1 REDUCED ORDER MODEL

For any unseen parameter value, our aim is to be able to assemble and solve a smaller algebraic problem, and yet to obtain a solution close enough to that of the Full Order Model. In order to do so, first we need to obtain certain algebraic structures which capture the essence of the problem at hand.

We do so by sampling the original problem at certain parameter values and processing snaphsots of the solution and the operators, obtained from the solution of the FOM problem. This is called the *offline phase*. Later on, we exploit these static structures to build reduced operators which capture the dynamics of the problem sufficiently well, solve a smaller algebraic system and then recover the solution in the original mesh variables, in order to postprocess it. This is called the *online phase*.

We will use the Reduced Basis Method (RB) to construct an ad-hoc problem-based basis to represent our ROM; the Discrete Empirical Interpolation Method (DEIM) and its matrix version (MDEIM) to build suitable approximations of the algebraic operators involved.

The continuous reduced problem formulation is skipped since it will not be used. Therefore, we jump directly into the discrete problem. We recall that we are focused at reducing the problem for the homogeneous component $\hat{u}(x)$ of our solution, that is, for the weak formulation given by the system of equations (**??**).

1.1 A Naive Galerkin Approach

We have included in the title the word *naive* because we are going to define the reduction problem in a very blunt way, where many ineffiencies will show up. We do so to motivate the operator reduction procedures we will include, especially regarding the operators.

In the reduced context, we use a finite space $V_N \subset V_h$, where we can represent the solution as the linear combination of a set of orthonormal[1] problem-based basis functions $\psi_i(x)$,

$$\hat{u}_N(x) = \sum_j^N \hat{u}_{N_j} \psi_j(x). \tag{1}$$

These basis functions $\psi_i(x)$ have global support, since their goal is to capture the problem dynamics. This is why we usually expect or desire to have $N \ll N_h$, since we want to reduce the number of basis functions we need to represent our solution.

To maintain focus and in favor of generality, let us assume at this point that the global basis functions are given, and that they are zero at the boundary. We will give details on how to obtain them later on.

1.1.1 Reduced Space Projection

Since we can represent any function in terms of our FE basis functions $\varphi_i(x)$, we have a linear mapping between the problem-based functions $\psi_i(x)$ and the nodal basis functions. This allows us to establish the following relation between the problem solution in the reduced and original spaces,

$$\hat{\mathbf{u}}_h = \mathbb{V}\hat{\mathbf{u}}_N. \tag{2}$$

The entries of the $\mathbb{V}$ matrix represent the coefficients of the global basis representation in the FE basis,

$$\psi_i(x) = \sum_j [\mathbb{V}]_{ji} \varphi_j(x). \tag{3}$$

1.1.2 Discrete Reduced Problem Assembly

In theory, to assemble the reduced problem operators, one could actually compute the inner products defined in Equations (**??**) with these new basis functions $\psi_i(x)$. In practice, it is more convenient to project the algebraic FOM operators with matrix-matrix and matrix-vector products into the reduced space,

$$\mathbf{X}_N^{n+1} = \mathbb{V}^T \mathbf{X}_h^{n+1} \mathbb{V}, \tag{4a}$$

$$\mathbf{F}_N^{n+1} = \mathbb{V}^T \mathbf{F}_h^{n+1}, \tag{4b}$$

1. If they were not, we can always make them so via Gram-Schmidt.

where $\mathbf{X}_h$ and $\mathbf{F}_h$ stand for each of the FOM operators (matrix and vector respectively). The assembly of matrices based on a FE basis can be easily done in parallel due to their local support, whereas the integration of functions with global support is not necessarily computationally efficient.

By doing so, we find the following ROM for the time evolution problem,

$$\begin{aligned} m_{\text{BDF}}\mathbf{M}_N^{n+1}\hat{\mathbf{u}}_N^{n+1} + \Delta t\mathbf{C}_N^{n+1}\hat{\mathbf{u}}_N^{n+1} + \Delta t\mathbf{A}_N^{n+1}\hat{\mathbf{u}}_N^{n+1} \\ +\Delta t\hat{\mathbf{N}}_N^{n+1}\hat{\mathbf{u}}_N^{n+1} + \Delta t\left[\mathbf{N}_N^{n+1}\left(\hat{\mathbf{u}}_N^{n}\right)\right]\hat{\mathbf{u}}_N^{n+1} \\ = \mathbf{F}_{\hat{\mathbf{u}}_N}^{n} + \Delta t\mathbf{F}_{g,N}^{n+1}, \end{aligned} \tag{5a}$$

$$\hat{\mathbf{u}}_N^0 = \hat{\mathbf{u}}_{N,0}. \tag{5b}$$

If we collect terms and factor out the unknowns we get a linear system, this time in the reduced space, to be solved for each timestep to advance the solution,

$$\mathbf{K}_N^{n+1}\hat{\mathbf{u}}_N^{n+1} = \mathbf{b}_N^{n+1}, \tag{6a}$$

$$\hat{\mathbf{u}}_N^0 = \hat{\mathbf{u}}_{N,0}, \tag{6b}$$

$$\begin{aligned} \mathbf{K}_N^{n+1} = \mathbf{M}_h^{n+1} + \Delta t\left[\mathbf{A}_h^{n+1} + \mathbf{B}_h^{n+1}\right. \\ \left.+\hat{\mathbf{N}}_h^{n+1} + \mathbf{N}_h^{n+1}\left(\hat{\mathbf{u}}_N^{n}\right)\right], \end{aligned} \tag{6c}$$

$$\mathbf{b}_N^{n+1} = \mathbf{F}_{\hat{\mathbf{u}}_N}^{n} + \Delta t\mathbf{F}_{g,N}^{n+1}. \tag{6d}$$

All of the previous has the same algebraic pattern as the FOM problem. The only difference is the size of the operators, much smaller due to the small size of N.

Time-Discretization Forcing Term

The forcing term $\mathbf{F}_{\hat{\mathbf{u}}_h}^n$ due to the time discretization has been intentionally left out in the previous section. We could naively project it too,

$$\mathbf{F}_{\hat{\mathbf{u}}_N}^N = \mathbb{V}^T\mathbf{F}_{\hat{\mathbf{u}}_h}^n, \tag{7}$$

but this would force us to reconstruct the FE vector of the ROM at each time-step, making the integration cumbersome. To go around this issue, we can exploit the algebraic representation of $\mathbf{F}_{\hat{\mathbf{u}}_h}^n$, given in Equation (**??**) in terms of the mass matrix. The forcing term due to the time discretization takes the following form in the ROM:

$$\mathbf{F}_{\hat{\mathbf{u}}_N}^n = \begin{cases} \mathbf{M}_N^{n+1}\hat{\mathbf{u}}_N^n, & \text{BDF-1,} \\ \frac{1}{2}\mathbf{M}_N^{n+1}\hat{\mathbf{u}}_N^n - \frac{3}{4}\mathbf{M}_N^{n+1}\hat{\mathbf{u}}_N^{n-1}, & \text{BDF-2.} \end{cases} \tag{8}$$

Again, these mimic the algebraic pattern obtained in the FOM.

1.1.3 Boundary and Initial Conditions

The computation of the initial condition $\hat{\mathbf{u}}_{N,0}$ is to be done in two steps. First, the initial condition $\hat{\mathbf{u}}_{h,0}$ in the FOM space needs to be computed as a FE vector via interpolation or projection. Then, this FE representation $\hat{\mathbf{u}}_{h,0}$ needs to be projected unto the reduced space. Since we are dealing with an orthonormal basis, we can use the matrix $\mathbb{V}$ to project the FE vector departing from the FOM-ROM relation given in Equation (2),

$$\mathbb{V}^T\hat{\mathbf{u}}_{h,0} = \underbrace{\mathbb{V}^T\mathbb{V}}_{\mathbb{I}}\hat{\mathbf{u}}_{N,0} \rightarrow \hat{\mathbf{u}}_{N,0} = \mathbb{V}^T\hat{\mathbf{u}}_{h,0}. \tag{9}$$

Regarding the spatial boundary conditions, at this point of the naive reduction scheme, two facts come into play, which we first present and then put together:

1) The weak form has homogeneous boundary conditions.
2) The ad-hoc basis elements $\psi_i(x)$ are zero at the boundaries,

$$\psi_i(x) = 0 \quad \forall\, x \in \partial\Omega.$$

These two facts imply that the projection of the operators, Equation (4), does not break the original constraint of homogeneous boundary conditions; and that the linear expansion of the solution $\hat{u}_N(x)$ in the span of V_N, Equation (1), is always true.

There was a paper which discussed this kind of details regarding homogeneous boundary projection.

Once the reduced homogeneous solution $\hat{u}_N(x)$ is obtained, it can be brought back to the original

space V_h via Equation (2), and then add on top of it the FE representation of the Dirichlet lifting, to obtain the final solution,

$$u(x,t;\mu) \simeq u_h(t,\mu) = \mathbb{V}\hat{u}_N(t,\mu) + g_h(t,\mu). \quad (10)$$

1.1.4 Wind-Up for the Naive Reduction Scheme

At this point, the naive reduction scheme for the RB-ROM has been defined and explained. However, some aspects of it remain unattended.

The construction of the problem-based basis functions $\psi_i(x)$: there are many methods to build them, and which combination of them we choose defines which reduction method we are applying, along with their advantages and requirements.

The *offline-online decomposition*: that is, to uncouple the usage of FOM operators in as much as possible from the integration of the ROM. Ideally, no FOM structures should be assembled during the online phase. In this naive scheme, we are clearly not meeting such requirement, since we need to assemble all the FOM operators and then project them *for each timestep*.

Hence, some additional sections need to be brought up, in order to complete our definition of the reducing scheme. We now treat the construction of the basis functions, Section 1.2; and the approximation of the algebraic operators, Section 1.3.

1.2 Reduced Basis Construction

Hereby we layout the details of the basis construction, that is, the definition of the $\psi_i(x)$ functions. There are many techniques to build such basis. We opt for an automatic and out-of-the-box technique: the nested POD approach.

On paper, the most simple basis anyone could come up with is a collection of solution snaphsots for different parameter values and timesteps,

$$\begin{aligned}\Psi_{\hat{u}} &:= [\psi_i] \\ &= [\hat{u}_h(t^0;\mu_0), \hat{u}_h(t^1;\mu_0), \ldots, \hat{u}_h(t^j;\mu_i), \ldots], \\ &= [\Psi_{\hat{u}}(\mu_0), \Psi_{\hat{u}}(\mu_1), \ldots, \Psi_{\hat{u}}(\mu_{N_\mu})] \end{aligned} \quad (11)$$

Yet, this basis $\Psi_{\hat{u}}$ is unpractical from a computational point of view: we could not possibly store all the basis vectors and neiter compute efficiently all the required algebraic operations with them. Additionally, it would probably lead to ill-posed linear systems, since the vectors are almost linearly dependent.

However, since all these vectors arise from the same PDE (although for different parametrizations of it), and for each timestep the solution is close to the previous one, in a way, we could expect there to be a lot of repeated information inside each $\Psi_{\hat{u}}(\mu_i)$, and consequently, inside $\Psi_{\hat{u}}$. We can exploit this fact by using a compression algorithm, such as the Proper Orthogonal Decomposition, to find a set of vectors which capture sufficiently good the span of $\Psi_{\hat{u}}$, and yet do so with less number of basis vectors.

We call this the *method of snapshots*.

1.2.1 POD Space Reduction

This section is left intentionally short for the moment. Many high quality references exist to explain what is the Proper Orthogonal Decomposition (POD), why does it work and which limitations does it have.

We can see it as an enhanced Gram-Schmidt orthonormalization procedure: not only an orthonormal basis is obtained, but additionally the output basis vectors recover hierarchically the span of the original space given by the input vectors.

Let us define the $POD : X \to Y$ function between two normed spaces, such that $Y \subseteq X$, which takes as inputs a collection of N_S vectors and a prescribed tolerance error ε_{POD},

$$[\psi_i]_{i=1}^{N_{POD}} = POD\left([\varphi_i]_{i=1}^{N_S}, \varepsilon_{\text{POD}}\right). \quad (12)$$

with $\varphi_i \in X$ and $\psi_i \in Y$. This function returns a collection of orthonormal N_{POD} vectors, whose span is a subset of the input vector span.

Internally, the POD is solving an optimality approximation problem in the L^2 norm. Therefore,

with a slight notation abuse, one could conceptually say

$$\text{span}\,(\varphi_i) = \text{span}\,(\psi_i) + \varepsilon_{\text{POD}}, \tag{13}$$

that is, the representation of any vector in the original space can be reconstructed to a point with the output POD basis, and the error in the L^2 norm should be less or equal to ε_{POD}.

There is a relation between the prescribed tolerance error $\varepsilon_{\text{POD}}$and the outcoming number of basis elements N_{POD},

$$N_{POD} = N_{POD}(\varepsilon_{\text{POD}}). \tag{14}$$

This functional relationship usually shows exponential decay, or a sharp drop beyond a given number of elements. That is, if a problem is reduceable, beyond a given number of elements, adding more will not improve significantly the approximation bondness.

Alternatively to a prescribed error, one could directly ask for a prescribed number of basis functions $N^*_{POD} \leq N_S$,

$$[\psi_i]_{i=1}^{N^*_{POD}} = POD\left([\varphi_i]_{i=1}^{N_S}, N^*_{POD}\right). \tag{15}$$

1.2.2 Nested POD Basis Construction

A simple procedure to leverage the compression properties of the POD function, and the nested structure of our PDE with respect to time and the parameters; is to first build certain POD basis from the snapshots of the solution at different timesteps for a fixed parameter value,

$$\begin{aligned}
\Psi_{\mu_0} &= POD_\varepsilon\left(\left[\hat{u}_h(t^0;\mu_0), \ldots, \hat{u}_h(t^T;\mu_0)\right]\right),\\
\Psi_{\mu_1} &= POD_\varepsilon\left(\left[\hat{u}_h(t^0;\mu_1), \ldots, \hat{u}_h(t^T;\mu_1)\right]\right),\\
&\ldots,\\
\Psi_{\mu_{N_\mu}} &= POD_\varepsilon\left(\left[\hat{u}_h(t^0;\mu_{N_\mu}), \ldots, \hat{u}_h(t^T;\mu_{N_\mu})\right]\right).
\end{aligned}$$

Then, all the μ-fixed POD basis, which sum up the information contained in the time evolution direction, are compressed again using a POD,

$$\mathbb{V} := \Psi = POD_\varepsilon\left(\left[\Psi_{\mu_0}, \Psi_{\mu_1}, \ldots, \Psi_{\mu_{N_\mu}}\right]\right).$$

In the end, this gives us a unique basis $\Psi = [\psi_i] = \mathbb{V}$, which contains information for parameter variations and time evolution. And above all, it has been built in a computationally efficienty manner, at least in terms of storage.

Different error tolerances could be prescribed at the time and parameter compression stages.

We say this to be an automatic and out-of-the-box procedure because it does not require further complications beyond the storage of the snapshots and the implementation of the POD algorithm. It only demands the definition of a collection of parameter values to solve for. This can be done with random sampling techniques, or if some physically-based knowledge is available, a custom selection of the parameters for ranges in which we know the solution will strongly vary.

Naturally, more involved procedures exist to create the final basis Ψ, but they demand the capacity to evaluate during the creation of the basis how good or bad the new basis is performing at capturing the problem dynamics, or the determination of sharp *a priori* error estimators.

We leave this for later.

1.3 System Approximation

Regarding the assembly of the reduced operators, if already during the offline stage, where the FOM problem is tackled, we had to assemble all the discrete operators for each timestep; during the online stage we have to additionally project them unto the reduced space too, as shown in Equations (4), increasing the overall cost of the integration procedure.

This will be our main motivation to include a system approximation procedure, with the goal of speeding up the construction of the operators.

We talk about *parameter and time separable* problems, or the existence of an *affine decomposition*, when the spatial operators (bilinear or linear forms), which depend on time and parameter values, present the

following functional form,

$$A_h(t,\mu) = \sum_q^{Q_a} \Theta_q^a(t,\mu) A_{h,q}, \tag{16}$$

where the coefficient functions are real-valued, $\Theta_q^a(t,\mu) \in \mathbb{R}$, and the operator basis elements $A_{h,q}$ are parameter-independent.

This expansion can be used for both matrices or vectors provided the topology of the mesh does not change in time. If this is the case, the matrices can be transformed into vectors, and later on brought back to matrix form once any necessary operation has been carried out.

If we had such a decomposition, once we had computed the basis matrix $\mathbb{V}$, we could project each element of the operator basis $A_{h,q}$ to obtain an expression for the reduced operator,

$$\begin{aligned} A_N(t,\mu) &= \mathbb{V}^T A_h(t,\mu)\mathbb{V} \\ &= \sum_q^{Q_a} \Theta_q^a(t,\mu)\mathbb{V}^T A_{h,q}\mathbb{V} \\ A_N(t,\mu) &= \sum_q^{Q_a} \Theta_q^a(t,\mu) A_{N,q}. \end{aligned} \tag{17}$$

Since $A_{N,q}$ is fixed, provided that we had a way to evaluate each $\Theta_q^a(t,\mu)$, we would be able to build the reduced operator for a given parameter for each timestep without having to use any FOM operator.

1.3.1 Discrete Empirical Interpolation Method

Naturally, not many problems are likely to present a separable form as the one shown above. Even our simple linear heat equation problem, due to the time-deformation of the mesh, cannot be presented in such a form.

To tackle this issue, we use the Discrete Empirical Interpolation Method (DEIM). This method is a numerical extension of its analytical sibling, the Empirical Interpolation Method (EIM). Basically, it mimicks the idea of creating a basis for the solution space, but this time centered around the operator space. By means of a nested POD as we explained in Section 1.2.2, if we replace the solution snapshots with operator snapshots, we can build the static and problem-dependent basis $A_{h,q}$.

Since we will be creating the operator basis with an approximation technique, an error is expected in the reconstruction of the actual operator, and so we introduce the notation $A_h^m(t,\mu)$ to reference the approximation of the operator via the (M)DEIM algorithm,

$$A_h(t,\mu) \simeq A_h^m(t,\mu) = \sum_q^{Q_a} \Theta_q^a(t,\mu) A_{h,q}. \tag{18}$$

Naturally, this idea leads to the concept of approximated reduced operators,

$$A_N(t,\mu) \simeq A_N^m(t,\mu) = \sum_q^{Q_a} \Theta_q^a(t,\mu) A_{N,q}, \tag{19}$$

which is the approximation of the reduced operators when the basis comes from an approximation of the operator snapshots.

1.3.2 Discussion on the Approximation Target

There are reasons to approximate A_h instead of directly A_N, but I still need to wrap my head around them.

1.3.3 Evaluation of the Coefficient Functions

To evaluate the $\Theta_q^a(t,\mu)$ functions, we set and solve an interpolation problem; that is, we enforce the approximation to actually match certain elements of the operator,

$$[A_h(t,\mu)]_k = [A_h^m(t,\mu)]_k = \sum_q^{Q_a} \Theta_q^a(t,\mu)[A_{h,q}]_k \tag{20}$$

for certain indices $k \in \mathcal{I}_a$ within a collection of Q_a selected indices $\mathcal{I}_a$. The notation $[A_h(t,\mu)]_k$ stands for the value of the operator at the given mesh node k. In the FE context it can be obtained by integrating the weak form locally.

This leads to a system with the same number of unknowns as equations, where each $\Theta_q^a(t,\mu)$ is unknown. The indices $\mathcal{I}_a$ are selected during the offline stage, according to error reduction arguments of the separable form (18),

$$e_a(t,\mu) = \|A_h(t,\mu) - A_h^m(t,\mu)\|. \tag{21}$$

1.3.4 Reduction of the Nonlinear Term

The basis for the nonlinear operator C_h can be built in many ways, but there is one that will allows us to make it as rich as possible in an efficient way.

The nonlinearity of this operator comes from the fact that it depends on the solution from previous timesteps. Thus, one way to approach the reduction of this operator would be to collect the operator snapshots during the offline phase of the FOM. The problem with this approach is that we would be tied by the parameter space used for the reduced basis.

Instead, if we use the reduced basis, we could span a larger parameter space. We run the offline phase for the nonlinear operator once we have obtained the reduced basis.

We use a nested POD approach again, althoguh this time we have three levels.

We fix a parameter, then for each reduced basis element we collect as many snapshots as timesteps. Then we compress these snapshots to obtain a collateral basis, which we store. When we are done with all the elements in the reduced basis, we have a collection of collateral basis, which we compress. This gives us a basis which contains all the information about the solution for that parameter. We repeat this procedure for each parameter, thus obtaining a collateral basis for each parameter. Finally, we compress these collateral basis to obtain the final collateral basis with information about the solution and the parameter space.

1.4 Hyper Reduced Basis Method

With an approximation of the reduced operators available, we can define yet another algebraic problem to integrate and obtain the reduced solution in time for a given parameter value,

$$\mathbf{M}_N^{m,n+1}\hat{\mathbf{u}}_N^{n+1} + \Delta t \mathbf{A}_N^{m,n+1}\hat{\mathbf{u}}_N^{n+1} = \mathbf{F}_{\hat{\mathbf{u}}_N}^n + \Delta t \mathbf{F}_N^{m,n+1} + \Delta t \mathbf{F}_{g,N}^{m,n+1}, \tag{22a}$$

$$\hat{\mathbf{u}}_N^0 = \hat{\mathbf{u}}_{N,0}. \tag{22b}$$

If we collect terms and factor out the unknowns we get a linear system, in the reduced space and with approximated operators, to be solved for each timestep to advance the solution,

$$\mathbf{K}_N^{m,n+1}\hat{\mathbf{u}}_N^{n+1} = \mathbf{b}_N^{m,n+1}, \tag{23a}$$

$$\mathbf{K}_N^{m,n+1} = \mathbf{M}_N^{m,n+1} + \Delta t \mathbf{A}_N^{m,n+1}, \tag{23b}$$

$$\mathbf{b}_N^{m,n+1} = \mathbf{F}_{\hat{\mathbf{u}}_N}^n + \Delta t \mathbf{F}_N^{m,n+1} + \Delta t \mathbf{F}_{g,N}^{m,n+1}, \tag{23c}$$

$$\hat{\mathbf{u}}_N^0 = \hat{\mathbf{u}}_{N,0}. \tag{23d}$$

Each of the operators present in the problem, $\mathbf{M}_N, \mathbf{A}_N, \mathbf{F}_N$ and $\mathbf{F}_{g,N}$ will have associated an operator basis and will require the solution of the interpolation problem (20) to be solved for each timestep and parameter value. Although this could still seem like a costly procedure, if the operators are actually reduceable, the number of basis functions $Q_m, Q_a, Q_f, Q_{f,g}$ should be small, and thus way simpler problems than assembling the whole operator and the carrying out the projection.

Once the reduced homogeneous solution $\hat{u}_N^m(x)$ is obtained with approximated operators, it can be brought back to the original space V_h via Equation (2), and then add on top of it the FE representation of the Dirichlet lifting, to obtain the final solution,

$$u(x,t;\mu) \simeq u_h^m(t,\mu) = \mathbb{V}\hat{u}_N^m(t,\mu) + g_h(t,\mu). \tag{24}$$

1.5 Certification of the Reduction Procedure

Computation of error bounds between FOM and ROM without actually computing the FOM.

1.5.1 Sacrifical Mode

We can use two ROMs to estimate the error with respect to the FOM solution.

$$\|u_h - \mathbb{V}_N u_N\| = \|u_h - \mathbb{V}_{N+1}u_{N+1} + \mathbb{V}_{N+1}u_{N+1} - \mathbb{V}_N u_N\|. \tag{25}$$

By the triangle inequality, we get

$$\|u_h - \mathbb{V}_N u_N\| \leq \|u_h - \mathbb{V}_{N+1} u_{N+1}\| + \|\mathbb{V}_{N+1} u_{N+1} - \mathbb{V}_N u_N\|. \tag{26}$$

If we define the error function $e_N = \|u_h - \mathbb{V}_N u_N\|$, we get an upper bound of the desired error in terms of the sacrificial ROM error and the error between ROMs,

$$e_N \leq e_{N+1} + \|\mathbb{V}_{N+1} u_{N+1} - \mathbb{V}_N u_N\|. \tag{27}$$

With sufficient ROB[2] elements, it should be safe to assert that the error made with an extra mode should be smaller or equal to the one made without it,

$$e_{N+1} \leq e_N. \tag{28}$$

Thus, we get the following error bound,

$$e_N \leq \|\mathbb{V}_{N+1} u_{N+1} - \mathbb{V}_N u_N\|. \tag{29}$$

It remains to determine how sharp this error estimate is.

ROMs error: Implementation Details

The error between the two ROMs,

$$\|\mathbb{V}_{N+1} u_{N+1} - \mathbb{V}_N u_N\|, \tag{30}$$

can be expressed as a sum in terms of the ROB elements. Due to the hierarchical character of the ROB elements, $\mathbb{V}_{N+1}$ contains the same elements up to N as $\mathbb{V}_N$. Thus, the error between ROMs can be expressed like

$$\|\mathbb{V}_{N+1} u_{N+1} - \mathbb{V}_N u_N\| = \left\| u_{N+1}^{N+1} \psi_{N+1} + \sum_{j}^{N} (u_j^{N+1} - u_j^N) \psi_j \right\| \tag{31}$$

The difference between ROM coefficients $(u_j^{N+1} - u_j^N)$ should be relatively small, since it represents the difference between the two coefficients associated to the same mode. If they were notably different, it would mean that the dynamics between the original ROM and the one carrying the sacrificial mode are different. Since the basis has a hierarchical character, this effect is unlikely to happen: adding a mode should only refine the solution, not change how the previous modes are scaled. They are not strictly the same because the ROM system entries change if more modes are added to the basis, but again, they should do so in a way that somehow preserves dynamics.

2. Reduced Order Basis