Dimensionality reduction of crash and impact simulations using LS-DYNA

C. Bach[1,2], L. Song[2], T. Erhart[3], F. Duddeck[1,4]

[1] Technische Universität München
[2] BMW AG
[3] Dynamore GmbH
[4] Queen Mary University of London

1 Motivation

Automotive crash simulations of full vehicle models still constitute a large computational effort which can be a major problem for applications requiring a large number of evaluations with varying parameter configurations. In some applications, highly similar simulations frequently need to be carried out multiple times with only minor local parameter modifications. At the same time, large amounts of numerical simulation data increasingly become available in industrial simulation databases as part of the progressive level of digitalization of automotive development processes. Data-driven modeling methods are an area of active research, aiming to exploit this new “treasure” in order to find interesting patterns and accelerate predictions.

In particular, projection-based nonlinear model order reduction (MOR) methods have recently been proposed for multi-query scenarios such as parametric optimizations or robustness analyses, and have shown promising results in literature [1, 2]. These methods aim to reduce the dimensionality of the original finite element model (hereafter referred to as the Full-Order Model, FOM) by finding a low-dimensional subspace approximation. The classical two-stage approach of nonlinear MOR methods shown in fig. 1 consists of a so-called offline stage, which incurs a one-time computational cost overhead due to training, and a subsequent online stage during which the generated reduced-order model (ROM) can repeatedly be used for rapid evaluations.

Due to the challenging nonlinearities and intrusive implementation, which requires access to parts of the finite element solver, hyper-reduced automotive crash simulations are very difficult to implement and have not been investigated in literature yet. This work presents a methodology for applying nonlinear MOR methods to highly nonlinear impact and crashworthiness simulations, and gives first promising numerical results which show that the hyper-reduced models remain numerically stable and are able to reproduce the original FOM simulations to a high level of accuracy. We make use of a classical Galerkin projection equipped with a cubature-based hyper-reduction of the nonlinear element forces [1, 2]. The method is complemented by an eigenvalue-based time step estimation as described in [3] for further speed-ups within a multi-query context. We analyze the accuracy and achievable speed-ups for an example problem and provide an outlook of necessary future implementation steps and research directions in order to apply nonlinear MOR methods to full-scale industrial problems.

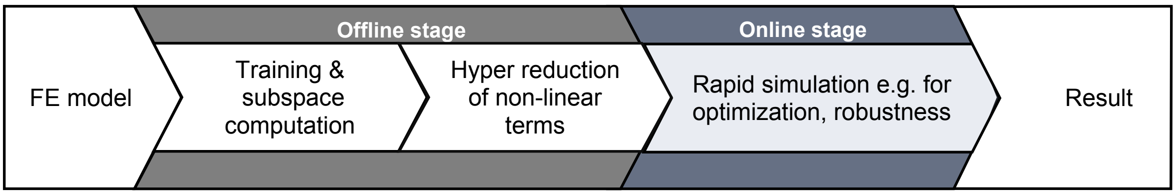
Offline stage
Online stage
FE model
Training & subspace computation
Hyper reduction of non-linear terms
Rapid simulation e.g. for optimization, robustness
Result

Fig.1: Workflow visualization of the classical online/offline decomposition of nonlinear MOR methods.

2 Dimensionality reduction via mathematical projection

A crucial step towards constructing a reduced-order model is to find a suitable low-dimensional representation of the system’s behavior. This work is concerned with projection-based reduction using a linear subspace basis $\boldsymbol{V} \in \mathbb{R}^{m\times k}$ with orthonormal columns, where m is the number of degrees of freedom (DOF) of the FOM, and $k \ll m$ is the dimension of the subspace. Such a basis $\boldsymbol{V}$ can for example be readily obtained using the first eigenmodes of the system as long as the system is linear. Even for weakly nonlinear problems, the eigenmodes can be used in combination with modal derivatives, joint interface modes, or similar techniques. Crash simulations, however, are strongly nonlinear and cannot be appropriately represented within a low-dimensional eigenmode subspace.

The subspace approximation presented in the following relies on existing training simulations of the FOM for different parameter configurations, from which the relevant "deformation modes" of the system can be extracted. The training simulations are usually considered to be the first part of the offline stage, but often existing simulations from previous studies are already available in the industry. Similar simulations from slightly different FOMs (e.g. not all parts in common) may also be re-used, but require some pre-processing of the results. The method of computing a reduced basis from full-scale training data is also known as the method of snapshots [4]. It is based on the singular value decomposition (SVD), equivalently known as Proper Orthogonal Decomposition (POD) or Principal Component Analysis (PCA). It is known that a matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ can be exactly factored into a left-singular matrix $\boldsymbol{U} \in \mathbb{R}^{m\times n}$ with orthonormal columns, a diagonal matrix of singular values $\boldsymbol{\Sigma} \in \mathbb{R}^{n\times n}$, and an orthogonal right-singular matrix $\boldsymbol{Z} \in \mathbb{R}^{n\times n}$, i.e.

$$\boldsymbol{A} = \boldsymbol{U}\,\boldsymbol{\Sigma}\boldsymbol{Z}^T \tag{1}$$

Further, it is known that truncating this factorization after a certain singular value σ_k yields an optimal linear low-rank approximation with respect to the Frobenius and L_2 norms. The truncated approximation of $\boldsymbol{A}$ via the SVD can be written as

$$\boldsymbol{A} \approx \boldsymbol{V}\,\boldsymbol{\Sigma}_k\boldsymbol{Z}_k^T. \tag{2}$$

where $\boldsymbol{\Sigma}_k = \mathrm{diag}(\sigma_1, \ldots, \sigma_k)$ contains the k first singular values of $\boldsymbol{A}$ in decreasing order, $\boldsymbol{Z}_k$ contains the first k columns of $\boldsymbol{Z}$, and $\boldsymbol{V} = \boldsymbol{U}_k$ contains the k first columns of $\boldsymbol{U}$. Here, $\boldsymbol{V}$ is also called the reduced SVD basis of the column space of $\boldsymbol{A}$, with $\boldsymbol{A} \approx \boldsymbol{V}\boldsymbol{V}^T\boldsymbol{A}$. The columns of $\boldsymbol{V}$ are referred to as the "basis vectors". The truncation rank k is either prescribed as fixed or determined via a criterion on the relative approximation error, e.g. with respect to the Frobenius norm

$$k = \min_{s\in\Phi} s, \quad \Phi = \left\{ s \in \mathbb{N}, \frac{\|\boldsymbol{A} - \boldsymbol{U}_s\boldsymbol{\Sigma}_s\boldsymbol{Z}_s^T\|_F}{\|\boldsymbol{A}\|_F} \le \varepsilon \right\} = \left\{ s \in \mathbb{N}, \frac{\sum_{i=1}^{s}\sigma_i^2}{\sum_{i=1}^{n}\sigma_i^2} \ge 1 - \varepsilon^2 \right\}. \tag{3}$$

The SVD is a versatile tool and can for instance be used to compress graphical images (fig. 1), but also numerical simulation data. The data obtained from the training simulations simply needs to be stored in a so-called snapshot matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, where each column of $\boldsymbol{A}$ represents an observed deformation state of the model during a training simulation, i.e. it contains all degrees of freedom (DoFs) of the model which should be reduced in a long flattened vector. In the case of crash simulation and structural dynamics, the DoFs consist of the individual nodal displacements and rotations of each node in the mesh that is intended to be part of the reduced mesh. The corresponding output from LS-DYNA can easily be obtained using the *DATABASE_NODOUT option.

The individual basis vectors, or "modes", can be thought of as deformation patterns similar to the vibration modes of a pre-tensioned Guitar string: the first basis vector describes the most important, typically global, deformation shape, while the subsequent vectors describe more and more high-frequent deformation patterns. In analogy to an image compression in figure 2, theprocess of snapshot collection and reduced basis computation for nonlinear simulation models is visualized in fig. 3. Naturally, strongly nonlinear problems with multiple training simulations across different parameter configuration will require a larger dimension of the subspace basis in order to be accurately represented.

One drawback of the SVD is its algorithmic complexity for large and detailed simulation models with a large number of training snapshots, as is typically the case for nonlinear explicit crash simulations which require small time step sizes. The "classical" divide-and-conquer SVD computes the full factorization (eq. 1) before truncating it, which scales quadratically with the smaller dimension of the snapshot matrix, and linearly with the larger dimension. This is inefficient if one only needs to compute a small number of basis vectors ($k \ll n, m$), and a number of alternative low-rank factorizations have been proposed to mitigate the computational costs. In particular, randomized low-rank approximations have been shown to be highly efficient for a number of nonlinear structural mechanics simulations. The interested reader is referred to [5] for an overview of randomized low-rank approximations, and to [6] for a comparison of their performance relative to other methods for nonlinear MOR applications. Incremental SVD or single-pass randomized SVD methods can be very useful to reduce the memory consumption for very large models, or to implement a streaming algorithm.

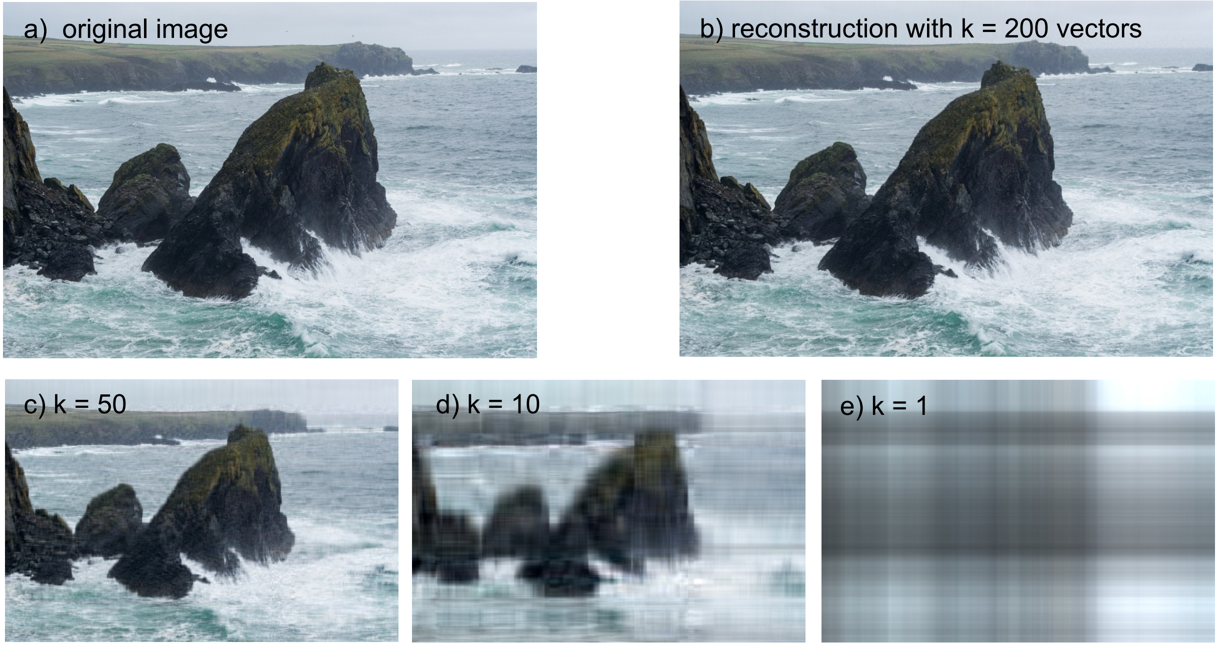

Fig.2: Compression of a 6000 x 4000 pixel color image using a randomized SVD. Each pixel contains three RGB (red, green, blue) color values, such that the image can be represented as a three-dimensional tensor of size 6000 x 4000 x 3. This tensor has been unfolded into two dimensions before computing the SVD. It can be seen that the first basis vectors globally approximate the colors and general shape of the rocks, while smaller details and the spray can only be approximated with a sufficiently high number of basis vectors.

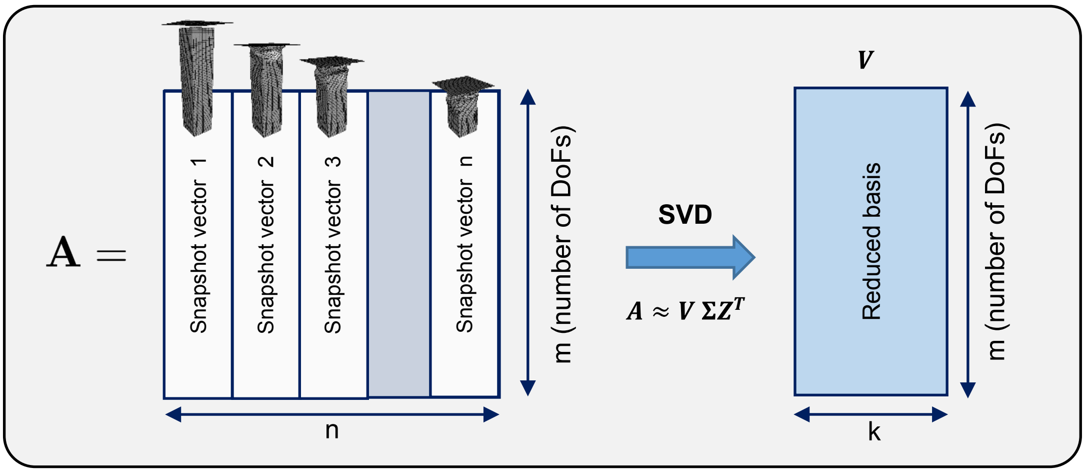

Fig.3: Schematic view of the snapshot matrix SVD for a single training simulation. Further training simulations with different parameters can simply be appended as additional columns, although an alternative pre-clustering of the snapshots for local reduced basis computation can also be beneficial (see [7, 8]).

3 Equations of motion and hyper-reduction of the nonlinear element forces

Consider the equations of motion describing a nonlinear problem solved by LS-DYNA

$$\boldsymbol{M}\ddot{\boldsymbol{x}} + \boldsymbol{f}_{int} = \boldsymbol{f}_{ext}, \tag{4}$$

where $\boldsymbol{x} \in \mathbb{R}^m$ is the DoF vector containing the nodal displacements and nodal rotations (if any), $\boldsymbol{M}$ is a diagonal mass matrix, $\boldsymbol{f}_{int}$ denotes the internal forces, and $\boldsymbol{f}_{ext}$ denotes the external forces. The subspace approximation via the reduced basis $\boldsymbol{V}$ now approximates $\boldsymbol{x}$ as $\boldsymbol{V}\widetilde{\boldsymbol{x}}$, where $\widetilde{\boldsymbol{x}} \in \mathbb{R}^k$ is the vector of the generalized DoFs in reduced space. The equations of motion are thus approximated as

$$\boldsymbol{M}\boldsymbol{V}\ddot{\widetilde{\boldsymbol{x}}} + \boldsymbol{f}_{int} = \boldsymbol{f}_{ext}, \tag{5}$$

The residual of this overdetermined system is then typically constrained to be orthogonal to the space spanned by the reduced basis, which is known as the Galerkin projection, resulting in the equations

$$\boldsymbol{V}^{\boldsymbol{T}}\boldsymbol{M}\boldsymbol{V}\ddot{\widetilde{\boldsymbol{x}}} + \boldsymbol{V}^{\boldsymbol{T}}\boldsymbol{f}_{int} = \boldsymbol{V}^{\boldsymbol{T}}\boldsymbol{f}_{ext}, \tag{6}$$

Notice that $\boldsymbol{V}$ is typically normalized with respect to the mass matrix, such that $\boldsymbol{V}^{\boldsymbol{T}}\boldsymbol{M}\boldsymbol{V} = \boldsymbol{I}$. In the following, we use $\boldsymbol{V}_M$ to denote the mass-orthonormalized version of $\boldsymbol{V}$.
While these equations are already of a much smaller dimension k, their evaluation still requires the computation of every single entry of the nodal force vector. This is problematic, because the computation of the nodal forces is typically the most costly step in the solution procedure for explicit integration. The introduction of a second reduction layer has therefore been proposed, a step which is typically referred to as the “hyper-reduction”. Nevertheless, a reduced model described by (6) can, depending on the problem, still achieve considerable speed-ups as is, due to typically larger stable time step sizes of the reduced model [2, 3, 9].

One class of hyper-reduction methods attempts to approximate the nodal force vector $\boldsymbol{f}$ (or individual contributions to $\boldsymbol{f}$ which are too expensive to evaluate) using another reduced basis – which may also be identical to $\boldsymbol{V}$ if no force snapshots are available. These methods then only evaluate $\boldsymbol{f}$ at a few pre-selected DoFs and try to reconstruct the high-dimensional vector $\boldsymbol{f}$ using this limited information. Examples for these methods are Gappy POD (which originally dates back to the completion of gappy image data, see [10]), and the Discrete Empirical Interpolation Method (DEIM) [11]. Unfortunately, it was found that these methods are often numerically unstable for second-order dynamical systems (see e.g. [3, 12]). In practice, we observed that these hyper-reduction methods required a very large number of evaluation points to remain stable within the limits of the simulation time, and otherwise produced negative volumes or out-of-range velocities within a few time steps.

In the following, we therefore employ a hyper-reduction method based on optimized cubature. This method was originally presented within a computer graphics context in [1] and has later been extended and introduced to the computational mechanics community in [2]. Optimized cubature methods try to approximate the sum of the *projected* unassembled forces via a set of non-negative element weighting factors ξ_e^*. The idea is to optimize the weights such that as many of them are reduced to zero, but a prescribed tolerance criterion is still fulfilled. Elements with a zero weight no longer need to be evaluated in the hyper-reduced simulations and can therefore be omitted from the online stage computations. The set of all elements with non-zero weights is called the reduced mesh. The optimized cubature approximation can be formulated as follows [2].

$$\boldsymbol{V}_M^T \boldsymbol{f}_{\text{node}} = \boldsymbol{V}_M^T \sum_{e \in \Omega} \boldsymbol{L}_e^T \boldsymbol{f}_e \approx \boldsymbol{V}_M^T \sum_{e \in \Omega_{\text{RM}}} \xi_e^* \boldsymbol{L}_e^T \boldsymbol{f}_e \tag{7}$$

$$\xi^* = \text{arg min}_{\xi \in \Phi} \|\xi\|_0; \Phi = \{\xi \in \mathbb{R}^{N_{elem}}, \xi_e \geq 0, \|\boldsymbol{G}\xi - \boldsymbol{b}\|_2 \leq \tau \|\boldsymbol{b}\|_2\} \tag{8}$$

Here, $\boldsymbol{f}_e$ denotes the vector of unassembled nodal force (or moment) contributions of the element e to its nodes, and $\boldsymbol{L}_e^T$ is the assembly operator which assembles the elemental forces of the element e into the correct positions of the global force vector (for details, see [13]). $0 \leq \tau < 1$ is the tolerance on the approximation. Lower values of τ will often produce a higher accuracy, but a larger reduced mesh. Ω is used to denote the set of all elements which should be considered for reduction, and Ω_{RM} is the set of elements which remain in the reduced mesh after computing the element weights (i.e. those elements with non-zero weights). The training data matrix $\boldsymbol{G}$ and right-hand side vector $\boldsymbol{b}$ are obtained as

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{G}_{1,1} & \cdots & \boldsymbol{G}_{1,N_{\mathrm{elem}}} \\ \vdots & \ddots & \vdots \\ \boldsymbol{G}_{n_t,1} & \cdots & \boldsymbol{G}_{n_t,N_{\mathrm{elem}}} \end{bmatrix}, \qquad \boldsymbol{G}_{i,e} = (\boldsymbol{V}_M)_e^T \boldsymbol{f}_e^{(i)} \tag{9}$$

Here, N_{elem} denotes the number of elements in the region of the model which should be reduced, and $(\boldsymbol{V}_M)_e^T$ is the transpose of the restriction of $\boldsymbol{V}_M$ to the rows which correspond to DoFs of the element with index e. Further, $\boldsymbol{f}_e^{(i)}$ denotes the i-th unassembled training force (or moment) snapshot at element e which was collected from the training simulations.

As finding the solution of equation (8) is an NP-hard problem [12], a greedy iterative algorithm is typically used to compute an approximate solution. Notice that penalty contact forces can in principle be hyper-reduced analogously, although the contact forces are not subject to hyper-reduction for the numerical examples presented in the following.

4 Hyper-reduced simulation of a bumper undergoing a small overlap crash

The described methods are now applied to an example problem. The implementation consists of a Python code, which is used to compute the reduced basis, element weights, as well as any other relevant information for the model reduction, and a special development interface to the LS-DYNA explicit FEM solver (version R9 SMP), which is used to implement the online methods and to export unassembled force snapshots from the training simulations. An efficient randomized SVD algorithm is used to compute the reduced basis, and an accelerated variant of the sparse non-negative least squares solver is used to determine the element weights. More details on the individual steps and algorithms will be provided in a forthcoming publication.

Notice that we have so far implemented the hyper-reduction only for a limited set of element types and formulations, and it is not yet possible with the current implementation to perform a hyper-reduction of a full car crash simulation. All simulations are carried out on a Linux workstation with an Intel Xeon E5-2637 v4 3.5 GHz processor and 64 GB of RAM, using four computational threads.

The bumper system model (fig. 4, 5) used as an example problem consists of 51,313 fully integrated shell elements (ELFORM 16), and 52,045 nodes. Of these, 28,817 shell elements and 29,498 nodes are part of a rigid small overlap crash test barrier, which is constrained not to move in any direction. The remaining 22,498 deformable elements and 22,547 nodes are part of the bumper system which is impacting the barrier at an initial speed of 21.1 km/h. The average element edge length is 5.3 mm.

To account for the omitted mass of the rest of the car as well as its rotational inertia about the z-axis, an *ELEMENT_INERTIA keyword is used to impose a concentrated nodal mass and inertia on node 400,000. A piecewise linear plasticity behavior is assumed for the material (figure 6), with a density of 7,830 kg/m³, a Young's modulus of 200 GPa, and a yield strength of 366 MPa. Material failure is not considered. The cross-sections of the bumper and the two crash boxes are rectangular, without any additional deformation triggers, and the wall thickness is set to 1.8 mm. Further model parameters are summarized in table 1.

concentrated mass	**concentrated inertia I_{zz}**	**initial velocity**	**simulation time**	**bumper cross-section ($l_x \times l_z$)**	**crash box cross-sections ($l_y \times l_z$)**
1.50 t	600 kg m²	21.1 km/h	100 ms	30 mm x 120 mm	63 mm x 120 mm

Table 1: Parameters of the bumper system small overlap model

All contacts, including self-contact and the contact between the bumper and the rigid barrier, are modeled using an automatic single surface penalty contact formulation. Notice that the contact forces are not hyper-reduced in this model due to limited access to the source code. However, it has been observed by the authors that penalty contact is in such cases often “well-behaved”, in a sense that contact forces between a subset of elements of the original contact interfaces are sufficient for the hyper-reduced simulations.

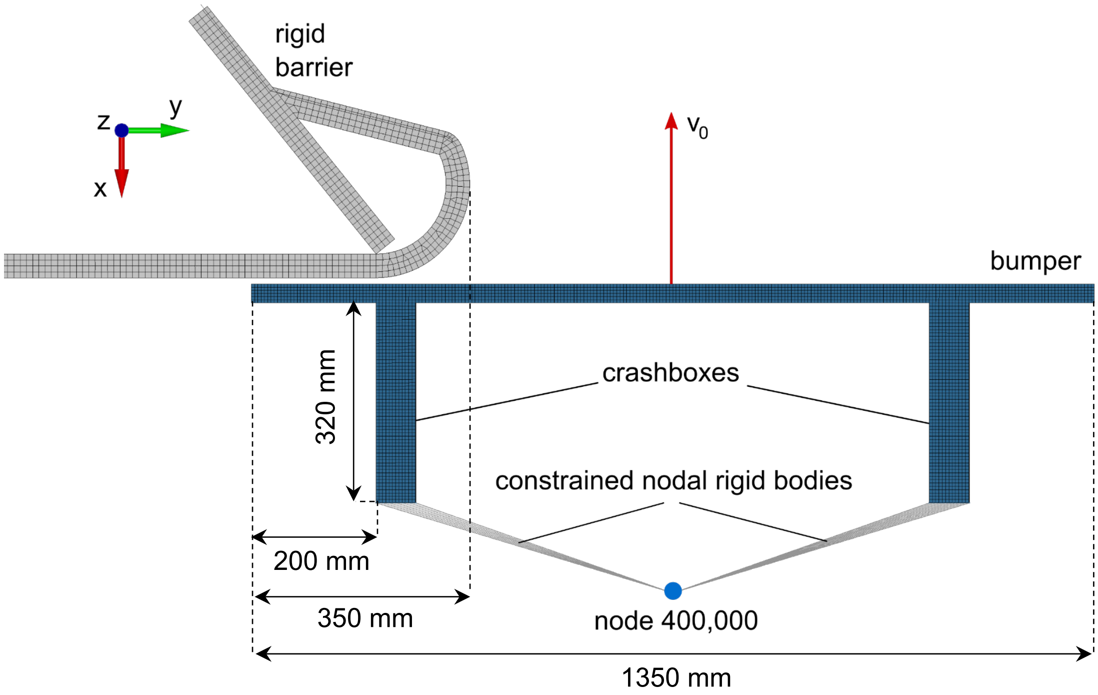

Fig.4: Initial state (t = 0) of the bumper system model and schematic view of the components.

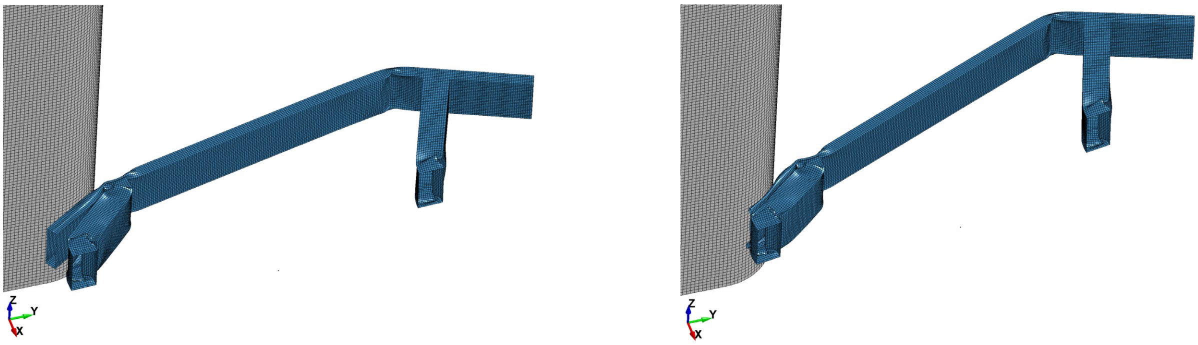

Fig.5: Bumper system reference simulation with the full model, at t=50 ms (a), and t = T = 100ms (b). The crash boxes exhibit folds in multiple places, while the bumper deformation is highly non-linear only in the contact zone close to the barrier, as well as the kink near the right crash box.

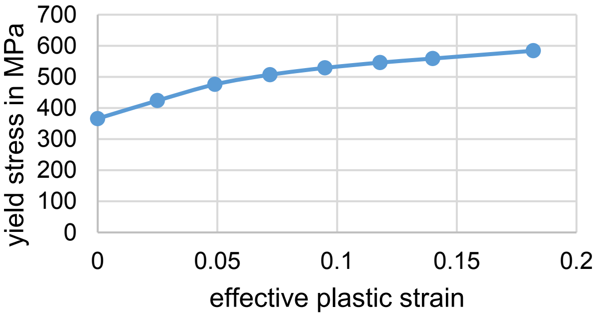

Fig.6: Yield stress vs. effective plastic strain of the material used for the bumper and crash boxes.

Next, two hyper-reduced models are constructed using 30 basis vectors and hyper-reduction tolerance levels $\tau \in \{0.02, 0.05\}$. Displacements and rotational DoFs are reduced individually. Rigid parts are left unreduced, as they already move within a low-dimensional rigid-body subspace. The resulting reduced meshes are shown in figure 7, key statistics are summarized in table 2. Notice that LS-DYNA's own time step control was used for the FOM simulation (TSSFAC = 0.9, no mass scaling), whereas a prescribed stable time step size based on the ROM eigenvalues [3] was used for the hyper-reduced models.

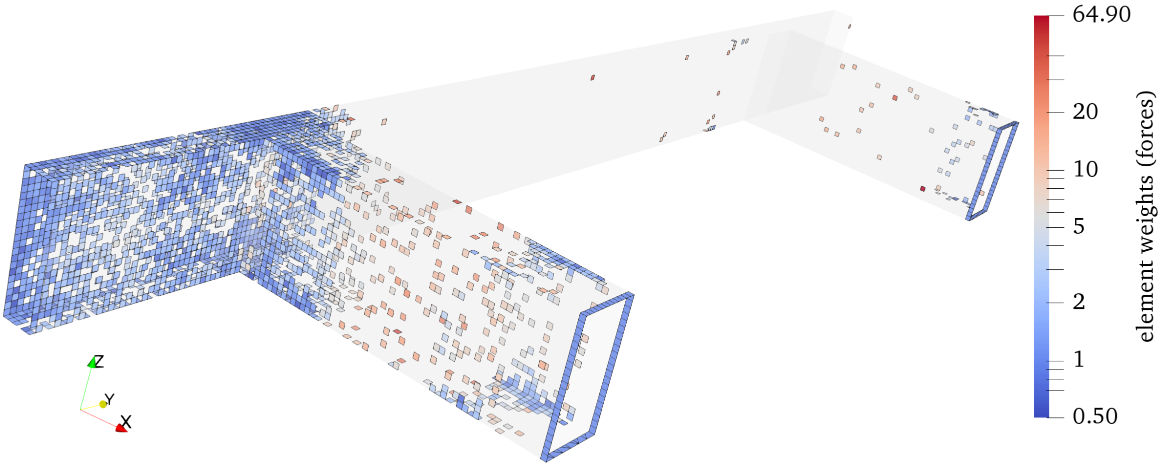

Fig.7: Reduced mesh of the hyper-reduced model with k=30, $\tau = 0.02$, with selected elements and associated element weights of the forces, as computed by the hyper-reduction sampling and weighting method (magnitude is color-coded).

The approximation errors of the reduced-order online simulations are measured using the global mean relative error of the nodal displacements of the solution. It is defined by [2]:

$$\varepsilon_{GMRE} := \frac{\sqrt{\sum_{t\in T_E}\left(\boldsymbol{x}(t) - \boldsymbol{V}_M\tilde{\boldsymbol{x}}(t)\right)^T\left(\boldsymbol{x}(t) - \boldsymbol{V}_M\tilde{\boldsymbol{x}}(t)\right)}}{\sqrt{\sum_{t\in T_E}\boldsymbol{x}^T(t)\boldsymbol{x}(t)}} \tag{7}$$

where $\boldsymbol{x}(t)$ at time t denotes the vector containing the nodal displacements of the original FOM, and $\tilde{\boldsymbol{x}}(t)$ denotes the reduced displacements of the ROM or hyper-reduced model. T_E is the set of time instants at which nodal displacement outputs are generated in the online simulations.

Model type	**number of deformable elements**	**online simulation time (8 threads)**	ε_{GMRE} **(displacements)**
Original model	22,498 (100 %)	49 minutes	n.a.
Galerkin ROM (k = 30, no hyper-reduction)	22,498 (100 %)	27 minutes	0.8 %
Hyper-reduced model (k = 30, $\tau = 0.05$)	778 (3.5 %)	5 minutes	2.5 %
Hyper-reduced model (k = 30, $\tau = 0.02$)	2,495 (11 %)	9 minutes	1.1 %

Table 2: Model key figures, including online run time and online errors.

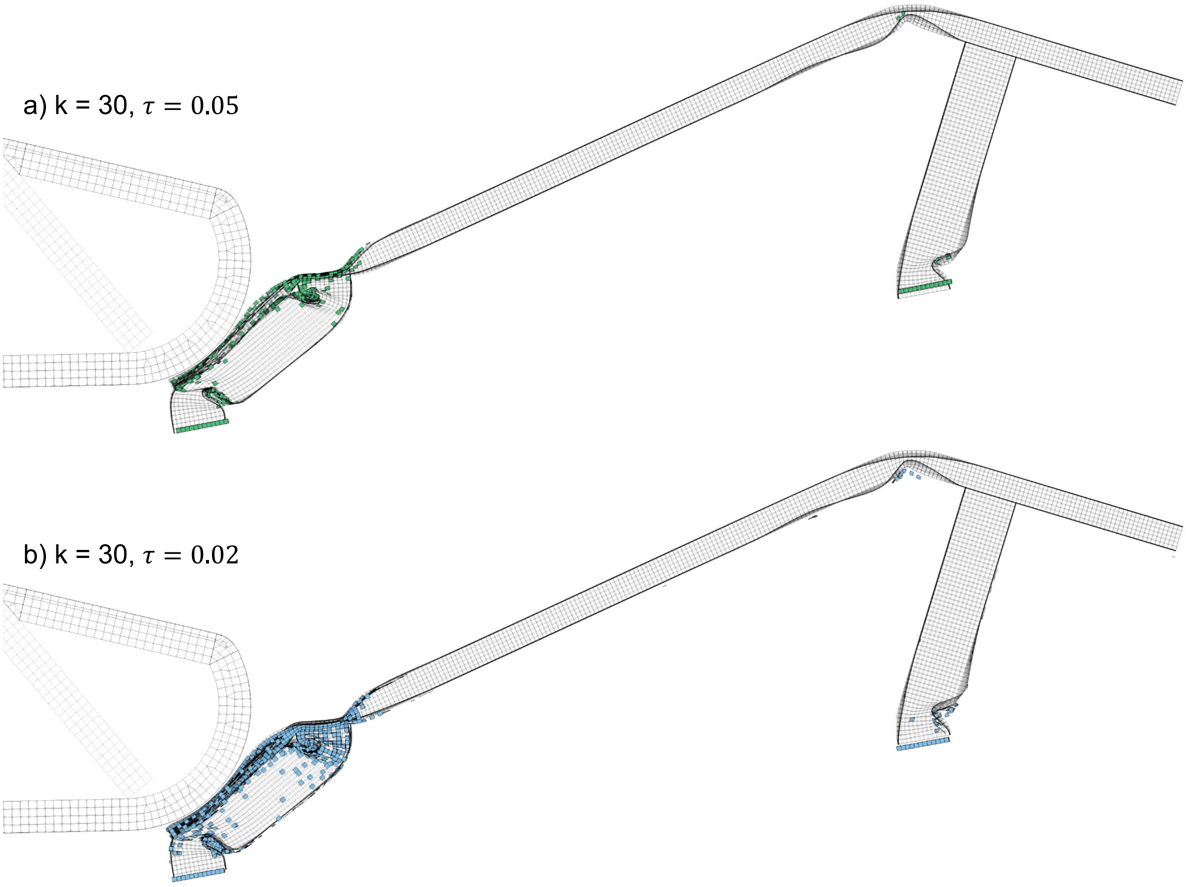

Fig.8: Hyper-reduced simulation models (green/blue) and FOM simulation (wireframe) overlaid at the final state (t = 100 ms) and showing a generally good agreement. Notice that a full output for all nodes can easily be reconstructed from the hyper-reduced solution.

5 Discussion

Observing fig. 7, one can see that most sampled elements are located within, or close to, the regions where large nonlinear deformations occur, e.g. the bumper zone which comes into direct contact with the barrier, the folds of the crash box, or the kink in the bumper. Thus, the hyper-reduction algorithm was able to automatically identify the most important regions of the model. It is also interesting that the largest element weights are assigned to elements in regions where only very few elements have been selected and which do not undergo large plastic deformations. This is related to the fact that these elements need to “replace” a larger number of omitted neighboring elements, whereas elements in zones of large deformations have retained more neighbors and are therefore mostly assigned smaller weights. As expected, the reduced mesh size increases with a larger number of basis vectors (table 2).

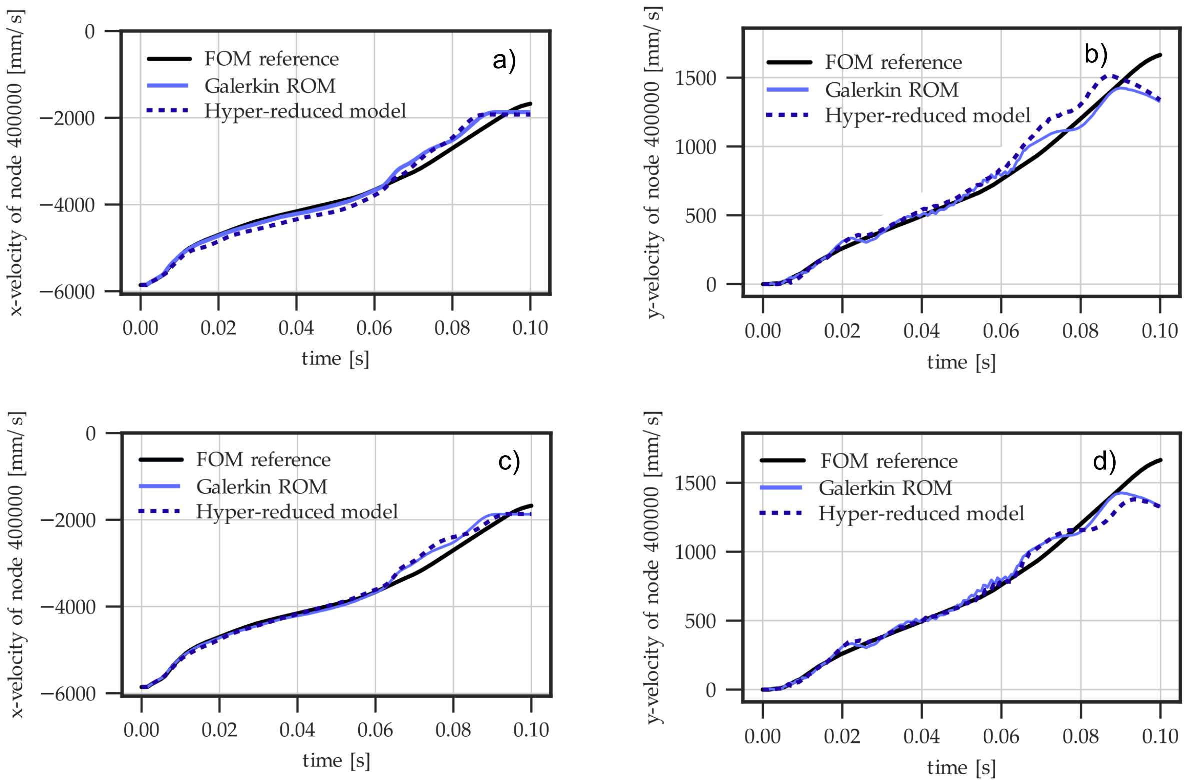

Fig.9: Accuracy in the approximation of the x- and y-velocities of node 400,000. Subfigures a-b: k = 30, $\tau = 0.05$, subfigures c-d: k = 30, $\tau = 0.02$.

The online simulations first of all revealed that all tested reduced and hyper-reduced models were numerically stable. The reduced and hyper-reduced simulations also showed a high level of accuracy, especially for $\tau = 0.02$, with displacement global mean relative errors of about 1-3 % and a good visual agreement with the FOM simulation. The accuracy of the velocity curves of node 400,000 is somewhat lower than that of the nodal displacements, as can be seen e.g. from figure 9 (particularly in the second half of the simulation). Even without hyper-reduction, the reduced models further achieved online speed-ups factors of roughly 1.8 due to larger stable time steps compared to the FOM. Hyper-reduction achieved online speed-ups of 5.4 to 9.8.

Finally, the hyper-reduction worked well even though the penalty contact forces themselves were not subject to hyper-reduction. This is perhaps surprising, because the initial contact forces between two interfaces will generally be smaller in the hyper-reduced models due to the “missing” elements. Penalty contact seems to be well-behaved in this regard, at least for the type of problem studied. A similar behavior has been remarked by [14] in the context of hyper-reduced tire simulations, who also reported that in some cases it can be beneficial to adjust the contact penalty parameter for hyper-reduced simulations (especially when static training simulations are used and the ROM is not well able to approximate the rich contact dynamics).

6 Summary and outlook

This paper discusses data-driven dimensionality reduction of explicit crash and impact simulations, with a particular focus on hyper-reduction and intrusive MOR to reduce the computing times with LS-DYNA. The method described uses data from training simulations to compute a subspace approximation which is able to capture nonlinear phenomena, and employs an optimized cubature scheme based on [1, 2] to further reduce the numerical effort. Since the implementation is partly intrusive, modifications of certain FE solver routines were implemented using a particular development interface to the LS-DYNA FEM code. The workflow also draws from previous work [3, 6], which has addressed questions of the scalability of the offline stage to highly detailed industrial FEM models, as well the impact of Galerkin and hyper-reduction methods on the stable time step size under explicit time integration. To the authors' knowledge, this is one of the first applications of such model reduction methods within an automotive crashworthiness context.

Hyper-reduction was applied to a numerical example problem involving multiple sources of nonlinearity, including contact, self-contact, plasticity, and geometric nonlinearities. It could be observed that the reduced and hyper-reduced models were able to reproduce the training simulations to a good level of accuracy, while providing speed-up factors of 5.4 to 9.8 for the problems studied. Even though the contact forces were not themselves subject to hyper-reduction, the penalty contact proved to be well-behaved for this problem. The results further demonstrated that rigid bodies can also be incorporated in this framework, and that hybrid models are possible which restrict the reduction to only part of the model.

While the results look promising, these are only the first steps towards using nonlinear model reduction or hyper-reduction methods productively for industrial crash simulations. The influence of various parameters on the computing times and resulting accuracies needs to be analyzed in further detail. The online accuracy of reduced and hyper-reduced models also remains to be tested for parameter configurations other than the training simulations, and further improvements which have been proposed in literature could be the subject of future work (e.g. local reduced basis approximations or parameter-adaptive reduced models). Moreover, hyper-reduction has not yet been implemented for all element types. Similar modifications to force assembly routines would be necessary to apply hyper-reduction to more complex full-scale automotive crash simulations.

Finally, low-dimensional subspace approximations of automotive crashworthiness models obtained by fast and scalable randomized or incremental SVD algorithms can also be of interest for future work in the area of non-intrusive model reduction (see e.g. [15, 16, 17]).

7 Literature

[1] An S S, Kim T, & James D L: "Optimizing Cubature for Efficient Integration of Subspace Deformations". ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH Asia 2008, 27(5), 2008.

[2] Farhat C, Avery P, Chapman T, & Cortial J: "Dimensional reduction of nonlinear finite element dynamic models with finite rotations and energy-based mesh sampling and weighting for computational efficiency". International Journal for Numerical Methods in Engineering, 98(9), 625–662, 2014.

[3] Bach C, Song L, Erhart T, Duddeck F: "Stability of explicit time integration for crashworthiness using nonlinear model reduction". arXiv preprint, 2018.

[4] Sirovich L: "Turbulence and the dynamics of coherent structures, part I". Quarterly of Applied Mathematics, XLV(3), 1987.

[5] Halko N, Martinsson P-G, & Tropp J A: Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions. SIAM Review, 53(2), 217–288, 2011.

[6] Bach C, Ceglia D, Song L, & Duddeck F: "Randomized low-rank approximation methods for projection-based model order reduction of large nonlinear dynamical problems". International Journal for Numerical Methods in Engineering, 118(4), 209–241, 2018.

[7] Amsallem D, Zahr M J, & Farhat C: „Nonlinear Model Order Reduction Based on Local Reduced-Order Bases". International Journal for Numerical Methods in Engineering, 92(10), 891–916, 2012.

[8] Amsallem D, Zahr M J, & Washabaugh K: „Fast local reduced basis updates for the efficient reduction of nonlinear systems with hyper-reduction". Advances in Computational Mathematics, 41(5), 1187–1230, 2015.

[9] Krysl P, Lall S, & Marsden J E: „Dimensional model reduction in non-linear finite element dynamics of solids and structures". International Journal for Numerical Methods in Engineering, 51, 479–504, 2001.

[10] Everson R, & Sirovich L: "Karhunen-Loève procedure for gappy data". Journal of the Optical Society of America A, 12(8), 1657–1664, 1995.

[11] Chaturantabut S & Sorensen D C: "Nonlinear Model Reduction via Discrete Empirical Interpolation". SIAM Journal on Scientific Computing, 32(5), 2737–2764, 2010.

[12] Farhat C, Chapman T, & Avery P. Structure-preserving, stability, and accuracy properties of the energy-conserving sampling and weighting method for the hyper reduction of nonlinear finite element dynamic models. International Journal for Numerical Methods in Engineering, 102(5), 1077–1110, 2015.

[13] Belytschko T, Liu W K, Moran B, & Elkhodary K I: “Nonlinear Finite Elements for Continua and Structures”, Second Edition. Chichester, UK. Wiley, 2014.

[14] De Gregoriis D, Naets F, Kindt P, & Desmet W: „A nonlinear a-priori hyper-reduction method for the dynamic simulation of a car tire rolling over a rough road surface”. ECCM-ECFD 2018. Glasgow, UK, 2018.

[15] Le Guennec Y, Brunet J-P, Daim F Z, Chau M, & Tourbier Y: “A parametric and non-intrusive reduced order model of car crash simulation”. Computer Methods in Applied Mechanics and Engineering, 338, 186–207, 2018.

[16] Borzacchiello D, Aguado J V, & Chinesta F: “Non-intrusive Sparse Subspace Learning for Parametrized Problems”. Archives of Computational Methods in Engineering, 26(2), 303–326, 2019.

[17] Guo M, & Hesthaven J S: “Data-driven reduced order modeling for time-dependent problems”. Computer Methods in Applied Mechanics and Engineering, 345, 75–99, 2019.