

PARALLEL HARMONIC BALANCE METHOD FOR ANALYSIS OF NONLINEAR DYNAMICAL SYSTEMS

J. Blahoš*, A. Vizzaccaro, L. Salles, F. El Haddad

Vibration University Technology Centre, Department of Mechanical Engineering
Imperial College London, London, United Kingdom, SW7 2AZ
Email: j.blahos@imperial.ac.uk

ABSTRACT

Controlling vibration in jet engine remains one of the biggest challenges in aircraft engine design and conception. Methods dealing with vibration modelling usually rely on reduced order modelling techniques. This paper aims to provide a high fidelity method to solve vibration problems. It presents a parallel harmonic balance method applied to a full size problem. In order to be computationally efficient, a parallel harmonic balance method is used for the first time in solid mechanics. First, the parallel implementation of harmonic balance method is described in detail. The algorithm is designed to minimize communication between cores. Then, the software is tested for both beam and blade geometries. Finally, a scalability study shows promising acceleration when increasing the number of cores.

INTRODUCTION

Designers of modern aircraft engines face two opposite challenges: increasing specific performance while assuring a higher level of safety compared to the previous generation of engines. High levels of vibration in different components of the engine may lead to high cycle fatigue. For this reason the study of vibration in gas turbines is of extremely high importance. Computer simulations are one of the essential tools in these studies. Contrary to an experiment, a simulation provides a cheap and fast way to assess the quality of a design. Ideally, a high-fidelity model of a whole engine should be used. Historically, this has

not been possible due to the high complexity of a turbine with all its components. Only individual parts have been modelled separately and techniques like reduced order modelling (ROM) [1], cyclic symmetry [2] and others have often been used to further reduce the problem complexity [3]. The state of the art nonlinear vibration analysis relies on nonlinear static analysis performed in a commercial finite element software followed by the computation of the SAFE diagram [4] and finally a nonlinear analysis on a reduced order model [5]. This approach has proven to be efficient and provides accurate results in case of localised nonlinearities, however the time spent by engineers to build the model and run the different steps of the analysis is very important. It also limits the type of analysis that can be run during the design process. In the present work we are targeting the development of a technique that could be used as a black box to build a nonlinear Campbell diagram. Due to the massive increase in computational power of high-performance computers in the last decades, running a simulation of a whole turbine structure at once on a fine scale is potentially within reach.

As a first step towards a full nonlinear harmonic balance finite element software (HBFEM), a solver for geometric nonlinearities in parallel HBM has been implemented. The geometric nonlinear effect needs to be taken into account when computing response of a blade under centrifugal loading that varies with rotational speed [6]. The present paper presents the implementation of HBFEM in C++ and the results obtained with the proposed methods.

Implementations of nonlinear HBM for large systems has al-

*Address all correspondence to this author.

ready been researched in electrical circuits [7] and electromagnetism [8]. However, this has not been the case in structural dynamics.

Geometric nonlinearities have been studied in the past using simple models and reduced order modelling. Most approaches to build the reduced order model with geometric nonlinear effect rely on non-intrusive techniques [9, 10]. Some researchers proposed intrusive techniques [11]. Other researchers directly implemented nonlinear finite element analysis but it was limited to simple elements [12] or a limited number of degrees of freedom [13]. Weegers [14] seems to be the only work that showed the implementation of finite element with harmonic balance method for nonlinear vibration analysis. However, there is no mention of parallel programming in their work and for large size models they rely on ROM.

In the present work we propose a parallel implementation of HBFEM, that allows to solve geometrically nonlinear unreduced large scale models. The importance of such a solver in relation to ROMs is twofold. Firstly, the validity range of ROMs is limited by the vibration amplitude in that they normally rely on the hypothesis of second order dynamics, as the quadratic manifold approach [10], or at most five order dynamics as the normal form approach [15]. Conversely, the range of validity of a full model computation is only limited by the number of harmonics used and this number can be increased with no change in the code by any user, at the only expense of the computational cost. Secondly, to validate newly proposed ROMs, researchers rely on results obtained with commercial softwares where FRFs can only be computed through time integration sweeps; depending on the size of the model, such procedures can be time consuming. Overall, the proposed parallel solver for unreduced models not only allows to improve and validate existing ROMs, but is also able to compete with them in terms of computational time.

The paper is organized as follows. First the proposed numerical methods are presented followed by their implementation in C++. Finally numerical results on different test cases illustrate the capability of the code. Preliminary results on the scalability of the proposed approach are given in the result section.

HARMONIC BALANCE METHOD

Harmonic Balance Method (HBM) is a common tool used in vibration analysis [16, 17]. By assuming a periodic motion of the system, the time variable is eliminated and therefore we can avoid performing potentially many computationally expensive time integration steps. The assumption of periodicity is reasonable since we are only interested in the steady state motion of the structure. We start with the standard discrete equations of motion arising from a finite element formulation:

$$M\ddot{u} + D\dot{u} + F_{int}(u, \dot{u}) = F_{exc}(t) + F^{nl}(u, \dot{u}) \quad (1)$$

where $F_{int}(u, \dot{u})$ represents the internal forces of the structure. It can be split into its linear and nonlinear component in u as:

$$F_{int}(u, \dot{u}) = Ku + F_{int}^{nl}(u, \dot{u}) \quad (2)$$

$F^{nl}(u, \dot{u})$ stands for possible external nonlinear terms. u is used instead of $u(t)$ for brevity. The K , M and D matrices are the stiffness, mass and damping matrix respectively. A periodic excitation is applied, i.e.:

$$F_{exc}(t) = A \cos(\omega t + \phi) \quad (3)$$

with ω being the excitation frequency and ϕ the excitation phase. We then rearrange (1) into residual form:

$$R(t) = M\ddot{u} + D\dot{u} + F_{int}(u, \dot{u}) - A \cos(\omega t + \phi) - F^{nl}(u, \dot{u}) \quad (4)$$

The periodic motion assumption is then formulated as follows:

$$u(t) = \tilde{u}_0 + \sum_{k=1}^m \tilde{u}_k^c \cos(k\omega t) + \tilde{u}_k^s \sin(k\omega t) \quad (5)$$

where \tilde{u}_0 , \tilde{u}_k^c and \tilde{u}_k^s are vectors of coefficients for constant, cosine and sine components of displacement. Collectively we will express all these coefficients by vector \tilde{u} . The higher harmonic terms ($k > 1$) are present to account for nonlinear behaviour. We can also express first and second derivative with respect to time as:

$$\begin{aligned} \dot{u}(t) &= \omega \sum_{k=1}^m k [-\tilde{u}_k^c \sin(k\omega t) + \tilde{u}_k^s \cos(k\omega t)] \\ \ddot{u}(t) &= -\omega^2 \sum_{k=1}^m k^2 [\tilde{u}_k^c \cos(k\omega t) + \tilde{u}_k^s \sin(k\omega t)] \end{aligned} \quad (6)$$

Projection on the same harmonic functions that were used in (5) is used to enforce residual orthogonality:

$$\begin{aligned} \frac{2}{T} \int_0^T R(t) dt &= 0 \\ \frac{2}{T} \int_0^T R(t) \cos(k\omega t) dt &= 0, & \text{for } k = 1..m \\ \frac{2}{T} \int_0^T R(t) \sin(k\omega t) dt &= 0, & \text{for } k = 1..m \end{aligned} \quad (7)$$

where $\frac{2}{T}$ is a scaling constant picked for convenience purposes. These three equations define constant (if present) and cosine and

sine components for each harmonic (k) of the force vectors in frequency domain. As the result we obtain nonlinear algebraic equations in frequency domain with unknowns being the coefficients \tilde{u} :

$$Z(\omega)\tilde{u} + \tilde{F}_{int}^{nl}(\tilde{u}, \omega) - \tilde{F}^{nl}(\tilde{u}, \omega) = \tilde{F}_{exc} \quad (8)$$

Z is the dynamic stiffness matrix which represents the linear part of the system. \tilde{F}_{nl} and \tilde{F}_{int}^{nl} include nonlinear effects. Z can be assembled from matrices K , M and D directly by analytically evaluating (7). When unknowns are grouped by harmonics, Z has a block diagonal structure where each block is in form:

$$\begin{bmatrix} K - (k\omega)^2 M & -k\omega D \\ k\omega D & K - (k\omega)^2 M \end{bmatrix} \quad (9)$$

where k stands for different harmonics as in (5). The exception is for $k = 0$ (i.e. for \tilde{u}_0) where the block is simply $2K$ instead. However, this ordering of dofs is not optimal from the communication perspective. Therefore, a different ordering was used and is discussed in the implementation section. \tilde{F}_{exc} can also be obtained analytically from $F_{exc}(t)$. It is important to note that the vectors of forces \tilde{F}_{exc} , $\tilde{F}_{int}^{nl}(\tilde{u}, \omega)$ and $\tilde{F}^{nl}(\tilde{u}, \omega)$ arise from projecting the nonlinear forces in time onto the harmonic functions base over the period T and are therefore only the best approximation within this base.

NONLINEARITY

We used the Green-Lagrange strain formulation for large deformations as our source of nonlinearity [18]:

$$E(u) = \frac{1}{2} \left(\nabla u + (\nabla u)^T + \nabla u (\nabla u)^T \right) \quad (10)$$

Here (only in this section) u stands for the displacement field as a function of space and time. Using the second Piola-Kirchhoff stress tensor, the finite element formulation for forces contribution of an element e to node i is:

$$F_{int}^{(e,i)}(u) = \int_{\Omega} S(u) \delta E(u, v_i) dV \quad (11)$$

where v_i is the corresponding shape function and Ω represents the element volume.

$$S(u) = CE(u) \quad (12)$$

where C is the tensor of coefficients describing the stress-strain relationship.

$$\delta E(u, v_i) = \frac{1}{2} \left((\nabla u)^T \nabla v_i + \nabla u (\nabla v_i)^T + \nabla v_i + (\nabla v_i)^T \right) \quad (13)$$

In (2), we split the force from (11) into its linear part Ku and the rest denoted by $F_{int}^{nl}(u, \dot{u})$.

NEWTON-RAPHSON

The equation (8) defines a system of nonlinear algebraic equations with unknowns \tilde{u} . We assume ω to be a fixed parameter for now. We use the basic version of the Newton-Raphson algorithm to solve this system. Putting all terms in (8) on one side and grouping them under one function G , we obtain:

$$G(\tilde{u}) = Z(\omega)\tilde{u} + \tilde{F}_{int}^{nl}(\tilde{u}, \omega) - \tilde{F}^{nl}(\tilde{u}, \omega) - \tilde{F}_{exc} \quad (14)$$

The Newton-Raphson iteration is then as follows:

$$\nabla G(\tilde{u}^k) \Delta \tilde{u}^k = \nabla G(\tilde{u}^k) (\tilde{u}^{k+1} - \tilde{u}^k) = -G(\tilde{u}^k) \quad (15)$$

with \tilde{u}^{k+1} being a new guess of the solution, based on the previous one \tilde{u}^k . The initial guess \tilde{u}^0 needs to be picked at the start of the iteration process. The matrix ∇G , i.e. the matrix of derivatives of G with respect to \tilde{u} , is:

$$\nabla G(\tilde{u}) = Z(\omega) + \nabla \tilde{F}_{int}^{nl}(\tilde{u}, \omega) - \nabla \tilde{F}^{nl}(\tilde{u}, \omega) \quad (16)$$

JACOBIAN MATRIX

In this section we address how the Jacobian matrix defined in (16) is evaluated. The constant part $Z(\omega)$ was already shown in (9) and is straightforward to assemble. We shall now show evaluation of $\nabla \tilde{F}_{int}^{nl}(\tilde{u}, \omega)$. Identical procedure can be applied to $\nabla \tilde{F}^{nl}(\tilde{u}, \omega)$. For example, from (7), the cosine components of second harmonic ($k = 2$) of the force vector $\tilde{F}_{int}^{nl}(\tilde{u}, \omega)$ are defined as:

$$\frac{2}{T} \int_0^T F_{int}^{nl}(u, \dot{u}) \cos(2\omega t) dt \quad (17)$$

Let's assume for simplicity now that $F_{int}^{nl}(u, \dot{u})$ only depends on u and not \dot{u} , i.e. in this section $F_{int}^{nl}(u, \dot{u}) = F_{int}^{nl}(u)$. We then differentiate (17) for example w.r.t. the first harmonic sine component

of \tilde{u} , i.e. \tilde{u}_1^s :

$$\frac{\partial}{\partial \tilde{u}_1^s} \left[\frac{2}{T} \int_0^T F_{int}^{nl}(u) \cos(2\omega t) dt \right] \quad (18)$$

By moving the derivative operator inside the integral and applying chain rule $\frac{\partial}{\partial \tilde{u}_1^s} = \frac{\partial}{\partial u} \frac{\partial u}{\partial \tilde{u}_1^s}$ we obtain:

$$\frac{2}{T} \int_0^T \frac{\partial}{\partial u} F_{int}^{nl}(u) \frac{\partial u}{\partial \tilde{u}_1^s} \cos(2\omega t) dt \quad (19)$$

From (5) we note that $\frac{\partial u}{\partial \tilde{u}_1^s} = \sin(\omega t)$. This gives us the final expression:

$$\frac{2}{T} \int_0^T \frac{\partial}{\partial u} F_{int}^{nl}(u) \sin(\omega t) \cos(2\omega t) dt \quad (20)$$

Same procedure can be applied to any combination of components of the force vector and \tilde{u} . We can express this collectively as:

$$\nabla \tilde{F}_{int}^{nl}(\tilde{u}, \omega) = \frac{2}{T} \int_0^T \nabla F_{int}^{nl}(u) B_i(t) B_j(t) dt \quad (21)$$

where $B_i(t)$ and $B_j(t)$ each stand for one of either 1, $\cos(k\omega t)$ or $\sin(k\omega t)$ functions. It is easy to see that if we included \dot{u} in $F_{int}^{nl}(u, \dot{u})$, the procedure would be analogous with the chain rule being extended to $\frac{\partial}{\partial \tilde{u}_1^s} = \frac{\partial}{\partial u} \frac{\partial u}{\partial \tilde{u}_1^s} + \frac{\partial}{\partial \dot{u}} \frac{\partial \dot{u}}{\partial \tilde{u}_1^s}$. It can be seen from (21) that the off diagonal elements of the Jacobian matrix (that is, elements belonging to a combination of different base harmonic functions) won't generally be zero. This creates a coupling between different harmonics in (15) which is not present in the linear part $Z(\omega)$.

Alternating Frequency-Time Procedure (AFT)

To obtain $\tilde{F}_{int}^{nl}(\tilde{u}, \omega)$ and $\tilde{F}_{int}^{nl}(\tilde{u}, \omega)$ from $F_{int}^{nl}(u, \dot{u})$ and $F^{nl}(u, \dot{u})$ respectively, in general case a numerical approximation has to be used when evaluating the integrals in (7). In this work, we use a simple rectangular integration scheme, using N_t integration points evenly distributed with step $h_t = \frac{T}{N_t}$:

$$\begin{aligned} \frac{2}{T} \int_0^T f(t) dt &\approx \frac{2}{T} \sum_{n=0}^{N_t-1} h_t f(t_n) \\ &= \frac{2}{T} \sum_{n=0}^{N_t-1} \frac{T}{N_t} f(nh_t) \\ &= \frac{2}{N_t} \sum_{n=0}^{N_t-1} f(nh_t) \end{aligned} \quad (22)$$

where $f(t)$ represents an arbitrary function. To compute $\tilde{F}_{int}^{nl}(\tilde{u}, \omega)$, we first need to reconstruct the displacement in time for each time sample using (5):

$$u(t_n) = u_n = \tilde{u}_0 + \sum_{k=1}^m \tilde{u}_k^c \cos(k\omega t_n) + \tilde{u}_k^s \sin(k\omega t_n) \quad (23)$$

where $t_n = nh_t$. The derivative \dot{u}_n can be obtained in the same way from (6). The nonlinear force in time is then evaluated for each time sample:

$$F_{int,n}^{nl} = F_{int}^{nl}(u_n, \dot{u}_n) \quad (24)$$

Components of $\tilde{F}_{int}^{nl}(\tilde{u}, \omega)$ are computed by (7) using $F_{int,n}^{nl}$ for numerical integration formulated by (22). This completes the cycle of transformations from frequency domain into time domain ($\tilde{u} \rightarrow u$) and back to frequency ($F^{nl} \rightarrow \tilde{F}^{nl}$).

An analogous approach can be used for the Jacobian matrix. Expression (21) can be evaluated numerically using the same integration procedure. We can use analytical Jacobian matrices of the force vectors in time domain, only the transformation to frequency has to be approximated numerically.

CONTINUATION

Previously we assumed that the frequency parameter ω in (8) was fixed. However, typically we are interested in finding a set of solution pairs $\{(\tilde{u}, \omega)\}$ for $\omega \in \langle \omega_{min}, \omega_{max} \rangle$. This set is commonly called a frequency response function (FRF).

For nonlinear systems, there can be generally multiple solutions for certain frequencies. If we use the Newton iterative procedure as described in (15), we might obtain one of these solutions, or the solver will not converge at all, depending on the initial guess \tilde{u}^0 . Therefore, the continuation algorithm uses an already known solution \tilde{u}_i , obtained for a certain frequency ω_i , to generate an initial guess for finding solution for a new

frequency ω_{i+1} which is in the proximity of ω_i . Using this approach repetitively, a discrete set of solutions (\tilde{u}_i, ω_i) is obtained as the FRF. The first solution of the FRF curve has to be computed without any previous known solution. This is usually done for a frequency where the system is believed to behave almost linearly and the linear solution is used as the initial guess. The continuation algorithm takes advantage of the fact that the Newton iteration has second order convergence when the initial guess is close to a solution. This means that usually only few iterations are required for convergence if a good initial guess is generated. An example of an FRF curve for nonlinear problem represented by a 2D graph can be seen in Fig. 1. The amplitude is computed as the maximum absolute displacement over the period T .

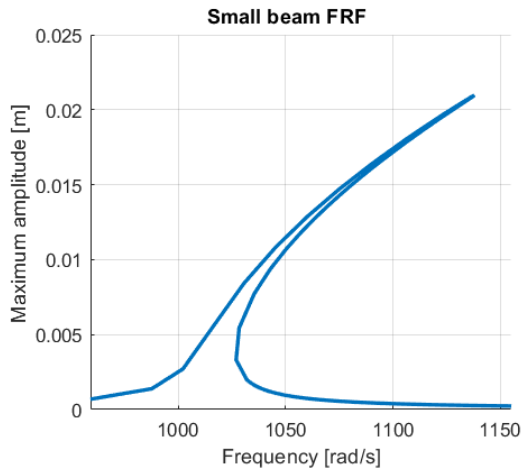


FIGURE 1: Example of an FRF curve of a clamped-clamped beam with geometric nonlinearity. The Y axis represents amplitude of displacement (as expressed in (5)) over the period $T = \frac{2\pi}{\omega}$.

In this work, a simple frequency stepping continuation technique was used. Solution \tilde{u}_i is used directly as the initial guess for frequency $\omega_{i+1} = \omega_i + h$, where h is the size of the step in frequency. If the Newton solver does not converge in given number of iterations, the step h is shortened and a new solve is attempted for a closer frequency. The value of h never changes its sign, the stepping always only goes in one direction. Therefore, the main disadvantage of this approach is that it cannot capture turning points, i.e. points where the curve changes its direction with respect to frequency. The advantages are the simplicity of its implementation and virtually no additional computational cost.

IMPLEMENTATION

In order to efficiently solve (8) with large number of unknowns, parallel algorithms need to be employed. We developed a C++ code using MPI (Message Passing Interface) to run on large distributed memory systems. Object oriented programming model was used for code modularity and clarity.

At input the code loads a configuration file and a mesh file. The mesh is loaded and processed by a 3rd party mesh processing package [19]. After loading the mesh, its elements and nodes are uniquely distributed among MPI ranks. For each node, 3 degrees of freedoms are registered - x, y and z displacement. All these dofs are collectively represented by the u vector in (1). A scheme illustrating a mesh distribution is in Fig. 2.

The vector u is then extended into the vector of frequency domain unknowns \tilde{u} . The way this extension is done can be seen in Fig. 3. This ordering ensures that each dof is owned by the same process as its corresponding node.

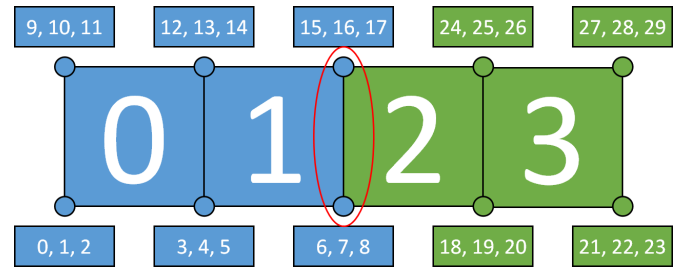


FIGURE 2: Mesh decomposition scheme. The four elements and their nodes are both uniquely distributed among 2 MPI ranks (represented by the blue and green colour). Numbers around the nodes show possible physical dof (vector u) assignment. The red ellipse highlights domain boundary nodes.

For matrices, the CSR (compressed sparse row) matrix format is used [20]. Each MPI rank owns matrix rows corresponding to its owned dofs. The matrices K , M , D and $\nabla F_{int}(u)$ for large deformations are assembled in parallel - each rank assembles element matrices for its elements and adds them into the global matrix, as shown in Fig. 2 and 4. Communication needs to be performed for rows corresponding to domain boundary dofs. Matrices K , M and D are assembled once at the beginning of the execution and buffered as they are constant.

Assembly of the Jacobian matrix in frequency domain (16) for the Newton algorithm is described by algorithm 1. Only F_{int}^{nl} is included in the algorithm since this stands for the Green-Lagrange strain nonlinearity which is the only nonlinearity used right now. However, the procedure would be the same for any other nonlinear term. Assembly of $\nabla \tilde{F}_{int,n}^{nl}$ from $\nabla F_{int,n}^{nl}$ at line 9 doesn't require any additional communication since matrices

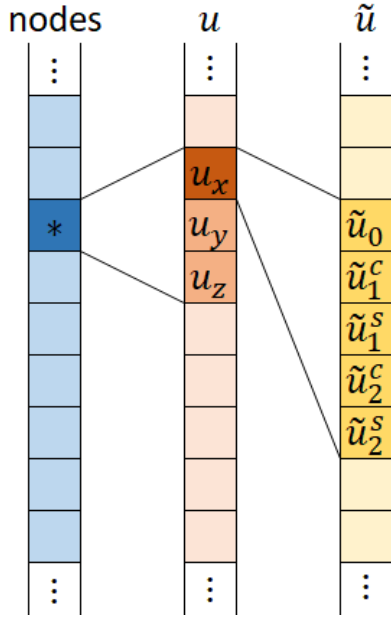


FIGURE 3: Extension of list of nodes into vector of physical dofs and subsequently into vector of frequency domain dofs.

in frequency domain maintain the same row distribution, only expanded accordingly to the dof expansion as shown in Fig. 3. Same applies to the assembly of $Z(\omega)$ from matrices K , M , D , performed at line 3.

Algorithm 1 Jacobian matrix assembly. MPI communication is performed at lines 5 and 8.

```

1: function COMPUTEJAC( $\tilde{u}$ ,  $\omega$ )
2:    $J = 0$   $\triangleright$  initialise the Jacobian matrix by zeros
3:    $J += Z(\omega)$   $\triangleright$  add linear part
4:   if use Green-Lagrange then
5:     communicate  $\tilde{u}$  on domain boundary nodes
6:     for  $n = 0$  to  $N_t - 1$  do
7:       evaluate  $u(t_n)$   $\triangleright$  see (23)
8:       evaluate  $\nabla F_{int}^{nl}(u(t_n))$ 
9:        $\nabla \tilde{F}_{int,n}^{nl} := \nabla F_{int}^{nl}(u(t_n))B_i(t_n)B_j(t_n)$   $\triangleright$  see (21)
10:       $J += \frac{2}{N_t} \nabla \tilde{F}_{int,n}^{nl}$   $\triangleright$  see (22)
11:     end for
12:   end if
13:   return  $J$ 
14: end function

```

For continuation, the code runs a while loop which iterates over a frequency range $\langle \omega_{min}, \omega_{max} \rangle$. Currently there is only a

Algorithm 2 Simple continuation loop

```

1:  $\omega = \omega_{min}$ 
2:  $h_\omega = h_{\omega,init}$ 
3:  $\tilde{u}_{prev} = 0$ 
4:  $\omega_{prev} = 0$ 
5: while  $\omega \leq \omega_{max}$  do
6:    $\tilde{u} = \text{NewtonSolve}(\tilde{u}_{prev}, \omega)$   $\triangleright \tilde{u}_{prev}$  used as initial guess
7:   if converged then
8:     save  $\tilde{u}$ 
9:      $\tilde{u}_{prev} = \tilde{u}$ 
10:    increase  $h_\omega$ 
11:     $\omega_{prev} = \omega$ 
12:     $\omega += h_\omega$ 
13:   else
14:     if  $\omega = \omega_{min}$  then
15:       break
16:     end if
17:     decrease  $h_\omega$ 
18:     if  $h_\omega < h_{\omega,min}$  then
19:       break
20:     end if
21:      $\omega = \omega_{prev} + h_\omega$ 
22:   end if
23: end while

```

simple version of a continuation algorithm implemented in the code. The code performs steps in frequency in one direction. During Newton iterations, the frequency is fixed. The step size is adapted based on the Newton convergence. Once a turning point is reached, the loop breaks. The procedure is described in algorithm 2.

Figure 5 shows collaboration of the main parts of the code. Each block represents a C++ class in the code. When the Newton-Raphson solver is called (line 6 in algorithm 2), it asks the HBM interface for evaluation of matrix $G(\tilde{u})$ and vector $\nabla G(\tilde{u})$ and feeds those into a linear solver. The code provides a general interface for linear solver that can be implemented using various packages or in house algorithms. The parallel sparse direct solver MUMPS based on LU factorisation [21] was used in this study.

TEST CASES

Four test cases were used for code validation and performance assessment. Their list and a brief description is given in table 1. All tests were run on the Salomon supercomputer at IT4Innovations, Ostrava, Czech Republic. Each computational node of Salomon has 2×12 computational cores (Intel Xeon(R) E5-2680 v3 2.50GHz), 32 MB of cache and 128 GB RAM memory. The nodes are interconnected by Infiniband FDR system.

For all tests, the stopping condition for the Newton solver

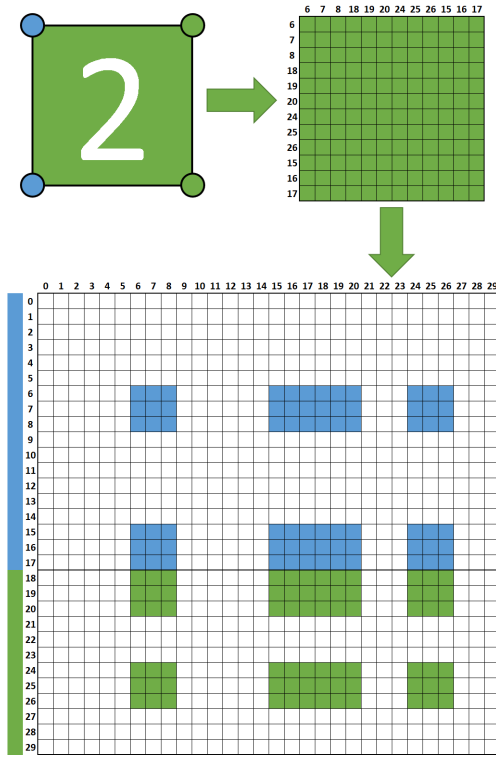


FIGURE 4: Assembly of a global matrix from element matrices, based on mesh example in Fig. 2. The top square represents a matrix for element 2 which is computed on green rank and added into the bottom global matrix. Some of its values need to be communicated to the blue rank since it owns some of the nodes incident with element 2.

was norm of the residual, more precisely:

$$\frac{\|G(\tilde{u})\|}{n} \leq 10^{-8} \quad (25)$$

where n here represents the number of equations in the algebraic system.

All test case meshes were composed solely of HEXA20 elements. All FRFs were computed around the first natural frequency of the corresponding structure.

Beam test-cases

The small beam and cantilever test cases were used for validation of the code results. The code performance on these test cases is not addressed as the meshes are very small. The clamped-clamped beam testcase is described in Fig. 6a and table 2. The cantilever test case is in Fig. 6b and table 3.

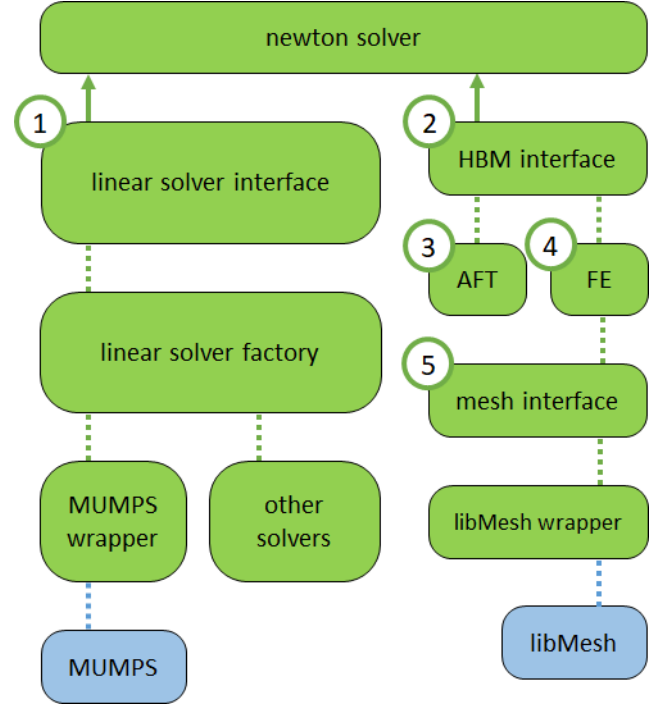


FIGURE 5: Collaboration diagram of the main parts of the code. 1) Interface to a linear solver implementation, 2) interface to HBM formulation, 3) provides AFT procedures, 4) interface to finite elements implementation that provides matrices and vectors in (1), 5) provides interface to mesh data

Name	Type	# elements	# nodes
Small beam	Clamped-clamped beam	60	471
Large beam	Clamped-clamped beam	17,024	81,621
Cantilever	Cantilever beam	400	3,093
Blade	Blade	7,722	41,021

TABLE 1: List of used test cases

The large beam test case was used for scalability assessment. It was run on 32 computational nodes. The parameters of this test case are the same as for the small beam, with only difference being the mesh size.

For all test cases, 3 harmonics with the constant term ($2 \times 3 + 1$ coefficients) were used. For the cantilever beam, we additionally also used 7 harmonics to study the convergence w.r.t. the number of harmonics and validate the implementation against the results available in the literature [22]. The displacement was

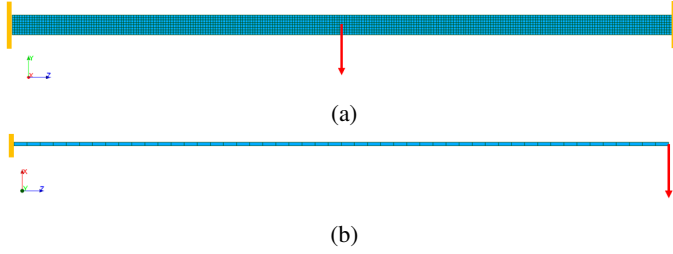


FIGURE 6: Clamped-clamped beam test case (a) and cantilever beam test case (b). The yellow lines highlight clamped faces. Excitation force (indicated by the red arrow) was applied to a single node in the middle (along all 3 axes) of the beam (a) and in the middle of the free end of the beam (b).

Size [m]	$1 \times 0.03 \times 0.03$
Density [kg/m^3]	7800
Young's modulus [N/m^2]	2.1×10^{11}
Poisson's ratio []	0.3
Damping matrix	$D = 3 \times M$
Excitation amplitude [N]	200

TABLE 2: Parameters of the clamped-clamped beam test case (applies to both small and large version)

Size [m]	$0.005 \times 0.1 \times 1$
Density [kg/m^3]	4400
Young's modulus [N/m^2]	1.4×10^{11}
Poisson's ratio []	0.3
Damping matrix	$D = 0.2467 \times M$
Excitation amplitude [N]	2
Normalising frequency f_0 [rad/s]	24.8945

TABLE 3: Parameters of the cantilever test case

measured in direction of the excitation force.

Fan blade test-case

The code was also tested on a fan blade geometry representative of an industrial application. For this test, 2 computational nodes and 48 MPI processes were used for each part of the FRF. The mesh is shown in Fig. 7. 3 harmonics plus constant term

were again used. The mesh size converts to around 125000 dofs. This model is quite small compared to a standard model used in industry for accurate linear modal analysis. A sector of a bladed-disk is studied and the cyclic boundaries are clamped, which is coloured in yellow in the Fig. 7. The blade is tied with the disk and no displacement is allowed at the interface. A single point harmonic excitation is applied in the y direction and is shown as a red arrow in the figure. The displacement was measured in direction of the surface normal for the blade case.

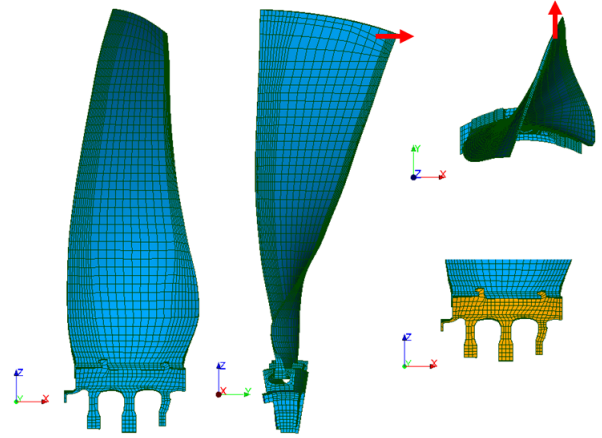


FIGURE 7: Mesh for the blade test case. The yellow faces in the bottom right picture indicate zero Dirichlet boundary condition. Same is set for the other side. The red arrow indicates the direction and approximate location of the applied excitation force.

RESULTS

Validation

The validation of the results obtained with the code is done both against existing results from the literature and against reduced order models. ROMs of the small beam test-case and of the cantilever beam test-case are built by means of static modal derivatives and quadratic manifold approach [10]. The static modal derivatives are computed in the commercial software Code-Aster and the reduced order model is build in Matlab; the continuation is performed with HBM in the open source code Manlab using 3 harmonics.

The frequency response around the first bending mode for the clamped-clamped beam is shown in Fig. 8. The obtained result agrees well with results of the literature [23, 13] and with the result of the ROM.

The FRF for the first flap of a cantilever beam is shown in Fig. 9. The cantilever test case was designed to have the same

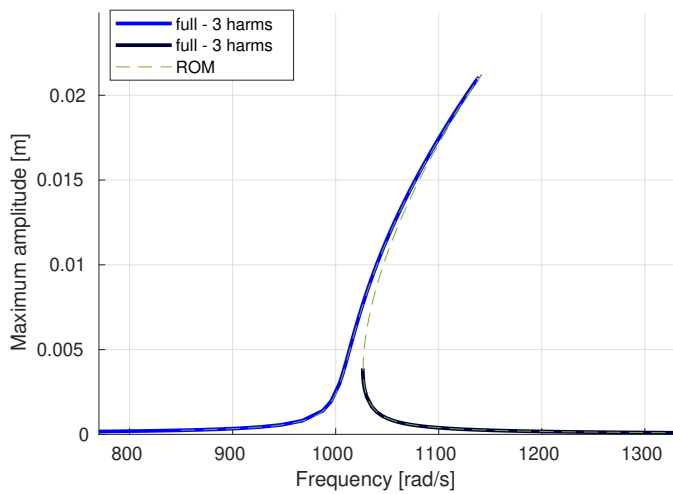


FIGURE 8: FRF obtained for the small clamped-clamped beam test case.

dimensions of the one in [22], where a finite element model composed of 200 beam elements and time integration were used. The obtained FRFs with 7 harmonics matches quite well with the one obtained by Thomas *et al.* with time integration. The difference in behaviour is justified by the different type of finite element used. Conversely, the result obtained with the ROM on the same mesh matches the result of the full model with 7 harmonics. It can be seen that, for a cantilever beam, a higher number of harmonics is necessary to accurately predict the dynamical behaviour. The explanation for this behaviour is that higher order dynamics are involved at the level of excitation tested [15].

Scalability test

Strong scalability tests have been performed on the large beam test case from table 1. The code was run using following number of MPI processes: 4, 8, 16, 32, 64, 128, 256, 384. The nonlinear problem has been solved for a single frequency and 4 Newton iterations were performed to obtain convergence. This means that the parts of the code that need to be performed only once (such as assembly of matrices K , M and D or MUMPS analysis phase) will not greatly influence the overall run time of the code. The obtained run time, speed-up and efficiency can be seen in Fig. 10, 11 and 12 respectively. Various parts of the code were measured individually to assess their performance separately. The MUMPS solver works in 3 stages - analysis (required only once for given matrix pattern), factorisation and solve. The pre- and post-processing tasks are related to data passing between the main code and the solver. The total run time decrease is from 1.47 hours for 4 MPIs to 0.18 hours for 384 MPIs. This corresponds to a speed-up of 7.94 or efficiency of

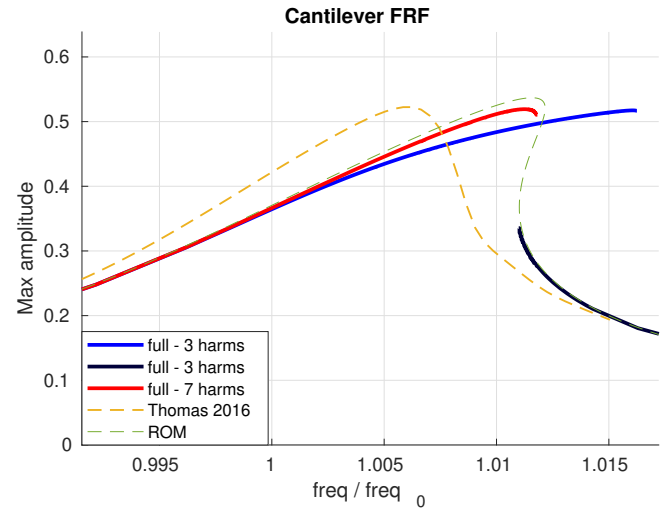


FIGURE 9: FRF obtained for the cantilever test case.

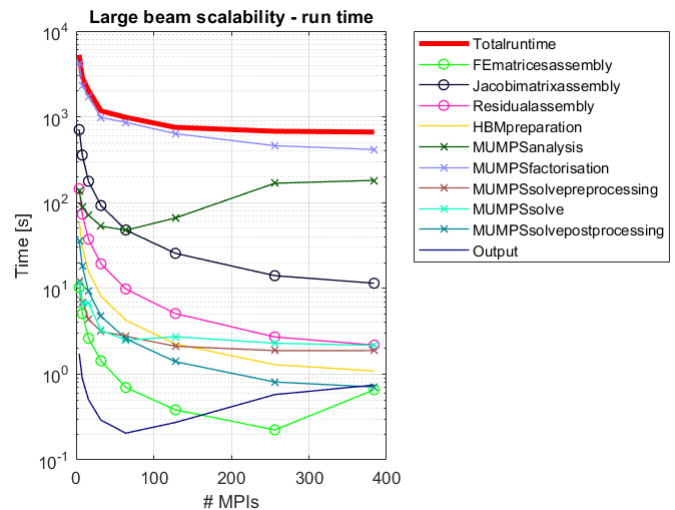


FIGURE 10: Wall time used for the individual parts of the code.

8%.

As expected, direct sparse factorization is the bottleneck for reaching good scalability. From Fig. 12 it can be concluded that for a model of this size running on more than 64 cores does not pay so much. It was not possible to increase the size of the tested model. During tests with larger meshes, the MUMPS solver crashed during the analysis stage when the number of MPI became too large. The issue was discussed with MUMPS developers and the conclusion is that parmetis allocates too much memory when used with this high number of MPIs. MUMPS supports

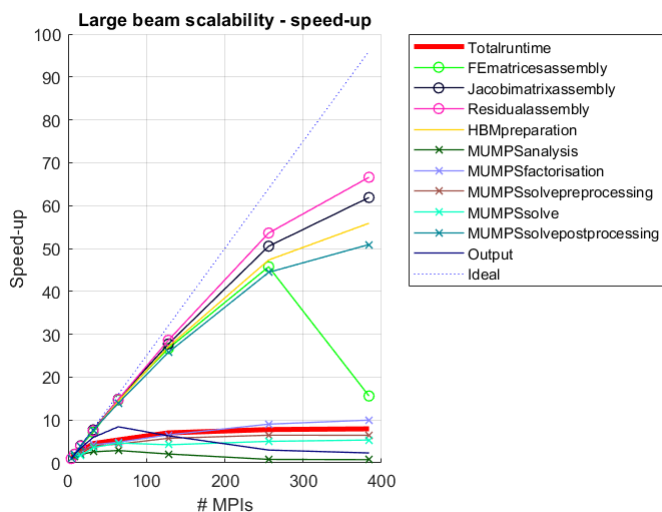


FIGURE 11: Strong scalability speed-up of the individual parts of the code.

besides MPI also OpenMP threads, which means the number of MPIs can be lower than the total number of cpus used. However, the code we developed currently only uses MPI so we are unable to take advantage of this option. In order to achieve better scalability it is necessary to switch to iterative solvers coupled with efficient preconditioners [24]. It will be the focus of the next development of our software.

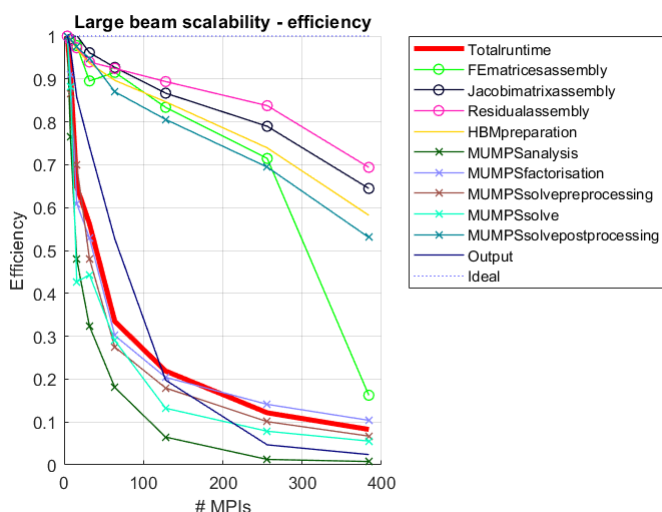


FIGURE 12: Efficiency plot of the individual parts of the code. Efficiency was computed as speed-up / ideal speed-up.

Curve	Total time [h]	# points	Avg. time / point [s]
4N	4.73	146	116.55
7N	6.62	147	161.99
10N (f)	14.23	120	426.75
10N (b)	6.32	85	267.65
20N (f)	106.73	906	424.08
20N (b)	6.42	80	288.84

TABLE 4: Computational time for the blade FRF. The (f) and (b) marks indicate forward and backward frequency sweep.

Fan bladed-disk

Finally, the code was used to build the FRF around the first flap mode of a fan blade taking into account the geometric nonlinearities. The different FRFs are shown in Fig. 13 for different amplitudes of excitation. As expected for a cantilever configuration a large excitation is needed to see the effect of the nonlinearities. The stiffening is quite strong for a 20 N excitation and is due to a modal interaction with other modes. Those results show the feasibility of nonlinear vibration analysis of a full model finite element model and their readiness for industrial application. The computational times needed for this application are reported in the table 4.

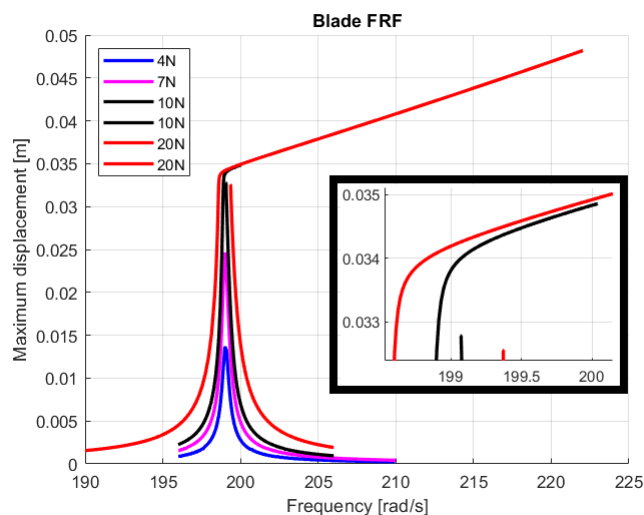


FIGURE 13: FRF obtained for the blade test case. Displacement was measured in direction of the normal vector.

CONCLUSIONS AND FUTURE WORK

We have developed a code capable of solving nonlinear vibration problems using HBM on large computer clusters, using some of the state of art algorithms and software for parallel computing. The code results have been validated with two standard test cases - a clamped-clamped beam and a cantilever beam. The nonlinear frequency response of a realistic fan bladed-disc has been built and it shows strong nonlinear behaviour for a large excitation. This application demonstrates the feasibility of harmonic balance finite element method for large scale industrial application. We are convinced that it will open a new area in nonlinear vibration analysis.

Use of parallel programming permits to solve larger systems in much faster times than it would be possible with a single cpu. However, the overall scalability of the code is far from ideal. Looking at the breakdown of performance of individual parts of the code, we can see that the most time consuming part of the code is the LU factorisation. Therefore, this part influences the scalability the most. Our next work will be therefore focused on finding a more optimal solver to improve the overall performance.

ACKNOWLEDGMENT

This project has received funding from the European Union's Horizon 2020 Framework Programme research and innovation programme under the Marie Skłodowska-Curie agreement No 721865. Dr Loïc Salles and Dr Fadi El Haddad thank Rolls-Royce plc and the EPSRC for the support under the Prosperity Partnership Grant "Cornerstone: Mechanical Engineering Science to Enable Aero Propulsion Futures", Grant Ref: EP/R004951/1. The authors thank the IT4Innovations National Supercomputing Center for their help with the compilation of the software ParHBM on the Salomon cluster and the granted core hours that made the scalability tests of ParHBM possible.

REFERENCES

- [1] Yuan, J., El-Haddad, F., Salles, L., and Wong, C., 2019. "Numerical assessment of reduced order modeling techniques for dynamic analysis of jointed structures with contact nonlinearities". *Journal of Engineering for Gas Turbines and Power*, **141**(3), p. 031027.
- [2] Petrov, E., 2004. "A method for use of cyclic symmetry properties in analysis of nonlinear multiharmonic vibrations of bladed disks". *J. Turbomach.*, **126**(1), pp. 175–183.
- [3] Seinturier, E., 2007. "Forced response computation for bladed disks industrial practices and advanced methods". *Lecture Series-Von Karman Institute For Fluid Dynamics*, **2**, p. 5.
- [4] Singh, M. P., Vargo, J. J., Schiffer, D. M., Dello, J. D., et al., 1988. "Safe diagram-a design and reliability tool for turbine blading". In *Proceedings of the 17th Turbomachinery Symposium*, Texas A&M University. Turbomachinery Laboratories.
- [5] Petrov, E., and Ewins, D., 2004. "State-of-the-art dynamic analysis for non-linear gas turbine structures". *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, **218**(3), pp. 199–211.
- [6] Sternchüss, A., and Balmes, E., 2006. "On the reduction of quasi-cyclic disk models with variable rotation speeds". In *ISMA*.
- [7] Mehrotra, A., and Somani, A., 2009. "A robust and efficient harmonic balance (hb) using direct solution of hb jacobian". In *Proceedings of the 46th Annual Design Automation Conference, DAC '09*, Association for Computing Machinery, p. 370–375.
- [8] Yao, W., Jin, J.-M., and Krein, P., 2015. "A 3d finite element analysis of large-scale nonlinear dynamic electromagnetic problems by harmonic balancing and domain decomposition". *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields*, **29**, 03.
- [9] Givois, A., Grolet, A., Thomas, O., and Deü, J.-F., 2019. "On the frequency response computation of geometrically nonlinear flat structures using reduced-order finite element models". *Nonlinear Dynamics*, **97**(2), pp. 1747–1781.
- [10] S., J., P., T., B., R. J., and J., R. D., 2017. "A quadratic manifold for model order reduction of nonlinear structural dynamics". *Computers and Structures*, **188**, pp. 80–94.
- [11] Touzé, C., Vidrascu, M., and Chapelle, D., 2014. "Direct finite element computation of non-linear modal coupling coefficients for reduced-order shell models". *Computational Mechanics*, **54**(2), pp. 567–580.
- [12] Lewandowski, R., 1992. "Non-linear, steady-state vibration of structures by harmonic balance/finite element method". *Computers & Structures*, **44**(1-2), pp. 287–296.
- [13] Martin, A., Chouvion, B., and Thouverez, F., 2017. "Harmonic study and reduction of 3d finite element structures subject to geometric nonlinearities". In *Congrès français de mécanique, AFM, Association Française de Mécanique*.
- [14] Weeger, O., Wever, U., and Simeon, B., 2014. "Nonlinear frequency response analysis of structural vibrations". *Computational Mechanics*, **54**(6), pp. 1477–1495.
- [15] Touzé, C., and Thomas, O., 2004. "Reduced-order modeling for a cantilever beam subjected to harmonic forcing". In *EUROMECH 457, Nonlinear modes of vibrating systems*.
- [16] Grolet, A., and Thouverez, F., 2010. *Vibration analysis of a nonlinear system with cyclic symmetry*.
- [17] Krack, M., and Gross, J., 2019. *Harmonic Balance for Non-linear Vibration Problems*. Springer.
- [18] Bhatti, M. A., 2006. *Advanced Topics in Finite Element Analysis of Structures: With Mathematica and MATLAB*.

- Computations*. John Wiley and Sons, Inc., USA.
- [19] Kirk, B. S., Peterson, J. W., Stonger, R. H., and Carey, G. F., 2006. “libmesh: A c++ library for parallel adaptive mesh refinement/coarsening simulations”. *Engineering with Computers*, **22**, pp. 237–254.
- [20] Heroux, M. A., Bartlet, R. A., Howle, V. E., Hoekstra, R. J., Hu, J. J., Kolda, T. G., Lehoucq, R. B., Long, K. R., Pawlowski, R. P., Phipps, E. T., Salinger, A. G., Thornquist, H. K., Tuminaro, R. S., Willenbring, J. M., Williams, A., and Stanley, K. S., 2005. “An overview of the trilinos project”. *ACM Transactions on Mathematical Software*, **31**(3), pp. 397–423.
- [21] Amestoy, P. R., Duff, I. S., L’Excellent, J.-Y., and Koster, J., 2001. “A fully asynchronous multifrontal solver using distributed dynamic scheduling”. *SIAM Journal on Matrix Analysis and Applications*, **23**(1), pp. 15–41.
- [22] Thomas, O., S  n  chal, A., and De  , J.-F., 2016. “Hardening/softening behavior and reduced order modeling of nonlinear vibrations of rotating cantilever beams”. *Nonlinear Dynamics*, **86**(2), Oct, pp. 1293–1318.
- [23] Grolet, A., and Thouverez, F., 2012. “On the use of the proper generalised decomposition for solving nonlinear vibration problems”. In ASME 2012 International Mechanical Engineering Congress and Exposition, American Society of Mechanical Engineers Digital Collection, pp. 913–920.
- [24] Riha, L., Merta, M., Vavrik, R., Brzobohaty, T., Markopoulos, A., Meca, O., Vysocky, O., Kozubek, T., and Vondrak, V., 2019. “A massively parallel and memory-efficient fem toolbox with a hybrid total feti solver with accelerator support”. *International Journal of High Performance Computing Applications*, **33**(4), pp. 660–677.