times M_j}$ defined as $Y_j = \left[\mathbf{u}_j^1\ \mathbf{u}_j^2\ ...\ \mathbf{u}_j^{M_j}\right]$;

3. write the reduced solution as:
$$\mathbf{y}_{R,j}(t) = Y_j\ \mathbf{z}_j(t) + \mathbf{y}_{R,j-1}(t_j), \tag{11}$$
for some $\mathbf{z}_j(t) \in \mathbb{R}^{M_j}$ to be determined;

4. rewrite system (1) over the time subinterval I_j by replacing the exact solution $\mathbf{y}$ with the reduced one $\mathbf{y}_{R,j}$ as follows:
$$\begin{cases} Y_j^T \dfrac{d\mathbf{y}_{R,j}}{dt} = Y_j^T\ \mathbf{f}(t, \mathbf{y}_{R,j}) & t \in I_j, \\ \mathbf{y}_{R,j}(t_j) = \mathbf{y}_{R,j-1}(t_j); \end{cases} \tag{12}$$

or equivalently:

$$\begin{cases} Y_j^T Y_j \dfrac{d\mathbf{z}_j}{dt} = Y_j^T \,\mathbf{f}(t, Y_j\mathbf{z}_j + \mathbf{y}_{R,j-1}(t_j)) & t \in I_j, \\ \mathbf{z}_j(t_j) = 0; \end{cases} \tag{13}$$

5. solve the M_j dimensional system (13) and get $\mathbf{y}_{R,j}$, for $j = 1, ..., J$.

Finally, by assembling the J reduced solutions $\mathbf{y}_{R,j}$, we obtain the complete reduced solution $\mathbf{y}_R(t)$, $t \in [0, T]$.

3.2 The RB$_{\boldsymbol{\mu}}$-PDEs method

We briefly report the mathematical formulation of the RB$_\mu$-PDEs. For an in-depth insight, we refer the reader to, e.g, [16].
The basic idea of the RB$_{\boldsymbol{\mu}}$-PDEs method consists in "assembling" a reduced parametrized problem, starting from the original parametrized PDEs, which allows to reproduce accurately the original solution while containing the computational costs. The solution of such reduced problem lies in a lower-dimensional space, whose basis is composed by M solutions of the original problem corresponding to selected values of the parameters. The main feature of the method consists in splitting the whole procedure in an *offline* step (computationally expensive) and an *online* step (rapid) in which the reduced parametrized problem is solved for a given value of the parameter. The RB$_\mu$-PDEs method has been proved to be convenient, accurate and reliable for the solution of parametrized PDEs and systems of PDEs on a broad range of Engineering applications [16, 23].
Let $\mathcal{Y}$ be a Hilbert space, $\mathcal{Y}'$ its dual space and ${}_{\mathcal{Y}'}\langle\cdot,\cdot\rangle_{\mathcal{Y}} \equiv \langle\cdot,\cdot\rangle$ the associated duality pair. Then, we define, for any $\boldsymbol{\mu} \in D^{\boldsymbol{\mu}} \subset \mathbb{R}^P$, where, once again, $D^{\boldsymbol{\mu}}$ is the parameter set and $P \in \mathbb{N}$ is the corresponding dimension, a parametrized linear operator $A(\boldsymbol{\mu}) : Y \to Y'$. We assume that $A(\boldsymbol{\mu})$ could be affinely decomposed as a combination of $Q \in \mathbb{N}$ linear parameter independent operators, $A_q : \mathcal{Y} \to \mathcal{Y}'$, weighed by parameter dependent functions $\Theta_q(\boldsymbol{\mu}) : \mathcal{D}^{\boldsymbol{\mu}} \to \mathbb{R}$, $q = 1, ..., Q$, s.t:

$$A(\boldsymbol{\mu}) = \sum_{q=1}^{Q} \Theta_q(\boldsymbol{\mu}) A_q. \tag{14}$$

We aim at finding the parameter dependent solution $y(\boldsymbol{\mu}) \in \mathcal{Y}$ of the following problem:

$$\text{find } y(\boldsymbol{\mu}) \in \mathcal{Y} \quad \text{s.t} \quad \langle A(\boldsymbol{\mu})y(\boldsymbol{\mu}), v\rangle = \langle F, v\rangle \qquad \forall\, v \in \mathcal{Y},\ \forall\, \boldsymbol{\mu} \in D^{\boldsymbol{\mu}}, \tag{15}$$

where $F \in \mathcal{Y}'$. As the exact solution of Eq.(15) is not always available, we require a finite-dimensional "truth" approximation of the infinite-dimensional space $\mathcal{Y}$, let say $\mathcal{Y}_h$, an N dimensional subspace of $\mathcal{Y}$ (in general we expect N to be very large). The approximated "truth" solution $y_h(\boldsymbol{\mu})$ lies in the space $\mathcal{Y}_h$ which is typically defined as a Finite Element (FE) subspace of $\mathcal{V}$ [16]. Hence, the approximated Galerkin-FE problem reads:

$$\text{find } y_h(\boldsymbol{\mu}) \in \mathcal{Y}_h \quad \text{s.t} \quad \langle A(\boldsymbol{\mu})y_h(\boldsymbol{\mu}), v_h\rangle = \langle F, v_h\rangle \qquad \forall\, v_h \in \mathcal{Y}_h,\ \forall\, \boldsymbol{\mu} \in D^{\boldsymbol{\mu}}. \tag{16}$$

It is convenient to express the solution of Eq.(16) in terms of the basis of the space $\mathcal{Y}_h$; we let $\phi_i \in Y_h$ be the generic characteristic Lagrangian basis function, such that

$\phi_i(x_i) = \delta_{ij}$, with $i, j = 1, ..., N$, being x_j the coordinate of the j-th node of the triangulation, and δ_{ij} the Kronecher delta. Hence, Eq.(16) reads as:

$$\text{find } \mathbf{y}_h(\boldsymbol{\mu}) \in \mathbb{R}^N \quad \text{s.t.} \quad A_h(\boldsymbol{\mu})\mathbf{y}_h(\boldsymbol{\mu}) = \mathbf{F}_h, \tag{17}$$

where $\mathbf{y}_h(\boldsymbol{\mu}) = \sum_{i=1}^{N} (\mathbf{y}_h)_i(\boldsymbol{\mu})\phi_i$, the matrix $A_h(\boldsymbol{\mu}) \in \mathbb{R}^{N\times N}$ is $[A_h(\boldsymbol{\mu})]_{ij} := \langle A(\boldsymbol{\mu})\phi_j, \phi_i\rangle$, with $i, j = 1, ..., N$ and $\mathbf{F}_h \in \mathbb{R}^N$ is $[\mathbf{F}_h]_i = \langle F, \phi_i\rangle$, $i = 1, ..., N$. We observe that the affine decomposition hypothesis (14) allows us to write:

$$A_h(\boldsymbol{\mu}) = \sum_{q=1}^{Q} \Theta_q(\boldsymbol{\mu})A_{h,q}, \tag{18}$$

being $[A_{h,q}(\boldsymbol{\mu})]_{ij} := \langle A_q(\boldsymbol{\mu})\phi_j, \phi_i\rangle$, $i, j = 1, ..., N$.
The FE approach could lead to high computational costs, if N is huge, especially in the many-query contest. Even if we can decompose the computation of the solution in an *offline* step and an *online* one, "too" high computational costs are still required at the *online* step, hence the RB$_\mu$-PDEs method can conveniently be considered.
In the RB$_\mu$-PDEs method the solution of the parametrized problem lies on a low-dimensional subspace $\mathcal{Y}_M$ of $\mathcal{Y}_h$. The elements of the basis of the reduced space $\mathcal{Y}_M$ are the Galerkin-FE solutions, say $\zeta_m = y_h(\boldsymbol{\mu}_m)$, corresponding to some values of the parameter $\boldsymbol{\mu}_m \in D^{\boldsymbol{\mu}}$. More precisely:

$$\mathcal{Y}_M := \text{span}\{\zeta_m \ : \ m = 1, ..., M\} \qquad M \in \mathbb{N}, \tag{19}$$

where $M \ll N$. Hence, the reduced solution solves the following problem:

$$\text{find } y_M(\boldsymbol{\mu}) \in \mathcal{Y}_M \quad \text{s.t} \quad \langle A(\boldsymbol{\mu})y_M(\boldsymbol{\mu}), v\rangle = \langle F, v\rangle, \qquad \forall\, v \in \mathcal{Y}_M,\ \forall\, \boldsymbol{\mu} \in D^{\boldsymbol{\mu}}. \tag{20}$$

Once again, it is possible to rewrite Eq.(20) in terms of the basis functions $\{\zeta_m\}_{m=1}^M$, which, in a matricial notation, reads:

$$\text{find } \mathbf{y}_M(\boldsymbol{\mu}) \in \mathbb{R}^M \quad \text{s.t.} \quad A_M(\boldsymbol{\mu})\mathbf{y}_M(\boldsymbol{\mu}) = \mathbf{F}_M, \tag{21}$$

where $y_M(\boldsymbol{\mu}) := \sum_{m=1}^{M} (\mathbf{y}_M)_m(\boldsymbol{\mu})\zeta_m$, the matrix $A_M(\boldsymbol{\mu}) \in \mathbb{R}^{M\times M}$ is $[A_M(\boldsymbol{\mu})]_{ij} := \langle A(\boldsymbol{\mu})\zeta_j, \zeta_i\rangle$, with $i, j = 1, ..., M$ and $\mathbf{F}_M \in \mathbb{R}^M$ is $[\mathbf{F}_M]_i = \langle F, \zeta_i\rangle$, $i = 1, ..., M$. We observe that $A_M(\boldsymbol{\mu})$ can be affinely decomposed similarly to Eq.(14), as:

$$A_M(\boldsymbol{\mu}) = \sum_{q=1}^{Q} \Theta_q(\boldsymbol{\mu})A_{M_q}, \tag{22}$$

being $[A_{M_q}(\boldsymbol{\mu})]_{ij} := \langle A_q(\boldsymbol{\mu})\zeta_j, \zeta_i\rangle$, $i, j = 1, ..., M$.
Due to the affine decomposition assumption the RB$_\mu$-PDEs method can be divided in two steps:

- at the *offline* step the reduced space $\mathcal{Y}_M$ is built, the parameter independent matrices A_{M_q} are assembled (this step typically requires high computational costs, even if is performed just once);
- at the *online* step given $\boldsymbol{\mu} \in D^{\boldsymbol{\mu}}$, the matrix $A_M(\boldsymbol{\mu})$ is assembled and the linear system (21) solved (the solution is typycally very rapid, being $M \ll N$.

Remark 3.1 *At the online step we are actually looking for the combination of some "truth" FE solutions. From an algebraic point of view the unknowns of our problem are the weights of the linear combination, i.e.* $(\mathbf{y}_M(\boldsymbol{\mu}))_m$, *for* $m = 1, ..., M$.

3.3 The RB$_{\boldsymbol{\mu}}$-DAEs method

In this Section we mathematically formulate our RB$_{\boldsymbol{\mu}}$-DAEs method for the solution of parametrized systems of DAEs.
At the basis of the RB$_{\boldsymbol{\mu}}$-DAEs method there is the assumption that the solution of a parametrized problem is not an arbitrary member of an infinite-dimensional solution space, but it resides (evolves) on a lower-dimensional subspace induced by the parameter dependence. In particular, we are interested in solving the system of parametrized DAEs defined in Eq.(1). The idea, which arises from the considerations reported in Remark 3.1, consists in looking for the reduced solution as the projection of the true solution onto a lower-dimension subspace, let say $\mathcal{S}_R$, whose basis is defined by some FE solutions (called "truth" solutions) corresponding to some values of the parameters. We write the reduced solution in a form similar to the one used for the PDEs case, i.e. as a combination of M non-reduced solutions:

$$\mathbf{y}_R(t, \boldsymbol{\mu}) = Y(t)\ \mathbf{a}(t, \boldsymbol{\mu}), \tag{23}$$

where the components of $\mathbf{a}(t, \boldsymbol{\mu}) \in \mathbb{R}^M$ represent the weights $((y_M(\boldsymbol{\mu}))_m$, $m = 1, ..., M$, in the PDEs case) associated with the basis (the "truth" solutions ζ_m in the PDEs case), which are the vectors contained in the time-independent matrix $Y(t) \in \mathbb{R}^{N\times M}$, $\forall\ t \in [0, T]$. We point out that, as in the RB-DAEs method, the reduced subspace is a function of the time, hence we write $\mathcal{S}_R = \mathcal{S}_R(t)$.
Two approaches, associated with two different choices of the reduced subspace $\mathcal{S}_R(t)$, can be implemented: the Reduce-and-Discretize (RD) approach and the Discretize-and-Reduce (DR) one.
The RD approach is similar to the one proposed by Porsching in [18]: i.e. given a set of parameters $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, ..., \boldsymbol{\mu}_M\} \subset D^{\boldsymbol{\mu}}$, we define the reduced subspace $\mathcal{S}_R(t)$ as:

$$\mathcal{S}_R(t) := \text{span}\{\mathbf{y}(t, \boldsymbol{\mu}_1), \mathbf{y}(t, \boldsymbol{\mu}_2), ..., \mathbf{y}(t, \boldsymbol{\mu}_M)\} \qquad \forall\ t \in [0, T], \tag{24}$$

where $\mathbf{y}(t, \boldsymbol{\mu}_m)$ is the exact solution of Eq.(1) corresponding to $\boldsymbol{\mu}_m$, with $m = 1, 2, ..., M$,.
By substituting $\mathbf{y}_R(t, \boldsymbol{\mu})$ (see Eq.(23)) in place of $\mathbf{y}(t, \boldsymbol{\mu})$ in Eq.(1), we obtain a reduced continuous model of dimension M. Then, by means of a time discretization scheme we discretize the reduced model and we get the approximated non-reduced solution.
In the DR approach, for a given partition of the whole time interval $[0, T]$ ($t^0, t^1, ..., t^F$, with $F \in \mathbb{N}$, s.t. $t^F = T$), we apply, first of all, a time discretization scheme to Eq.(1), yielding a nonlinear system of algebraic equations at each time-step t^n, $n = 0, ..., F$. We define the reduced subspace at each time-step as:

$$\mathcal{S}_R(t^n) := \text{span}\{\mathbf{y}_h(t^n, \boldsymbol{\mu}_1), \mathbf{y}_h(t^n, \boldsymbol{\mu}_2), ..., \mathbf{y}_h(t^n, \boldsymbol{\mu}_M)\} \qquad \forall\ n = 1, ..., F, \tag{25}$$

where $\mathbf{y}_h(t^n, \boldsymbol{\mu}_m)$, for $m = 1, ..., M$ and $n = 1, ..., F$, is the non-reduced approximate solution obtained in correspondence of $\boldsymbol{\mu}_m$. By means of Eq.(23) we perform the projection onto the reduced subspace at each time-step and we get the reduced solution $\mathbf{y}_R(t^n, \boldsymbol{\mu})$ by solving the M-dimensional algebraic system for $n = 1, ..., F$.

Remark 3.2 *Numerical tests carried out by means of the RB_μ-DAEs method have revealed that the DR approach is more efficient w.r.t. the RD one in terms of computational costs for a prescribed accuracy [7]. For this reason, in this work we detail the $RB_{\boldsymbol{\mu}}$-DAEs method according to the DR approach.*

3.3.1 The DR approach

The first step of the DR approach consists in choosing the time discretization scheme. The corresponding approximate solution will be regarded as the "truth" solution. For the sake of comparison, it is advisable to use the same discretization scheme when solving both the non-reduced and reduced problems, in order to better appreciate the effectiveness of the reduction. In this work we will use the Backward-Euler (BE) method, an implicit one-step method [21], in order to skip stability troubles both for the reduced and the non-reduced problems. We present here the formulation of the method in the case of a linear parametric dependence of the nonlinear function $\mathbf{g}$ (i.e. we define $\mathbf{f}(t, \mathbf{y}; \boldsymbol{\mu}) = -R(\boldsymbol{\mu})\mathbf{y} + \mathbf{g}(t, \mathbf{y})$, see Sec.1). In the case of a nonlinear parametric dependence we move from the same formulation. Then, let $\mathbf{y}_h^n(\boldsymbol{\mu})$ denote the non-reduced BE approximation of the exact solution $\mathbf{y}(t^n, \boldsymbol{\mu})$ at time step t^n, for $n = 0, ..., F$, which is the solution of the following problem:

$$C(\mathbf{y}_h^{n+1} - \mathbf{y}_h^n) = -\Delta t \; R(\boldsymbol{\mu})\mathbf{y}_h^{n+1} + \Delta t \; \mathbf{g}(t^{n+1}, \mathbf{y}_h^{n+1}) \quad \forall \; n = 0, ..., F-1, \tag{26}$$

being $\mathbf{y}_h^0 = \mathbf{y}_0$. In order to perform a projection, at each time-step of the discretization scheme, we have to compute the vectors of the basis of the reduced subspace; given a suitable set of M parameters $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, ..., \boldsymbol{\mu}_M\} \subset D^{\boldsymbol{\mu}}$, $\mathcal{S}_R(t^n)$ assumes the following form $\mathcal{S}_R(t^n) := \text{span}\{\mathbf{y}_h(t^n, \boldsymbol{\mu}_1), \dots, \mathbf{y}_h(t^n, \boldsymbol{\mu}_M)\}$, for $n = 1, \dots, F$. Then, we write the reduced solution as a combination of the basis vectors, s.t. $\mathbf{y}_R(t^n, \boldsymbol{\mu}) = Y(t^n)\mathbf{a}(t^n, \boldsymbol{\mu})$, where the matrix $Y(t^n)$ is formed by the basis vectors of the reduced subspace and we substitute this expression in Eq.(26). We obtain a $N \times M$ dimensional system at each time-step:

$$C(Y^{n+1}\mathbf{a}^{n+1}(\boldsymbol{\mu}) - Y^n\mathbf{a}^n(\boldsymbol{\mu})) = -\Delta t \; R(\boldsymbol{\mu})Y^{n+1}\mathbf{a}^{n+1}(\boldsymbol{\mu}) + \Delta t \; \mathbf{g}(Y^{n+1}\mathbf{a}^{n+1}(\boldsymbol{\mu}), t^{n+1}) \qquad \forall \; n = 0, \dots, F-1, \tag{27}$$

where $\mathbf{a}^n(\boldsymbol{\mu}) \in \mathbb{R}^M$ is defined as $\mathbf{a}^n(\boldsymbol{\mu}) := \mathbf{a}(t^n, \boldsymbol{\mu}) \; \forall n = 0, \dots, F$. We can rewrite Eq.(26) as:

$$A^{n+1}(\boldsymbol{\mu})\mathbf{a}^{n+1}(\boldsymbol{\mu}) - \Delta t \; \mathbf{g}(t^{n+1}, Y^{n+1}\mathbf{a}^{n+1}(\boldsymbol{\mu})) = \mathbf{f}^n(\boldsymbol{\mu}) \qquad \forall \; n = 0, ..., F-1, \tag{28}$$

where $A^n(\boldsymbol{\mu}) \in \mathbb{R}^{N \times M}$ is $A^n(\boldsymbol{\mu}) := [CY^n + \Delta t \; R(\boldsymbol{\mu})Y^n]$, while $\mathbf{f}^n(\boldsymbol{\mu}) \in \mathbb{R}^N$ reads $\mathbf{f}^n(\boldsymbol{\mu}) := CY^n\mathbf{a}^n(\boldsymbol{\mu})$, $n = 1, ..., F$.
Firstly, we consider the linear case, i.e. $\mathbf{g}(t, \mathbf{y}) \equiv \mathbf{0}$, and we discuss a suitable technique for the solution of the rectangular system (28).
We multiply both sides by $(A^{n+1}(\boldsymbol{\mu}))^T$, thus getting the squared linear system:

$$D^{n+1}(\boldsymbol{\mu})\mathbf{a}^{n+1}(\boldsymbol{\mu}) = \mathbf{F}^{n+1}(\boldsymbol{\mu}), \tag{29}$$

where $D^{n+1}(\boldsymbol{\mu}) = (A^n(\boldsymbol{\mu}))^T A^{n+1}(\boldsymbol{\mu})$ and $\mathbf{F}^{n+1} = (A^n(\boldsymbol{\mu}))^T \mathbf{f}^n(\boldsymbol{\mu})$. We notice that for any given n the vectors of the basis could be linearly dependent and, as a consequence, $\text{rank}(D^n(\boldsymbol{\mu})) \leq M$. In particular, we observe that, $\forall \; \boldsymbol{\mu} \in D^{\boldsymbol{\mu}}$, $\text{rank}(D^n(\boldsymbol{\mu})) = \min\{P, M\}$, $\forall \; n = 1, ..., F-1$, where P and M are the dimensions of the space of parameters and the reduced subspace, respectively. In order to solve the system (29) according with the values of P and M we use two different techniques depending on the value assumed by the rank of the matrix $D^n(\boldsymbol{\mu})$. When $D^n(\boldsymbol{\mu})$ has full rank ($= M$) we perform an LU factorization of $D^n(\boldsymbol{\mu})$ at each time-step. Otherwise (i.e. $\text{rank}(D^n(\boldsymbol{\mu})) < M$), we use the generalized inverse $\widetilde{D}^n(\boldsymbol{\mu})$ of $D^n(\boldsymbol{\mu})$ obtained by means of the Singular Value Decomposition [21]. In practise, we find two orthogonal matrices $U(\boldsymbol{\mu}) \in \mathbb{R}^{N\times N}$ and $V(\boldsymbol{\mu}) \in \mathbb{R}^{M\times M}$ s.t.: $U(\boldsymbol{\mu})^T D^n(\boldsymbol{\mu})V(\boldsymbol{\mu}) = \Sigma(\boldsymbol{\mu})$, $n = 1,\dots,F$, where $\Sigma(\boldsymbol{\mu}) = \text{diag}(\sigma_1,...,\sigma_k,0,0...) \in \mathbb{R}^{N\times M}$, with $k = \text{rank}(D^n(\boldsymbol{\mu}))$ and σ_i, for $i = 1,...,k$, are the singular values of $D^n(\boldsymbol{\mu})$. By defining $\widetilde{\Sigma}(\boldsymbol{\mu}) = \text{diag}(\sigma_1^{-1},...,\sigma_k^{-1},0,0,...)$, $\widetilde{D}^n(\boldsymbol{\mu})$ reads $\widetilde{D}^n(\boldsymbol{\mu}) = V(\boldsymbol{\mu})\widetilde{\Sigma}(\boldsymbol{\mu})(U(\boldsymbol{\mu}))^T$. Finally, the solution weights $\mathbf{a}^n(\boldsymbol{\mu})$ are given by $\mathbf{a}^n(\boldsymbol{\mu}) = \widetilde{D}^n(\boldsymbol{\mu})\mathbf{F}^n(\boldsymbol{\mu})$. It is important to point out that the matrix $D^n(\boldsymbol{\mu})$ can be affinely decomposed, $\forall\, n = 0,...,F$, similarly to the PDEs case, as a combination of parameter independent matrices, say D_q^n, with parameter dependent weights, say $\Theta_q(\boldsymbol{\mu})$:

$$D^n(\boldsymbol{\mu}) = \sum_{q=1}^{Q} \Theta_q(\boldsymbol{\mu}) D_q^n, \qquad Q \in \mathbb{N}. \tag{30}$$

This allows to extend to the RB$_\mu$-DAEs method the *offline-online* decomposition, with a very rapid *online* step, being $M \ll N$. Hence, the computation of all the matrices A_q^n can be done *offline* and just once and, in correspondence of any new parameter, the *online* step consists only in the assembling of the whole matrix $D^n(\boldsymbol{\mu})$ and solving the reduced problem.
In the nonlinear case, i.e. for $\mathbf{g}(t,\mathbf{y}) \neq \mathbf{0}$, similar considerations could be done. For the solution of the nonlinear system (28), we use the Newton method [21], an iterative solver requiring at each iteration r the solution of the following linear system (which is rectangular in our case):

$$J_r(\boldsymbol{\mu})\mathbf{a}_{r+1}^{n+1}(\boldsymbol{\mu}) = \mathbf{b}_r(\boldsymbol{\mu}); \tag{31}$$

where $J_r(\boldsymbol{\mu}) \in \mathbb{R}^{N\times M}$ is defined as $J_r(\boldsymbol{\mu}) = A^{n+1}(\boldsymbol{\mu}) - \Delta t\ \nabla\mathbf{g}(t^{n+1}, Y^{n+1}\mathbf{a}_r^{n+1}(\boldsymbol{\mu}))$ and $\mathbf{b}_r(\boldsymbol{\mu}) \in \mathbb{R}^N$ is defined as $\mathbf{b}_r(\boldsymbol{\mu}) = -A^{n+1}(\boldsymbol{\mu})\mathbf{a}_r^{n+1}(\boldsymbol{\mu}) + \Delta t\ \mathbf{g}(t^{n+1}, Y^{n+1}\mathbf{a}_r^{n+1}(\boldsymbol{\mu})) - \mathbf{f}^{n+1}(\boldsymbol{\mu})$, with $A^n(\boldsymbol{\mu})$ and $\mathbf{f}^n(\boldsymbol{\mu})$ defined as in the linear case $\forall\, n = 1,...,F$. Once again, with a technique similar to the one adopted in the linear case, we multiply both sides of Eq.(31) by $(J_r^{n+1})^T(\boldsymbol{\mu})$ in order to get an $M \times M$ dimensional problem whose matrix can have rank less than M. Unfortunately, unlike the linear case where the rank was known a priori, it is not possible to find a relation for the rank of the matrix. Hence, we should check at each iteration of the Newton method the value of the rank and perform an LU factorization or compute the generalized inverse matrix. This implies that computational cost of the nonlinear case is greater than in the linear one. Also, only the linear part of the system can be affinely decomposed and, as a consequence, we should evaluate *online* the nonlinear term.

Remark 3.3 *An important issue of the RB$_\mu$-DAEs method consists in the choice of the vectors of the basis, since the projection onto the reduced space could lead to an ill-conditioned and, even worse, to an ill-posed systems. Thus, the choice of the parameters $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, ..., \boldsymbol{\mu}_M\}$, and hence of the basis, is crucial and will be discussed in Sec.3.3.2.*

3.3.2 The generation of the basis

We propose an adaptive procedure for the choice of the basis vectors in analogy with the RB$_\mu$-PDEs method (see [16]). Actually, different approaches can be implemented (e.g.

random generation from a Gaussian or uniform distribution [7]), but in spite of their easy implementation and rapid computation, experimental tests have revealed that they work only in the linear case. Hence, in the nonlinear case a generation approach similar to the one used for the RB$_\mu$-PDEs method is proposed and described in the following steps.
The basis parameters are chosen among a finite set of parameters, let say $\mathcal{D} \subset D^{\boldsymbol{\mu}}$. Let $\mathbf{y}_{h,j}^n = \mathbf{y}_h(t^n, \boldsymbol{\mu}_j)$ be the non-reduced solution associated with the j-th parameter in $\mathcal{D}$ at the time-step t^n and $\mathbf{y}_{R,j}^n$ be the reduced one in correspondence of the same parameter and time-step. Our procedure reads as follows:

1. we choose randomly $\boldsymbol{\mu}_1 \in \mathcal{D}$; then we compute $\mathbf{y}_{h,1}$ and set $Y = [\mathbf{y}_{h,1}]$;
2. for $m = 1, ..., M_{max}$, with $M_{max} \ll N$ to be fixed a priori:
   (a) $\forall\ \boldsymbol{\mu}_j \in \mathcal{D}$ such that $\boldsymbol{\mu}_j \neq \boldsymbol{\mu}_k,\ \ k = 1, ..., i-1$, compute: $e_j = \max\limits_{n=0,...,F} \dfrac{\|\mathbf{y}_{h,j}^n - \mathbf{y}_{R,j}^n\|}{\|\mathbf{y}_{h,j}^n\|}$,
   where $\mathbf{y}_{R,j}$ is the reduced solution computed with the actual matrix Y;
   (b) if $e_j < \varepsilon\ \forall j$, with ε fixed a priori, stop the procedure, else go to 2c;
   (c) set $\boldsymbol{\mu}_i = \text{argmax}_{S\setminus\{\boldsymbol{\mu}_1,...\boldsymbol{\mu}_{m-1}\}}\, e_j$;
   (d) set $Y = [\mathbf{y}_{h,1}\ ...\ \mathbf{y}_{h,m}]$.

This procedure is computationally expensive, since for each parameter in $\mathcal{D}$, a non-reduced solution is required. However, this procedure is performed only once *offline*. Also, this procedure can be improved in terms of accuracy adding a control on the rank of the matrix Y each time a new basis vector is introduced.
Once provided the reduced subspace S_R and the matrix Y, the CPU times associated with the computation of the reduced solution at the *online* step are significantly smaller than the ones required while computing the corresponding non-reduced solution. Nevertheless, it is important to point out that the real gain in the RB$_\mu$-DAEs method cannot be appreciated in the computation of a single solution, but rather in a many-query context: if the time required to perform the adaptive procedure and to compute the reduced solutions is smaller than the one required to compute the non-reduced solutions, then, the RB$_{\boldsymbol{\mu}}$-DAEs method is convenient (see also Sec.4).

3.4 A priori error estimate

In this Section we derive an *a priori* error estimate for the RB$_\mu$-DAEs method.
Typically, in the context of the time-discretization schemes an a priori estimate of the local error, i.e. the approximation error generated at each time-step, is derived in order to obtain the estimate of the global error, i.e. over the whole time interval; we apply the same approach also to the analysis of the RB$_\mu$-DAEs method. In view of the error estimate, we recall that the proposed DR approach is structured in two consequent steps: a discretization step and a reduction one (see Sec.3.3.1). Since we consider the non-reduced approximate solution as a “truth” approximation of the exact one, we are mostly interested in estimating the error committed in the reduction step of our method performed after a discretization in time. For this reason, we will only consider the estimation of the *reduction error*, which is defined as:

$$e_R(t^n, \boldsymbol{\mu}) := \|\mathbf{y}_h^n(\boldsymbol{\mu}) - \mathbf{y}_R^n(\boldsymbol{\mu})\|, \qquad n = 1, ..., F, \tag{32}$$

where $\mathbf{y}_h^n(\boldsymbol{\mu})$ and $\mathbf{y}_R^n(\boldsymbol{\mu})$ are respectively the non-reduced and reduced solutions at time t^n. In this Section we provide a bound for the global reduction error:

$$E_R(\boldsymbol{\mu}) := \max_{n=1,\ldots,F} e_R(t^n, \boldsymbol{\mu}). \tag{33}$$

For the sake of simplicity, we provide the estimate for the system (1) and we consider the parameter vector as a single parameter $\boldsymbol{\mu} = \mu \in \mathbb{R}$; also, the matrix $C \in \mathbb{R}^{N\times N}$ is defined as $C = I_N$, the identity matrix.
The local error $e_R(t^n, \mu)$ (32) is estimated in the following Theorem.

Theorem 3.1 *If $\mathbf{f}(t, \mathbf{y}; \mu)$ satisfies the assumptions of Sec.1 with Lipschitz constant L, $\forall \mu \in D^\mu$, then there exists a positive constant $k \in \mathbb{R}$ such that:*

$$e_R(t^{n+1}, \mu) \leq \overline{C}^n(\Delta t, \mu) \max_{i=1,\ldots,n} \left(k \inf_{\mathbf{w}\in\mathcal{S}_R(t^i)} \|\mathbf{y}_h^i(\mu) - \mathbf{w}\| \right) \qquad n = 1, \ldots, F, \tag{34}$$

where $\overline{C}^n(\Delta t, \mu) := [C_{LU}(\mu)\ C_Y(\mu)(1 + \Delta t\ L) + 1]\ \dfrac{(C_{LU}(\mu))^n - 1}{C_{LU}(\mu) - 1}$ for $n = 1, ..., F$ with $C_{LU}(\mu) := C_L\, C_U(\mu)$; the constant C_L depends on the Lipschitz constant L, while $C_U(\mu)$ and $C_Y(\mu)$ are defined as follows:

$$C_U(\mu) := \max_{n=1,\ldots,F} \|U^n(\mu)\|, \qquad C_Y(\mu) := \max_{n=1,\ldots,F} \|Y^n(\mu)\|, \tag{35}$$

being $U^n(\mu) \in \mathbb{R}^{N\times M}$ chosen such that $(U^n(\mu))^T Y^n(\mu) = I_M$ $\forall n = 1, \ldots, F$, with I_M the identity matrix [12].

Proof. For the sake of clearness, we omit the explicit dependence on the parameter for $\mathbf{y}_h^n$, $\mathbf{y}_R^n$ and the associated variables. If we apply to Eq.(1) the BE scheme we can rewrite system (1) at each time-step as follows:

$$\mathbf{F}(\mathbf{y}_h^{n+1}; \mu) := \mathbf{y}_h^{n+1} - \Delta t\, \mathbf{f}(t^{n+1}, \mathbf{y}_h^{n+1}; \mu) - \mathbf{y}_h^n = 0 \qquad n = 0, ..., F-1, \tag{36}$$

where $\mathbf{y}_h^n = \mathbf{y}_h^n(\mu)$. Also, if we apply the DR approach described in Sec.3.3 we can write the reduced system as:

$$\mathbf{y}_R^{n+1} - \Delta t\, \mathbf{f}(t^{n+1}, \mathbf{y}_R^{n+1}; \mu) - \mathbf{y}_R^n = 0 \qquad n = 0, ..., F-1; \tag{37}$$

where $\mathbf{y}_R^n = \mathbf{y}_R^n(\mu)$. Equivalently:

$$\mathbf{F}(\mathbf{y}_R^{n+1}; \mu) + (\mathbf{y}_h^n - \mathbf{y}_R^n) = 0 \qquad n = 0, ..., F-1. \tag{38}$$

We note that the Lipschitz-continuity of the function $\mathbf{f}$ is inherited by the function $\mathbf{F}$, which is endowed with a Lipschitz constant $(1 + \Delta t\, L)$ [7]. Eq.s (36) and (38) allow to write:

$$\mathbf{F}(\mathbf{y}_R^{n+1}; \mu) + \mathbf{y}_h^n - \mathbf{y}_R^n = \mathbf{F}(\mathbf{y}_h^{n+1}; \mu) = 0 \qquad n = 0, ..., F-1. \tag{39}$$

By introducing the projector $P^n \in \mathbb{R}^{N\times N}$ of $\mathbb{R}^N$ onto $S_R(t^n)$, defined as $P^n := Y^n(U^n)^T$ for $n = 1, ..., F$ and by subtracting $\mathbf{F}(P^{n+1}\mathbf{y}_h^{n+1}; \mu)$ from Eq.(39), we have:

$$-\mathbf{F}(P^{n+1}\mathbf{y}_h^{n+1}; \mu) + \mathbf{F}(\mathbf{y}_R^{n+1}; \mu) + \mathbf{y}_h^n - \mathbf{y}_R^n = -\mathbf{F}(P^{n+1}\mathbf{y}_h^{n+1}; \mu) + \mathbf{F}(\mathbf{y}_h^{n+1}; \mu) \qquad n = 0, ..., F-1. \tag{40}$$

Then, by multiplying both sides of Eq.(40) by $(U^{n+1})^T$ and by recalling the definition of the projector P^{n+1}, we can write:

$$\begin{aligned}(U^{n+1})^T[\mathbf{F}(Y^{n+1}(U^{n+1})^T\mathbf{y}_h^{n+1}; \mu) - \mathbf{F}(\mathbf{y}_R^{n+1}; \mu) - \mathbf{y}_h^n + \mathbf{y}_R^n] =& \\ (U^{n+1})^T[\mathbf{F}(P^{n+1}\mathbf{y}_h^{n+1}; \mu) - \mathbf{F}(\mathbf{y}_h^{n+1}; \mu)] \qquad n = 0, ..., F-1.&\end{aligned} \tag{41}$$

We introduce $\forall \mu \in D^{\mu}$ the vector $\mathbf{w}_h(\mu) \in \mathbb{R}^M$ defined as:

$$\mathbf{w}_h^{n+1}(\mu) := (U^{n+1})^T \mathbf{y}_h^{n+1}; \tag{42}$$

then, by substituting this last expression in Eq.(41) and writing $\mathbf{y}_R^n = Y^n \mathbf{a}^n(\mu)$, we get:

$$\begin{aligned} (U^{n+1})^T[\mathbf{F}(Y^{n+1}\mathbf{w}_h^{n+1}(\mu);\mu) - \mathbf{F}(Y^{n+1}\mathbf{a}^{n+1};\mu)] + (U^{n+1})^T(\mathbf{y}_R^n - \mathbf{y}_h^n) = \\ (U^{n+1})^T[\mathbf{F}(P^{n+1}\mathbf{y}_h^{n+1};\mu) - \mathbf{F}(\mathbf{y}_h^{n+1};\mu)]. \end{aligned} \tag{43}$$

We define the mapping $H : \mathbb{R}^M \to \mathbb{R}^M$ such that $\forall\, \mathbf{x} : (0,T) \to \mathbb{R}^M$ and $\forall \mu \in D^{\mu}$ we have:

$$H(\mathbf{x}(t^{n+1});\mu) := (U^{n+1})^T \mathbf{F}(Y^{n+1}\mathbf{x}(t^{n+1});\mu) \qquad n = 0, ..., F-1. \tag{44}$$

Since $\mathbf{F}(\mathbf{y};\mu)$ is a Lipschitz-continuous function, H inherits the same property with a Lipschitz constant $C_U\ C_Y\ (1 + \Delta t\ L)$ [7]. By recalling the definition of $\mathbf{w}_h^n$ given in Eq.(42), we define:

$$\begin{aligned} \mathbf{u}^n(\mu) := \mathbf{u}(t^n,\mu) = H(\mathbf{w}_h^n(\mu);\mu) \qquad n = 0, ..., F-1, \\ \mathbf{v}^n(\mu) := \mathbf{v}(t^n,\mu) = H(\mathbf{a}^n(\mu);\mu) \qquad n = 0, ..., F-1. \end{aligned} \tag{45}$$

Then, we rewrite Eq.(43) as:

$$\begin{aligned} \mathbf{u}^{n+1}(\mu) - \mathbf{v}^{n+1}(\mu) = (U^{n+1})^T\ e_R(t^n,\mu) \\ -(U^{n+1})^T[\mathbf{F}(P^{n+1}\mathbf{y}_h^{n+1};\mu) - \mathbf{F}(\mathbf{y}_h^{n+1};\mu)] \qquad n = 0, ..., F-1, \end{aligned} \tag{46}$$

and we find the following upper bound:

$$\begin{aligned} \|\mathbf{u}^{n+1}(\mu) - \mathbf{v}^{n+1}(\mu)\| \le \|U^{n+1}\|\ e_R(t^n,\mu) \\ +\|U^{n+1}\|\ \|\mathbf{F}(P^{n+1}\mathbf{y}_h^{n+1};\mu) - \mathbf{F}(\mathbf{y}_h^{n+1};\mu)\| \qquad n = 0, ..., F-1. \end{aligned} \tag{47}$$

Introducing in the previous expression the Lipschitz constant of $\mathbf{F}$, $(1 + \Delta t\ L)$, we write:

$$\|\mathbf{u}^{n+1}(\mu) - \mathbf{v}^{n+1}(\mu)\| \le \|U^{n+1}\|\ e_R(t^n,\mu) + (1+\Delta t\ L)\|U^{n+1}\|\ \|P^{n+1}\mathbf{y}_h^{n+1} - \mathbf{y}_h^{n+1}\| \qquad n = 0, ..., F-1. \tag{48}$$

Moreover, since H is continuously differentiable, we can write:

$$\begin{aligned} \|\mathbf{w}_h^{n+1}(\mu) - \mathbf{a}^{n+1}(\mu)\| = \|H^{-1}(\mathbf{u}^{n+1}(\mu);\mu) - H^{-1}(\mathbf{v}^{n+1}(\mu);\mu)\| \le \\ C_L\|\mathbf{u}^{n+1}(\mu) - \mathbf{v}^{n+1}(\mu)\| \qquad n = 0, ..., F-1, \end{aligned} \tag{49}$$

where C_L is the Lipschitz constant of H^{-1}; by using Eq.s (35) and (48) we can write:

$$\|\mathbf{w}_h^{n+1}(\mu) - \mathbf{a}^{n+1}(\mu)\| \le C_{LU}\ (1+\Delta t\ L)\|P^{n+1}\mathbf{y}_h^{n+1} - \mathbf{y}_h^{n+1}\| + C_{LU}\ e_R(t^n,\mu) \qquad n = 0, ..., F-1, \tag{50}$$

where $C_{LU} := C_L\ C_U \in \mathbb{R}$. Now we can find a bound for the local error $e_R(t^{n+1},\mu)$, which we write as:

$$e_R(t^{n+1},\mu) = \|\mathbf{y}_h^{n+1} - \mathbf{y}_R^{n+1} \pm P^{n+1}\mathbf{y}_h^{n+1}\| \qquad n = 0, ..., F-1, \tag{51}$$

by means of the triangular inequality:

$$e_R(t^{n+1},\mu) \le \|P^{n+1}\mathbf{y}_h^{n+1} - \mathbf{y}_h^{n+1}\| + \|P^{n+1}\mathbf{y}_h^{n+1} - \mathbf{y}_R^{n+1}\| \qquad n = 0, ..., F-1. \tag{52}$$

For every $n = 0, \ldots, F$, we have:

$$\begin{aligned} \|P^{n+1}\mathbf{y}_h^{n+1} - \mathbf{y}_R^{n+1}\| = \|Y^{n+1}(\mathbf{w}_h^{n+1}(\mu) - \mathbf{a}^{n+1}(\mu))\| \le \\ \|Y^{n+1}\|\ \|\mathbf{w}_h^{n+1}(\mu) - \mathbf{a}^{n+1}(\mu)\| \qquad n = 0, ..., F-1; \end{aligned} \tag{53}$$

then, by means of Eq.s (35) and (50) we have:

$$\|P^{n+1}\mathbf{y}_h^{n+1}-\mathbf{y}_R^{n+1}\| \leq C_Y\ C_{LU}\ (1+\Delta t\ L)\|P^{n+1}\mathbf{y}_h^{n+1}-\mathbf{y}_h^{n+1}\|+C_{LU}\ e_R(t^n,\mu) \qquad n=0,...,F-1. \tag{54}$$

By using Eq.(54), we can write:

$$e_R(t^{n+1},\mu) \leq [C_{LU}\ C_Y\ (1+\Delta t\ L)+1]\,\|P^{n+1}\mathbf{y}_h^{n+1}-\mathbf{y}_h^{n+1}\|+C_{LU}\ e_R(t^n,\mu) \qquad n=0,...,F-1. \tag{55}$$

If we define:

$$\epsilon^{n+1}(\mu) := [C_{LU}\ C_Y(1+\Delta t\ L)+1]\,\|P^{n+1}\mathbf{y}_h^{n+1}-\mathbf{y}_h^{n+1}\| \qquad n=0,...,F-1, \tag{56}$$

we can write the following recursive inequality:

$$e_R(t^{n+1},\mu) \leq \epsilon^{n+1}(\mu)+C_{LU}\ e_R(t^n,\mu) \qquad n=0,...,F-1. \tag{57}$$

By using a recursive strategy over the time (similar to the one presented in [17]) and by recalling that $e_R(t^0,\mu)=\mathbf{y}_h^0-Y^0\mathbf{a}^0=0$, we can write:

$$\begin{aligned} e_R(t^n,\mu) \leq \sum_{i=0}^{n-1}\epsilon^{n-i-1}(\mu)(C_{LU})^i \leq \max_{i=0,...,n-1}\epsilon^i(\mu)\sum_{i=0}^{n-1}(C_{LU})^i = \\ \max_{i=1,...,n-1}\epsilon^i(\mu)\tfrac{(C_{LU})^n-1}{C_{LU}-1} \qquad n=1,...,F; \end{aligned} \tag{58}$$

then, by recalling Eq.(56), it follows:

$$e_R(t^n,\mu) \leq \overline{C}^n(\Delta t)\max_{i=1,\dots,n}\|P^i\mathbf{y}_h^i(\mu)-\mathbf{y}_h^i(\mu)\| \qquad n=1,\dots,F. \tag{59}$$

where $\overline{C}^n(\Delta t)=\overline{C}^n(\Delta t,\mu)$. In order to make effective the estimate (59), we need to bound the vector norm $\|P^i\mathbf{y}_h^i(\mu)-\mathbf{y}_h^i(\mu)\|$, for $i=1,\dots,n$ and $n=1,\dots,F$. With this aim, let us recall a standard result on projectors (see [17]) for which, given a subspace $S\subset\mathbb{R}^N$ and a projector B, there exists a positive constant c_B s.t. $\|B\mathbf{x}-\mathbf{x}\|\leq c_B\inf_{\mathbf{w}\in S}\|\mathbf{x}-\mathbf{w}\|\ \forall\mathbf{x}\in\mathbb{R}^N$. By applying this property to our problem and by introducing the positive constant $k\in\mathbb{R}$, the result (34) follows. □

We need now a bound for the following term in Eq.(34):

$$\inf_{\mathbf{w}\in\mathcal{S}_R(t^i)}\|\mathbf{y}_h^i(\mu)-\mathbf{w}\| \qquad \forall i=1,...,F. \tag{60}$$

With this aim, we introduce the following Definitions and an associated Proposition which will be the basis of the main result of this Section.

Definition 3.1 *Let us suppose that for $j=1,...,J$ the parameters λ_j are distinct and evenly distributed and that $\mathbf{x}(\lambda_j)\in\mathbb{R}^N$ is known for $j=1,...,J$; the Lagrangian subspace is defined as:*

$$S_L := span\,\{\mathbf{u}^j \mid \mathbf{u}^j=\mathbf{x}(\lambda_j),\ j=1,...,J\}, \tag{61}$$

which is the set of linear combinations of J points in $\mathbb{R}^N$.

Definition 3.2 *Given a scalar parameter $\lambda\in\mathbb{R}$, the Lagrangian interpolating polynomial [21] is defined by:*

$$\mathbf{x}_L(\lambda)=U\ \mathbf{w}_L(\lambda); \tag{62}$$

where

$$[\mathbf{w}_L(\lambda)]_j := \prod_{k=1,k\neq j}^{J}\frac{\lambda-\lambda_k}{\lambda_j-\lambda_k}. \tag{63}$$

and $U:=\left[\mathbf{u}^1\ \mathbf{u}^2\ ...\ \mathbf{u}^M\right]$, being $\mathbf{u}^j$, for $j=1,...,J$, members of the basis of $\mathcal{S}_L$.

Remark 3.4 *The Lagrangian subspace S_L coincides with the reduced subspace $S_R(t^n)$ at each time-step with $J = M$: in fact the basis vectors $\mathbf{u}^j$, $j = 1, ..., M$, assume the form of $\mathbf{y}(t^n, \mu_j)$ and, hence, U plays the role of the matrix Y^n. The Lagrangian interpolating polynomial associated with a particular choice of the parameter $\lambda = \mu \in \mathbb{R}$ is a representation of the reduced solution (it is in fact a combination of the basis vectors). This allows the following result (see [17] or [21] for the proof) on the Lagrangian polynomial which will be useful in the estimate of an upper bound for the expression* (60).

Proposition 3.1 *Let $\mathbf{x} : \mathbb{R} \to \mathbb{R}^N$ be a continuous function and let S_L be the Lagrangian subspace of dimension J, defined in Eq.*(61)*. If $\mathbf{x}(\lambda) \in \mathcal{C}^{J+1}(\mathbb{R})$, the space of $(J+1)$ times differentiable functions, for each parameter $\lambda \in \mathbb{R}$ there exists another parameter $\widetilde{\lambda} \in \mathbb{R}$ s.t. the approximation error of the Lagrangian interpolating polynomial could be estimated as:*

$$\mathbf{x}_i(\lambda) - \mathbf{x}_{L,i}(\lambda) = \frac{\prod_{k=0,k\neq j}^{J} (\lambda - \lambda_k)}{(J+1)!} \frac{d^{(J+1)}\mathbf{x}_i(\widetilde{\lambda})}{d\lambda^{J+1}} \qquad \forall\, i = 1, ..., N. \tag{64}$$

Hence, we have the following upper bound of the Lagrangian approximation error:

$$\|\mathbf{x}(\lambda) - \mathbf{x}_L(\lambda)\| \leq \frac{2^{J+1}}{(J+1)!} \left\| \frac{d^{(J+1)}\mathbf{x}}{d\lambda^{J+1}} \right\| \frac{|D|}{J}, \tag{65}$$

where $|D| = \max_{1 \leq i,j \leq J,\ i\neq j} |\lambda_i - \lambda_j|$.

If we combine the results shown in Theorem 3.1 and Proposition 3.1 we can find the following upper bound for the reduction-error.

Theorem 3.2 *Let $e_R(t^n, \mu)$ be the reduction error defined in Eq.*(32)*. Under the assumptions of Theorem 3.1 and by keeping the same notation, $e_R(t^n, \mu)$ satisfies the following inequality:*

$$e_R(t^{n+1}, \mu) \leq \overline{C}^n(\Delta t, \mu)\; k\; |D|\; \frac{2^{M+1}}{M(M+1)!} \left(\max_{i=1,\dots,n} d^i_{M+1}(\mu) \right) \qquad n = 0, \dots, F-1; \tag{66}$$

where $d^i_{M+1}(\mu) := \dfrac{d^{(M+1)}\mathbf{y}^i_h}{d\mu^{M+1}}$ and $|D| = \max_{1\leq i,j\leq J,\ i\neq j} |\mu_i - \mu_j|$ depends on the selected parameters for the generation of the basis.

Proof. For the sake of simplicity we omit the parameter dependence for all variable depending on $\mathbf{y}_h$ and $\mathbf{y}_R$. By recalling Remark 3.4 we can apply the result given in Proposition 3.1 to Eq.(34), thus obtaining:

$$e_R(t^{n+1}, \mu) \leq \overline{C}^n(\Delta t, \mu)\; k \max_{i=1,\dots,n} \left\| \frac{d^{(M+1)}\mathbf{y}_h}{d\mu^{M+1}} \right\| \frac{2^{M+1}}{(M+1)!} \frac{|D|}{M}. \tag{67}$$

The definition of $d^i_{M+1}(\mu)$ leads to the thesis. □

By using the results of Theorem 3.2 concerning the local reduction error, it is possible to estimate the global reduction error $E_R(\mu)$ as reported in the following Theorem.

Theorem 3.3 *If the assumptions in Theorem 3.2 hold for the global reduction error (Eq.(33)), the following a priori estimate holds:*

$$E_R(\mu) \leq \widetilde{C}(\Delta t, \mu)\; d_{max}(\mu) \frac{2^{M+1}}{(M+1)!} \frac{|D|}{M}, \tag{68}$$

where $d_{max}(\mu) := \max\limits_{n=1,\ldots,F} d^n_{M+1}(\mu)$ *and* $\widetilde{C}(\Delta t, \mu) := \max\limits_{n=1,\ldots,F} \overline{C}^n(\Delta t, \mu)$.

Proof. The thesis follows from the inequality $(C_{LU})^n \leq (C_{LU})^F$, $\forall n = 1, ..., F$, and from the definition of d_{max}. $\square$

Theorem 3.3 ensures that $E_R(\mu) \to 0$ as $M \to \infty$, $\forall \mu \in D^\mu \subset \mathbb{R}$ because of the factorial term and the fact that $|D| \to 0$ as $M \to \infty$; when the number of basis vectors tends to infinity, S_R covers the whole $\mathbb{R}^N$, i.e the reduced solution $\mathbf{y}_R(t^n, \mu)$ is actually a member of the basis itself and $e_R(t^n, \mu) = 0$ at each time-step.

4 Numerical tests

In this Section we provide some numerical tests for problem (1) with both a linear and nonlinear parameter dependence. For the numerical solution we use the proposed RB$_\mu$-DAEs and, for the sake of comparison, the POD$_\mu$ method presented in Sec.2.1. We analyze the performances of the RB$_\mu$-DAEs method from two standpoints: the error behavior and the computational cost.
Numerical tests for the linear problem are reported in [7]; for the linear formulation of the problem the RB$_\mu$-DAEs method gives optimal results: the matrix of the reduced system allows an affine decomposition similar to the one presented in Sec.3.2 for the PDEs case. In this way, most of the computational effort resides in the *offline* step; the *online* step only requires the assembling of the matrix and the solution of an M dimensional system, which is actually very fast. Such decomposition is not possible when dealing with parametric nonlinearity. Even larger savings in computational costs are allowed by the reduced method in the nonlinear case.

4.1 Test1: linear parameter dependence

In this Section we compare the numerical results obtained by using the RB$_\mu$-DAEs and the POD$_\mu$ methods on problem (1); with the choice of $\mathbf{f}(t, y; \boldsymbol{\mu}) = -R(\boldsymbol{\mu})\mathbf{y} + \mathbf{g}(t, \mathbf{y}; \boldsymbol{\mu})$, this is a nonlinear problem with a nonlinear parameter dependence. Moreover, we choose a scalar parameter $\boldsymbol{\mu} = \mu \in D^\mu \subset \mathbb{R}$, which represents the resistance associated with each resistor of the chain.

4.1.1 Implementation

Let us report some details on the implementation of the methods.
Both methods require the computation of some non-reduced solutions for the assembling of the projection matrices. Those solutions are obtained by solving problem (1), with $N = 200$, using the BE method for the time discretization with a time-step $\Delta t = 5 \cdot 10^{-10}$ *sec* and the Newton method for the solution of the nonlinear system at each time-step with a tolerance of $5 \cdot 10^{-5}$ on the residual.
In the implementation of the adaptive procedure and for building the cubic spline we use $D = 40$ non-reduced solutions computed in correspondence of D parameters randomly

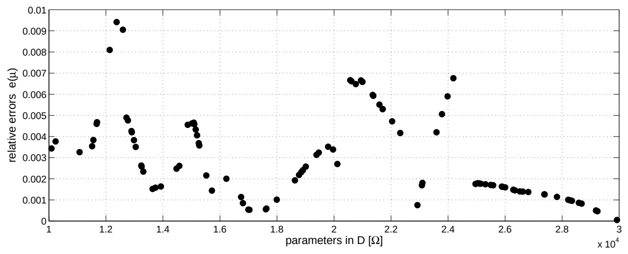


Figure 2: Test 1: relative errors $e(\mu)$ vs $\mu \in D$ in correspondence of $M = 8$.

chosen in the set $D^{\boldsymbol{\mu}} = [10^4, 3 \cdot 10^4]$ Ω. Both reduced solutions are computed using the BE scheme with the same time-step used for the non-reduced system and the Newton scheme for the nonlinear system, now with a tolerance of $5 \cdot 10^{-4}$ on the residual. These reduced solutions are computed in correspondence of a set $\overline{D}^{\boldsymbol{\mu}}$ of 100 new parameters randomly chosen from the same set $D^{\boldsymbol{\mu}}$.

Remark 4.1 *In the implementation of the adaptive procedure for the RB$_\mu$-DAEs method we add a control on the rank of the current Y: we allow a solution $\mathbf{y}_h$ to concur to the new basis vectors only if at each time-step the matrix Y is not rank deficient.*

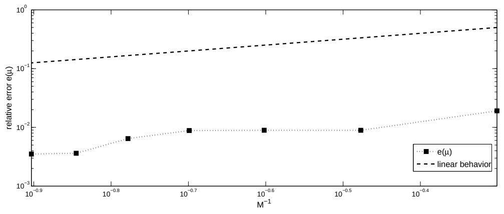
4.1.2 Error behavior

In this Section we discuss the error behavior associated with the RB$_\mu$-DAEs method, in particular, we analyze the error over the set $\overline{D}^{\boldsymbol{\mu}}$ with a fixed dimension of the basis and, then, the error behavior as the number of basis vectors changes.
For testing these properties we perform the adaptive procedure over a set of D parameters and we compute the reduced solutions over the set $\overline{D}^{\boldsymbol{\mu}}$. The relative error reads:

$$e(\mu) = \max_{n=1,\dots,F} \frac{e_R(t^n,\mu)}{\|\mathbf{y}_h(t^n,\mu)\|}, \tag{69}$$

where $e_R(t^n,\mu)$ is the local error defined in Eq. (32). Fig.2 displays the relative errors vs the set of parameters $\overline{D}^{\boldsymbol{\mu}}$. The reduced solutions are computed using a projection matrix formed by $M = 8$ basis vectors: this error behavior can be explained by the adaptive procedure, as the current parameter is close to a basis parameter the error goes to zero since a great part of the information concerning the exact solution is stored in the projection matrix. On the other hand, a larger error occurs when the current parameter is far from the ones in the basis. In Fig.3 we report the error behavior in correspondence of two different parameter values as M changes. As expected from Theorem 3.3, as M increases, the error decreases almost linearly: this is due to the low number of basis vectors M used; however we expect an improvement of the convergence rate as M tends to infinity.

Remark 4.2 *Simulations performed using a vector parameter with a linear parameter dependence highlight a similar error behavior and the same conclusions can be drawn; for more details on this issue see [7].*



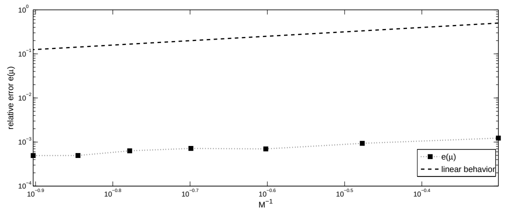


Figure 3: Test 1: relative errors vs M^{-1} in correspondence of $\mu = 19479\ \Omega$ (left) and $\mu = 29196\ \Omega$ (right) in a logarithmic scale; a linear reference behavior is also reported with a dotted line.

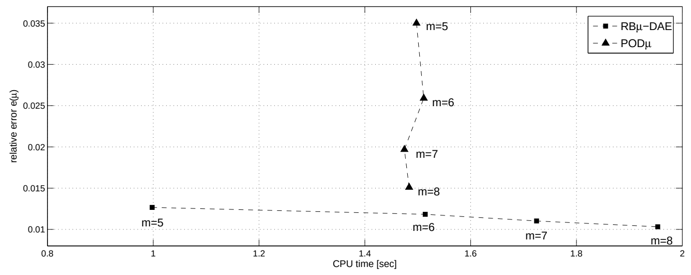


Figure 4: Test 1: relative errors $e(\mu)$ vs CPU time [sec] in correspondence of different numbers m of basis vectors.

4.1.3 Comparison with the POD$_\mu$ method

In this Section we compare the numerical results obtained on the same problem by the RB$_\mu$-DAEs method and the POD$_\mu$ method. As already discussed in Sec.4.1.1 the same set of parameters is used for the generation of both the projection matrices; the results reported refer to the set $\overline{D}^{\boldsymbol{\mu}}$. In this Section and in the following ones we will use m to indicate the number of basis vectors both for the RB$_\mu$-DAEs (M) and POD$_\mu$ (K) methods. Fig.4 displays for different values of m the relative errors versus the CPU time. In particular, each dot in the plot represents the mean value of both the displayed variables over $\overline{D}^{\boldsymbol{\mu}}$. These results show that the two methods have a completely different behavior. In the RB$_\mu$-DAEs method the computational cost increases with m, while in the POD$_\mu$ method it is almost constant. This can be explained by the condition number of the reduced systems; in the first method as m increases the condition number of the linear system (at each Newton iteration) increases and more Newton iterations are required to reach the prescribed tolerance. This does not occur with the POD$_\mu$ method where the condition number is almost fixed. The RB$_\mu$-DAEs method requires less computational time only for $m = M < 6$ basis vectors; besides that, also the relative error should be taken into account. Increasing the number of basis vectors, both methods show improvements in the accuracy: in the RB$_\mu$-DAEs method we have a slower error decrease but the level of accuracy is always better than the POD$_\mu$ one. In Fig.s 5 and 6 we report in correspondence of a fixed value of m the relative error and the computational times for both the methods. In these plots the different behavior is still evident: we can notice again the "jumping" behavior of the RB$_\mu$-DAEs method (already explained in the previous paragraph) and the smooth behavior of the POD$_\mu$ one. This can be explained by the theory at the basis of the method itself: as the

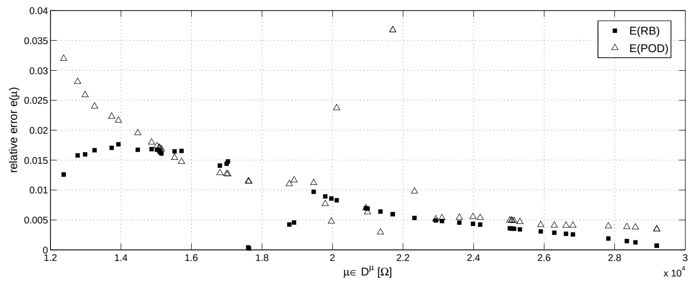


Figure 5: Test 1: comparison of the RB_μ-DAEs method and the POD_μ one; relative errors $e(\mu)$ vs $\mu \in D$ are reported in correspondence of $m = 5$.

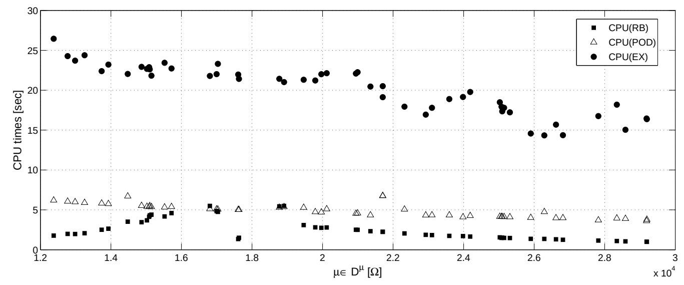


Figure 6: Test 1: comparison of the RB_μ-DAEs method and the POD_μ one; CPU times [sec] vs $\mu \in D$ are reported in correspondence of $m = 5$.

parameter becomes larger (i.e. larger resistance), then the transient of the circuit is dominated by a linear dynamics which requires few basis vectors (associated with the energy of the system) to be described.

4.2 Test 2: nonlinear parameter dependence

In this Section we consider the nonlinear parameter dependence; we show results obtained solving the problem (1) with $\mathbf{f}(t, \mathbf{y}; \boldsymbol{\mu}) = -R\mathbf{y} + \mathbf{g}(t, \mathbf{y}; \boldsymbol{\mu})$. The scalar parameter $\boldsymbol{\mu} = \mu \in D^{\boldsymbol{\mu}} \subset \mathbb{R}$ represents the thermal voltage of the diode whose constitutive law has an exponential dependence on this parameter; this fact makes the basis generation more complex: the condition number of the problem can increase, thus requiring higher computational costs. The implementation techniques for the solution of this problem are the same described in Sec.4.1.1.

4.2.1 Error behavior and comparison with the POD method

Let us consider the error behavior of the RB_μ-DAEs method and we compare its performances w.r.t. the POD_μ ones for a nonlinear parameter dependence. We perform the adaptive procedure for the basis generation over a set of $D = 40$ parameters randomly chosen in the set $D^{\boldsymbol{\mu}} = [0.02, 0.05]$ V. Fig.7 displays the relative errors, defined in Eq.(69), as the parameter μ changes; we can notice again the typical "jumping" behavior: the error tends to zero when the current parameter is close to a basis one. In Fig.8

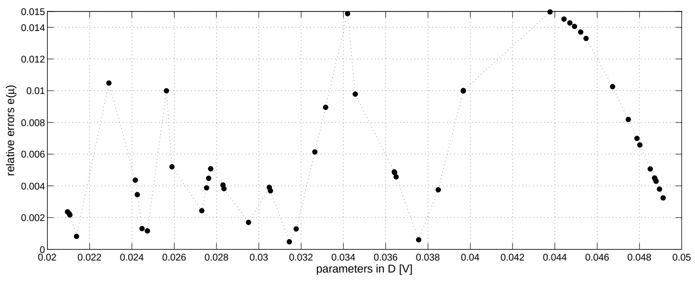


Figure 7: Test 2: relative errors $e(\mu)$ vs $\mu \in D$ in correspondence of $M = 8$.

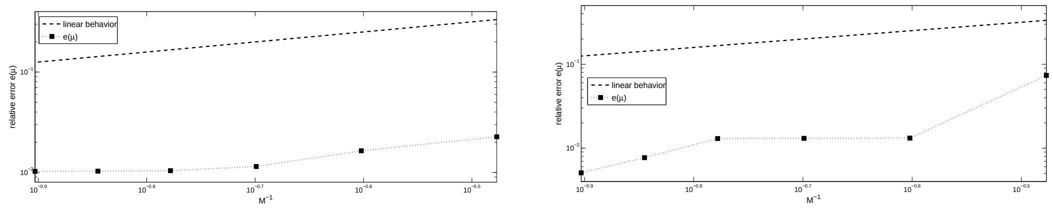


Figure 8: Test 2: relative errors vs M^{-1} in correspondence of $\mu = 0.046729$ V (left) and $\mu = 0.027724$ V (right) in a logarithmic scale; a linear reference behavior is also reported with a dotted line.

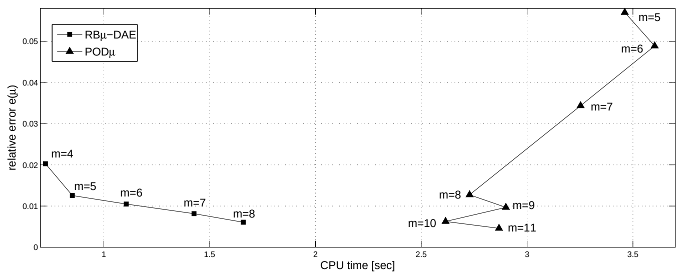


Figure 9: Test 2: relative errors $e(\mu)$ vs CPU time [sec] in correspondence of different numbers m of basis vectors.

we report the relative error as M changes for a fixed value of the parameter: again, we have a monotone decreasing of the error ad M increases.
Regarding the POD$_\mu$ method: for the basis generation we use the same set of parameters used in the adaptive procedure and we test both the methods on the same set $\overline{D}^{\boldsymbol{\mu}}$ of 50 new parameters chosen randomly in $D^{\boldsymbol{\mu}}$. Fig.9 shows the relative errors vs the CPU time in correspondence of different numbers m of basis vectors. In the nonlinear case the CPU time required by the RB$_\mu$-DAEs method for the solution of the reduced system is always lower than the POD$_\mu$ one. Again, the curve corresponding to the POD$_\mu$ method in the plot has a vertical displacement, also, the CPU time can decrease with m since the conditioning of the system can get better when adding some new basis vectors. We report results obtained for $m = M = 4, ..., 8$ for the RB$_\mu$-DAEs method: an $m = M = 9$ vector basis is hard to be built due to instabilities caused by the bad conditioning of

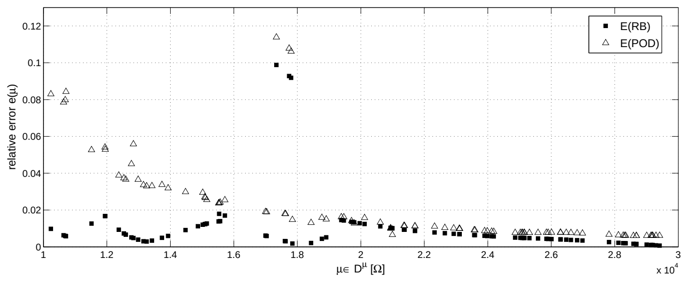


Figure 10: Test 2: comparison of the RB$_\mu$-DAEs method and the POD$_\mu$ one; relative errors $e(\mu)$ vs $\mu \in D$ are reported in correspondence of $m = 5$.

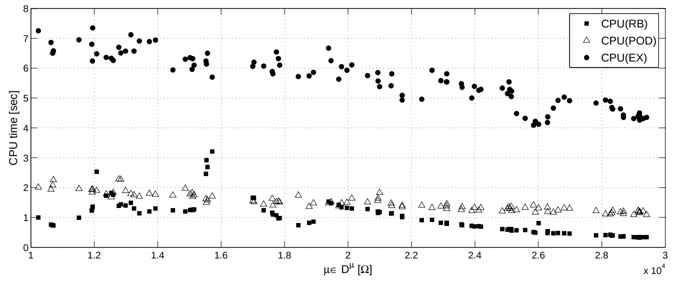


Figure 11: Test 2: comparison of the RB$_\mu$-DAEs method and the POD$_\mu$ one; CPU times [sec] vs $\mu \in D$ are reported in correspondence of $m = 5$.

the reduced problem; for the POD$_\mu$ method we report results for $m = K = 5, ..., 11$ since the value of the relative errors and the CPU times become almost stationary for $m = K > 11$. The relative error achieved with the POD$_\mu$ method becomes competitive with the RB$_\mu$-DAEs one only in correspondence of about $m = K = 10$ basis vectors for the POD$_\mu$ method. Fig.s 10 and 11 show a comparison of the two methods in terms of relative errors and the computational times respectively: we can draw similar conclusions as those reported for the linear parameter dependence; also, in this case the computational saving of the RB$_\mu$-DAEs method is definitely lower.

5 Conclusions

In this work we have proposed a Reduced Basis method (RB$_\mu$-DAEs) for the solution of parametrized systems of DAEs. In particular, we have considered a Microelectronic application, even if the method is applicable to other Engineering problems as well. We have proposed an *a priori* estimate for the error associated with the solution of parametrized problems via the RB$_\mu$-DAEs method, thus highlighting that the error rapidly decreases as the number of basis vectors used for the reduction increases. Numerical tests reveal that the proposed RB$_\mu$-DAEs method, in comparison with a POD approach for parametrized DAEs (POD$_\mu$), is efficient for the solution of such problems. This is more evident in the case of nonlinear problems with nonlinear parameter dependence; e.g. for the numerical test discussed in this work, the computational costs of the online step associated with the RB$_\mu$-DAEs method are even more than three times lower than those of the POD$_\mu$ approach, for a prescribed error level.

Acknowledgements

The third Author (A.Q.) is indebted with E.J.W. ter Maten and W.H.A. Schilders (both from NXP Semiconductors Research and Technical University, Eindhoven) for many useful discussions at the early stage of this research project. The Authors acknowledge G. Rozza (École Politechnique Fédérale de Lausanne, Switzerland) for the suggestions and helpful discussions.

References

[1] P. Astrid. Reduction of process simulation models: a proper orthogonal decomposition approach. PhD Thesis, Eindhoven University of Technology, 2004.

[2] P. Astrid, A. Verhoeven. Application of Least Square MPE technique in the reduced proper modeling of electrical circuits. In *Proceedings of the International Symposium on MTNS*, Kyoto, Japan, July 24-28 (2006).

[3] P. Astrid, S. Weiland, K. Willcox, A.C.P.M. Backx. Missing point estimation in models described by proper orthogonal decomposition. In *Proceedings of the 43rd IEEE Conference on Decision and Control*, Paradise Island, Bahamas (2004).

[4] B.N. Bond, L. Daniel. Parameterized model order reduction of nonlinear dynamical systems. In *Proceedings of the IEEE Conference on computer-aided design*, San Jose, California, November (2005).

[5] T. Bui-Thanh, M. Damodaran, K. Willcox. Proper orthogonal decomposition extensions for parametric applications in compressible aerodynamics. In *Proceedings of 21st AIAA Applied Aerodynamics Conference*, Orlando, Florida, June 23-26 (2003).

[6] C. De Boor. *A Practical Guide to Splines*. Springer-Verlag, New York, 1978.

[7] M. D'Elia. Reduced-basis method and Model Order Reduction for initial value problems of differential algebraic equations. Master Thesis, Politecnico di Milano, 2007. `http://mox.polimi.it`

[8] R.W. Freund. Model reduction methods based on Krylov subspaces. *Acta Numerica* **12**, 267-319 (2003).

[9] R.W. Freund. Reduced-order modeling techniques based on Krylov subspaces and their use in circuit simulation. *Applied and Computational Control, Signals and Circuits* **1**, 435-498, Birkhäuser, Boston (1999).

[10] E.J. Grimme. Krylov projection methods for model reduction. PhD Thesis, University of Illinois at Urbana-Champaigne, 1994.

[11] S. Lall, J. E. Marsden, S. Glavaski. A subspace approach to balanced truncation for model reduction of nonlinear systems. *International Journal of robust and nonlinear control* **12**, 519-535 (2002).

[12] P. Lancaster. *Theory of Matrices*. Academic Press, New York, 1984.

[13] M. Lin Lee. Estimation of the error in the reduced basis method solution of DAE systems. *SIAM Journal of Numerical Analysis* **28**(2), 512-528 (1991).

[14] D.A. Nagy. Model representation of geometrically nonlinear behavior by the finite element method. *Computers and Structures* **10**, 683-688 (1977).

[15] A.K. Noor, J.M. Peters. Reduced basis technique for nonlinear analysis of structures. *American Institute of Aeronautics and Astronautics Journal*, **18**, 455-462 (1980).

[16] A.T. Patera, G. Rozza. *Reduced Basis Approximation and A Posteriori Error Estimation for Parametrized Partial Differential Equations.* Version 1.0, Copyright MIT 2006-2007. To appear in MIT Pappalardo Graduate Monographs in Mechanical Engineering. `http://augustine.mit.edu/`

[17] T.A. Porshing. Estimation of the error in the reduced basis method solution of nonlinear Equations. *Mathematics of Computation* **45**(172), 487-496 (1985).

[18] T.A. Porsching, M. Lin Lee. The reduced basis method for initial value problems. *SIAM Journal of Numerical Analysis* **24**, 1277-1287 (1987).

[19] C. Prud'homme, D. Rovas, K. Veroy, Y. Maday, A.T. Patera, G. Turinici. Reliable real-time solution of parametrized partial differential equations: reducedbasis output bound methods. *Journal of Fluids Engineering* 2002; **124**(1), 70-80.

[20] A. Quarteroni, G. Rozza, L. Dedè, A. Quaini. Numerical approximation of a control problem for advection-diffusion processes. In *System modeling and optimization, IFIP Int. Fed. Inf. Process.* **199**, 261-273, Springer, New York (2006).

[21] A. Quarteroni, R. Sacco, F. Saleri. *Numerical Mathematics.* Springer-Verlag, Berlin, 2006.

[22] M.J. Rewienski. A Trajectory piecewise-linear approach to model order reduction of nonlinear dynamical systems. PhD Thesis, Massachusetts Institute of Technology, 2003.

[23] G. Rozza. Reduced-basis methods for elliptic equations in sub-domains with a posteriori error bounds and adaptivity. *Journal of Applied and Numerical Mathematics* **55**(4), 403-424 (2005).

[24] A. Verhoeven, M. Striebel, J. Rommes, J. ter Maten, T. Bechtold. Proper Orthogonal Decomposition Model Order Reduction of nonlinear IC models. *TUe CASA Report*, 2008. `http://www.win.tue.nl`

[25] A. Verhoeven, J. ter Maten, M. Striebel, R. Mattheij. Model Order Reduction for nonlinear IC models. *TUe CASA Report*, 2007.
`http://www.win.tue.nl`

[26] T. Voss, A. Verhoeven, T. Bechtold, J. ter Maten. Model Order Reduction for Nonlinear DAE in Circuit Simulation. *TUe CASA Report*, 2006.
`http://www.win.tue.nl`