

Article

# Toward optimality of proper generalised decomposition bases

Shadi Alameddin <sup>1,\*</sup> , Amélie Fau <sup>1</sup>, David Néron <sup>2</sup>, Pierre Ladevèze <sup>2</sup> and Udo Nackenhorst <sup>1</sup><sup>1</sup> IBNM, Leibniz Universität Hannover, Appelstraße 9a, 30167 Hannover, Germany<sup>2</sup> LMT, ENS Paris-Saclay, CNRS, Université Paris Saclay, 61 avenue du Président Wilson, 94235 Cachan, France

\* Correspondence: shadi.alameddin@ibnm.uni-hannover.de; Tel.: +49-511-762-2286

**Abstract:** The solution of structural problems with nonlinear material behaviour in a model order reduction framework is investigated in this paper. In such a framework, greedy algorithms or adaptive strategies are interesting as they adjust the reduced order basis (ROB) to the problem of interest. However, these greedy strategies may lead to an excessive increase in the size of the reduced basis, i.e. the solution is no more represented in its optimal low-dimensional expansion. Here, an optimised strategy is proposed to maintain, at each step of the greedy algorithm, the lowest dimension of the PGD basis using a randomised SVD algorithm. Comparing to conventional approaches such as Gram-Schmidt orthonormalisation or deterministic SVD, it is shown to be very efficient both in terms of numerical cost and optimality of the reduced basis. Examples with different mesh densities are investigated to demonstrate the numerical efficiency of the presented method.

**Keywords:** model order reduction (MOR); low-rank approximation; proper generalised decomposition (PGD); PGD compression; randomised SVD; nonlinear material behaviour.

## 1. Introduction

Numerical simulations appeal as an attractive augmentation to experiments to design and analyse mechanical structures. Despite the recent improvements in computational resources that makes it feasible to solve systems with a substantial number of degrees of freedom efficiently, it is of common interest to reduce the numerical cost of numerical models throughout model order reduction (MOR) strategies [1]. The performance of MOR techniques has been shown in different fields such as their application to nonlinear problems [2,3], real-time computations [4] or for performing cyclic, parametric or probabilistic computations in which the information provided by some queries can be efficiently reused to respond to other queries that exhibit some similarities [5,6].

A posteriori model reduction techniques such as the Proper Orthogonal Decomposition (POD) is based on an offline training computations which extract a reduced order basis (ROB) from the solution of a high fidelity model. This optimal basis is practically built through a singular value decomposition (SVD) of a snapshot matrix. The singular vectors corresponding to the highest singular values are used to build the ROB [7]. Then, the problem of interest is confined to this ROB resulting in a drastic reduction in the numerical cost [1,8]. However, since the ROB has been defined as an optimal basis for the training stage, some advanced adaptive approaches are required to enrich the basis to tackle nonlinearities [9]. On another hand, *a priori* MOR techniques such as the Proper Generalized Decomposition (PGD) are based on the assumption that the quantities of interest can be written as a finite sum of products of separated functions, of generalised coordinates, which are sought in online computations [8,10]. No prior knowledge of the system is required in such a case and the ROB is directly adapted to the problem of interest by using a greedy algorithm, which enriches the basis when required [3,11]. However, an issue may be caused by the rapid growth of the ROB basis, whereas the primary interest of MOR is to benefit from a small sized ROB which provides a nondemanding temporal updating step. This step is equivalent to a POD step where the spatial modes are fixed and only the temporal ones are updated. It has been observed that the basis can increase to count some

hundreds of modes for parametric studies of nonlinear cyclic loading [12], or some thousands for parametric computations [13]. In [5], some advanced strategies have been proposed to use optimal parametric path allowing to control the basis expansion optimally.

Another approach to control the increase of the basis is by compressing the ROB. In the context of reusing a ROB from a previous computation, a learning strategy has been proposed in [14] to extract an optimal basis from the ROM through a Karhunen-Loève expansion. In a PGD framework, recompression based on SVD has been evaluated in [15]. However, the SVD step turns out to be numerically expensive prohibiting its implementation at each iteration. Therefore, it is common to let the basis increase and compress the results only at convergence to decrease their storage requirements. Therefore, it appears of interest to test advanced computer science tools to compress the ROB on-the-fly without creating a bottleneck in the ROM. A detailed review of the most established algorithms to compute an SVD is provided in [16,17]. These algorithms are not limited to conventional deterministic methods such as truncated, incremental or iterative SVD but also randomised algorithms [18]. Different algorithms have been tested for POD application in the case of dynamical problems in [17]. It has been noticed that randomised SVD algorithms can reduce drastically the numerical cost of the decomposition required after the training stage. Even if this step occurs only once in the offline stage of POD based ROM, for nonlinear problems, the number of degrees of freedom and time steps can be vast for the high fidelity model so that the decomposition process can be a bottleneck.

Our goal here is to maintain the flexibility of the greedy algorithm through the usage of PGD while controlling the size of the reduced basis with a minimal numerical cost, by proposing to use a randomised SVD algorithm that provides a nondemanding compressive step after each enrichment of the basis. The numerical approach will be herein exemplified for the specific case of a fatigue computation based on continuum damage mechanics in a large time increment (LATIN) framework. However, the proposed numerical strategy can be generally used to optimise efficiently of the PGD basis for any application.

This paper is structured as follows. An overview of the LATIN-PGD scheme is provided in Section 2, followed by a discussion on the optimality of the PGD modes and the different algorithms to ensure that in Section 3. Lastly, in Section 4, different numerical examples are presented to illustrate the robustness and efficiency of the proposed algorithm.

### 1.1. Notation

The notation used in this paper is summarised in Table 1.

**Table 1.** Symbols and their representation.

Symbolic representation	Verbal representation
$a, \varphi$	scalars: lowercase letters
$\mathbf{u}, \mathbf{x}$	first-order tensors: lowercase boldface letters
$\mathbf{I}, \mathbf{N}$	second-order tensors: uppercase boldface letters
$\boldsymbol{\sigma}, \boldsymbol{\varepsilon}$	second-order tensors: Greek boldface letters
$\mathbb{C}, \mathbb{H}$	fourth-order tensor: blackboard bold letters
$\underline{a}$	column vector
$\underline{\underline{A}}$	matrix

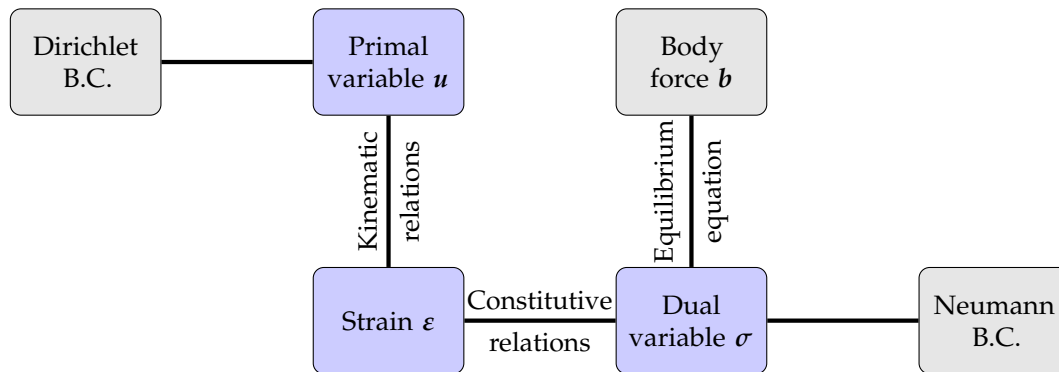
## 2. An overview of the LATIN-PGD method

LATIN is a linearisation scheme that makes it easier to introduce PGD in nonlinear mechanical computations. A review of the LATIN-PGD method and some of its recent extensions to nonlinear solid mechanics problems can be found in [8,11].

The LATIN method is a fully discrete non-incremental solution scheme that inherits its efficiency for mechanical problems from incorporating *a priori* model order reduction technique, namely PGD. It is shown in [19], that functions defined over space-time domain, with some regularities, may be

approximated by PGD. However, it is vital that the number of modes (approximating functions) is small and the approximation error is low. A summary of the implemented framework is provided below.

For a generic structural problem defined over space-time domain  $\Xi = \Omega \times \mathcal{I} = \Omega \times [0, T]$  in an infinitesimal strain and quasi-static framework, the strong form to be solved is represented in Figure 1 [20,21]. The equilibrium equation is linear in terms of the stress and the nonlinearity, in this case, is introduced through the constitutive model, i.e. in the stress-strain  $(\sigma, \epsilon)$  relationship.



**Figure 1.** Graphical representation of the strong form of a structural problem (Tonti Diagram).

In a standard incremental Newton-Raphson scheme, the constitutive relations along with the kinematic relations are substituted into the balance equation resulting in a nonlinear problem in terms of the primal variable. However, a different linearisation strategy (LATIN) would be to solve the equilibrium equations along with kinematic relations in one step and to solve the constitutive relations in the following step, then to seek a solution that satisfies both of these systems simultaneously. In such a framework, two sets of equations are distinguished, the local equations described by constitutive relations (evolution and state laws) and the global equilibrium equation along with the kinematic compatibility. Data flow between these two systems is required, i.e. to get statically admissible stress and kinematically permissible strain or displacement, a relation between the stress and the strain should be assumed. In the same manner, a decision should be taken on what data to pass back from the global system to the local one; these relations are referred to as search direction equations because they are nothing more than a line equation in a 12-dimensional space hosting the stress and strain fields. The main advantage of the LATIN linearisation scheme is confining the computational cost to the solution of a global linear equation which allows for introducing a model order reduction technique such as the PGD to reduce this numerical cost [8].

PGD is often used in many query context and quick response simulations where the solution is approximated by a finite sum of separated functions on each of the problem generalised coordinates, e.g. the displacement field may be approximated by a finite sum of globally spatial and temporal functions as

$$\mathbf{u}(\mathbf{x}, t) \approx \sum_{j=1}^N \mathbf{v}_j(\mathbf{x}) \circ \lambda_j(t), \quad (1)$$

where  $N \in \mathbb{N}$  and  $\circ$  is the entry-wise Hadamard or Schur multiplication of vectors [8,22]. It is shown in [8] that a small number of pairs/modes is sufficient to approximate the solution of many problems with substantial savings in terms of CPU time and memory. In contrary to POD based techniques (that include a preliminary learning phase), PGD defines the basis of the problem on-the-fly using a greedy algorithm such that additional pairs are added if necessary, i.e. the approximation error is controlled by the successive enrichment of the generated basis [23].

The LATIN solution algorithm starts with an elastic initialisation followed by a sequence of two stages, namely the local and the global ones. These two steps form one LATIN iteration, and they

are repeated until convergence is reached. Note that at every local and global step, the quantities of interest over all the space-time points are approximated. The space that belongs to the solution manifold of the constitutive relations is denoted by  $\Gamma$  while  $\mathcal{A}$  represents the admissible space that satisfies the equilibrium equation (static admissibility) along with the kinematic relations (kinematic admissibility). Hence, the exact solution is defined as a set  $s = \{X, \dot{Y}\} \in \Gamma \cap \mathcal{A}$ , where  $X$  is the dynamic conjugate variables and  $\dot{Y}$  represents the evolution of the internal variables. For discussions on the LATIN convergence behaviour, refer to [8,19,24].

The elastic solution  $s_0 = \{X_0\}$  takes all the boundary conditions into account, and the following solutions are computed in terms of corrections to  $s_0$ . Then, the constitutive model, consisting of the nonlinear evolution equations in addition to the state equations, is solved and integrated within the local stage at every space-time point. The outcome of this stage, at the  $i^{\text{th}}$  iteration, is the solution  $\hat{s}_i = \{\hat{X}_i, \hat{Y}_i\}$  which is used in the following global stage to obtain  $s_{i+1}$ . The admissibility equations are the only ones left to be solved in the global stage. The kinematic admissibility is satisfied by deriving the strain as the symmetric gradient of the displacement field  $\varepsilon = \nabla^s \mathbf{u}$  and the static admissibility condition is obtained from the equilibrium equation which reads [25]

$$\nabla \cdot \boldsymbol{\sigma}(\mathbf{x}, t) + \mathbf{b} = \mathbf{0} \quad \forall (\mathbf{x}, t) \in \Omega \times \mathcal{I}, \quad (2)$$

with  $\mathbf{u} = \bar{\mathbf{u}}$  on  $\partial\Omega_D$ ,  $\boldsymbol{\sigma} \cdot \mathbf{n} = \bar{\mathbf{t}}$  on  $\partial\Omega_N$ ,  $\boldsymbol{\sigma}$  is Cauchy stress and  $\mathbf{b}$  is the body force in the spatial domain  $\Omega$ . The use of the Hamilton law of varying action, which is the principle of virtual work integrated over time [26], leads to the following weak form

$$\int_{\Omega \times \mathcal{I}} \boldsymbol{\sigma} : \boldsymbol{\varepsilon}(\mathbf{u}^*) \, d\Omega \, dt = \int_{\Omega \times \mathcal{I}} \mathbf{b} \cdot \mathbf{u}^* \, d\Omega \, dt + \int_{\partial\Omega_N \times \mathcal{I}} \bar{\mathbf{t}} \cdot \mathbf{u}^* \, dS \, dt \quad \forall \mathbf{u}^* \in \mathcal{U}_0^{\mathcal{I}}, \quad (3)$$

where  $\mathcal{U}_0^{\mathcal{I}} = \{\mathbf{u}(\mathbf{x}, t) \mid \mathbf{u}(\mathbf{x}, t) \in \mathcal{H}^1(\Omega) \times \mathcal{C}^0(\mathcal{I}), \mathbf{u} = \mathbf{0} \text{ on } \partial\Omega_D \times \mathcal{I}\}$ . As long as the boundary conditions are satisfied by the elastic initialisation, the corrections in each iteration, in terms of displacement, are defined as  $\Delta \mathbf{u}_{i+1} = \mathbf{u}_{i+1} - \mathbf{u}_i$ , where the  $i$  and  $i + 1$  subscripts refer to the previous and the current global stage, respectively. The solution of Equation (3) is computationally expensive due to the integration over the spatial domain. Therefore, the kinematically and statically admissible fields are computed for the whole space-time domain with the help of PGD, where a separate representation of the displacement and consequently the strain corrections is introduced as

$$\Delta \mathbf{u} = \mathbf{v}(\mathbf{x}) \circ \boldsymbol{\lambda}(t), \quad \Delta \boldsymbol{\varepsilon} = \nabla \mathbf{v}(\mathbf{x}) \circ \boldsymbol{\lambda}(t). \quad (4)$$

Note that the subscript  $i + 1$  is dropped to simplify the notations, and it is assumed that only one PGD term/pair is generated within one LATIN iteration. Following the derivations in [3,27] by writing Equation (3) in terms of corrections and introducing the aforementioned PGD scheme results in a spatial and a temporal problem. These two problems are solved iteratively in a staggered manner using a fixed-point, alternated directions algorithm [8]. After introducing a finite element discretisation, for the spatial and the temporal domains, this algorithm renders a space problem, with homogeneous boundary conditions,

$$\boldsymbol{\gamma} \underline{\underline{K}} \underline{\underline{v}} = \underline{\underline{f}} \quad \boldsymbol{\gamma} \in \mathbb{R} \quad \underline{\underline{K}} \in \mathbb{R}^{n \times n} \quad \underline{\underline{v}}, \underline{\underline{f}} \in \mathbb{R}^n, \quad (5)$$

and a temporal problem, with zero initial conditions,

$$a \underline{\underline{\lambda}} = \underline{\underline{b}} \quad a \in \mathbb{R} \quad \underline{\underline{\lambda}}, \underline{\underline{b}} \in \mathbb{R}^{n_t}, \quad (6)$$

where  $(n, n_t)$  are the spatial and temporal degrees of freedom and  $(\underline{\underline{v}}, \underline{\underline{\lambda}})$  are the spatial and temporal functions. The stiffness matrix is defined as  $\underline{\underline{K}} = \int_{\Omega} \underline{\underline{B}}^T \underline{\underline{C}} \underline{\underline{B}} \, d\Omega$  where  $\underline{\underline{B}}$  is a globally assembled matrix containing the derivatives of the shape functions and  $\underline{\underline{C}}$  is a block diagonal matrix with  $6 \times 6$  diagonal blocks representing the elasticity tensor at each integration point. The scaling factor

in front of the stiffness is defined as  $\gamma = \int_{\mathcal{I}} \underline{\lambda}^T \underline{\lambda} dt$  and the right-hand side  $\underline{f} = - \int_{\Omega \times \mathcal{I}} \underline{B}^T (\underline{\hat{f}} \underline{\lambda}) d\Omega dt$  where  $\underline{\hat{f}}$  is a residual obtained from the previous local stage. The temporal problem is defined by  $a = \int_{\Omega} (\underline{B} \underline{v})^T \underline{C} (\underline{B} \underline{v}) d\Omega$  and  $\underline{b} = - \int_{\Omega} \underline{\hat{f}}^T (\underline{B} \underline{v}) d\Omega$ . Using  $\mu$  modes at iteration  $i + 1$ , the displacement field is approximated by

$$\underline{u}_{i+1} = \underline{u}_0 + \sum_{j=1}^{\mu} \underline{v}_j \underline{\lambda}_j^T, \quad (7)$$

where  $\underline{u}_0$  corresponds to the elastic solution. It is seen that the cost of the global stage is dominated by the computational cost of the spatial problem, Equation (5), that has an identical dimension to the linear elastic problem associated with the finite element discretisation. Thus, a trial, POD-like, step is introduced at the beginning of the global stage that consists of reusing the previously generated spatial modes while updating the temporal ones [28].

### 2.0.1. Temporal modes update

Starting from a certain number ( $\mu$ ) of previously generated PGD pairs, the displacement correction is written as

$$\Delta \underline{u}_{i+1} = \sum_{j=1}^{\mu} \underbrace{\underline{v}_j}_{\text{known}} \Delta \underline{\lambda}_j^T, \quad (8)$$

where  $\Delta \underline{\lambda}_j(t)$  is the correction added to the temporal function  $\underline{\lambda}_j(t)$ . Introducing this assumption into the temporal problem, Equation (6), leads to

$$\underline{\tilde{A}} \underline{\tilde{\Lambda}}^T = \underline{\tilde{B}} \quad \underline{\tilde{A}} \in \mathbb{R}^{\mu \times \mu} \quad \underline{\tilde{B}} \in \mathbb{R}^{\mu \times n_t}, \quad (9)$$

where

$$\tilde{A}_{kl} = \int_{\Omega} (\underline{B} \underline{v}_k)^T \underline{C} (\underline{B} \underline{v}_l) d\Omega, \quad \tilde{\Lambda} = [\Delta \underline{\lambda}_1, \dots, \Delta \underline{\lambda}_{\mu}], \quad \tilde{B}_{kl} = \int_{\Omega} (\underline{B} \underline{v}_k)^T \underline{\hat{f}}_{-t_l} d\Omega. \quad (10)$$

The cost of this step depends only on the temporal discretisation  $n_t$  and the number of already generated modes  $\mu$ . If the computed approximation introduces a significant change to the original temporal modes, measured by  $(\|\Delta \underline{\lambda}_j\| / \|\underline{\lambda}_j\|)$ , then no further enrichment to the spatial modes is required and the algorithm continues to the next local stage. Otherwise, this update is ignored as not to introduce unwanted numerical errors into the temporal functions and a new pair of temporal and spatial modes is generated.

### 3. Optimality of the generated reduced basis

Recall that the correction/solution at the  $i^{\text{th}}$  iteration of the LATIN algorithm, in matrix notation, reads

$$\underline{\tilde{U}} = \sum_{j=1}^{\mu} \underline{v}_j \underline{\lambda}_j^T = \underline{V} \underline{\Lambda}^T \in \mathbb{R}^{n \times n_t}, \quad (11)$$

where  $\underline{V} = [\underline{v}_1, \dots, \underline{v}_{\mu}] \in \mathbb{R}^{n \times \mu}$  and  $\underline{\Lambda} = [\underline{\lambda}_1, \dots, \underline{\lambda}_{\mu}] \in \mathbb{R}^{n_t \times \mu}$ . The representation in Equation (11) is referred to as an outer-product form [29], such a form requires the storage of  $\mu(n + n_t)$  entries only to represent  $\underline{\tilde{U}}$  with  $nn_t$  entries. It is practical to orthonormalise the spatial functions  $\underline{v}_j$  before generating the temporal ones in order to limit the reduced basis size, i.e. the PGD expansion.

This is traditionally done via a Gram-Schmidt (GS) procedure [3]. An orthonormalisation scheme based on a GS procedure is summarised in Algorithm 1, where  $\underline{v}_i^\top \underline{v}_m = \delta_{lm}$  is the inner product between the spatial modes,  $\delta_{lm}$  is the Kronecker delta and  $\|\underline{v}_j\|_2 = \underline{v}_j^\top \underline{v}_j$ .

---

**Algorithm 1:** Gram-Schmidt based orthonormalisation procedure
 

---

**Data:** Previously generated modes  $\{\underline{v}_j, \underline{\lambda}_j\} (j = 1, \dots, \mu)$  with  $\underline{v}_i^\top \underline{v}_m = \delta_{lm}$   
 New pair of modes  $\{\underline{v}_{\mu+1}, \underline{\lambda}_{\mu+1}\}$

**Result:** Enriched basis  $\{\underline{v}_j, \underline{\lambda}_j\} (j = 1, \dots, \mu + 1)$  with  $\underline{v}_i^\top \underline{v}_m = \delta_{lm}$

**for**  $j \leftarrow 1$  **to**  $\mu$  **do**

    Calculate the inner product with an existing mode via  $p = \underline{v}_j^\top \underline{v}_{\mu+1}$

    Subtract the projection from the new mode via  $\underline{v}_{\mu+1} \leftarrow \underline{v}_{\mu+1} - p \underline{v}_j$

    Update existing temporal mode  $\underline{\lambda}_j = \underline{\lambda}_j + p \underline{\lambda}_{j+1}$

**end**

Normalise the new spatial mode  $\underline{v}_{\mu+1} \leftarrow \underline{v}_{\mu+1} / \|\underline{v}_{\mu+1}\|_2$

Update the new temporal mode  $\underline{\lambda}_{\mu+1} \leftarrow \underline{\lambda}_{\mu+1} / \|\underline{v}_{\mu+1}\|_2$

---

While experimenting on the LATIN-PGD scheme in a three-dimensional finite element framework, it has been noticed that reaching a small error required generating many modes, further discussion about the computational cost is provided in Section 4. This confirms the findings in [2] that orthonormality of the spatial modes is not enough to confine the PGD expansion, i.e. compressing the spatial modes only, leaves the temporal ones susceptible to redundancy.

### 3.1. SVD compression of PGD

As long as PGD is not a unique decomposition and does not ensure the optimality of the generated modes in terms of a minimal expansion, an optimal decomposition can be obtained via an SVD of the full solution [30]. An SVD of the solution provides a straightforward scheme to compress both spatial and temporal information into a minimal set of modes, following Algorithm 2. This is similar to compressing information from different spatial directions into a single spatial mode.

---

**Algorithm 2:** SVD compression of a PGD expansion
 

---

**Data:** Previously generated modes  $\{\underline{v}_j, \underline{\lambda}_j\} (j = 1, \dots, \mu)$   
 New pair of modes  $\{\underline{v}_{\mu+1}, \underline{\lambda}_{\mu+1}\}$   
 Required number of modes / truncation threshold  $k < \mu + 1, \epsilon_{\text{tol}}$

**Result:** Enriched basis  $\{\underline{v}_j, \underline{\lambda}_j\} (j = 1, \dots, k)$  with  $\underline{v}_i^\top \underline{v}_m = \delta_{lm}$

Compute the full solution  $\underline{\tilde{U}} = \underline{V} \underline{\tilde{\Lambda}}^\top$

Compute a thin/truncated SVD of the solution  $\underline{\tilde{U}}^{(\mu+1)} = \sum_{j=1}^{\mu+1} \tilde{s}_j \tilde{\underline{v}}_j \tilde{\underline{\lambda}}_j^\top$

Truncate the decomposition based on  $\tilde{s}_{k+1} / \tilde{s}_1 < \epsilon_{\text{tol}}$  or directly using  $k$

Recover the outer-product representation:

$$\underline{V} \leftarrow [\tilde{\underline{v}}_1, \dots, \tilde{\underline{v}}_k] \in \mathbb{R}^{n \times k}$$

$$\underline{\tilde{\Lambda}} \leftarrow [\tilde{s}_1 \tilde{\underline{\lambda}}_1, \dots, \tilde{s}_k \tilde{\underline{\lambda}}_k] \in \mathbb{R}^{n_t \times k}$$


---

It is known via the Schmidt-Eckart-Young theorem that the solution  $\underline{\tilde{U}}$  has an optimal approximation of rank  $k \leq \mu + 1$  with respect to the Frobenius norm that satisfies [29]

$$\|\underline{\tilde{U}} - \underline{\tilde{U}}^{(k)}\|_F = \sum_{j=k+1}^{\mu+1} \tilde{s}_j^2. \quad (12)$$



The corresponding approximation error in terms of the spectral norm reads

$$\|\underline{\tilde{U}} - \underline{\tilde{U}}^{(k)}\|_2 = \tilde{s}_{k+1}. \quad (13)$$

Hence, the PGD expansion may be restricted to a maximum number of modes and Equation (12) will give a measure of the approximation error due to this enforced truncation. Another way is to prescribe a subjectively acceptable tolerance  $\epsilon_{\text{tol}}$  that the approximation error should not exceed, e.g. in the spectral norm this renders to

$$\frac{\|\underline{\tilde{U}} - \underline{\tilde{U}}^{(k)}\|_2}{\|\underline{\tilde{U}}\|_2} = \frac{\tilde{s}_{k+1}}{\tilde{s}_1} < \epsilon_{\text{tol}}. \quad (14)$$

The computation of a full SVD, in case of  $n > n_t$ , requires  $\mathcal{O}(nm_t^2)$  floating point operations (flops) while seeking a truncated SVD requires  $\mathcal{O}(nn_t k)$  flops. Due to the high computational cost of applying an SVD at each enrichment step in a PGD context, a quasi-optimal iterative orthonormalisation scheme was proposed in [2,15]. However, another appealing straightforward approach to provide a direct compression of the PGD modes into a minimal set is utilised here. It consists of using a randomised SVD algorithm [18] to compress the PGD expansion.

### 3.2. Randomised SVD (RSVD) compression of PGD

Low-rank matrix decompositions may be computed efficiently using randomised algorithms as illustrated in this section for an SVD case. Such methods are based on random sampling to approximate the range of the input matrix, i.e. a subspace that captures most of the matrix effect. Then, the matrix is restricted to this subspace, and the low-rank approximation of this reduced matrix is sought using classical deterministic schemes. If  $\underline{\tilde{U}}$  is a dense matrix, the required flops are reduced from  $\mathcal{O}(nn_t k)$  to  $\mathcal{O}(nn_t \log(k))$ , where  $k$  is the number of the sought dominant singular values of an  $n \times n_t$  matrix. It is worth mentioning that even when randomised algorithms require a higher number of flops, they exploit modern multi-processors architecture more efficiently than standard deterministic schemes [18]. An overview of the randomised SVD algorithm applied in a PGD context is briefed in Algorithm 3.

---

#### Algorithm 3: RSVD compression of a PGD expansion

---

**Data:** Previously generated modes  $\{\underline{v}_j, \underline{\lambda}_j\} (j = 1, \dots, \mu)$   
 New pair of modes  $\{\underline{v}_{\mu+1}, \underline{\lambda}_{\mu+1}\}$

**Result:** Enriched basis  $\{\underline{v}_j, \underline{\lambda}_j\} (j = 1, \dots, k)$  with  $\underline{v}_l^T \underline{v}_m = \delta_{lm}$

Compute the full solution  $\underline{\tilde{U}} = \underline{V} \underline{\Lambda}^T \in \mathbb{R}^{n \times n_t}$

Approximate a basis  $\underline{E}$  of  $\text{range}(\underline{\tilde{U}})$  via  $\underline{E} \leftarrow \underline{\tilde{U}} \underline{Q} \in \mathbb{R}^{n \times \tilde{k}}$  with

$\underline{Q} \in \mathbb{R}^{n_t \times \tilde{k}}$  is a random matrix,  $\tilde{k} = k + p$  and  $p$  is an oversampling factor taken experimentally to be in the range of  $5 \sim 10$  [18].

Orthonormalise the columns of  $\underline{E}$  such that  $\underline{\tilde{U}} \approx \underline{E} \underline{E}^T \underline{\tilde{U}}$ .

Restrict  $\underline{\tilde{U}}$  to the  $\text{span}\{\text{col}(\underline{E})\}$  to get a small matrix  $\underline{S} = \underline{E}^T \underline{\tilde{U}} \in \mathbb{R}^{\tilde{k} \times n_t}$

Compute a truncated SVD  $\underline{S} \approx \underline{S}^{(k)} = \underline{\tilde{V}} \underline{\tilde{S}} \underline{\tilde{\Lambda}}^T$  with  $k < \mu + 1$

Expand  $\underline{S}$  to  $\text{span}\{\text{col}(\underline{\tilde{U}})\}$ , i.e.  $\underline{\tilde{U}} \approx \underline{E} \underline{S} \approx \underline{E} \underline{\tilde{V}} \underline{\tilde{S}} \underline{\tilde{\Lambda}}^T = \underline{\tilde{V}} \underline{\tilde{S}} \underline{\tilde{\Lambda}}^T$

Recover the outer-product representation:

$$\underline{V} \leftarrow [\underline{\tilde{v}}_1, \dots, \underline{\tilde{v}}_k] \in \mathbb{R}^{n \times k}$$

$$\underline{\Lambda} \leftarrow [\underline{\tilde{s}}_1 \underline{\tilde{\lambda}}_1, \dots, \underline{\tilde{s}}_k \underline{\tilde{\lambda}}_k] \in \mathbb{R}^{n_t \times k}$$


---

Algorithm 3 can be straightforwardly extended to sample the rows of  $\underline{\tilde{U}}$  when  $n_t$  is large. However, this is not the case in the current study. It is also possible to exploit the PGD decomposition of the solution when computing its SVD or RSVD [29], see Algorithm 4 for details.

**Algorithm 4:** RSVD compression that exploits the PGD expansion (RSVD-PGD)

Previously generated modes  $\{\underline{v}_j, \underline{\lambda}_j\} (j = 1, \dots, \mu)$   
**Data:** New pair of modes  $\{\underline{v}_{\mu+1}, \underline{\lambda}_{\mu+1}\}$   
 Required number of modes / truncation threshold  $k < \mu + 1, \epsilon_{\text{tol}}$

**Result:** Enriched basis  $\{\underline{v}_j, \underline{\lambda}_j\} (j = 1, \dots, k)$  with  $\underline{v}_i^T \underline{v}_m = \delta_{im}$

QR-decomposition:

$$\underline{V} = \underline{Q} \underline{R}_v \sim \mathcal{O}((\mu + 1)^2 n)$$

$$\underline{\Lambda} = \underline{Q} \underline{R}_\lambda \sim \mathcal{O}((\mu + 1)^2 n_t)$$

Compute  $\underline{R}_v \underline{R}_\lambda^T \in \mathbb{R}^{(\mu+1) \times (\mu+1)}$

Apply Algorithm 3 to approximate  $\underline{R}_v \underline{R}_\lambda^T$  as  $\sum_{j=1}^k \tilde{s}_j \tilde{\underline{v}}_j \tilde{\underline{\lambda}}_j^T$

Recover the outer-product representation:

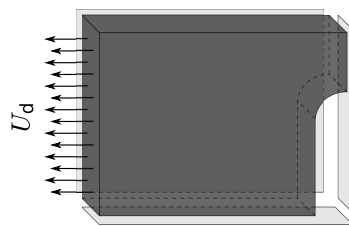
$$\underline{V} \leftarrow \underline{Q} \underline{\tilde{V}} \in \mathbb{R}^{n \times k}$$

$$\underline{\Lambda} \leftarrow \underline{Q} \underline{\tilde{\Lambda}} \underline{\tilde{S}}^T \in \mathbb{R}^{n_t \times k}$$

Algorithm 4 utilises a rank revealing QR-decomposition in order not to rebuild the full matrix. Further algorithmic details of the presented deterministic and randomised algorithms may be found in [16–18]. However, the goal of this study is to investigate the behaviour, robustness and efficiency of the presented algorithms in a PGD framework.

#### 4. Numerical results

The different algorithms are tested in the case of a modified unified viscoplastic viscodamage model, in an infinitesimal strain settings, derived from [3,31–33]. The analysis is carried out on a three-dimensional plate made of Cr-Mo steel at 580°C [34] with a central groove. One-eighth of the plate with symmetric boundary conditions is shown in Figure 2. The plate geometry is defined by its length, width and depth being (40, 20, 2) mm while the length and width of the groove are (10, 4) mm. This plate is subjected to a uniformly distributed displacement field of the form  $U_d = U_0 \sin\left(\frac{2\pi}{T}t\right)$  with  $t$  and  $T$  being the time and the time period, respectively.



**Figure 2.** A plate with a central groove subjected to cyclic loading

Three examples are discussed below. Firstly, the effect of the temporal functions update is investigated, see Section 2.0.1. Then the PGD behaviour with different orthonormalisation schemes is analysed to illustrate the optimality of the reduced basis. Lastly, the computational requirements of the SVD schemes are illustrated throughout a synthetic example.

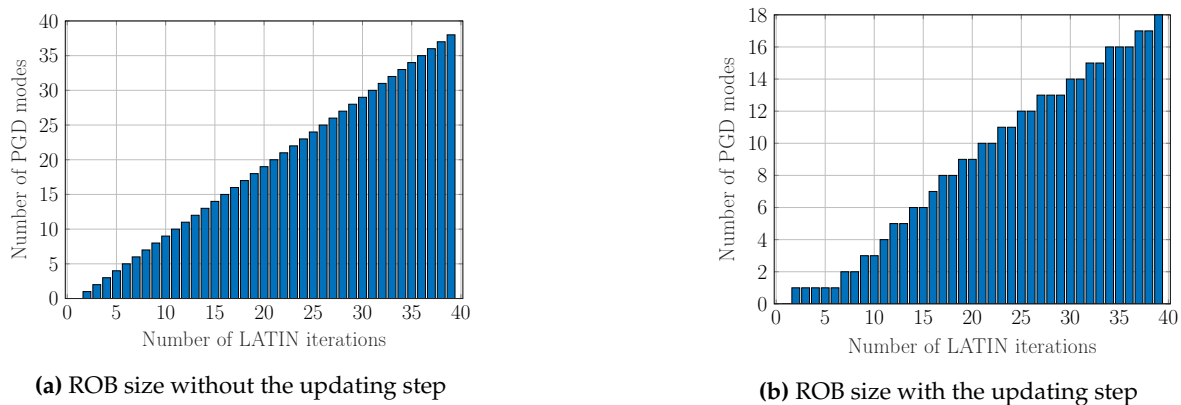
##### 4.1. POD-like temporal functions update

The analysis of the plate, shown in Figure 2, is carried out on a mesh that consists of 387 hexahedron elements, with eight integration points in each element, resulting in 1884 spatial displacement degrees of freedom. The model is subjected to a uniformly distributed displacement field with an amplitude  $U_0 = 0.00606$  mm and a time period  $T = 10$  sec. The temporal discretisation



is chosen such that the domain  $[0, T]$  is discretised into 33 time steps. Since the whole time domain is computed at once, a total of 62172 degrees of freedom are being sought. The commonly used GS scheme (Algorithm 1) is utilised in this example and the convergence criterion is considered to be  $10^{-10}$ .

The purpose of this test case is to evaluate the importance of the updating step in a PGD approach. Hence, the number of generated modes along with the number of the LATIN iterations, with and without this POD-like step, is illustrated in Figure 3.



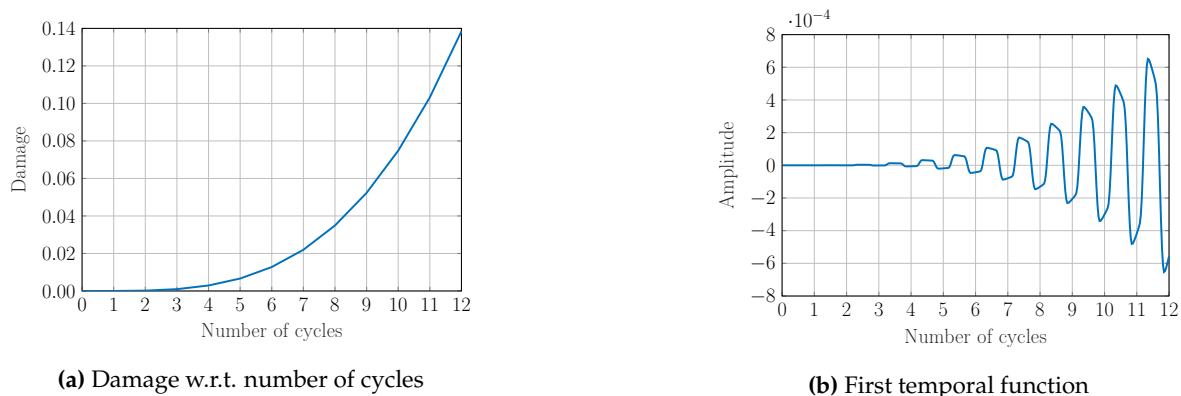
**Figure 3.** The size of the generated reduced basis.

It is seen in Figure 3 the convergence rate of LATIN is not affected by this updating step but the computational cost is sharply decreased. Moreover, such a step is crucial to limit the size of the PGD expansion. With the updating step, only half the number of modes were generated in comparison with the approach without any update. Due to this favourable nature, the updating step is implemented in the rest of the examples.

#### 4.2. PGD behaviour with different orthonormalisation schemes

The previous example with the same spatial discretisation is subjected to 12 load cycles with different amplitudes, in the range of  $[0.0033, 0.0066]$  mm. The temporal domain is divided into 12 intervals, each corresponding to one cycle, and the ROB generated within one cycle is reused in the following cycles. The convergence criterion is considered to be  $10^{-4}$ .

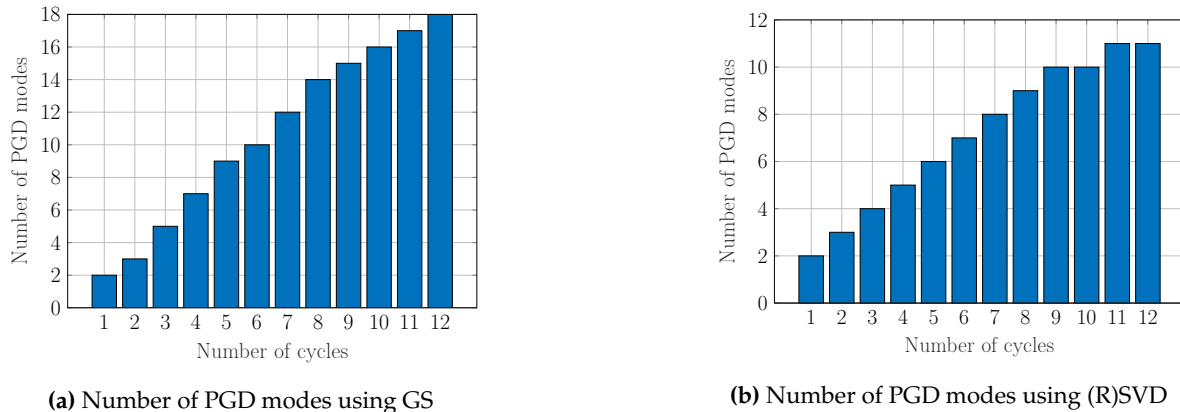
The nonlinearity and the rapid damage evolution can be seen in Figure 4a where the damage value at the end of each cycle is plotted with respect to the number of cycles. The first PGD temporal function of each cycle, after convergence, is illustrated in Figure 4b.



**Figure 4.** Damage evolution and first PGD temporal mode in each cycle.

The simulation is carried out using Algorithms 1 to 4 and the resulting number of PGD modes with respect to the number of cycles is depicted in Figure 5. It is shown in Figure 5a that using

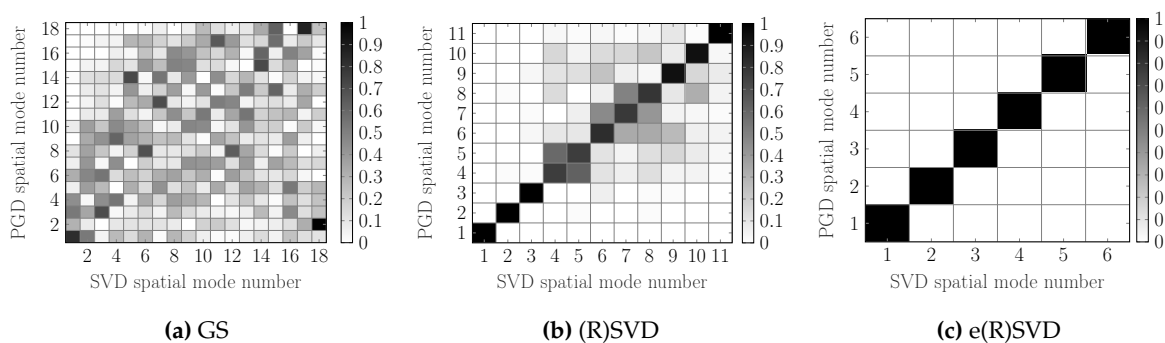
Algorithm 1 resulted in a ROB with 18 modes while Algorithms 2 to 4 reduced this number to 11 modes by adding a maximum of one supplementary mode for each cycle. It is emphasised that Algorithms 2 to 4 provide the same ROB. However, their computational cost differs as illustrated in Section 4.3.



**Figure 5.** Number of PGD modes in each cycle using different orthonormalisation schemes.

It is observed that an SVD compression provides optimality of the ROB. It also has interesting properties such as not rejecting any mode in the current example. In other words, due to the optimality of the generated ROB, the POD-like step plays a noticeable role in convergence and there is no need for further enrichment of the ROB.

The inner product of the spatial modes in each case, after the last cycle, with their corresponding SVD of the acquired solution is shown in Figure 6. As expected, the GS modes are far from the optimal SVD ones while, trivially, Figure 6b depicts an almost diagonal matrix. The off-diagonal entries are caused by the temporal functions update at the final iterations. It is of interest to point out that an excessive RSVD (eRSVD) step after the temporal functions update, which seems to be an unnecessary step, restricts the ROB to six modes only as illustrated in Figure 6c.

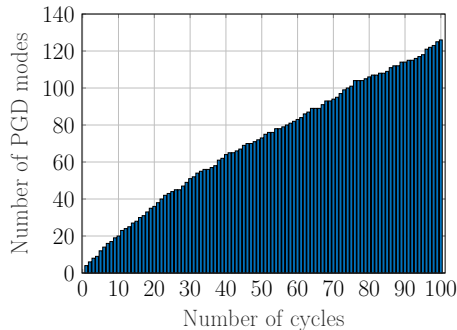


**Figure 6.** ROB optimality w.r.t. an SVD of the resulting solution.

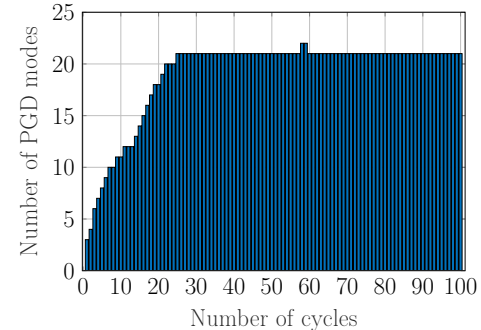
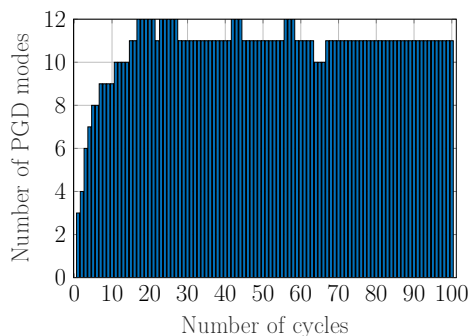
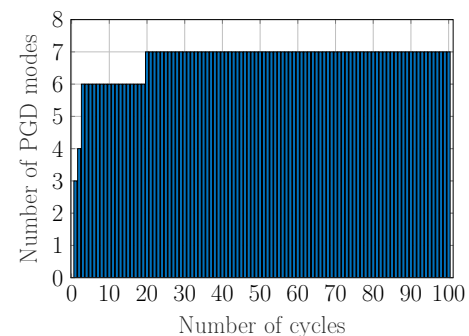
The ensured optimality of the ROB is of interest when used with challenging examples such as in many-query context, due to the expected slow growth of the ROB. In order to investigate the robustness and the behaviour of the ROB, in a many-query context with a large number of degrees of freedom, the plate model is discretised into 13812 hexahedron elements, with eight integration points in each element, resulting in 50547 spatial displacement degrees of freedom. The temporal discretisation consists of 33 time step in each cycle resulting in 1668051 degrees of freedom in each cycle. The plate is subjected to a uniformly distributed displacement field with a uniformly distributed random amplitudes in the range of  $[18, 22] \cdot 10^{-5}$  mm and a time period  $T = 10$  sec. The convergence criterion is considered to be  $10^{-4}$ .

The resulting number of PGD modes with respect to the number of cycles using GS and SVD algorithms is illustrated in Figure 7. It is seen that using a GS algorithm allows the ROB to grow to

contain 126 pairs of modes while SVD algorithms confine this size to 21 modes, using a truncation threshold of  $10^{-8}$ . Accepting a bigger approximation error with a truncation threshold of  $10^{-5}$  reduces the number of modes to 11 pairs while an e(R)SVD scheme introduces further reduction to seven modes without any rejection or truncation due to the maintained optimality of the ROB.



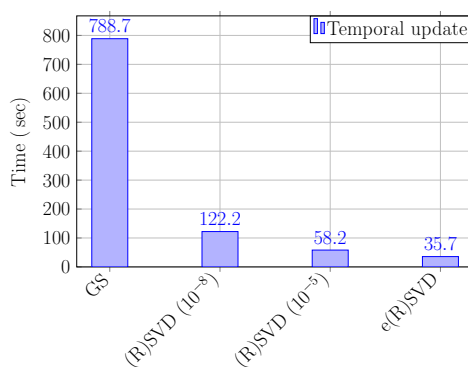
(a) Number of PGD modes using GS

(b) Number of PGD modes using (R)SVD ( $\epsilon_{\text{tol}} = 10^{-8}$ )(c) Number of PGD modes using (R)SVD ( $\epsilon_{\text{tol}} = 10^{-5}$ )

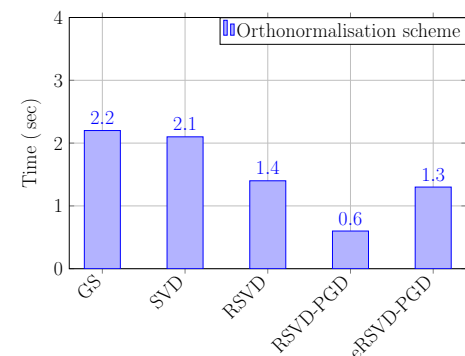
(d) Number of PGD modes using e(R)SVD

Figure 7. ROB size using different orthonormalisation algorithms.

It is worth noting that in this example the SVD orthonormalisation schemes, other than e(R)SVD, were invoked 53 times only compared to 125 times with the GS algorithm. Hence, this explains the low computational requirements of Algorithms 2 to 4 in comparison with Algorithm 1 as summarised in Figure 8b. The eRSVD scheme was invoked in each LATIN iterations. However, due to the small number of generated modes, the required time to update the temporal functions is drastically decreased in comparison with the other schemes, see Figure 8a for a profiler summary.



(a) Timing of the temporal functions update

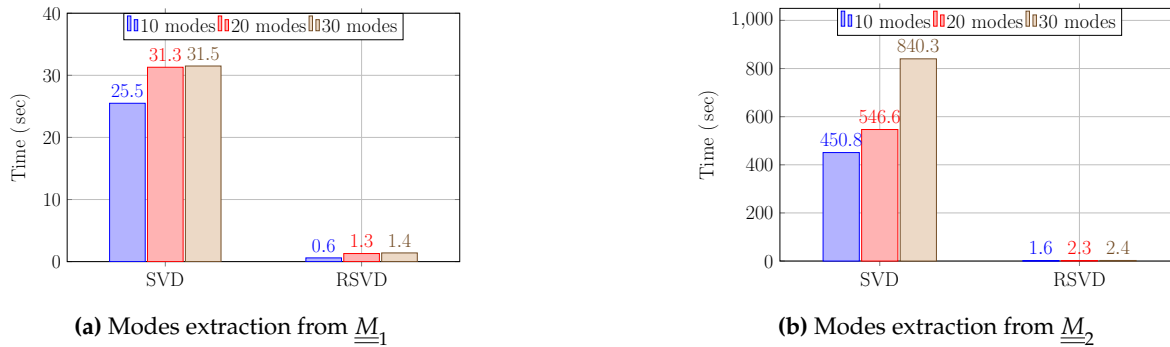


(b) Timing of orthonormalisation schemes

Figure 8. The required time to perform the temporal update and the orthonormalisation steps.

#### 4.3. Comparison between deterministic and randomised SVD schemes

Since it is clear from Figure 8 that all deterministic and randomised SVD schemes are of interest to compress the PGD expansion on-the-fly. Following is an example that illustrates the difference between Algorithms 2 and 3 in case of two synthetic matrices  $\underline{M}_1 \in \mathbb{R}^{10^6 \times 10^2}$  and  $\underline{M}_2 \in \mathbb{R}^{10^6 \times 10^3}$ . The cost to extract the first 10, 20 and 30 singular vectors is recorded and summarised in Figure 9.



**Figure 9.** The required time to extract the first 10,20 and 30 singular vectors using SVD and RSVD.

It is worth noting that the timing for each algorithm depends on the available computational resources but their relative performance is what we are after. The RSVD algorithm is implemented in MATLAB® and uses its built-in SVD routine.

#### 5. Conclusion and further research

Different orthonormalisation techniques were investigated to ensure the optimality of the PGD decomposition. These techniques and their effect on the PGD greedy algorithm are illustrated throughout examples with a varying number of degrees of freedom. It is found that a randomised SVD algorithm is a promising scheme to ensure the optimality of PGD expansions. It introduces beneficial time saving by limiting the number of modes compared to a Gram-Schmidt procedure and, at the same time, it shows a drastic speed-up compared to a deterministic SVD scheme. Another promising approach is proposed here where the randomised SVD scheme is invoked at each LATIN iteration, after the temporal update or the basis enrichment. This approach is referred to, in the current work, as (eSVD/eRSVD) and it shows desired properties such as ensuring an optimal basis in each iteration and eliminating the need to enrich the basis more than needed, i.e. no modes are rejected. The proposed numerical strategy, though it is presented in a LATIN-PGD framework, can be used to optimise PGD basis for any application.

**Author Contributions:** Conceptualization, Shadi Alameddin, Amélie Fau and Udo Nackenhorst; Funding acquisition, Udo Nackenhorst; Investigation, Shadi Alameddin; Methodology, Shadi Alameddin, David Néron and Pierre Ladevèze; Project administration, Amélie Fau and Udo Nackenhorst; Resources, Udo Nackenhorst; Software, Shadi Alameddin; Supervision, Amélie Fau, David Néron, Pierre Ladevèze and Udo Nackenhorst; Validation, Shadi Alameddin; Visualization, Shadi Alameddin; Writing – original draft, Shadi Alameddin and Amélie Fau; Writing – review & editing, Shadi Alameddin.

**Funding:** This research was funded by the German Research Foundation/Deutsche Forschungsgemeinschaft (IRTG 1627).

**Conflicts of Interest:** The authors declare no conflict of interest.

#### Abbreviations

The following abbreviations are used in this manuscript:

flops	Floating point operations
GS	Gram-Schmidt
MOR	Model order reduction
ROM	Reduced order model
ROB	Reduced basis
POD	Proper orthogonal decomposition
PGD	Proper generalised decomposition
SVD	Singular value decomposition
RSVD	Randomised singular value decomposition
e(R)SVD	Excessive SVD/RSVD applied at each iteration after ROB enrichment or temporal update
LATIN	Large time increment

1. Chinesta, F.; Huerta, A.; Rozza, G.; Willcox, K., *Encyclopedia of Computational Mechanics*; John Wiley & Sons, 2018; Vol. 3, chapter Model Reduction Methods.
2. Giacomini, A.; Dureisseix, D.; Gravouil, A.; Rochette, M. Toward an optimal a priori reduced basis strategy for frictional contact problems with LATIN solver. *Comput. Methods Appl. Mech. Eng.* **2015**, *283*, 1357–1381. doi:10.1016/j.cma.2014.09.005.
3. Bhattacharyya, M.; Fau, A.; Nackenhorst, U.; Néron, D.; Ladevèze, P. A LATIN-based model reduction approach for the simulation of cycling damage. *Comput. Mech.* **2018**, *62*, 725–743. doi:10.1007/s00466-017-1523-z.
4. Niroomandi, S.; González, D.; Alfaro, I.; Bordeu, F.; Leygue, A.; Cueto, E.; Chinesta, F. Real-time simulation of biological soft tissues: A PGD approach. *Int. j. numer. method. biomed. eng.* **2013**, *29*, 586–600. doi:10.1002/cnm.2544.
5. Heyberger, C.; Boucard, P.A.; Néron, D. A rational strategy for the resolution of parametrized problems in the PGD framework. *Comput. Methods Appl. Mech. Eng.* **2013**, *259*, 40–49. doi:10.1016/j.cma.2013.03.002.
6. Bhattacharyya, M.; Fau, A.; Nackenhorst, U.; Néron, D.; Ladevèze, P. A multi-temporal scale model reduction approach for the computation of fatigue damage. *Comput. Methods Appl. Mech. Eng.* **2018**, *340*, 630–656. doi:10.1016/j.cma.2018.06.004.
7. Cline, A.; Dhillon, I. Computation of the Singular Value Decomposition. In *Handb. Linear Algebra*; CRC Press, 2013; pp. 1027–1039.
8. Chinesta, F.; Ladevèze, P. *Separated Representations and PGD-Based Model Reduction*; Vol. 554, *CISM International Centre for Mechanical Sciences*, Springer Vienna: Vienna, 2014; p. 227. doi:10.1007/978-3-7091-1794-1.
9. Kerfriden, P.; Gosselet, P.; Adhikari, S.; Bordas, S. Bridging proper orthogonal decomposition methods and augmented Newton–Krylov algorithms: An adaptive model order reduction for highly nonlinear mechanical problems. *Comput. Methods Appl. Mech. Eng.* **2011**, *200*, 850–866. doi:10.1016/j.cma.2010.10.009.
10. Ladevèze, P. Large time increment method for the analysis of structures with non-linear behavior caused by internal variables (La méthode a grand increment de temps pour l'analyse de structures a comportement non lineaire decrit par variables internes). *Acad. des Sci. Comptes Rendus, Ser. II-Mecanique, Phys. Chim. Sci. l'Univers, Sci. la Terre* **1989**, *309*, 1095–1099.
11. Ladevèze, P. On reduced models in nonlinear solid mechanics. *Eur. J. Mech. - A/Solids* **2016**, *60*, 227–237. doi:10.1016/j.euromechsol.2016.08.005.
12. Nasri, M.A.; Robert, C.; Ammar, A.; El Arem, S.; Morel, F. Proper Generalized Decomposition (PGD) for the numerical simulation of polycrystalline aggregates under cyclic loading. *Comptes Rendus Mécanique* **2018**, *346*, 132–151. doi:10.1016/j.crme.2017.11.009.
13. El Halabi, F.; González, D.; Sanz-Herrera, J.A.; Doblaré, M. A PGD-based multiscale formulation for non-linear solid mechanics under small deformations. *Comput. Methods Appl. Mech. Eng.* **2016**, *305*, 806–826. doi:10.1016/j.cma.2016.03.039.
14. Ryckelynck, D. A priori hyperreduction method: an adaptive approach. *J. Comput. Phys.* **2005**, *202*, 346–366. doi:10.1016/j.jcp.2004.07.015.

15. Giacomini, A.; Dureisseix, D.; Gravouil, A. An efficient quasi-optimal space-time PGD application to frictional contact mechanics. *Adv. Model. Simul. Eng. Sci.* **2016**, *3*. doi:10.1186/s40323-016-0067-7.
16. Golub, G.H.; Van Loan, C.F. *Matrix Computations*; The Johns Hopkins University Press, 2013.
17. Bach, C.; Ceglia, D.; Song, L.; Duddeck, F. Randomized low-rank approximation methods for projection-based model order reduction of large nonlinear dynamical problems. *Int. J. Numer. Methods Eng.* **2019**, pp. 1–33. doi:10.1002/nme.6009.
18. Halko, N.; Martinsson, P.G.; Tropp, J.A. Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions. *SIAM Rev.* **2011**, *53*, 217–288. doi:10.1137/090771806.
19. Ladevèze, P. *Nonlinear Computational Structural Mechanics*; Mechanical Engineering Series, Springer New York: New York, NY, 1999; pp. XII, 222. doi:10.1007/978-1-4612-1432-8.
20. Steinke, P. *Finite-Elemente-Methode*; Springer Berlin Heidelberg: Berlin, Heidelberg, 2015; p. 391. doi:10.1007/978-3-642-53937-4.
21. Felippa, C.A. Recent developments in parametrized variational principles for mechanics. *Comput. Mech.* **1996**, *18*, 159–174. doi:10.1007/BF00369934.
22. Cueto, E.; González, D.; Alfaro, I. *Proper Generalized Decompositions*; SpringerBriefs in Applied Sciences and Technology, Springer International Publishing: Cham, 2016. doi:10.1007/978-3-319-29994-5.
23. Chinesta, F.; Keunings, R.; Leygue, A. *The proper generalized decomposition for advanced numerical simulations: A primer*; SpringerBriefs in Applied Sciences and Technology, Springer International Publishing: Cham, 2014; pp. iv–v. doi:10.1007/978-3-319-02865-1.
24. Ladevèze, P.; Perego, U. Duality preserving discretization of the large time increment methods. *Comput. Methods Appl. Mech. Eng.* **2000**, *189*, 205–232. doi:10.1016/S0045-7825(99)00310-2.
25. Wunderlich, W.; Pilkey, W.D. Mechanics of Structures. Variational and Computational Methods. *Meccanica* **2004**, *39*, 291–292. doi:10.1023/B:MECC.0000023038.64148.bc.
26. Gellin, S.; Pitarresi, J.M. Nonlinear analysis using temporal finite elements. *Eng. Anal.* **1988**, *5*, 126–132. doi:10.1016/0264-682X(88)90002-0.
27. Allix, O.; Ladevèze, P.; Gilletta, D.; Ohayon, R. A damage prediction method for composite structures. *Int. J. Numer. Methods Eng.* **1989**, *27*, 271–283. doi:10.1002/nme.1620270205.
28. Bhattacharyya, M. A model reduction technique in space and time for fatigue simulation. PhD thesis, Leibniz Universität Hannover, Université Paris Saclay, 2018.
29. Bebendorf, M. *Hierarchical Matrices*; Vol. 63, 2008; pp. 1–302. doi:10.1007/978-3-540-77147-0.
30. Eckart, C.; Young, G. The approximation of one matrix by another of lower rank. *Psychometrika* **1936**, *1*, 211–218. doi:10.1007/BF02288367.
31. Chaboche, J.L.; Nouailhas, D. A Unified Constitutive Model for Cyclic Viscoplasticity and Its Applications to Various Stainless Steels. *J. Eng. Mater. Technol.* **1989**, *111*, 424. doi:10.1115/1.3226490.
32. de Souza Neto, E.A.; Peric, D.; Owen, D.R.J. *Computational methods for plasticity: theory and applications*; John Wiley & Sons, 2011.
33. Lemaitre, J. *A Course on Damage Mechanics*; Springer Berlin Heidelberg: Berlin, Heidelberg, 1992. doi:10.1007/978-3-662-02761-5.
34. Lemaitre, J.; Desmorat, R. *Engineering Damage Mechanics*; Springer-Verlag: Berlin/Heidelberg, 2005. doi:10.1007/b138882.