Efficient basis updating for parametric nonlinear model order reduction

Christian H. Meyer[1]*, **Christopher Lerch**[2], **Morteza Karamooz Mahdiabadi**[1], and **Daniel Rixen**[1]

[1] Technical University of Munich, Chair of Applied Mechanics, Boltzmannstr. 15, 85748 Garching b. München
[2] Technical University of Munich, Chair of Automatic Control, Boltzmannstr. 15, 85748 Garching b. München

Nonlinear model reduction is used to speed up the numerical solution of finite element models for vibration analysis of structures undergoing large deflections. This speed up is highly desired in design and optimization applications where parametric models are considered and the outcoming high-dimensional differential equation must be solved multiple times. A first step is to approximate the solution vector i.e. the displacements of the nodes by a linear combination of some basis vectors. One common choice for these basis vectors is a combination of vibration modes static modal derivatives. However, these vectors depend on parameter values of the parameterized system. This contribution shows how these basis vectors can be updated efficiently. The vibration modes are updated by an inverse free preconditioned Krylov subspace method while the static derivatives are updated by a preconditioned conjugate gradient solver. A case study with a parametric beam gives a first insight into the performance of the proposed method.

1 Introduction and fundamentals

Finite element analysis of parameterized structures that undergo large deformations leads to the differential equation

$$\boldsymbol{M}(\boldsymbol{p})\,\ddot{\boldsymbol{u}}(t) + \boldsymbol{f}(\boldsymbol{p},\boldsymbol{u}(t)) = \boldsymbol{F} \tag{1}$$

with paremeters $\boldsymbol{p}$, nodal displacements $\boldsymbol{u}$, mass matrix $\boldsymbol{M}$, internal nonlinear restoring force $\boldsymbol{f}$ and external loads $\boldsymbol{F}$. Time and parameter dependecy will be omitted from now on for the sake of readability. A Galerkin projection

$$\boldsymbol{V}^T\boldsymbol{M}\boldsymbol{V}\,\ddot{\boldsymbol{q}} + \boldsymbol{V}^T\boldsymbol{f}(\boldsymbol{V}\boldsymbol{q}) = \boldsymbol{V}^T\boldsymbol{F} \tag{2}$$

that approximates the solution vector by $\boldsymbol{u} \approx \boldsymbol{V}\boldsymbol{q}$ and a hyperreduction (cf. [1]) that accelerates the evaluation of the nonlinear term $\boldsymbol{f}$ is applied to speed up numerical time integration of (1). One opportunity for choosing $\boldsymbol{V}$ is a concatenation of two kinds of vectors: First, some vibration modes $\boldsymbol{v}_i$ that are computed by solving the eigenvalue problem

$$(\boldsymbol{K} - \omega_k^2\boldsymbol{M})\boldsymbol{v}_k = \boldsymbol{0} \qquad \text{with} \quad \boldsymbol{K} = \left.\frac{\partial \boldsymbol{f}(\boldsymbol{u})}{\partial \boldsymbol{u}}\right|_{\boldsymbol{u}=\boldsymbol{u}_{\mathrm{static}}} \tag{3}$$

of the around $\boldsymbol{u}_{\mathrm{static}}$ linearized system and, second, some static modal derivatives $\boldsymbol{\theta}_{ij}$ [2] that are computed by solving

$$\boldsymbol{\theta}_{ij} = -\boldsymbol{K}^{-1}\left(\nabla_{\boldsymbol{v}_\mathrm{i}}\boldsymbol{K}(\boldsymbol{u})\right)\boldsymbol{v}_j\ . \tag{4}$$

However, these vectors are parameter-dependend since the system matrices $\boldsymbol{K}$ and $\boldsymbol{M}$ depend on the parameters $\boldsymbol{p}$. If the parameters $\boldsymbol{p}$ change, a new computation of $\boldsymbol{V}$ can be necessary but very costly if obtained by direct approaches.

2 Updating approach for modes and static derivatives

One approach to accelerate the computation of new basis vectors are iterative updating techniques. Instead of computing the reduction vectors for each new parameter values $\{\hat{\boldsymbol{p}}_0, \hat{\boldsymbol{p}}_1, \hat{\boldsymbol{p}}_2, \ldots\}$ directly, one only computes the vectors of the first system $\hat{\boldsymbol{p}}_0$ directly. The vectors of the other systems $\{\hat{\boldsymbol{p}}_1, \hat{\boldsymbol{p}}_2, \ldots\}$ are updated iteratively by using the results of the first system.

Both, modes and static derivatives must be updated. An updating algorithm for the modes has been published in [3]. It avoids the decomposition of the stiffness matrix $\boldsymbol{K}(\boldsymbol{p})$ which is the most costly part in eigensolvers such as [5]. It uses instead the decomposition of $\boldsymbol{K}(\hat{\boldsymbol{p}}_0)$ which is available since the modes of the first system have been computed directly. The reader is refered to [3] for a detailed discussion of the algorithm. We use a slightly modified version which is shown Algorithm 1. The algorithm is augmented by a singular value decomposition that is used to deflate the Krylov subspace $\mathcal{K}_r(\boldsymbol{K}^{-1}(\hat{\boldsymbol{p}}_0)(\boldsymbol{K} - \boldsymbol{\rho}(k)\boldsymbol{M}), \boldsymbol{X})$ that is often poor conditioned.

The static modal derivatives are updated by using a preconditioned conjugate gradient algorithm that also avoids the decomposition of the stiffness matrix. Again the stiffness matrix of the first system is used as preconditioner $\boldsymbol{P} = \boldsymbol{K}^{-1}(\hat{\boldsymbol{p}}_0)$. Thus, only sparse backward- and forward- subsitutions are needed to apply preconditioning.

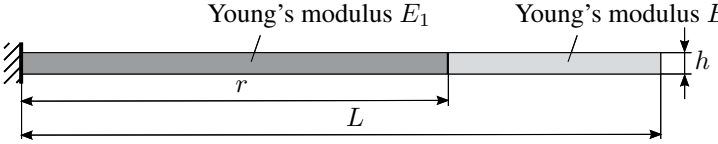
* Corresponding author: e-mail christian.meyer@tum.de, phone +49 89 289 15202, fax +49 89 289 15213

Algorithm 1 Algorithm for updating modes (modified version of [3])

Input: $K, M, P = \boldsymbol{K}^{-1}(\hat{\boldsymbol{p}}_0), X_0, r, tol$
$X \leftarrow X_0$ with $X_0^T M X_0 = I,\ k \leftarrow 0,\ \rho(0) \leftarrow X^T K X$ ▷ initialize variables
while $\|\rho(k) - \rho(k-1)\| > tol \cdot \|\rho(k)\|$ **do** ▷ while rho is not converged
  $Z \leftarrow \mathcal{K}_r(P(K - \rho(k)M), X)$ ▷ build Krylov subspace
  $Z \leftarrow \text{deflate(Z)}$ ▷ assign first significant left singular vectors using a svd
  $K_{red} \leftarrow Z^T K Z,\ M_{red} \leftarrow Z^T M Z$ ▷ compute reduced matrices
  $\omega^2, Q \leftarrow \text{eig}(K_{red}, M_{red})$ ▷ solve interaction problem
  $X \leftarrow Z \cdot Q$ ▷ expand
  M-Normalize columns of X ▷ Normalize eigenvectors
  $\rho(k+1) \leftarrow X^T K X,\ k \leftarrow k+1$ ▷ increment k and assign new rho
return $X, \rho(k)$

3 Case study and conclusion

Figure 1 shows a cantilever beam that is used for a case study. The beam is divided into two parts that are made of hyperelastic materials with different Young's modulus. The parameter vector is defined as $\boldsymbol{p} = E_1$ which is the Young's modulus of the left part. Modes and static derivatives are computed directly for one reference version (0) and iteratively for five updated versions (1-5). Parameter values for E_1 are listed in Figure 1.



Version	0	1	2	3	4	5
E_1/GPa	210	200	190	180	165	150

Fig. 1: Beam for case study with $L = 3\,\text{m}$, $r/L = 2/3$, $h/L = 1/30$, $t/L = 1/3$, where t is thickness in third dimension. Density $\rho = 10^4\,\text{kg/m}^3$, Young's modulus $E_2 = 210\,\text{GPa}$, Poisson's ratio $\nu_1 = \nu_2 = 0.3$, Young's modulus E_1 is parametric and changes according to the table on the right. Version 0 is the reference version for which the modes and static derivatives are computed directly.

Figure 2 shows the results of the relative wall time savings of Algorithm 1 compared to the direct computation with a Lanczos algorithm for different number of computed modes, degrees of freedom and parameter E_1 (version). The algorithm is written in Python and is compared with the `eigsh`-algorithm of the SciPy package [4] that is a wrapper to [5]. The savings do not strongly depend on E_1 but on the number of modes that are updated. Updating more modes leads to less relative savings. However, this dependency is less observable for higher numbers of degrees of freedom.

Updating static derivatives by using the preconditioned conjugate gradient algorithm give remarkable savings only for small models but is not advantageous for large models. The reason for this is that the computation of the tangent stiffness matrices which is not accelerated by the updating algorithm consumes the largest amount of computation time during the update.

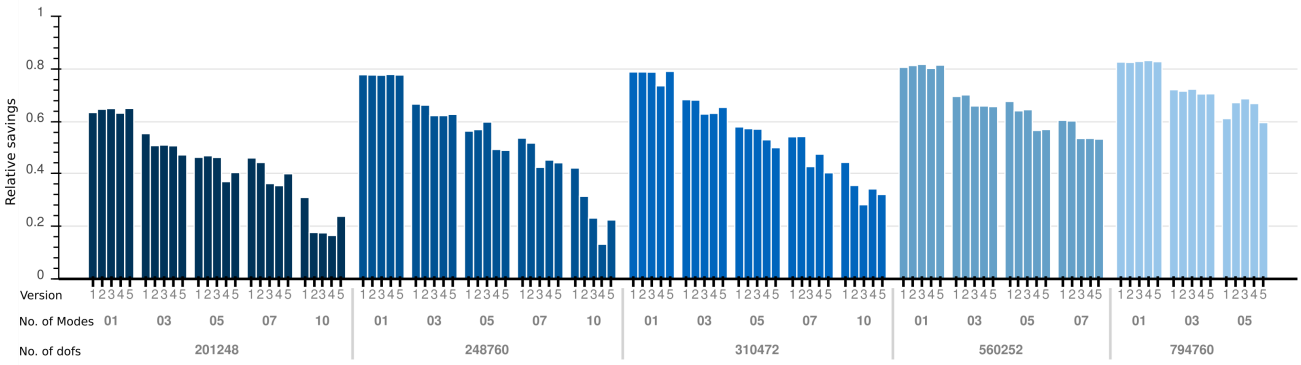


Fig. 2: Relative savings $(t_{\text{direct}} - t_{\text{updated}})/t_{\text{direct}}$ when modes are updated by Alg. 1 with $\text{tol} = 10^{-6}$ compared to a direct computation.

References

[1] C. H. Meyer, C. Lerch, B. Lohmann, and D. J. Rixen, PAMM **17**(1), 37–40.
[2] P. Slaats, J. De Jongh, and A. Sauren, Computers & structures **54**(6), 1155–1171 (1995).
[3] S. Voormeeren and D. Rixen, Computer Methods in Applied Mechanics and Engineering **253**, 39 – 59 (2013).
[4] E. Jones, T. Oliphant, P. Peterson et al., SciPy: Open source scientific tools for Python, 2001–, [Online; accessed 2018-05-15].
[5] R. B. Lehoucq, D. C. Sorensen, and C. Yang, ARPACK users' guide (Siam, 1998).