Computer Assisted Mechanics and Engineering Sciences, 8: 121–139, 2001.

Inconsistency errors of constant velocity multi-time step integration algorithms

Marek Klisiński
Division of Structural Mechanics, Department of Civil and Mining Engineering,
Luleå University of Technology, SE-971 87, Luleå, Sweden

(Received October 14, 1999)

Previously known multi-time step integration methods for finite element computations in structural dynamics have been shown to be unstable due to interpolation error propagation. New algorithms of multi-time step integration based on constant velocity during subcycling are investigated. The assumption of constant velocity gives linear variation of displacements so the errors connected to interpolation at the interface between different time step partitions are eliminated. As a consequence, the new constant velocity algorithms give bounded solutions and have been shown to be conditionally stable by their authors. However, numerical investigation demonstrates that if time steps close to the stability limit are used, the errors for higher natural modes are so huge that the obtained solutions can only be considered as incorrect. The main reason for this behaviour is that the constant velocity time integration algorithms are inconsistent. Displacements can be calculated either by direct integration or from the equation of motion leading to different solutions. Based on the numerical results it is concluded that use of time steps below stability limit is insufficient to assure proper solutions. Therefore, significant time step reductions are often required to assure acceptable error levels. As a consequence, the new subcycling algorithms can be more expensive than ordinary time integration. Because they also lead to larger errors the constant velocity subcycling algorithms are useless from practical point of view. Since subcycling is available as an option in LS-DYNA a serious warning is issued to potential users.

1. Introduction

Multi-time step (subcycling) integration methods, that were introduced by Belytschko et al. in 1979, have been developed for the last 20 years. Different schemes can be found e.g. in Belytschko and Lu (1993) where also stability of multi-time step methods for second order systems were discussed. However, the presented conclusions were incorrect. Klisiński et al. (1998) (see also a conference contribution from 1994) have shown that the interpolation errors introduced at the interface between different time step domains propagate leading to unbounded solutions. Therefore, it has been concluded that multi-time step integration methods are inherently unstable for second order systems. As a response to these findings a new explicit algorithm was introduced by Smoliński et al. (1996). A major difference, comparing to the previous algorithms, is that the algorithm preserves constant velocity during the subcycling. As a result the displacements change linearly and no interpolation error is introduced. Therefore, the assumptions on which the stability analysis is based in Klisiński et al. (1998) are no longer valid. Smoliński et al. (1996) proved by analysing the amplification matrix that the algorithm is conditionally stable. Another version of the constant velocity algorithm which possesses the same stability properties was presented by Daniel (1998). These two algorithms are analysed in the numerical way in the paper.

2. Basic formulation

The semidiscrete finite element formulation of structural transient dynamic problems without damping results in the following system of second order ordinary differential equations,

$$\mathbf{Ma}+\mathbf{Kd}=\mathbf{f} \tag{1}$$

where $\mathbf{M}$ denotes lumped mass matrix; $\mathbf{K}$ – stiffness matrix; $\mathbf{d}$ – vector of nodal displacements; $\mathbf{a}$ – vector of nodal accelerations and $\mathbf{f}$ – vector of external forces. In addition also a vector of nodal velocities $\mathbf{v}$ is introduced. The above matrices and vectors can be partitioned as follows,

$$\mathbf{M}=\begin{bmatrix}\mathbf{M}^A & \mathbf{0}\\ \mathbf{0} & \mathbf{M}^B\end{bmatrix},\qquad \mathbf{K}=\begin{bmatrix}\mathbf{K}^{AA} & \mathbf{K}^{AB}\\ \mathbf{K}^{BA} & \mathbf{K}^{BB}\end{bmatrix},$$
$$\mathbf{a}=\begin{bmatrix}\mathbf{a}^A\\ \mathbf{a}^B\end{bmatrix},\qquad \mathbf{v}=\begin{bmatrix}\mathbf{v}^A\\ \mathbf{v}^B\end{bmatrix},\qquad \mathbf{d}=\begin{bmatrix}\mathbf{d}^A\\ \mathbf{d}^B\end{bmatrix},\qquad \mathbf{f}=\begin{bmatrix}\mathbf{f}^A\\ \mathbf{f}^B\end{bmatrix}, \tag{2}$$

where A and B – two separate mesh domains. To simplify further considerations it is assumed that smaller time step Δt is used in domain A, whereas larger time step used in domain B is twice as long, i.e. $= 2\Delta t$.

3. Smoliński et al. algorithm

The explicit algorithm proposed by Smoliński et al. (1996) is based on the following equations

$$\mathbf{d}_{k+1}=\mathbf{d}_k+\Delta t\,\mathbf{v}_k+\frac{\Delta t^2}{2}\mathbf{a}_k, \tag{3}$$

$$\mathbf{v}_{k+1}=\mathbf{v}_k+\frac{\Delta t}{2}(\mathbf{a}_k+\mathbf{a}_{k+1}), \tag{4}$$

$$\mathbf{Ma}_{k+1}+\mathbf{Kd}_{k+1}=\mathbf{f}_{k+1}, \tag{5}$$

where the subscripts denote the time step, e.g. $\mathbf{d}_k$ means $\mathbf{d}(t_k)$ and $t_k=k\Delta t$ is the time station k.

In the actual algorithm Eqs. (3), (5) and a similar equation of motion written for time station k are combined to eliminate the displacements giving

$$\mathbf{Ma}_{k+1}=\left(\mathbf{M}-\frac{\Delta t^2}{2}\mathbf{K}\right)\mathbf{a}_k-\Delta t\,\mathbf{K}\mathbf{v}_k+\mathbf{f}_{k+1}-\mathbf{f}_k\,. \tag{6}$$

The algorithm was described by Smoliński et al. (1996) and after necessary adjustments to account for two substeps only ($m=2$ in the original description) it looks as follows:

1. initial conditions: $\mathbf{d}_0=\mathbf{d}(0)$, $\mathbf{v}_0=\mathbf{v}(0)$,
2. initial accelerations: $\mathbf{a}0=\mathbf{M}^{-1}(\mathbf{f}_0-\mathbf{Kd}_0)$,

4 updates of subdomain A with smaller time step Δt

3. $\mathbf{a}^A_{k+1}$ from $\mathbf{M}^A\mathbf{a}^A_{k+1}=\left([\mathbf{M}^A\,\mathbf{0}]-\frac{\Delta t^2}{2}[\mathbf{K}^{AA}\,\mathbf{K}^{AB}]\right)\mathbf{a}_k-\Delta t[\mathbf{K}^{AA}\,\mathbf{K}^{AB}]\,\mathbf{v}_k+\mathbf{f}^A_{k+1}-\mathbf{f}^A_k$,
4. $\mathbf{a}^B_{k+1}=-\mathbf{a}^B_k$,
5. $\mathbf{v}_{k+1}=\mathbf{v}_k+\frac{\Delta t}{2}(\mathbf{a}_k+\mathbf{a}_{k+1})$,

increment k by 1,

2 updates of subdomain B with larger time step $2\Delta t$

6. $\mathbf{a}^B_{k+2}$ from $\mathbf{M}^B\mathbf{a}^B_{k+2} = \left([\mathbf{0}\,\mathbf{M}^B] - \frac{\Delta t^2}{2}[\mathbf{K}^{BA}\,\mathbf{0}] - 2\Delta t^2[\mathbf{0}\,\mathbf{K}^{BB}]\right)\mathbf{a}_k$
$$- 2\Delta t\,[\mathbf{K}^{BA}\,\mathbf{K}^{BB}]\,\mathbf{v}_k + \mathbf{f}^B_{k+2} - \mathbf{f}^B_k$$
7. $\mathbf{a}^A_{k+2} = -\mathbf{a}^A_k$,
8. $\mathbf{v}_{k+2} = \mathbf{v}_k + \Delta t(\mathbf{a}_k + \mathbf{a}_{k+2})$,

repeat the updates in subdomains A and B.

The only purpose of changing the sign of accelerations is to keep the velocities constant in one of the partitions when updating the second one.

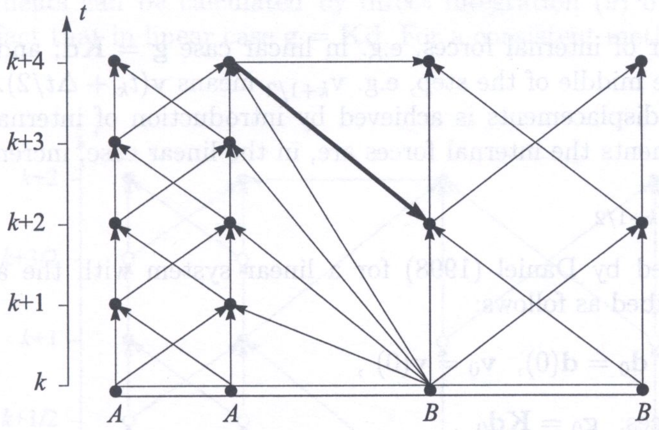

Fig. 1. Flow of information in the Smoliński et al. algorithm (Fig. 1 in Smoliński et al. 1996)

Unfortunately, the diagram provided by Smoliński et al. (1996) (see Fig. 1) suggests a slightly different procedure and also reveals a rather surprising fact that the values calculated in subdomain A for time station $k+4$ are used to calculate values in subdomain B at **earlier** time station $k+2$. According to this diagram the acceleration and velocity vectors at the beginning of the updates of subdomain B should be understood as

$$\mathbf{a}_k = \begin{bmatrix} \mathbf{a}^A_{k+4} \\ \mathbf{a}^B_k \end{bmatrix}, \qquad \mathbf{v}_k = \begin{bmatrix} \mathbf{v}^A_{k+4} \\ \mathbf{v}^B_k \end{bmatrix}, \tag{7}$$

and not as

$$\mathbf{a}_k = \begin{bmatrix} \mathbf{a}^A_k \\ \mathbf{a}^B_k \end{bmatrix}, \qquad \mathbf{v}_k = \begin{bmatrix} \mathbf{v}^A_k \\ \mathbf{v}^B_k \end{bmatrix}. \tag{8}$$

A simple comparison between the results shows that the interpretation according to (7) is the correct one, because use of the values (8) leads to directly unacceptable outcomes even if some other modifications are made. However, notice that such a procedure violates the concept of causality. For example, a sudden load applied later in partition A has an influence on the behaviour of partition B at earlier times. This gives rise to a serious suspicion about the correctness of the proposed method. In spite of this, let us check the algorithm assuming that the errors introduced in this way are small. In particular if free vibration problems are studied, $\mathbf{f} = \mathbf{0}$, and no sudden load changes occur.

Even if the actual algorithm does not use displacements they are vital in many applications. The basic equations used in the method allow to calculate them in two different ways. The first one applies direct integration (3) where different time steps can be used in both mesh partitions. The second method is based on the equation of motion (5) in which displacements are obtained from known accelerations. If the method was consistent both calculations would give the same result.

4. Daniel algorithm

Daniel (1998) proposed another version of the constant velocity subcycling algorithm, where oscillating accelerations are avoided. The algorithm uses velocities calculated in the middle of the time step.

Unfortunately, the basic equations on which the method is based are not directly provided, so they have to be reconstructed from the information given in his paper. They can be written as

$$\mathbf{d}_{k+1} = \mathbf{d}_k + \Delta t\, \mathbf{v}_{k+1/2}\,, \tag{9}$$

$$\mathbf{v}_{k+1/2} = \mathbf{v}_k + \frac{\Delta t}{2}\mathbf{a}_k\,, \qquad \mathbf{v}_{k+1} = \mathbf{v}_{k+1/2} + \frac{\Delta t}{2}\mathbf{a}_{k+1}\,, \tag{10}$$

$$\mathbf{M}\mathbf{a}_{k+1} + \mathbf{g}_{k+1} = \mathbf{f}_{k+1}\,, \tag{11}$$

where $\mathbf{g}$ denotes vector of internal forces, e.g. in linear case $\mathbf{g} = \mathbf{K}\mathbf{d}$; and the subscript $k+1/2$ denotes the time in the middle of the step, e.g. $\mathbf{v}_{k+1/2}$ means $\mathbf{v}(t_k + \Delta t/2)$.

The elimination of displacements is achieved by introduction of internal forces and instead of updating the displacements the internal forces are, in the linear case, incremented according to

$$\mathbf{g}_{k+1} = \mathbf{g}_k + \Delta t\, \mathbf{K}\, \mathbf{v}_{k+1/2} \tag{12}$$

The algorithm proposed by Daniel (1998) for a linear system with the assumption of just two subcycles can be described as follows:

1. initial conditions: $\mathbf{d}_0 = \mathbf{d}(0),\ \ \mathbf{v}_0 = \mathbf{v}(0)$,
2. initial internal forces: $\mathbf{g}_0 = \mathbf{K}\mathbf{d}_0$,
3. initial accelerations: $\mathbf{a}_0 = \mathbf{M}^{-1}(\mathbf{f}_0 - \mathbf{g}_0)$,

1 update of subdomain A with smaller time step Δt

4. $\mathbf{v}^A_{k+1/2} = \mathbf{v}^A_k + \dfrac{\Delta t}{2}\mathbf{a}^A_k$,
5. $\mathbf{g}^A_{k+1} = \mathbf{g}^A_k + \Delta t\left(\mathbf{K}^{AA}\mathbf{v}^A_{k+1/2} + \mathbf{K}^{AB}\mathbf{v}^B_k\right)$,
6. $\mathbf{a}^A_{k+1}$ from $\mathbf{M}^A\mathbf{a}^A_{k+1} + \mathbf{g}^A_{k+1} = \mathbf{f}^A_{k+1}$,
7. $\mathbf{v}^A_{k+1} = \mathbf{v}^A_{k+1/2} + \dfrac{\Delta t}{2}\mathbf{a}^A_{k+1}$,

1 update of subdomain B with larger time step $2\Delta t$

8. $\mathbf{v}^B_{k+1} = \mathbf{v}^B_k + \Delta t\, \mathbf{a}^B_k$,
9. $\mathbf{g}_{k+2} = \mathbf{g}_k + \Delta t \mathbf{K}\mathbf{v}_{k+1}$,
10. $\mathbf{a}^B_{k+2}$ from $\mathbf{M}^B\mathbf{a}^B_{k+2} + \mathbf{g}^B_{k+2} = \mathbf{f}^B_{k+2}$,
11. $\mathbf{v}^B_{k+2} = \mathbf{v}^B_{k+1} + \Delta t\, \mathbf{a}^B_{k+2}$,

1 more update of subdomain A with smaller time step Δt

12. $\mathbf{v}^A_{k+3/2} = \mathbf{v}^A_{k+1} + \dfrac{\Delta t}{2}\mathbf{a}^A_{k+1}$,
13. $\mathbf{g}^A_{k+2} = \mathbf{g}^A_{k+1} + \Delta t\left(\mathbf{K}^{AA}\mathbf{v}^A_{k+3/2} + \mathbf{K}^{AB}\mathbf{v}^B_{k+2}\right)$,

14. $\mathbf{a}^A_{k+2}$ from $\mathbf{M}^A\mathbf{a}^A_{k+2} + \mathbf{g}^A_{k+2} = \mathbf{f}^A_{k+2}$,

15. $\mathbf{v}^A_{k+2} = \mathbf{v}^A_{k+3/2} + \dfrac{\Delta t}{2}\mathbf{a}^A_{k+2}$,

increment k by 2,

and repeat the entire update process.

The flow of information diagram for this algorithm is shown in Fig. 2. At some times only velocities are computed (indicated by hollow circles) whereas at other times all quantities are calculated (indicated by filled circles). Notice that this algorithm is conceptually much better than the previous one, because causality is treated correctly. Its structure is also more clear and easier to implement. However, the same problem of retrieving the displacements appears and once again it can be solved in two ways. Displacements can be calculated by direct integration (9) or from the equation of motion (11) using the fact that in linear case $\mathbf{g} = \mathbf{Kd}$. For a consistent method both results should be the same.

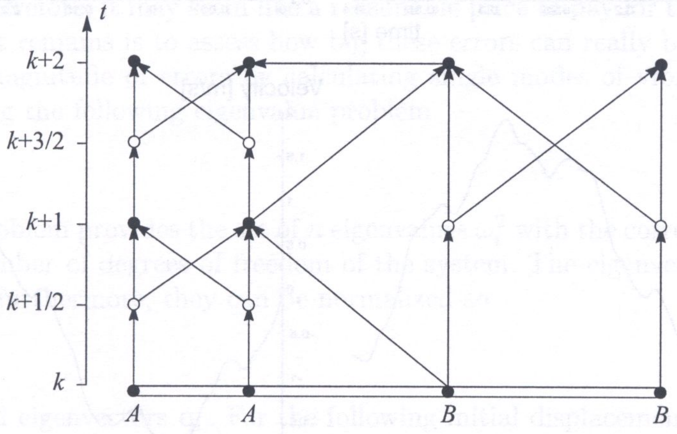

Fig. 2. Flow of information in the Daniel algorithm

5. Numerical evaluation

The same test example as in Klisiński et al. (1998) is used to check both algorithms. It is the cantilever beam with the following properties: length $L = 6$ m, density $\rho = 8000$ kg/m^3, Young modulus $E = 210$ GPa, cross-sectional area $A = 0.012$ m^2, moment of inertia $I = 1.44 \cdot 10^{-5}$ m^4. Initially the free end of the beam is statically displaced in the transverse direction by $d_e = 0.1$ m and instantaneously released. Initial velocities are set to zero. Free transverse vibrations are studied, so no external forces are applied to the system and no physical damping is present. The beam is divided into 6 elements of equal length, 1 m, where 3 elements ments counting from the free end belong to mesh domain A and the remaining 3 to mesh domain B. The interface between these two domains is placed in the middle node. A standard beam element with a diagonal mass matrix is used in the computations. The diagonal mass matrix is obtained by the HRZ lumping scheme as described, for example, in Cook et al. (1989).

The required stability limits for both schemes are found from the eigenvalue analysis as $\Delta t^A_s = 5.22 \cdot 10^{-4}$ s and $\Delta t^B_s = 5.46 \cdot 10^{-4}$ s so the larger time step was set to $5 \cdot 10^{-4}$ s and the smaller one to $\Delta t = 2.5 \cdot 10^{-4}$ s. For reference purposes calculations have also been performed without subcycling and then the larger time step $2\Delta t$ has been applied, because the stability limit for the entire beam is $\Delta t_s = 5.18 \cdot 10^{-4}$ s. The results from the Smoliński et al. algorithm and the Daniel algorithm are shown in Fig. 3 as the left and right columns, respectively. The displacement diagrams contain three curves: solid one represents the displacements obtained by integration, dashed curve

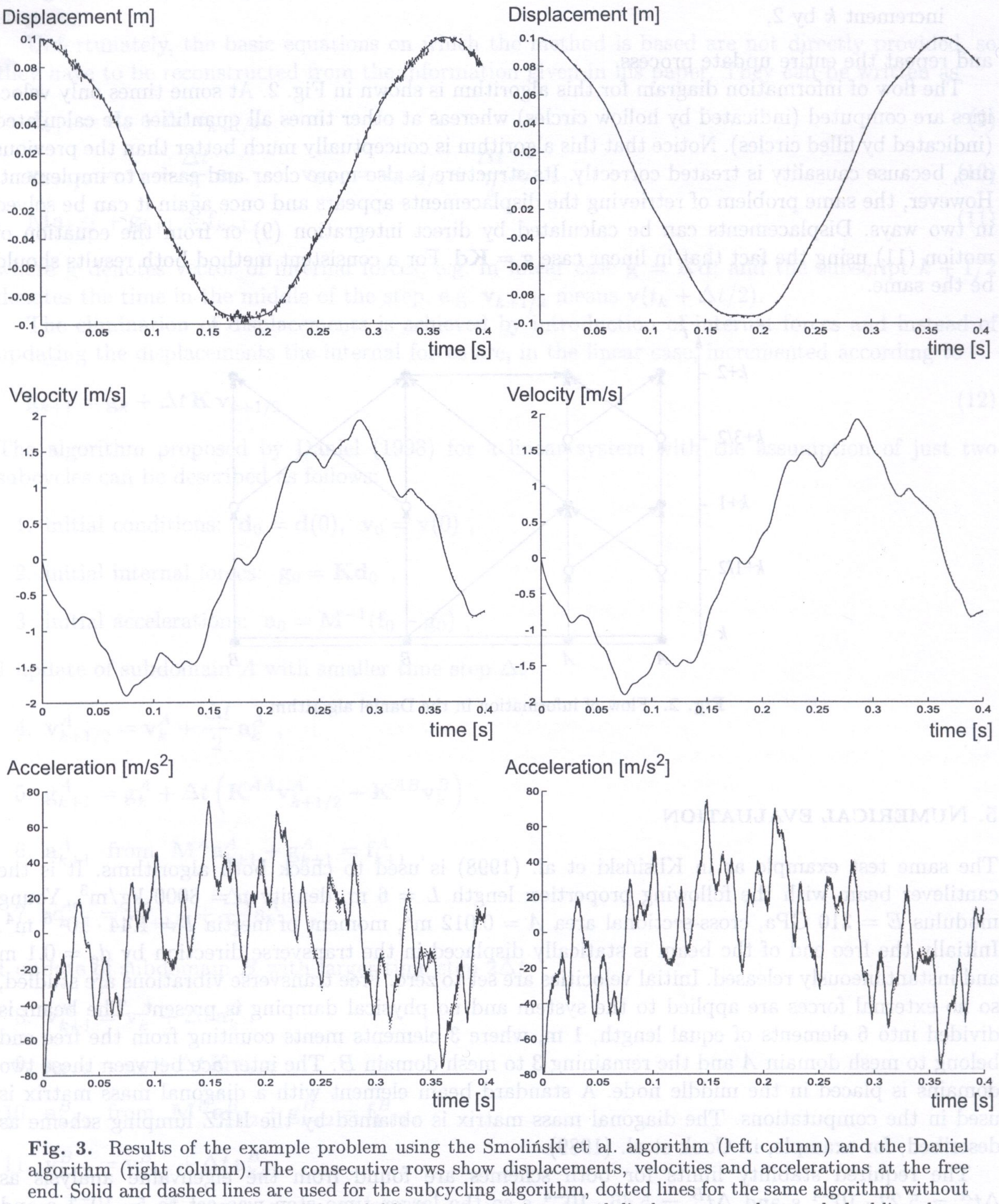

Fig. 3. Results of the example problem using the Smoliński et al. algorithm (left column) and the Daniel algorithm (right column). The consecutive rows show displacements, velocities and accelerations at the free end. Solid and dashed lines are used for the subcycling algorithm, dotted lines for the same algorithm without subcycling. In the displacement plot solid line represents integrated displacements whereas dashed line shows displacements obtained from the equation of motion

shows the displacements calculated from the equation of motion, both for the subcycling algorithm, and the dotted line represents the results from the same algorithm, but without subcycling. The velocity and acceleration diagrams consist of just two curves: a solid one for subcycling algorithm and a dotted one for the same algorithm, but without subcycling. The results are quite accurate and show no tendency of amplitude increase when calculations are carried out longer. The only problem that can be noticed is that the displacements obtained in the two described ways are not the same. It simply shows that both methods are inconsistent. While the assumption of constant velocities is beneficial to eliminate interpolation errors it requires zero accelerations to obtain a consistent method. Obviously accelerations are not zero and because of this inconsistency errors are introduced. One can notice that the direct integration results in smooth curves, whereas the equilibrium calculations lead to additional small oscillations. It is rather transparent, that accelerations appearing in the equation of motion vary much more than velocities. The amplitude of these oscillations, treated as an inconsistency error in displacements, can be compared with the amplitude of beam oscillations and in case of the Smoliński et al. algorithm it gives 6.8% maximum relative error, whereas it is 1.8% for the Daniel algorithm. What is more important the errors show no tendency to increase with time. Therefore, it may seem like a reasonable price to pay for the increased efficiency. The only problem that remains is to assess how big these errors can really become.

Let us check the magnitude of errors by calculating single modes of vibration. Natural modes are obtained by solving the following eigenvalue problem

$$\mathbf{K}\mathbf{q} = \omega^2\mathbf{M}\mathbf{q}. \tag{13}$$

The solution of this problem provides the set of n eigenvalues ω_i^2 with the corresponding eigenvectors $\mathbf{q}_i$, where n is the number of degrees of freedom of the system. The eigenvectors either are or can be made orthogonal. Furthermore, they can be normalized so

$$\boldsymbol{\Phi}_i^T\mathbf{M}\boldsymbol{\Phi}_i = 1 \tag{14}$$

where $\boldsymbol{\Phi}_i$ – normalized eigenvectors $\mathbf{q}_i$. For the following initial displacements and velocities

$$\mathbf{d}_0 = \boldsymbol{\Phi}_i\,, \qquad \mathbf{v}_0 = \mathbf{0}, \tag{15}$$

the analytical solution contains a single mode only

$$\mathbf{d} = \boldsymbol{\Phi}_i \cos\omega_i t. \tag{16}$$

Therefore, if the initial conditions are chosen according to (15) a numerical method should also give a solution corresponding to a single mode i.

Figures 4–7 present the results for the Smoliński et al. algorithm, whereas Figures 8–11 for the Daniel algorithm. Each figure shows in the first two rows displacement and velocity diagrams at the free end with their spectral analysis. Accelerations of the free end and the relative inconsistency errors are shown in the third row. The relative inconsistency error is calculated according to

$$e = \frac{d_e^{int} - d_e^{eom}}{\max|d_e^{cor}|}\cdot 100\% \tag{17}$$

where d_e^{int} denotes current displacement of free end obtained by integration, d_e^{eom} – current displacement of free end calculated from the equation of motion, $\max|d_e^{cor}|$ – correct amplitude of oscillations of free end calculated without subcycling. The same lines as before are used: solid and dashed for constant velocity algorithms with subcycling, dotted for the same algorithms without subcycling; solid for integrated displacements and dashed for displacements calculated from the equation of motion. The figures show clearly that only the first natural mode is calculated correctly by the constant velocity subcycling algorithms using a time step close to the stability limit. The inconsistency errors for the first mode are around 3% for the Smoliński et al. algorithm and 5 times smaller for the Daniel algorithm. However, the results for all other modes must be considered as

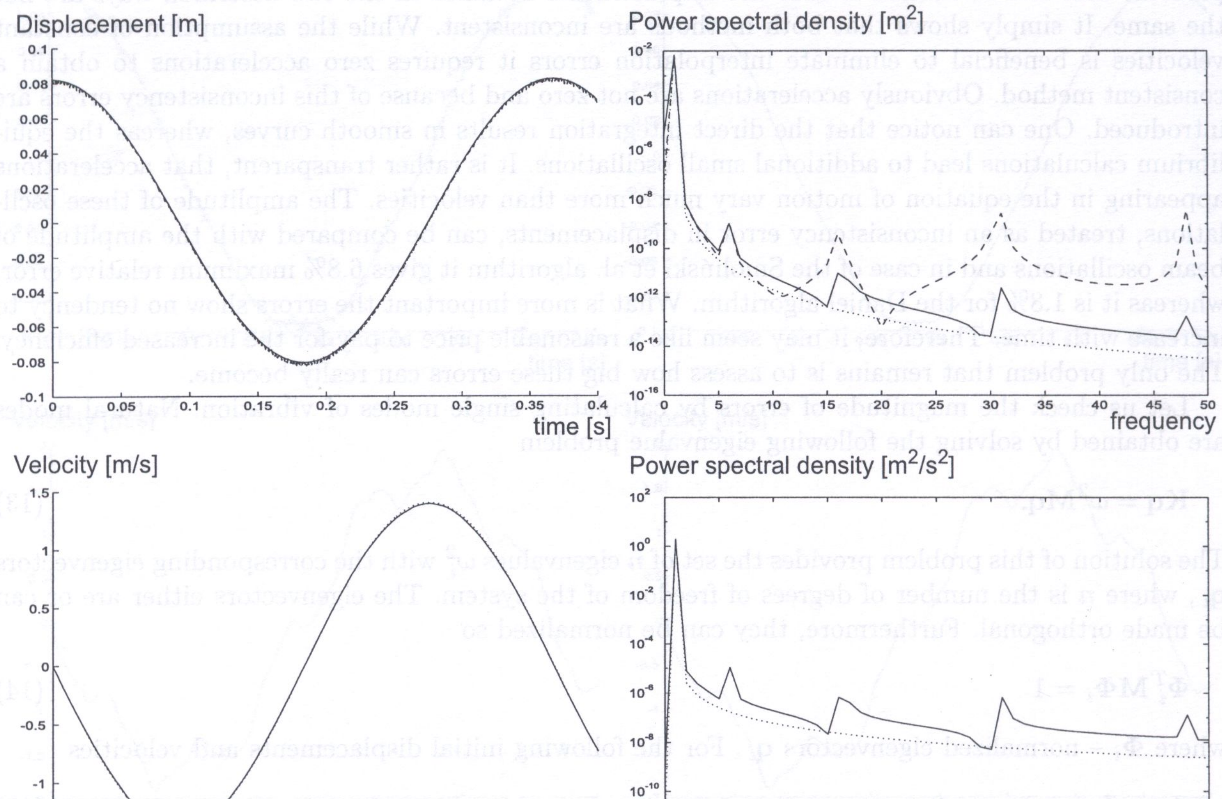

Fig. 4. The first natural mode of vibrations using the Smoliński et al. algorithm

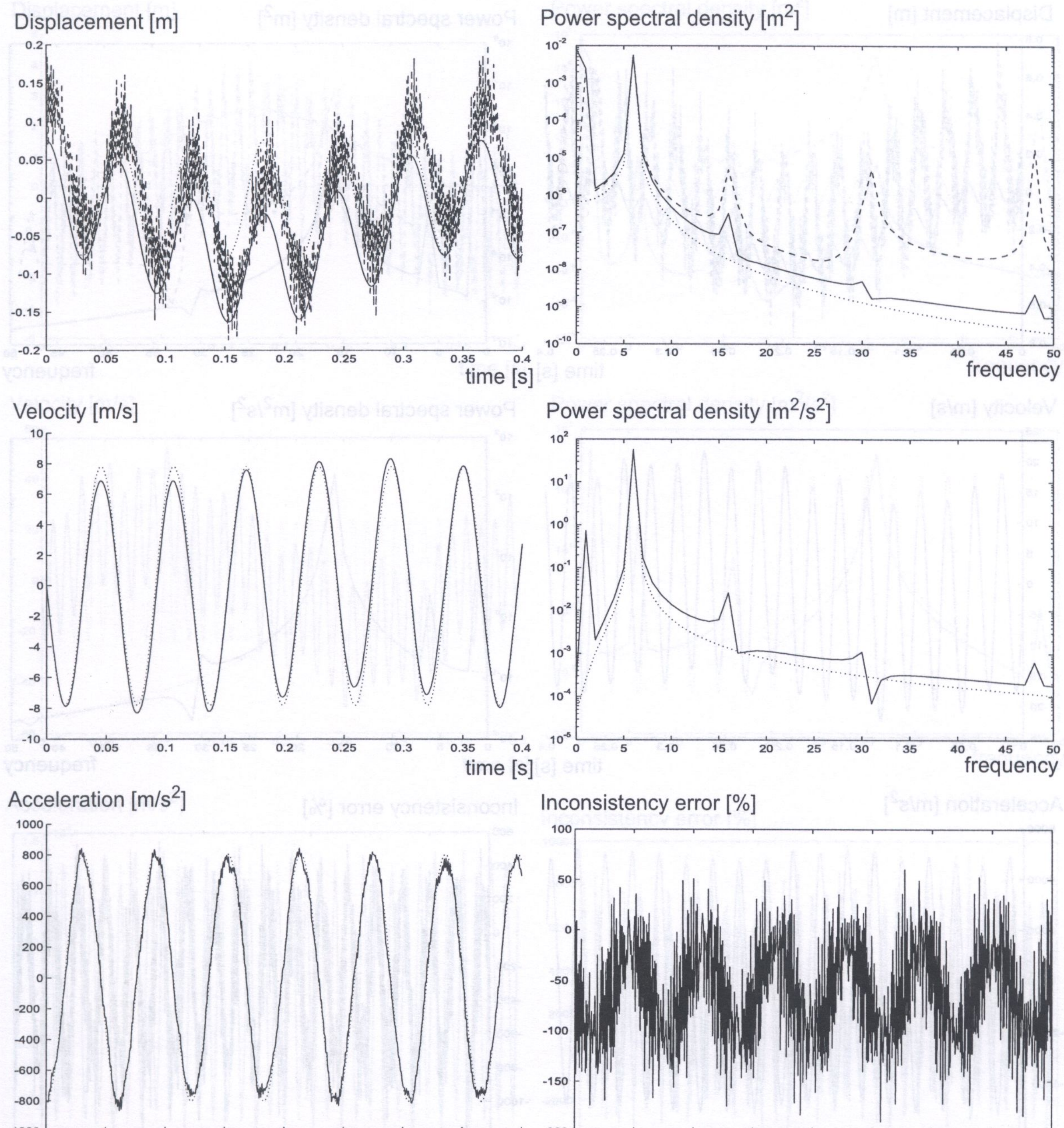

Fig. 5. The second natural mode of vibrations using the Smoliński et al. algorithm

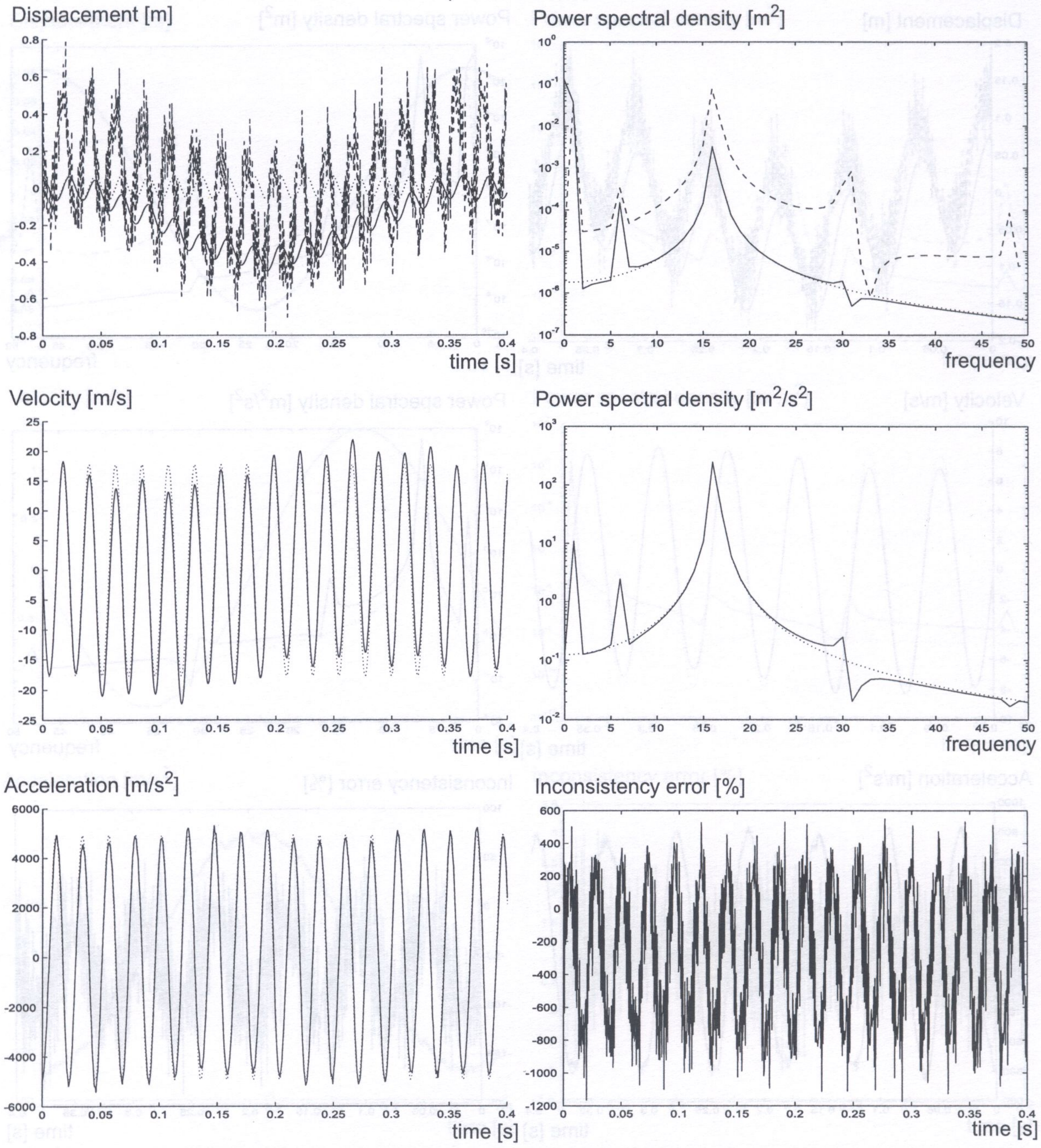

Fig. 6. The third natural mode of vibrations using the Smoliński et al. algorithm

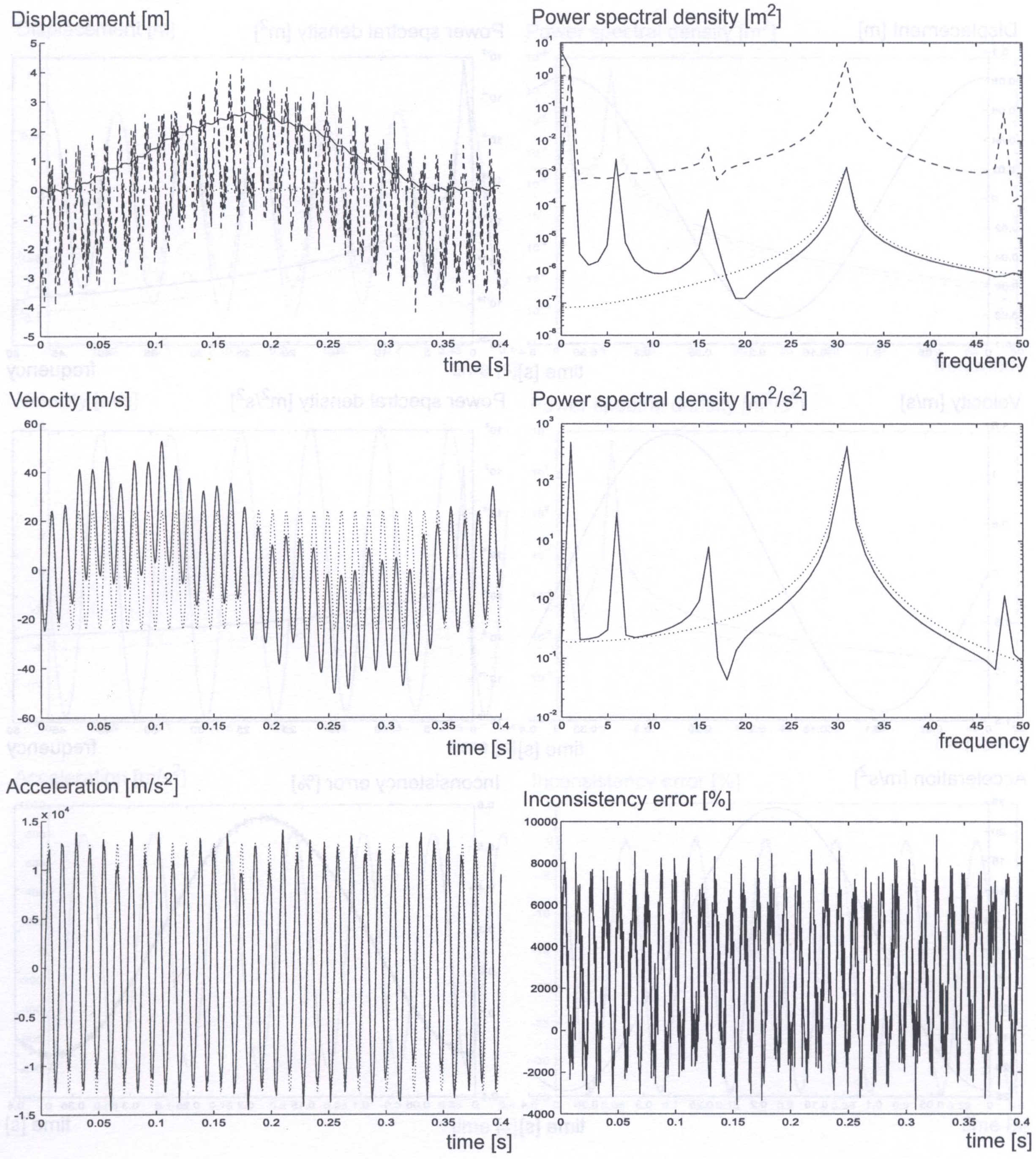

Fig. 7. The fourth natural mode of vibrations using the Smoliński et al. algorithm

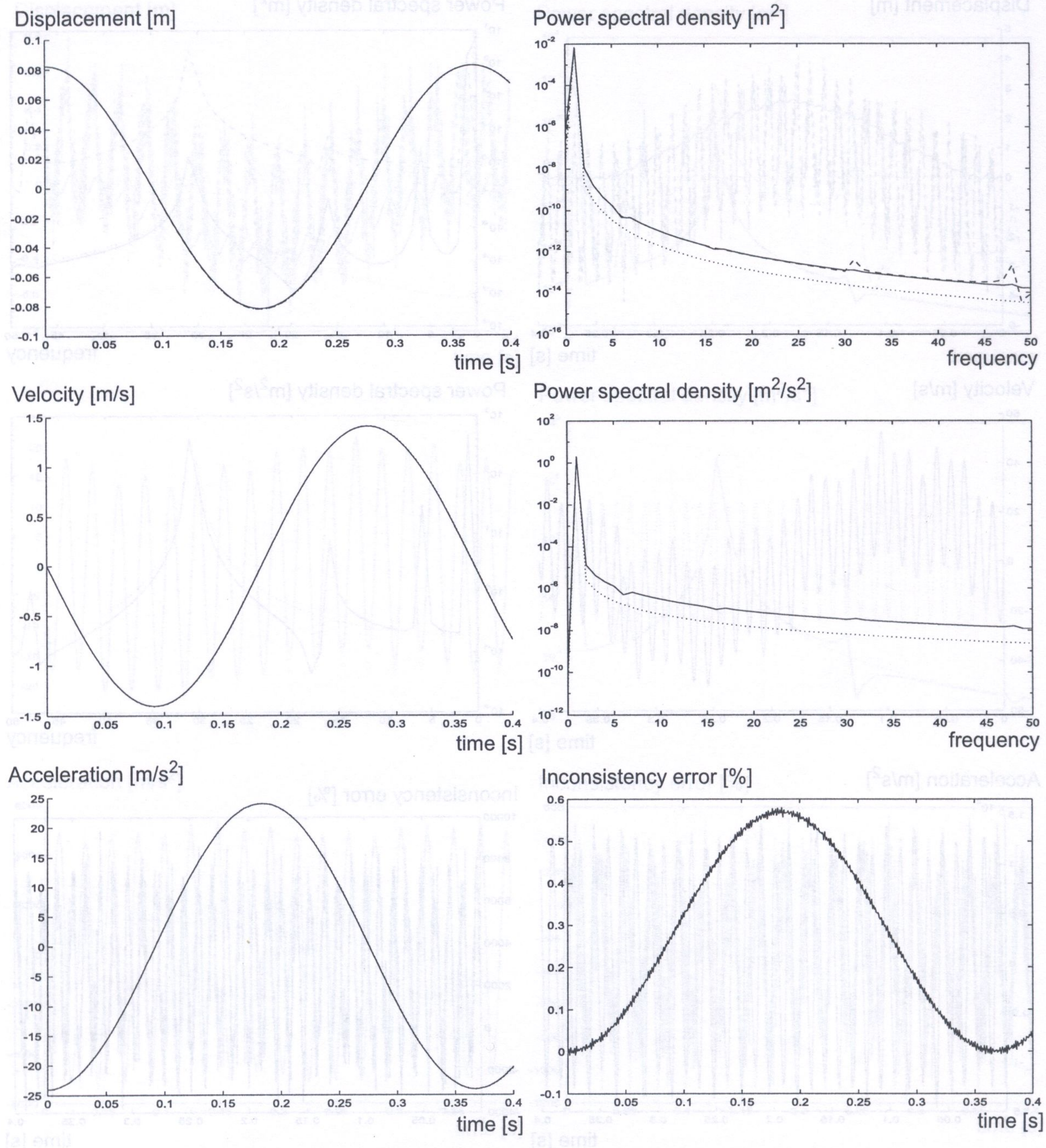

Fig. 8. The first natural mode of vibrations using the Daniel algorithm

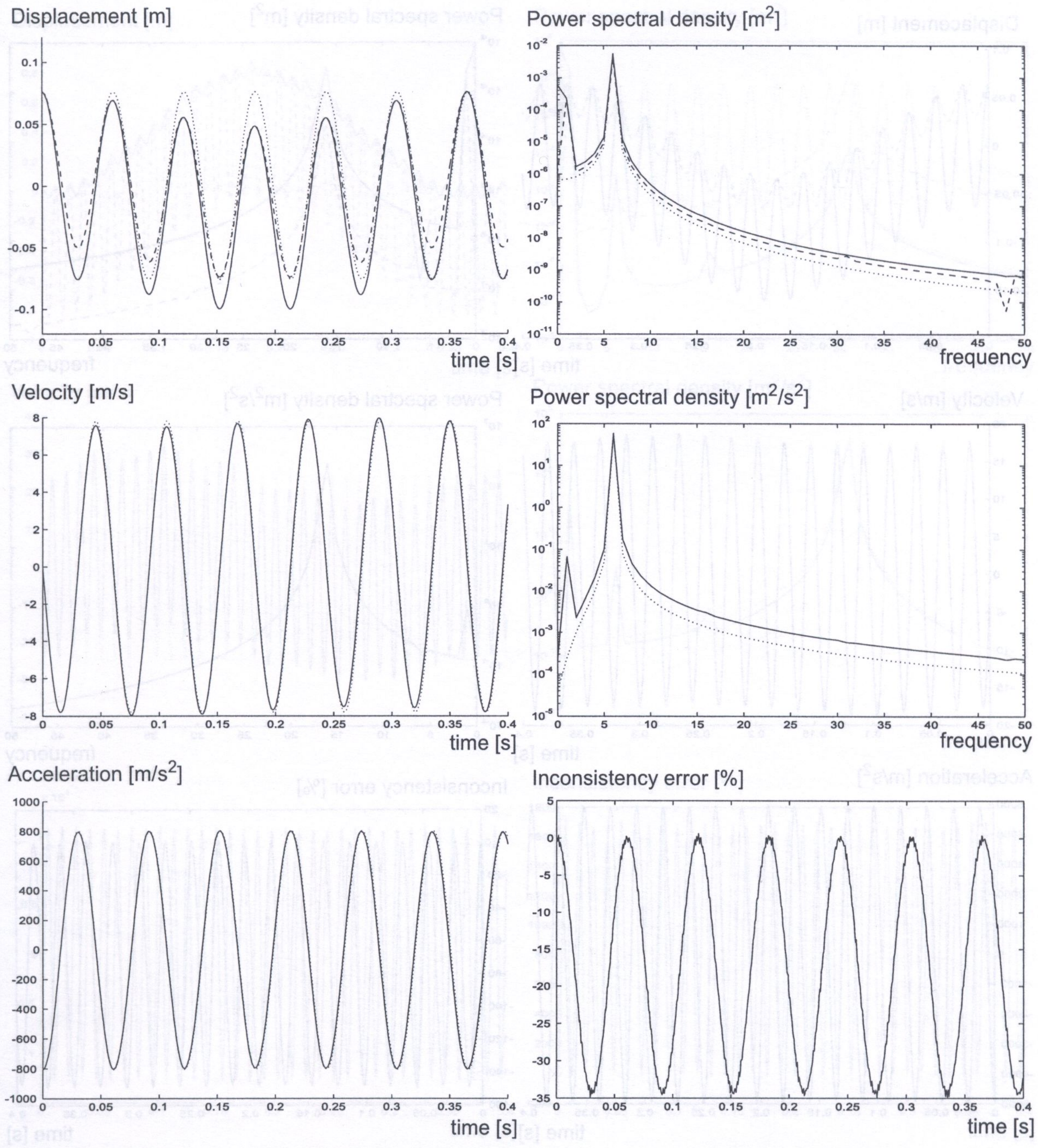

Fig. 9. The second natural mode of vibrations using the Daniel algorithm

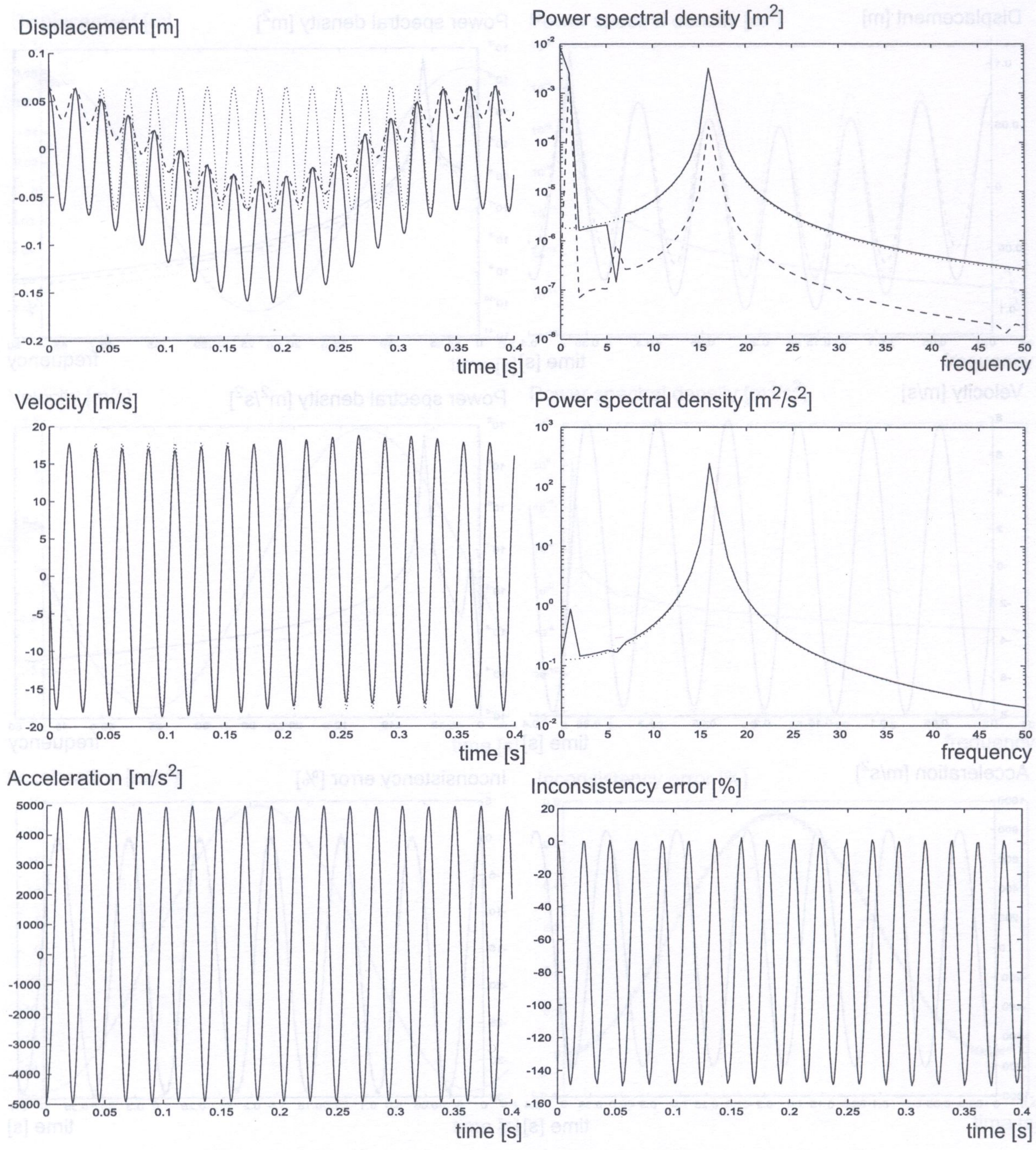

Fig. 10. The third natural mode of vibrations using the Daniel algorithm

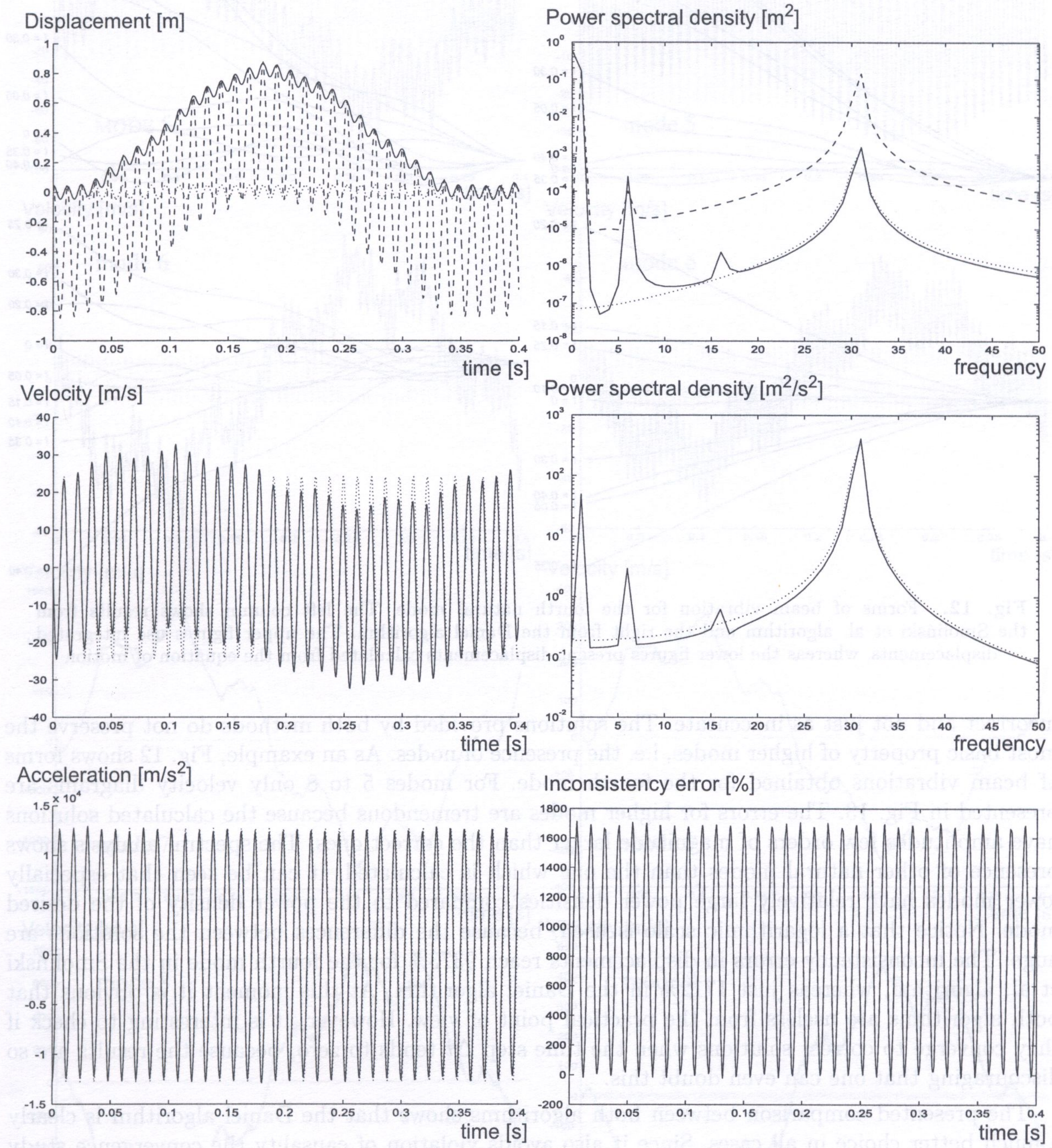

Fig. 11. The fourth natural mode of vibrations using the Daniel algorithm

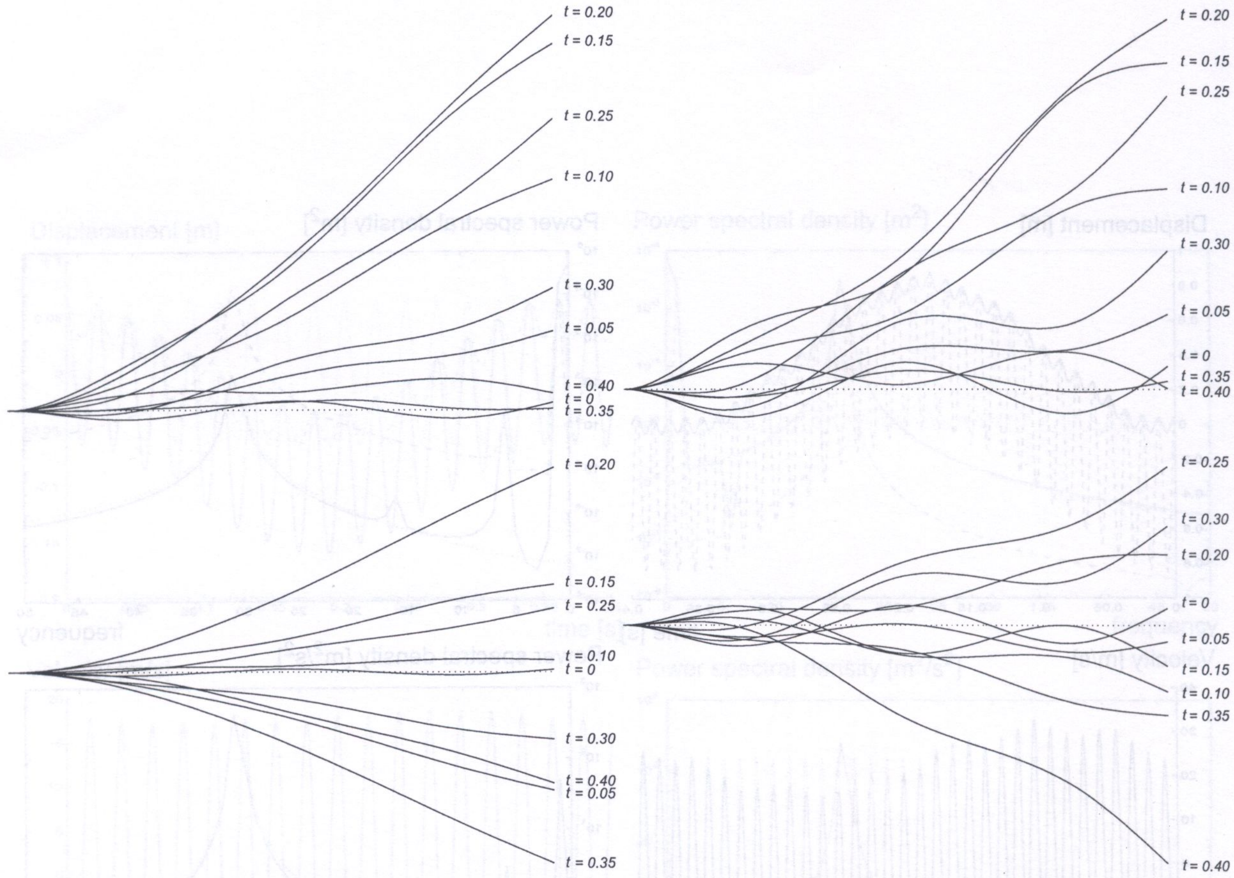

Fig. 12. Forms of beam vibration for the fourth natural mode. The left column shows results from the Smoliński et al. algorithm and the right from the Daniel algorithm. The upper figures use integrated displacements, whereas the lower figures present displacements calculated from the equation of motion

incorrect and not just as inaccurate. The solutions provided by both methods do not preserve the most basic property of higher modes, i.e. the presence of nodes. As an example, Fig. 12 shows forms of beam vibrations obtained for the fourth mode. For modes 5 to 8 only velocity diagrams are presented in Fig. 13. The errors for higher modes are tremendous because the calculated solutions have amplitudes few orders of magnitude larger than the correct ones. The spectral analysis shows presence of other natural modes than the one which is calculated. It can be seen that especially lower modes have relatively large power densities compared to the power density of the desired mode. Notice that a logarithmic scale is used, because the differences between the solutions are huge. The inconsistency errors in displacements reach 9373% for the fourth mode in the Smoliński et al. algorithm, whereas just 1712% in the Daniel algorithm. At this moment it is obvious that both algorithms are useless from the practical point of view. However, it is interesting to check if they converge to correct solutions when the time step Δt tends to zero, because the results are so discouraging that one can even doubt this.

The presented comparison between both algorithms shows that the Daniel algorithm is clearly a much better choice in all cases. Since it also avoids violation of causality the convergence study will be done for this algorithm alone. The same calculations are made with 10 times smaller step, i.e. larger time step $5 \cdot 10^{-5}$ s and smaller $\Delta t = 2.5 \cdot 10^{-5}$ s. The maximum relative inconsistency errors are presented in Table 1 and compared with the previous values.

There is no longer doubt that the algorithm converges to the correct solutions. Moreover, the inconsistency errors are roughly proportional to the square of the time step, because 10 times step reduction results in approximately 100 times smaller maximum errors. The registered error reduction is consistent with the second order of the applied integration method. The observed rate

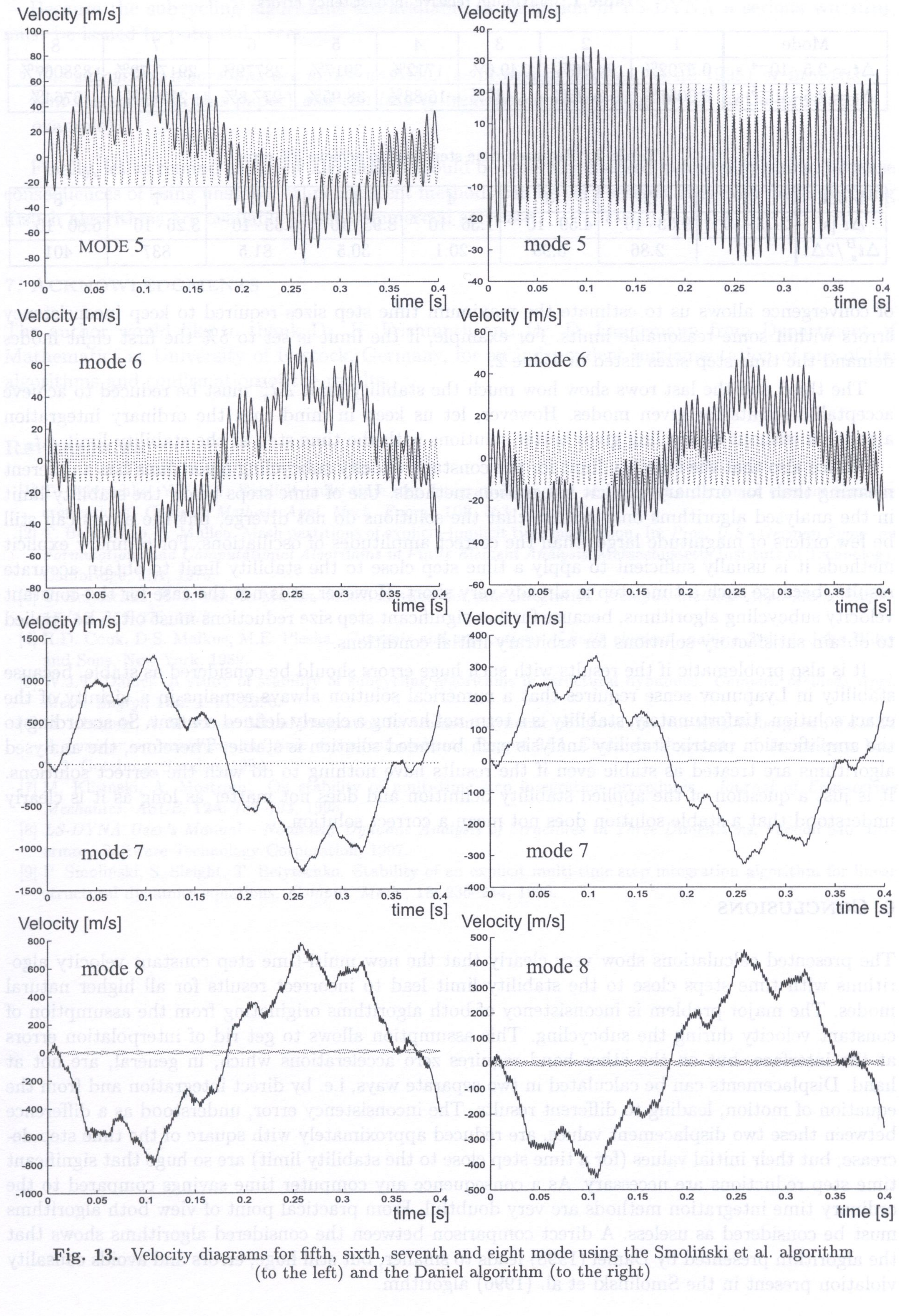

Fig. 13. Velocity diagrams for fifth, sixth, seventh and eight mode using the Smoliński et al. algorithm (to the left) and the Daniel algorithm (to the right)

Table 1. Maximum relative inconsistency errors

Mode	1	2	3	4	5	6	7	8
$\Delta t = 2.5 \cdot 10^{-4}$ s	0.5792%	34.86%	149.6%	1712%	3917%	28779%	2917179%	838067%
$\Delta t = 2.5 \cdot 10^{-5}$ s	0.0057%	0.3426%	1.482%	16.83%	38.95%	277.8%	29264%	6764%

Table 2. Required time step sizes to achieve 5% error

Mode	1	2	3	4	5	6	7	8
Δt [s]	$2.73 \cdot 10^{-4}$	$9.53 \cdot 10^{-5}$	$4.59 \cdot 10^{-5}$	$1.36 \cdot 10^{-5}$	$8.95 \cdot 10^{-6}$	$3.35 \cdot 10^{-6}$	$3.26 \cdot 10^{-7}$	$6.80 \cdot 10^{-7}$
$\Delta t_s^B / 2\Delta t$	1	2.86	5.95	20.1	30.5	81.5	837	401

of convergence allows us to estimate the maximum time step sizes required to keep inconsistency errors within some reasonable limits. For example, if the limit is set to 5% the first eight modes demand the time step sizes listed in Table 2.

The third and the last rows show how much the stability limit Δt_s^B must be reduced to achieve acceptable results for given modes. However, let us keep in mind that the ordinary integration algorithm without subcycling gives better solutions with the time step at the stability limit Δt_s.

Notice also that the stability limit for the constant velocity subcycling algorithms has a different meaning than for ordinary explicit integration methods. Use of time steps below the stability limit in the analysed algorithms only assures that the solutions do not diverge, but the errors can still be few orders of magnitude larger than the correct amplitudes of oscillations. For ordinary explicit methods it is usually sufficient to apply a time step close to the stability limit to obtain accurate results, because such a time step is already very short. However, it is not the case for the constant velocity subcycling algorithms, because further significant step size reductions must often be applied to obtain satisfactory solutions for arbitrary initial conditions.

It is also problematic if the results with such huge errors should be considered as stable, because stability in Lyapunov sense requires that a numerical solution always remains in a vicinity of the exact solution. Unfortunately, stability is a term not having a clearly defined content. So according to the amplification matrix stability analysis each bounded solution is stable. Therefore, the analysed algorithms are treated as stable even if the results have nothing to do with the correct solutions. It is just a question of the applied stability definition and does not matter as long as it is clearly understood that a stable solution does not mean a correct solution.

6. Conclusions

The presented calculations show very clearly that the new multi-time step constant velocity algorithms with time steps close to the stability limit lead to incorrect results for all higher natural modes. The major problem is inconsistency of both algorithms originating from the assumption of constant velocity during the subcycling. This assumption allows to get rid of interpolation errors at the interface, but on the other hand requires zero accelerations which, in general, are not at hand. Displacements can be calculated in two separate ways, i.e. by direct integration and from the equation of motion, leading to different results. The inconsistency error, understood as a difference between these two displacement values, are reduced approximately with square of the time step decrease, but their initial values (for a time step close to the stability limit) are so huge that significant time step reductions are necessary. As a consequence any computer time savings compared to the ordinary time integration methods are very doubtful. From practical point of view both algorithms must be considered as useless. A direct comparison between the considered algorithms shows that the algorithm presented by Daniel (1998) leads to smaller, but still huge, errors and avoids causality violation present in the Smoliński et al. (1996) algorithm.

Because the subcycling algorithms are available as an option in LS-DYNA a serious **warning** must be issued to potential users:

> *The subcycling algorithms are either unstable* (as proved in Klisiński et al. 1998) *or inconsistent* (as shown in this paper) *and can lead to tremendous errors* (e.g. $3 \cdot 10^6$ in the presented example).

Finally, let us remind that new algorithms should be carefully checked before publishing, because consequences of using unstable or inconsistent methods can be very serious. It is even more worrying if such algorithms are available in the commercial software.

7. Acknowledgments

The author would like to thank Dr. K. Frishmuth and Dr. D. Langemann from Department of Mathematics at University of Rostock, Germany, for an independent implementation of one of the algorithms and confirmation of the results.

References

[1] T. Belytschko, Y.Y. Lu. Explicit multi-time step integration for first and second order finite element semidiscretizations. *Comput. Methods Appl. Mech. Engrg.*, **108**: 353–383, 1993.

[2] T. Belytschko, R. Mullen. Mesh partitions of explicit–implicit time integration. In: *Proc. U.S.–German Symp. on Formulations and Computational Algorithms in Finite Element Analysis*, Massachussetts Institute of Technology, Cambridge, MA, 1976.

[3] T. Belytschko, H.-J. Yen, R. Mullen. Mixed methods for time integration. *Comput. Methods Appl. Mech. Engrg.*, **17/18**: 259–275, 1979.

[4] R.D. Cook, D.S. Malkus, M.E. Plesha. *Concepts and applications of finite element analysis*, 3rd ed. John Wiley and Sons, New York, 1989.

[5] W.J.T. Daniel. A study of stability of subcycling algorithms in structural dynamics. *Comput. Methods Appl. Mech. Engrg.*, **156**: 1–13, 1998.

[6] M. Klisiński, A. Mostrom. Mixed integration for transient dynamic problems. In: Nils-Erik Wiberg, ed., *FEM-94 A seminar on nonlinear and time dependent problems*, Report 94:1. Chalmers University of Technology, May 2–3, Goteborg, Sweden, 1994.

[7] M. Klisiński, A. Mostrom. On stability of multi-time step integration procedures. *Journal of Engineering Mechanics, ASCE*, **124**: 783–793, 1998.

[8] *LS-DYNA User's Manual - Nonlinear Dynamic Analysis of Structures in Three Dimensions*, Version 940. Livermore Software Technology Corporation, 1997.

[9] P. Smoliński, S. Sleight, T. Belytschko. Stability of an explicit multi-time step integration algorithm for linear structural dynamics equations. *Comput. Mech.*, **18**: 236–244, 1996.