AN ALGORITHM TO MAKE YOUR IMPLICIT CODE COMPETITIVE WITH AN EXPLICIT CODE FOR LARGE SCALE PROBLEMS

T. Meinders, A.H. van den Boogaard, J. Huétink

University of Twente, P.O.box 217, 7500AE Enschede, The Netherlands
URL: www.tm.ctw.utwente.nl/ timo e-mail: v.t.meinders@ctw.utwente.nl

ABSTRACT: To intensify the use of implicit finite element codes for solving large scale problems, the computation time of these codes has to be decreased drastically. A method is developed which decreases the computational time of implicit codes by factors. The method is based on introducing inertia effects into the implicit finite element code in combination with the use of iterative solvers. Deep drawing simulations are performed to investigate the performance of the dynamics contributions in combination with iterative solvers. It is concluded that the computation time can be decreased by a factor 5-10.

Key words: implicit code, iterative solvers, dynamics contributions

1 INTRODUCTION

At present time the competition between implicit and explicit finite element codes is still in full swing, where explicit codes are favored for solving large problems although implicit codes yield more accurate results. To intensify the use of implicit codes for solving large problems, the computation time of these codes has to be decreased drastically. A method is developed which decreases the computational time of implicit codes by factors, which makes the implicit finite element code competitive with explicit finite element codes for large scale problems. This method is based on introducing dynamics contributions into the implicit finite element code. Using a lumped mass matrix approach, this method yields an improved condition of the system matrix, which makes an effective use of iterative solvers possible. Another advantage of introducing dynamics contributions into an implicit finite element code is that it stabilizes the computation, especially when the problem is under-constrained.

This paper starts with the basics of the implementation of the dynamics contributions into the implicit finite element code DiekA, Section 2. In sheet metal forming, shell elements are used which have three displacement d.o.f. and three rotational d.o.f. per node. These rotational d.o.f. give rise to complications in the implementation of dynamics contributions. Therefore we start with the implementation of dynamics contributions in the two-dimensional plane strain element, which has only displacement d.o.f., Section 3. Then the implementation is proceeded for shell elements, Section 4. In Section 5 the performance of the dynamics contributions in combination with the use of iterative solvers is investigated for a simple deep drawing simulation. In Section 6 the results of the simulation of a front door panel are discussed [1]. The article is closed with some concluding remarks.

2 DYNAMICS CONTRIBUTIONS

Omitting damping influences, the discretized equations of motion in the linear case read:

\[ M \ddot{u}_n + Ku_n = F_n \]  

(1)

For non-linear computations the second term of the left hand side is replaced by the internal force vector.

The displacements and velocities can be determined by an integration method. Using the Newmark integration method, equation 1 reads in incremental form:

\[
\left( \frac{M}{\beta \Delta t^2} + K \right) \delta \Delta \dot{u}_n^k = F_n - R_n^k - \frac{M}{\beta \Delta t^2} \Delta \dot{u}_n^{k+1} \\
\frac{M}{\beta \Delta t} \dot{u}_{n-1} + \left( \frac{1}{2\beta} - 1 \right) M \ddot{u}_{n-1}
\]  

(2)

where the parameter \( \beta \) is free to choose. In the linear case, The Newmark integration method is unconditionally stable for \( \beta = 0.25 \). However, the scheme may lose stability in the nonlinear case [2]. For \( \beta = 1 \) the Newmark integration method degenerates to the Euler backward integration method.

The consistent mass matrix can be calculated
through the spatial discretization of the weak form of equilibrium:

$$\rho \int_V (\delta \ddot{u}) \ddot{u} dV = \rho \sum_{\alpha} \sum_{\beta} \int_V \delta \ddot{u}^{\alpha \beta} N^\alpha N^\beta \ddot{u} dV =$$

$$\delta \ddot{u}^\alpha \rho \int_V N^{\alpha \beta} dV \ddot{u}^{\beta} = \delta \ddot{u}^\alpha M^{\alpha \beta} \ddot{u}^{\beta} \quad (3)$$

The lumped mass matrix is determined by a diagonalization of the consistent mass matrix. The lumped mass matrix method is generally used in explicit codes. The main advantages of this method are that the inverse of this matrix is easily calculated and that it improves the condition number of the system matrix.

3 IMPLEMENTATION OF DYNAMICS CONTRIBUTIONS FOR PLANE STRAIN ELEMENTS

To validate the implementation of the dynamics contributions, a plane strain beam bending test is performed, see Figure 1. The beam length is 50 mm and is modeled with 20 elements. The density $\rho$ is $7.8 \cdot 10^{-9}$ tons/mm$^3$, the Poisson ratio $\nu$ is 0 and the Young’s modulus $E$ is 210000 N/mm$^2$.

Figure 1: Beam modeled with plane strain elements.

Two simulations are performed in which a certain displacement is prescribed where after the beam is released, using the backward integration scheme and the Newmark integration scheme, respectively. The results of these simulation are given in Figure 2.

Figure 2 clearly shows that several natural frequencies of the beam are activated, in case the Newmark integration scheme is applied. In other words, the first natural frequency of the beam is superimposed by higher order frequencies. Note that, in case the Backward integration scheme is applied, the higher order terms will damp more, and as a result the beam will further vibrate in its first natural frequency.

The analytical solutions for the natural frequencies of the beam are [3]:

$$f_n = \frac{C_n}{2\pi} \sqrt{\frac{EI}{\rho Al^4}} \quad (4)$$

with $C_1 = 3.52$, $C_2 = 22.4$, $C_3 = 61.7$ and $C_4 = 121.0$. The first natural frequency is 335.2 Hz for this beam (analytical). However, the simulation gives a first natural frequency of 676.7 Hz. The deviation between these values is due to the fact that the plane strain element acts too stiff for bending modes when the element size is not small enough (caused by an overestimation of the shear stress). For the given geometry, the overestimation of the stiffness of the beam is a factor 4.124 (analytical deflection versus simulated deflection). Consequently, the simulated natural frequencies will be a factor 2.03 to high. Modification of the simulated natural frequencies with this factor gives:

<table>
<thead>
<tr>
<th>frequency</th>
<th>analytical [Hz]</th>
<th>simulation [Hz]</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_1$</td>
<td>335.2</td>
<td>333.2</td>
</tr>
<tr>
<td>$f_2$</td>
<td>2133.5</td>
<td>2085.0</td>
</tr>
<tr>
<td>$f_3$</td>
<td>5875.6</td>
<td>5755.3</td>
</tr>
<tr>
<td>$f_4$</td>
<td>11522.6</td>
<td>11093.2</td>
</tr>
</tbody>
</table>

Note that the results are less accurate for increasing frequency, since the wavelength will be shorter and as a result more difficult to describe with 20 elements.

From these results, it can be concluded that the natural frequencies are simulated sufficiently accurate. Consequently, the implementation of the dynamics contributions is proceeded for shell elements.

4 IMPLEMENTATION OF DYNAMICS CONTRIBUTIONS FOR SHELL ELEMENTS

Three approaches can be considered to determine the mass contributions for both the displacement and rotational d.o.f. for a 3-node shell element. The first approach only takes into account the lumped mass contribution to the displacement d.o.f. The second approach makes use of a consistent mass
matrix, derived by Hermitian polynomials. The third approach uses mass moments of inertia to take into account the mass contributions for the rotational d.o.f. In Section 4.1, the different approaches will be explained in detail and compared with each other. Subsequently one of the approaches is chosen to be used, where after this section continues, using the chosen approach, Section 4.2. In Section 4.3 a comment will be made on the drilling d.o.f.’s of a triangular discrete shear element.

4.1 Three approaches to implement the dynamics contributions

4.1.1 Approach 1: Only displacement d.o.f.

Beforehand, it is not clear what the influence will be of taking into account mass contributions for the rotational d.o.f. Therefore only the displacement d.o.f. are focused on. Using the lumped mass matrix approach and the evaluation of area integrals according to [4], the mass matrix for a linear triangular element can be written as:

\[ M = \begin{bmatrix}
\frac{1}{h} & 0 & 0 & 0 & \ldots \\
0 & \frac{1}{h} & 0 & 0 & \ldots \\
0 & 0 & \frac{1}{h} & 0 & \ldots \\
0 & 0 & 0 & \frac{1}{h} & \ldots \\
\vdots & \vdots & \vdots & \vdots & \ddots \\
\end{bmatrix} \rho A \]

with \( h \) the thickness of the element. The first 3 d.o.f. represent the translations of the first node, the following 3 d.o.f. represent the rotations of the first node. Subsequently, node 2 and 3 follow.

4.1.2 Approach 2: Hermitian polynomials

The rotational d.o.f. influence the displacement normal to the element plane. Therefore a coupling between these d.o.f. can be generated through the interpolation functions. A neat coupling method is based on Hermitian polynomials [5], which yields a consistent mass matrix. However, lumping this consistent mass matrix gives negative values on the diagonal of the lumped mass matrix which is very bad for the condition number of the mass matrix.

4.1.3 Approach 3: Mass moment of inertia

Another method to implement mass contributions to the rotational d.o.f. is based on mass moments of inertia:

\[
\delta W = \delta \omega \delta I_n \delta \omega \approx \delta \omega \delta \rho h I \delta \omega \approx \delta \omega \rho h \left( \frac{1}{3} A^2 \right) \delta \omega 
\]

with \( A \) the element area. In case of linear triangular elements, this approach yields the following lumped mass matrix:

\[
M = \begin{bmatrix}
\frac{1}{h} & 0 & 0 & 0 & \ldots \\
0 & \frac{1}{h} & 0 & 0 & \ldots \\
0 & 0 & \frac{1}{h} & 0 & \ldots \\
0 & 0 & 0 & \frac{1}{h} & \ldots \\
\vdots & \vdots & \vdots & \vdots & \ddots \\
\end{bmatrix} \rho h A
\]

4.1.4 Comparison of the three approaches

The selection criterion to choose the right approach will be which one yields the highest increase of the condition number of the system matrix while preserving an accurate solution. From this point of view, approach 2 can already be dropped. The remaining two approaches are compared, using the beam bending test. The results showed that only a slight difference in amplitude was observed when mass is added to the rotational d.o.f. Based on these results and looking at the usage of iterative solvers, the mass moment of inertia approach is preferred, since this approach yields the highest increase of the condition number of the system matrix.

4.2 Beam bending test

Two simulations are performed, using the Newmark and backward integration scheme, respectively, Figure 3.

![Figure 3: Beam modeled with Mindlin elements.](image)

It is clear that also the higher order frequencies are spotted when Mindlin elements are used. The first natural frequency, calculated with the Mindlin beam is 328.9 Hz, which shows a good agreement with the analytical solution (335.2 Hz, see Section 3).
4.3 Drilling d.o.f’s of a discrete shear element

The in-plane behavior of the triangular discrete shear element is represented by a constant strain triangle, [6]. This element has 6 d.o.f. in each of the three nodes. The drilling d.o.f. has no inherent stiffness and therefore the resulting stiffness matrix becomes singular. One way to avoid this is to add a small stiffness to the diagonal element that represents this d.o.f. [7]. If a direct solver is used, this approach leads to realistic results. However, the very small diagonal terms lead to ill-conditioned matrices that are detrimental for iterative solvers. The approach of Zienkiewicz [5] may therefore be beneficial in combination with iterative solvers. Here a coupling between all drilling d.o.f. $\phi_z$ is achieved by adding the following relation to the stiffness matrix:

$$
\begin{bmatrix}
M_{z1} \\
M_{z2} \\
M_{z3}
\end{bmatrix}
= \alpha
\begin{bmatrix}
2 & -1 & -1 \\
-1 & 2 & -1 \\
-1 & -1 & 2
\end{bmatrix}
\begin{bmatrix}
\phi_{z1} \\
\phi_{z2} \\
\phi_{z3}
\end{bmatrix}
$$

where $\alpha$ is a factor that scales with the stiffness of the element. In this paper, $\alpha$ is set to 0.1 times the average of the other rotational stiffness diagonal terms. The results are not influenced much by this contribution, but the condition number of the matrix may improve considerably, [8].

5 DEEP DRAWING OF A RECTANGULAR PRODUCT

Section 4 showed that the dynamics contributions for shell elements are correctly implemented in the implicit finite element code. In this section the influence of the dynamics contributions on the deep drawing of a rectangular product is investigated. Since it is expected that the dynamics contributions will improve the condition of the matrix (and thus improve the convergence behavior of iterative solvers), the convergence behavior of different iterative solvers is investigated. The iterative solvers used are the Conjugate Gradient (CG) solver, the Generalized Minimum Residual (GMRES) solver and the Biconjugate Gradient Stabilized (Bi-CGSTAB) solver, all in combination with a Symmetric Successive Over Relaxation (SSOR) preconditioner [9]. Several simulations are performed in which the punch velocity is varied between 0 m/s (no dynamics contributions) and 50 m/s. The system matrix at step 40 is dumped where after the system is solved, using the different iterative solvers. The unbalance criterion is set to $10^{-5}$, the maximum number of iterations is set to 500. The performance of these iterative solvers is given in Figure 4.

![Figure 4: Convergence behavior of different iterative solvers, dynamics contributions on displacement and rotational d.o.f.](image1)

![Figure 5: Convergence behavior of different iterative solvers, dynamics contributions on displacement d.o.f. only.](image2)

It can be concluded that the convergence rate drastically increases with an increase of the deep drawing velocity. Since, the results between a simulation performed with dynamics contributions until a deep drawing speed of 25 m/s and one without dynamics contributions do not differ significantly [10], it is possible to decrease the computation time by factors without affecting the results. In case of the GMRES solver the computation time is reduced by more than a factor 12, for the CG solver with more than a factor 6 and for the Bi-CGSTAB solver with more than a factor 2.

Finally, a set of simulations is performed without dynamics contributions for the rotational d.o.f. The results of these simulations are shown in Figure 5. This figure shows that it is important to take into account the dynamics contributions for the rotational d.o.f., since they have a large influence on the convergence behavior.
In the last section, the forming of an AUDI-front door panel is discussed. This product served as a benchmark for the Numisheet conference, held in 1999. For more details concerning the geometry of the drawing tools, the reader is referred to [1]. The front view of the drawing tools and the initial blank is given in Figure 6. Note that the blankholder is doubly curved which gives rise to instabilities when dynamics contributions are not taken into account. This automotive product will be used to investigate the performance of the improved implicit code with respect to the conventional implicit code.

Figure 6: Drawing tools and initial blank for front door panel.

Two simulations are performed. For both simulations, the dynamics contributions are switched on, gravity loads are applied and automatic refinement is used. Following the strategy used in explicit codes (artificial scaling of punch velocity to benefit from dynamic contributions), the deep drawing velocity is set to 10 m/s. One simulation is performed in which the Bi-CGSTAB iterative solver with SSOR preconditioning is used (which shows a good performance, see Section 5), and one simulation is performed with a direct solver (Cholesky decomposition).

The discussion of the results is started with the simulation, using the iterative solver. In the first 90 steps, the blankholder is closed. Then the punch moves downwards, until the desired deep drawing depth is reached in step 191. The final deformed mesh is given in Figure 7.

During deep drawing, the mesh is refined on areas with locally high curvatures [11]. The initial mesh contains 17244 d.o.f., whereas the mesh ends up with 79344 d.o.f. The generation of new elements during the simulation are graphically represented in Figure 8. The total computation time for the entire simulation took 5.5 hours on a HP8000 workstation. This computation time is similar to the computation time of an explicit code for this simulation [1].

Figure 7: Final shape of front door panel.

Figure 8: Increase of d.o.f. during the simulation.

Then, a simulation is started, using a direct solver. Again, in the first 90 steps the blankholder is closed. Then the punch moves downwards, until the mesh contains 36876 d.o.f. At this moment the hardware of the current workstation is not sufficient anymore to proceed the simulation due to insufficient internal memory. Therefore, to make a comparison between the necessary computation time for both solvers, the computation time for one iteration is looked at. The computation time for one iteration strongly depends on the number of d.o.f., see Figure 9. This figure shows the computation time for one iteration for both the simulations with the iterative and direct solver, during the simulation. Besides, the analytical graph for the increase of computation time for the direct solver is added, $n \times n_b^2$ [9], where $n$ is the number of d.o.f. and $n_b$ is the bandwidth. In case of a planar uniform mesh, this graph can be represented by $n^2$. 
The figure clearly shows that the iterative solver is significantly faster than the direct solver. For 17244 d.o.f. the iterative solver is 10 times faster than the direct solver while this factor increases up to 25 for 36876 d.o.f. For the direct solver, it is also observed that the computation time increases more than quadratically, which is due to the non-uniformity of the refined mesh. The computation time of the iterative solver increases almost linear. If the analytical approach is taken as a lower bound for the prediction of the computation time for the direct solver in case of 79344 d.o.f., the computation time will increase by at least a factor 40 with respect to the iterative solver. Subsequently it is concluded that the computation time for the iterative solver in combination with dynamics contributions is factors smaller that the computation time when the direct solver is used. However, the total simulation time is not only determined by solving the system matrix, also the creation of the system matrix and the contact search algorithm are time consuming (linear with the number of d.o.f.). Therefore, for this simulation it is stated that when the hardware would be sufficient to perform the entire simulation when using the direct solver, the simulation time would increase by a factor 5-10.

7 CONCLUSIONS

The dynamics contributions are successfully implemented in the implicit finite element code DiekA for both the plane strain element (only displacement d.o.f.) and the shell element (displacement and rotational d.o.f.).

The computation time of a deep drawing simulation with an implicit finite element code is drastically decreased when dynamics contributions in combination with an iterative solver are used. For large problems, the computation time to solve the system is decreased by a factor 25. As a result, the computation time for the entire simulation (including the contact search algorithm and creation of the system matrix) can be decreased by a factor 5-10.

REFERENCE