Finite element analysis of a 2-D physical model of the human head

Technische Universiteit Eindhoven
Faculteit Werktuigbouwkunde
Vakgroep Fundamentele Werktuigkunde
Sectie Dynamica

Finite Element Analysis of a 2-D physical model of the human head.

D.W.A. Brands
W.F.W report nr. 95.178

stage verslag

door: D.W.A. Brands
id nr.: 333241
begeleiders: Dr. Ir. A.A.H.J. Sauren
Ir. M.H.A. Claessens

Summary

It is widely accepted that deformations of the brain mass, as a result of an impact to the head or as a result of inertial loading, lead to brain injury. That is why the biomechanical response of the brain is to be determined under extreme loading conditions. Because of the absence of measuring data from human models, physical or numerical models are used.
In this report a two-dimensional F.E. model of a mid-sagittal cross-section of the human head is presented. This model is a simplified version of the physical model described in the report by Pape (1994). It consists of a circular mesh with plane strain, second order, isoparametric, distorted quadrilateral elements that represent the brain tissue. The angular displacements of the outer nodes of this mesh are prescribed using a polynomial approximation of the angular displacement of the physical model. The radial displacements of these nodes are zero. This means that they rotate around the centre of the mesh as if they were part of a rigid body. For this reason no skull is modelled. The brain tissue is modelled as a homogeneous isotropic linear elastic material.
When looking at contour plots of the displacements in x and y direction a strange irregular shape appears. To investigate this the spatial and temporal discretization of the model is investigated to see if this phenomenon is a result of wave propagation. Therefore some wave propagation properties are determined. Also an assumption about the shape of the waves has been made. It is investigated if these properties and the assumed shapes of the wave fronts really appear in the model. From this investigation it seems that there occurs an oscillation in the tangential nodal displacements that cannot be explained using wave propagation theory. Some properties of this oscillation will be determined and some suggestions will be done for future investigations including one that is a possible explanation for this phenomenon.

Table of contents

1. Introduction

The project described in this report was conducted in the field of head injury research. It is widely accepted that deformations of the brain mass, as a result of an impact to the head or as a result of inertial loading, lead to brain injury. For this reason it is necessary to know what the biomechanical response of the brain (that is the mechanical behaviour as a result of a mechanical load) is under extreme loading conditions. For the determination of this biomechanical response one can use numerical methods as well as experimental methods

A problem for the determination of this response is the absence of experimental data with real human beings. Therefore physical models are being used as a substitute. These usually have a simplified geometry and represent a coronal or sagittal cross-section of the human head. Often the model consists of a cylindrical shell representing the skull which is filled with a silicone gel that represents the brain tissue. The model is subjected to an angular and/or linear acceleration and the deformation of the gel is measured. To model relative motion between skull and brain, skull-brain decoupling can be applied by putting, for instance, a layer of Teflon between the shell and the gel as is reported by Pape (1994).

Another way to analyse the response of the human head to an extreme loading condition is by making use of mathematical models. In this project the Finite Element Method is used to create a mathematical model. Other possibilities are lumped parameter models and continuum models [Sauren and Claessens (1993)]. The model used here is a two-dimensional plane strain representation of the mid-sagittal cross-section of the human head. As a first approach this cross-section was assumed to be circular. The model consists of a 2-D mesh with quadratic quadrilateral elements that represent the brain tissue. The elements have linear elastic material properties. The angular acceleration will be forced upon this model by prescribing the motion in every node at the boundary of the mesh. As a result the boundary of the mesh behaves like a rigid shell with a thickness of one node. This is the reason why no skull is modelled, it would just behave like a rigid body.

The numerical model is based on a physical model as described in a report by Pape (1994). In that report a cylindrical physical model was subjected to a rotational acceleration around a rotational axis in the centre of the cylinder. This physical model consisted of a cylindrical container with an inner geometry that resembles a two dimensional representation of the real anatomic configuration of a mid-sagittal cross-section of a human head. In the cylinder a gel represented the brain tissue.

In this report first the numerical model and its boundary conditions will be introduced in chapter 2. Chapter 3 deals with the effects of spatial discretization. From the displacements calculated to investigate the effects of mesh refinement it seems that there is a possibility of longitudinal and/or transverse wave propagation in the model. Therefore some basic wave

propagation theory will also be given in this chapter. Chapter 4 contains an investigation of the effects of temporal discretization, in relation with wave propagation. Chapter 5 gives some conclusions and hints for future projects in a summary of this project and a discussion.

2. The Finite Element Model

In this chapter the numerical model will be presented.First the geometry of the mesh and material properties are given, then the boundary conditions that define the motion of the mesh will be discussed. Also there will be a short review about the integration method used. The calculations are performed with the finite element package MARC version k6.1.

2.1 Introduction

The numerical model is based on a physical model as described by Pape (1994). This cylindrical physical model was subjected to a rotational acceleration around a rotational axis in the centre of the cylinder. The physical model consisted of a cylindrical container with an inner geometry that resembles a two dimensional representation of the real anatomic configuration of a mid-sagittal cross-section of a human head. In the cylinder the brain tissue is represented by a gel. The deformations of the gel as a result of the rotational acceleration were measured using high-speed cinematography recordings of markers that were placed in the gel. The stiffness of the cylinder and its inner geometry is such that it can be considared rigid. The height of the container is such that it may be assumed that the material in the plane of interest behaves as if the cylinder was of infinite height. The influence of slip and non-slip boundary conditions at the skull-brain interface were also considered in the report by Pape.

2.2 Mesh geometry and material properties

The model used in this report is a simplified version of the physical model used by Pape, because no inner geometry was modelled. This was done to keep the numerical costs low and to get a better understanding of the phenomena that occur in this model. The numerical model is a representation of a cross-section of a cylinder of infinite height so plane strain theory may be applied. This means that no motion is allowed perpendicular to this cross-section. The model used represents a two-dimensional circular flat surface with a radius of 98 *mm*. The element distribution in this mesh is the same as used by Kessels and Peerlings (1993) and is shown in Fig 2.1. There is one difference. In the model by Kessels and Peerlings the outer element ring of the mesh was used for modelling the skull. Now the whole mesh is used for modelling the brain and no elements are used for modelling the cylindrical container. This has been done for two reasons.
The first reason is that, as mentioned in section 2.1, the cylindrical container of the physical model behaves like a rigid body. This container will show almost no deformations in comparison with the gel and is therefore not of interest.
The second reason is that in the numerical model is assumed that there is no relative motion between the gel and the cylinder. In the real human head relative motion between the brains

The second reason is that in the numerical model is assumed that there is no relative motion between the gel and the cylinder. In the real human head relative motion between the brains and the skull is possible. This motion could be modelled with contact algorithms in a Finite Element package but there are no reliable contact algorithms for transient dynamic analysis in the Finite Element package MARC. The modelling of a rigid cylinder to use these algorithms is therefore not necessary.

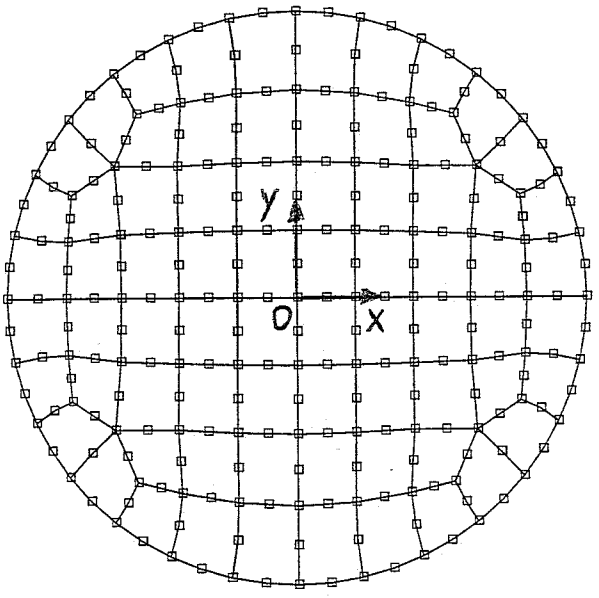


Figure 2.1: Mesh shape of unrefined mesh.

The mesh is located in the plane defined by the x-axis and the y-axis of an orthogonal coordinate system. The z-axis' direction is perpendicular to this plane. The origin of the coordinate system lies in the centre of the mesh.
The elements used are plane strain, second order, isoparametric, distorted quadrilateral elements (element number 27 in MARC). This element type has eight nodes and nine integrationpoints. Their configuration is given in Fig. 2.2. Four noded linear elements have also been considered and tested but they did not give results that were accurate enough.

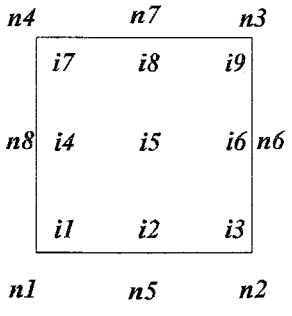


*Figure 2.2: Element number 27, **n** denotes node numbers, **i** denotes integration point numbers.*

- Young's modulus : $E = 1{,}0 \cdot 10^5 \left[{}^{N}/_{m^2} \right]$ used by Lee (1990)
- Poison's ratio : $\nu = 0.48$ used by Ruan (1991)
- Density : $\rho = 1000 \left[{}^{kg}/_{m^3} \right]$ commonly used

2.3 Kinematic boundary conditions

As mentioned before the model is of the plane strain type so no displacements in z-direction are allowed. The nodes have no constrains for displacements in the x,y-plane except those situated at the mesh boundary. These nodes move according to prescribed displacements.
Because the cylindrical container of the physical model behaves like a rigid body, and while no relative motion is allowed between the container and the gel, the nodes at the border of the mesh have all the same tangential displacements. These nodes have no displacements in radial direction. This means that the only displacement that has to be prescribed is an angular one. The angular displacement is represented in Fig. 2.3 by angle α.

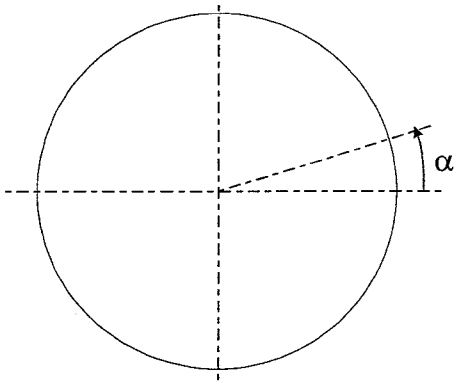


Figure 2.3: Definition of direction of prescribed angle α.

To get a realistic prescribed displacement, the data of the angular displacement of the cylindrical container in experiment M10, as described by Pape (1994), has been used for the displacement of the numerical model.
A problem with this displacement data is that it contains a finite number of data points. Another problem is that the only source of data is a graph in which the angular displacement against time is plotted. If this angular displacement is to be prescribed with very small time steps it is necessary to know the displacement at all these points in time. Therefore it is useful to have a function that can describe the angular displacement for every moment in time.
To get to know this function first some displacements and their points in time are measured from the graph. Then a least square approximation is performed using MATLAB, to get a polynomial that describes the measured points as a function of time.
Another problem is that the angular displacement reported by Pape starts at *-0,3 rad* and ends at zero *rad*. This leads to numerical problems at the beginning of the experiment. The displacement used in this report starts at zero *rad* and ends at *0,3 rad*. This is done by giving

the first polynomial coefficient the value zero instead of the value of *-0,3*. The polynomial approximation and the data measured from the graph are displayed in Fig. 2.4.

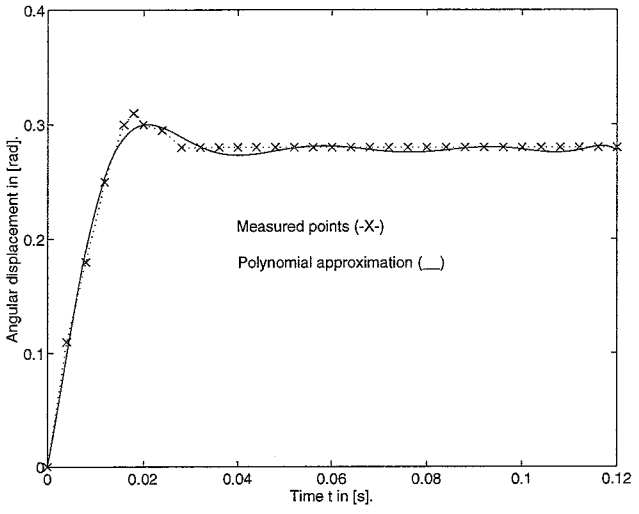


Figure 2.4: Polynomial approximation of degree 9 for angular displacement of vessel in experiment M10 in Pape (1994).

The Finite Element package MARC offers the possibility of making a user subroutine, *forcdt.f*, in which nodal displacements as a function of time can be defined. The MARC program calls this routine each time step for a selected number of nodes, and forces the defined incremental displacements on the selected nodes. The user subroutine used here is based on a routine programmed by B. Michielsen. This routine is capable of calculating the angular displacement, at any moment in time, using the polynomial coefficients of the polynomial approximation. It then calculates for every desired node the incremental displacements during one time step out of the difference of the calculated angles at the beginning and the end of the increment, and out of the initial coordinates of that node. This subroutine was altered in such way that it now also offers the possibility of prescribing a rotation with an non-central positioned rotation axis. This may be useful in future research. For more information about this subroutine the reader is referred to Appendix A.

2.4 The integration method

The analysis that is to be performed is a dynamic transient analysis. This means that instead of only a spatial discretization also a temporal discretization is necessary. These discretizations can be described independently.

With linear elastic materials and small displacements, the system of equations resulting from spatial discretization becomes:

$$\underline{M} \cdot \ddot{\underline{u}} + \underline{K} \cdot \underline{u} = \underline{f}(t) \qquad (2.1)$$

These are the equations of motion in which the columns $\underline{u}$ and $\underline{f}$ contain the nodal displacements and the nodal external forces, respectively. $\underline{M}$ is the kinematically consistent mass matrix and $\underline{K}$ the stiffness matrix. These matrices are independent of $\underline{u}$ and its time derivatives because of the assumption that the problem is linear.
Temporal discretization can be applied next. The system of differential equations in time, eq.(2.1), can be solved by using direct time integration schemes. These calculate the relevant quantities such as displacements, velocities and accelerations at a number of equally spaced discrete points in time. The integration scheme used in this report is the Newmark-β method. It consists of the following equations:

$$\underline{u}_{n+1} = \underline{u}_n + \Delta t \underline{\dot{u}}_n + \frac{\Delta t^2}{2}\left[(1-2\beta)\underline{\ddot{u}}_n + 2\beta \underline{\ddot{u}}_{n+1}\right] \tag{2.2}$$

$$\underline{\dot{u}}_{n+1} = \underline{\dot{u}}_n + \Delta t\left[(1-\gamma)\underline{\ddot{u}}_n + \gamma \underline{\ddot{u}}_{n+1}\right] \tag{2.3}$$

Here the following abbreviations are used; $\underline{u}_n = \underline{u}(t_n)$, $\underline{\dot{u}}_n = \underline{\dot{u}}(t_n)$ and $\underline{\ddot{u}}_n = \underline{\ddot{u}}(t_n)$. The time step in one increment is $\Delta t = t_{n+1} - t_n$. With these abbreviations the equation of motion at time t_{n+1} is

$$\underline{M} \cdot \underline{\ddot{u}}_{n+1} + \underline{K} \cdot \underline{u}_{n+1} = \underline{f}_{n+1} \tag{2.4}$$

The eq. (2.2) and (2.3) only hold for $0 \le \beta \le \frac{1}{2}$ and $0 \le \gamma \le 1$. If $\beta = \frac{1}{4}$ and $\gamma = \frac{1}{2}$ this method is also known as the *(Constant) average acceleration method* or *trapezoidal method.* The method then is implicit, unconditionally stable and it introduces no numerical damping [Hoof, van (1994)]. This means that large time steps can be used, although smaller time steps lead to more accurate solutions. A drawback of an implicit integration method is that because it solves the system of equations for every increment it is a numerically expensive method. An explicit integration method uses extrapolation and is therefore numerically cheaper.
Applying the values $\beta = \frac{1}{4}$ and $\gamma = \frac{1}{2}$ in eqs. (2.2) and (2.3) and using the equation of motion (2.4), at time t_n yields:

$$\left[\frac{4}{\Delta t^2}\underline{M} + \underline{K}\right]\Delta \underline{u} = \Delta \underline{f} + \underline{M}\left[2\underline{\ddot{u}}_n + \frac{4}{\Delta t}\underline{\dot{u}}_n\right] \tag{2.5}$$

In which the following holds, $\Delta \underline{u} = \underline{u}_{n+1} - \underline{u}_n$ and $\Delta \underline{f} = \underline{f}_{n+1} - \underline{f}_n$. This system is easy solvable especially when Δt has a constant value.
In the first increment the right hand side of eq. (2.5) contains initial velocities and accelerations. Usually only initial velocities and initial displacements can be defined in a Finite Element package. Therefore the initial accelerations are calculated with the following algorithm.

$$\underline{M} \cdot \underline{\ddot{u}}_0 + \underline{K} \cdot \underline{u}_0 = \underline{f}_0 \tag{2.6}$$

Summarising, when the stiffness and mass matrices are known from the spatial discretization and the initial velocities and displacements of the nodes in the mesh are also known, it is possible to calculate the initial accelerations from eq. (2.6) and solve eq. (2.5) for every next time step. This will be done using the MARC default solver that uses direct back-substitution.

3. The effects of spatial discretization

To analyse the effects of the spatial discretization on the results the same simulation has been performed with three different meshes. The first mesh is a coarse mesh. The second mesh is a version that has been refined once. The third mesh is a refined version of the second one. The way in which this refinement is performed is treated first. Next two simulations are described. The first one is performed with a time step of *0,6 ms* and serves mainly to get a qualitative indication of the quality of the mesh used in a numerical cheap manner. The results obtained from this first experiment are examined more closely by performing a second simulation in which nodal displacements in the different meshes are compared with each other using a smaller time step of *0,2 ms*. In the last section of this chapter the results of the two simulations are discussed. For this some wave propagation properties will be determined.

3.1 Mesh refinement

To get a mesh refinement all the elements of the coarse mesh that is shown in Fig. 2.1, are divided into four new elements. This is done by dividing the sides of the elements into two parts of equal length. By connecting the points that divide the element sides in the way as shown in Fig 3.1 four new elements are generated.

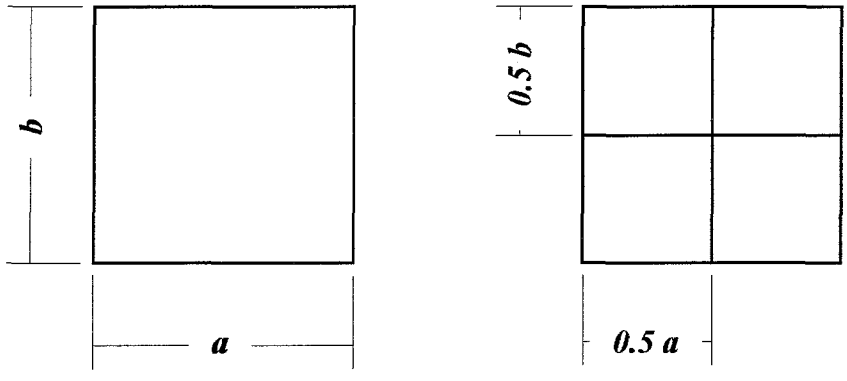


Figure 3.1: Manner in which mesh refinement has been performed on one element.

The second mesh is obtained by applying this refinement once on the original coarse mesh. The third mesh is a refined version of the second one obtained in the same way.

3.2 First simulation with large time step

The main objective of this run is to get a first impression of the quality of the meshes used with low numerical costs. This means that the number of calculations has to be as low as possible.
An important criterion for the quality of the mesh is the rotational symmetry. Because of the motion forced upon each of the meshes is a rotation around the centre of the mesh, as

discussed in chapter 2 and the model itself is axi-symmetric, this means that the displacements in the mesh have to satisfy rotational symmetry around this point too.

3.2.1 Description of numerical experiments

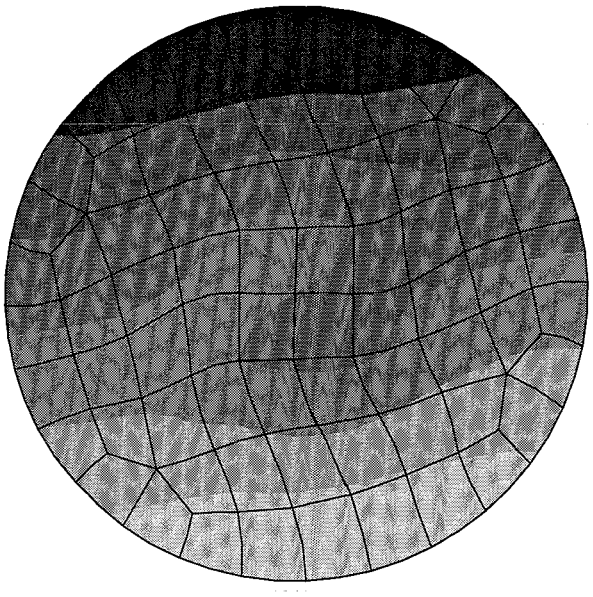
Factors that determine the numerical costs are among others the choice of the integration method, the element size in the mesh, the number of time increments and the number of quantities that have to be calculated (for instance strains, pressures, forces, displacements, and so on). Not all these factors can be chosen in the numerically cheapest way because the choice of some of them is gouverned by other criteria. The choice of the factors that determine the numerical costs will be discussed below.

- *The integration method*; This is chosen to be the Newmark-β method, because it is always unconditionally stable no matter what time step size or element size is chosen.
- *The element size in the mesh*; Larger elements will result in lower costs but also in a lower accuracy. The element size cannot be chosen in the numerically cheapest way, because this factor is to be varied to see what happens to the accuracy.
- *The number of time increments*; If the number of increments is small the number of times that the solution of the equations of motion has to be solved is small. A drawback is that if a relatively large time period has to be described a large time step is needed. This leads to a lower accuracy of the integration method. Nevertheless a small number of increments with a large time steps is used.
- *The number of quantities calculated*; The displacements are the primary variables calculated by the Finite Element method; all other output quantities are derived from them. To keep numerical costs low the output of this simulation contains only displacements in global x- and y-direction.

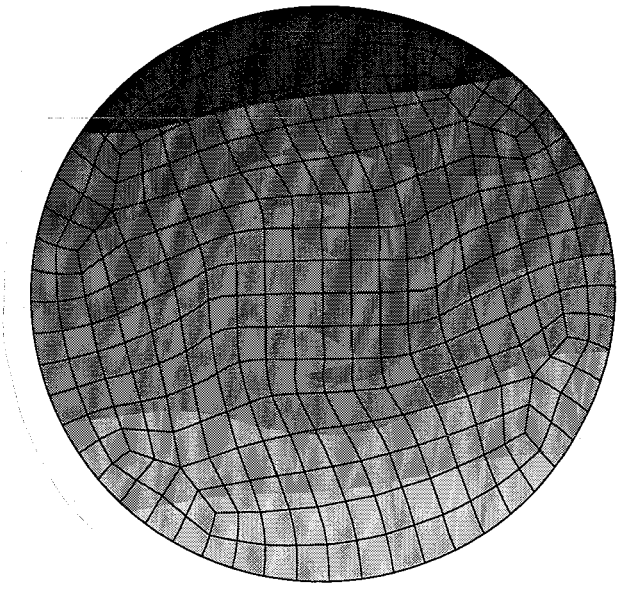
For the reasons mentioned above a run of 25 time increments of *0.6 ms* each is chosen for each mesh. The prescribed motion in each run is the one that is explained in section 2.3. The results of these runs contain only displacements in x- and y-direction. In appendix B a MARC input file and the user subroutine used are given for one mesh.

3.2.2 Results

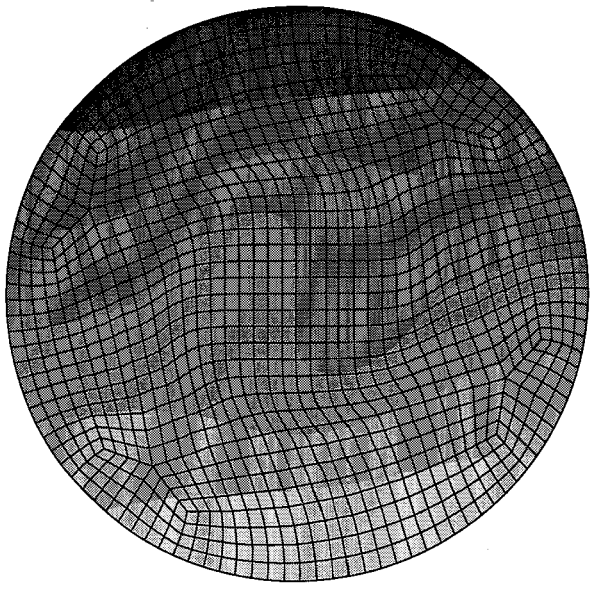
The output is plotted in contour plots for certain moments in time. The plots for the displacements in x- or y-direction of the three meshes for several moments in time have been compared with each other. Fig. 3.2 contains contour plots of the displacement in x-direction after the eighteenth time increment for each mesh.



First mesh



Second mesh



Third mesh

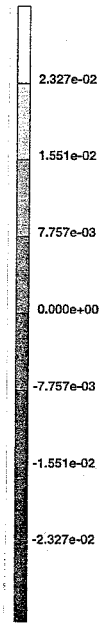


Figure 3.2: Displacements in x-direction on time = 10.8 ms (= 18^{th} increment) for the three meshes. The grey scales have the same values for all pictures in this figure.

When comparing the meshes one notices that for all meshes the shape of the contours in the outer area of the mesh are the same. Also it can be seen that there is a symmetry in the displacements. The displacements in the upper part of the meshes have opposite signs as the displacements in the lower part of the mesh. This means that the displacements in x-direction satisfy rotational symmetry.

The contour plots of the displacements in the inner area of the three meshes do not have the same shape. The line on which the displacement in x-direction equals zero has a very different shape for the different meshes. In the finest mesh it has a very smooth shape while in the two

coarser meshes it has an irregular shape. But this irregular shape also satisfies rotational symmetry. Because the large numerical costs involved no further mesh refinement is performed but a second series of experiments with a smaller time step to investigate the appearance of this irregular shape and to see if it is the result of numerical inaccuracies.

3.3 Second simulation with small time step

The results of the simulations mentioned above are probably not very accurate because of the large time step taken. This is the reason why a new series of simulations has been performed with a smaller time step. As mentioned in chapter 2 the Newmark-β method becomes more accurate for smaller time steps.

3.3.1 Description of numerical experiments

This simulation consists of 100 increments of *0.2 ms* each for the second and third mesh. The total simulation time then becomes *20 ms* which is longer than in the first series (the total simulation time there was *15 ms*). For the first mesh the simulation consists of 50 increments (the total simulation time is then *10 ms*). The displacement forced upon the geometry is the same as used in the first experiment and is discussed in chapter 2. Again the output only consists of the displacements.

3.3.2 Results

From the contour plots of the different meshes it can be concluded again that the displacements in *x*- and *y*-direction satisfy rotational symmetry. But the irregular shape is still there and appears now also in the third mesh.
To see more of the effects of mesh refinement the displacements per node are plotted against time for every mesh. This is done for eight nodes all situated on a line between the centre of the mesh and the border of the mesh in *y*-direction. These nodes are shown in Fig. 3.3 for the first mesh and are marked with a big dot. In the other meshes nodes with the same initial coordinates are used.

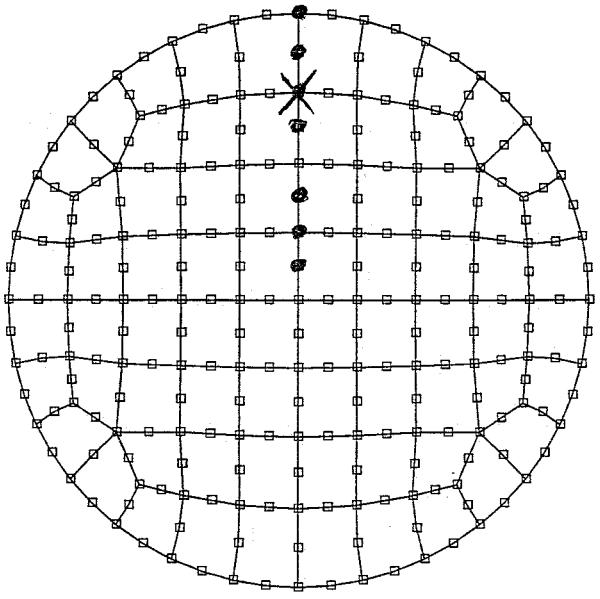


Figure 3.3: First mesh to illustrate the positions of the observed nodes.

The node marked with a cross will be used as an example. The observations made for this node hold also for the other nodes marked in Fig 3.3.
First will be looked upon the displacement in x-direction. This displacement will be called u in the remainder of this report. Fig 3.4a contains the displacements u of the three meshes for the time interval from *0* to *20 ms* (*0* to *10 ms* for the first mesh).

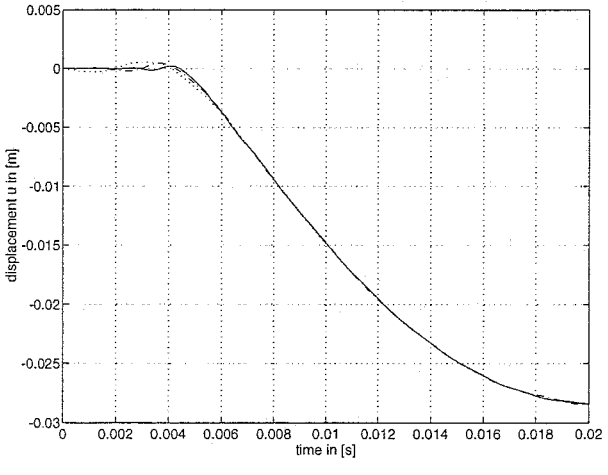


Figure **a**

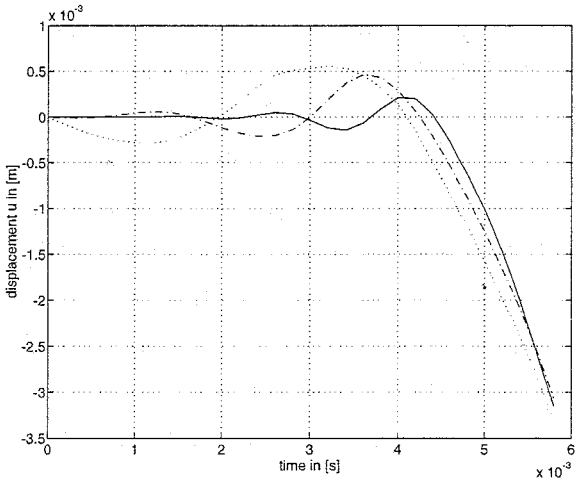


Figure **b**

Figure 3.4: Displacement in x-direction in m, for first (. . .),second(- . -) and third mesh (__) for the node denoted with a cross in fig 3.3.

***a**: time interval is 0 to 20 ms*

***b**: time interval is 0 to 6 ms*

It can be seen that the displacements differ a lot until approximately $t = 6$ *ms*. To get a better look at the displacements before $t = 6$ *ms*, a graph, in which only the first *6 ms* are plotted, is shown in Fig 3.4b. In this figure an oscillation in the displacements is visible. This oscillation

lasts up to about *4 ms* for the coarse mesh and up to about *4,5 ms* for the third mesh. The oscillations have different shapes for each mesh and it is not obvious that they converge to a certain unique solution. After $t = 6$ *ms* the difference between the displacements of a node in two meshes with successive fineness is of the order 10^{-4}. The displacement u then is about 10^{-2} thus the change as a result of mesh refinement is in both cases about 1 %.

The plots for the displacement in y-direction, which will be called v, are shown in Fig 3.5.

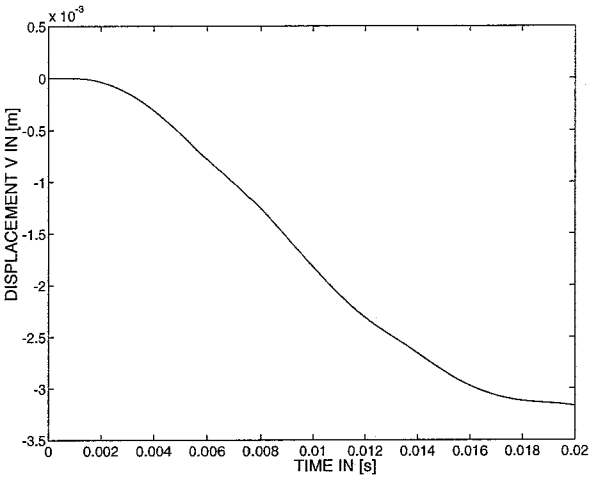


Figure **a**

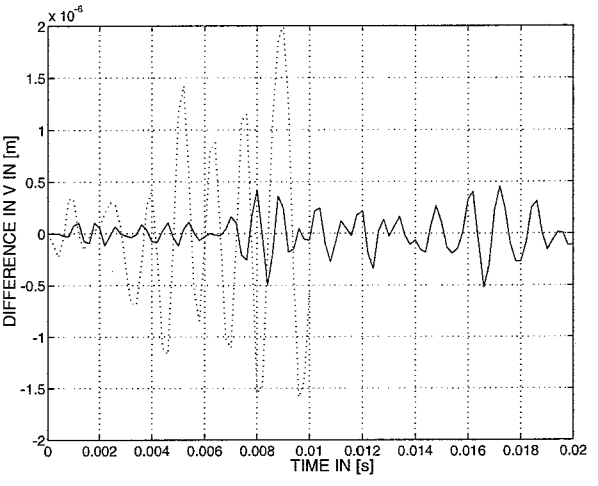


Figure **b**

Figure 3.5: ***a****: Displacement in y-direction for coarse (. . .), middle (- . -) and finest mesh(—) for the node denoted with a cross in fig 3.3.*

b*: Differences in displacements in y-direction.* v_{first} - vs_{econd} (. . .) *and* v_{second} - v_{third} (—)

Figure 3.5*a* contains the displacements v for each mesh in a time interval from *0* to *10 ms* for the coarse mesh and from *0* to *20 ms* for the two more refined meshes. There is hardly any difference visible for the displacement v of the node in the different meshes.
This is the reason why in Fig 3.5b the differences between the displacements versus the time are shown. This figure contains respectively v_{first}-v_{second} and v_{second}-v_{third}. v_{first} is the displacement in y-direction of the node in the coarse mesh. v_{second} is the displacement of the node in the mesh that is refined once, and v_{third} the one of the node in the mesh refined twice.
The difference v_{first}-v_{second} has order 10^{-6} *m*. The difference v_{second}-v_{third} has order 10^{-7} *m*. The displacements v have order 10^{-3}. This means roughly that the largest improvement because of mesh refinement is about 0,1 % for displacements in y-direction.

3.4 Discussion of the results

When only looking at the change of the displacements in y-direction as a result of mesh refinement it seems that no mesh refinement is necessary. The changes are about *0,1* % of the displacements when replacing the coarse mesh with the second mesh. When replacing the

latter mesh with the one that has been refined twice the change has order 10^{-7} *m*, that is about *0,01 %* of the displacements.
The displacement in *x*-direction shows for all the examined nodes first an oscillation that seems to disappear after a certain moment in time. The displacement then becomes negative. This oscillation in the displacement in *x*-direction of the nodes could be the result of wave propagation in the mesh. To check if this is true, some wave propagation properties of this mesh will be determined. Also it will be made clear which shape the waves are likely to have in the model.

3.4.1 Determination of wave propagation properties.

All the equations and mathematical theory used in the determination of the wave propagation properties are taken from the report by van Hoof (1994). The theory presented here holds for an unbounded elastic medium. In such a medium only two types of waves can propagate, *dilatational* waves and *rotational* waves.
In a dilatational wave the material particles of the medium move in the same direction as the wave it self, this means in a direction perpendicular to the wave front. This wave is also called *longitudinal* or *compressive* wave. The latter because the velocity of this wave is also the velocity of the propagation of a change in volume in the medium.
In a rotational wave the particles in the medium move in a direction perpendicular to the direction of the wave propagation, i.e. in a direction parallel to the wave front. This wave type is also called *shear*, *distortional* or *transverse* wave.
The longitudinal disturbance in an unbounded elastic material will propagate at the velocity

$$c_l = \sqrt{\frac{\lambda + 2\mu}{\rho}} \tag{3.1}$$

Transverse waves propagate with velocity

$$c_t = \sqrt{\frac{\mu}{\rho}} \tag{3.2}$$

In these two equations μ and λ are the Lamé constants. The mass per unit volume of the material is ρ. The Lamé constants are defined as

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)} \quad \text{and} \quad \mu = \frac{E}{2(1+\nu)} \tag{3.3}$$

Now it is possible to determine the wave propagation velocities in the material that is modelled in the mesh. Substitution of the material parameters $E = 1{,}0 \cdot 10^5 \left[{}^{N}/_{m^2}\right]$, ν = *0,48* and $\rho = 1000 \left[{}^{kg}/_{m^3}\right]$ in eq. (3.3) and substituting this in eqs. (3.1) and (3.2) leads to the velocities $c_l = 29{,}637 \left[{}^{m}/_{s}\right]$ and $c_t = 5{,}8124 \left[{}^{m}/_{s}\right]$.

To determine the shapes of the wave fronts in the model used in this report a simplified geometry is considered first. This geometry consists of a piece of a flat surface of infinite size that contains one node. It is plotted in Fig 3.6.

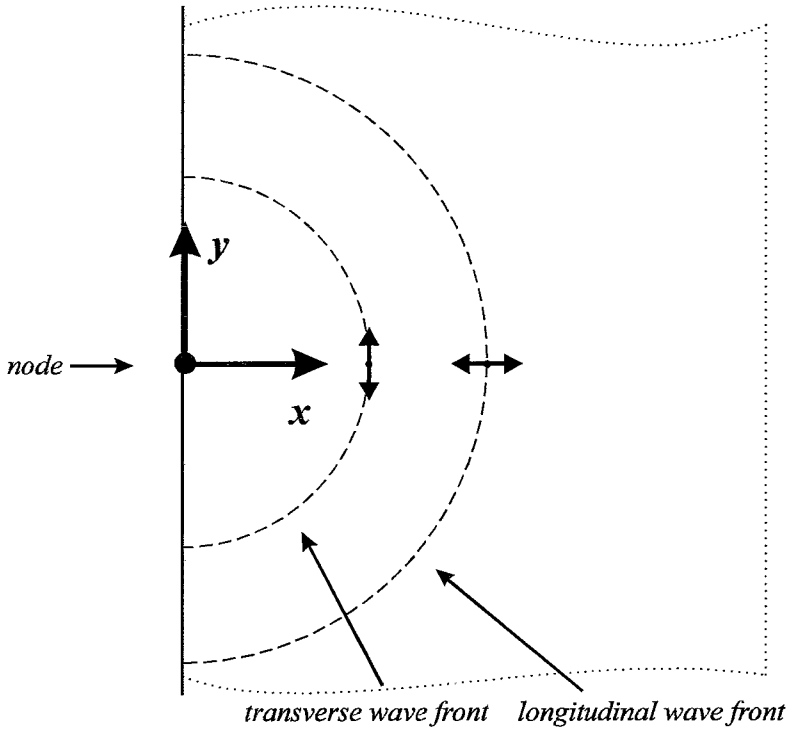


Figure 3.6: The wave propagation in a flat surface caused by one node.

The assumption is made that motion is only possible in the plane defined by the x- and the y-axis (plane strain theory). The waves then also propagate in this plane only. If the node that is situated at the border of this surface is excited two waves will propagate in radial direction away from that node. The direction of the propagation is the same for both wave types. As seen before the velocity of the longitudinal wave is higher than the velocity of the transverse wave. Therefore two wave fronts will appear in this geometry.
The model used in this report has the shape of a flat circle that is excited at certain points at its border. The wave propagation in the model is caused by several nodes at the border of the model and can be approximated by super-positioning the shapes of the waves in the flat surface with one node. This is illustrated in Fig 3.7.

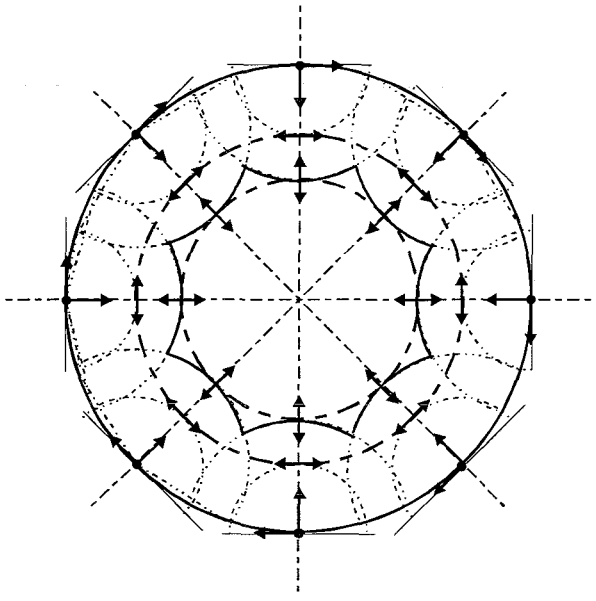


Figure 3.7: Approximation of the geometry of the model by super positioning of the shape of the wave fronts caused by each node.

If there are any waves they should propagate in the plane defined by the x-axis and the y-axis because of the assumed plane strain theory. The nodes move as being fixed on a rigid, circular outer geometry. This means that every node will be excited with the same radial displacement at the same moment in time. As can be seen in Fig. 3.7, interference between the waves originating from different nodes occurs as a result of the spatial discretization used. Here a discretization of only eight nodes is used but in the real numerical model the number of nodes is much larger and therefore more interference will occur. This interference is of no importance because in this report the wave fronts are used only to determine the wave propagation properties. The wave fronts originating from each node individually will be called nodal wave fronts from now on. The shape of the wave fronts in the model is determined by the shape of the nodal wave fronts. By connecting the nodal wave fronts of one type of wave, the wave front for the entire wave propagating from all the nodes at the border of the mesh is obtained. This is illustrated for the longitudinal wave front in Fig. 3.7 by a bold drawn line. This front is star shaped because only eight nodes are used in the spatial discretization of this figure. If more nodes will be used the shape of the wave front will converge to the circular shape that is to be expected to appear in the model when the border as a whole is excited. In that case it is expected that the wave fronts will start propagating at the border and move towards the centre of the model in radial direction. Because the velocity of one type of wave is the same in every direction the wave fronts will have the shape of a circle. Both type of waves have different velocities. This means that two wave fronts can be seen, one of the longitudinal wave with nodes moving in radial direction and one of the transverse wave with nodes moving

in tangential direction. These idealised wave front shapes will be assumed to be the shapes of the wave fronts that propagate in the model in the rest of this report.
The shape of the wave fronts in the model as derived above yields until the wave fronts of the longitudinal wave reach the centre of the geometry. What happens there after cannot be predicted in this quallitative manner.

3.4.2 Interpretation of experimental results

In the previous section is explained that the wave fronts are assumed to propagate from the border of the model in radial direction towards the centre. As a result of the longitudinal wave the nodes are expected to move in radial direction. Because the longitudinal waves move with a higher velocity than the transverse waves the longitudinal wave front will arrive before the transverse wave front arrives at a certain node. This means that the radial direction will coincide with the y-direction for the considered nodes in Fig. 3.3, until the transverse wave arrives. As a result of the transverse wave the nodes will move in tangential direction, which is the global x-direction at the beginning of the experiment.
The fact that there is wave propagation means also that no motion can be found until at least the longitudinal wave has arrived.
When looking at the displacements in y-direction in Fig 3.5 it can be seen that this displacement stays zero until a certain moment in time. Whether this moment in time is the same as the moment in time when the longitudinal wave arrives at the node considered, is determined in the next chapter. The displacements in x-direction do not have such a clear shape as can be seen in Fig 3.4. The considered node oscillates immediately when the experiment starts except maybe in the third mesh. Also Fig 3.4b shows that the period time of an oscillation in the coarse mesh is about twice as long as the period time in the second mesh. The period time of the third mesh is again about half the period time of the second mesh. This indicates that the period time of the oscillation depends on a mesh property like the distance between the nodes or the distance between the integration points. Because the values of these properties are also approximately divided by two for every mesh refinement. The fact that these period times do depent of a mesh property proves that the observed oscillation cannot be a physical phenomenon.
Summarising; The contour plots of the displacements in the mesh show an irregularity on the contour level zero. A closer look at certain nodes shows that when looking in x-direction an oscillation appears. This could be the result of wave propagation in the mesh. If this is the case then displacements in x-direction are likely to be the result of transverse waves and displacements in y-direction could be the result of longitudinal waves. If there is wave propagation the displacements of a node that is not situated at the border of the mesh should be zero until a wave has arrived. This seems to be true for displacements v in y-direction but not for the displacements u in x-direction. Also it can be seen that the period times of these oscillations are mesh dependent. From this it can be concluded that the oscillation cannot exist in the physical model. The oscillation could be the result of the finite numerical accuracy of the

integration method used. To investigate this oscillation and to see if it has anything to do with numerical inaccuracies, the effects of temporal discretization will be treated in chapter 4. Also the wave propagation velocities in the model will be determined under the assumption of section 3.4.1 to see if the assumed shape of the waves is correct.

4. The effects of temporal discretization

In this chapter an effort will be made to get more information about the wave propagation in the model and especially about the oscillations that occur in the displacements in x-direction. This is done by varying the time step size. The mesh used to investigate the effect of temporal discretization is the mesh that has been refined once. The prescribed angular displacement of the nodes at the border of the mesh is the one described in section 2.3. The simulation time is *20 ms*. First the time steps used will be determined, then the numerical results will be presented and the differences of the nodal displacements as a result of different time step sizes will be discussed. Finally the wave velocities will be determined by combining the numerical results of several nodes under the assumption that the waves propagate as seen in section 3.4.1. This is also a check if this assumption applies for the model used here.

4.1 Determination of time step sizes

To investigate the wave propagation it is necessary to use a small time step size. The criterion used here for the choice of the time step size is taken from van Hoof (1994). This criterion is based on wave propagation theory in which the rise time and duration of the load do not exceed several traversal times of the wave across the body of interest. He states that the maximum allowable time step size to maintain the wave front is bounded by the wave speed in such a way that

$$dt_{max} \leq K \frac{L^e_{min}}{c_{max}} \qquad (4.1)$$

were K is a generic constant depending on the kind of element type and integration scheme used. c_{max} is the maximum wave speed in the system and L^e_{min} is the smallest distance between any two of the nodes that lie in the direction of the wave propagation.
In the model in this report the duration and rise of the load does exceed several traversal times of the wave across the model. Therefore the time step value from eq. (4.1) will be smaller than necessary. This means that the values entered for K and L^e_{min} do not have to be determined exactly. The values entered in eq. (4.1) are,

$$L^e \approx \frac{r}{2 \cdot n_{elem}} = \frac{0.098\,[m]}{16} = 6{,}125 \cdot 10^{-3}\ m$$

$c_{max} = c_l = 29{,}637\ {}^{m}/_{s}$ and
$K = 1{,}0$
The value for K is arbitrarily chosen and L^e is the average distance between two nodes in y-direction. With these values dt will be the time necessary for a longitudinal wave to propagate across the average distance between two nodes. The value for the maximum time step size is then $dt_{max} = 0{,}21\ ms$.

To investigate the influence of the temporal discretization three time step sizes are chosen. The first one, $dt = 0,6\ ms$, lies above the maximum time step from eq. (4.1). If there is any wave propagation it should not be visible with this time step. The second time step $dt = 0,2\ ms$ is just a fraction smaller than dt_{max}. Wave propagation should now be visible. The simulation with the third and smallest time step $dt = 0,1\ ms$ is performed to see if the results of the second run converge to the results of this one.

4.2 Numerical results

As in chapter 3 again the nodes situated on the line between the centre of the mesh and the border of the mesh in y-direction are considered (see Fig 3.3). The effects of taking smaller time steps are illustrated with the same node as in chapter 3.
First the displacements in x-direction are considered.

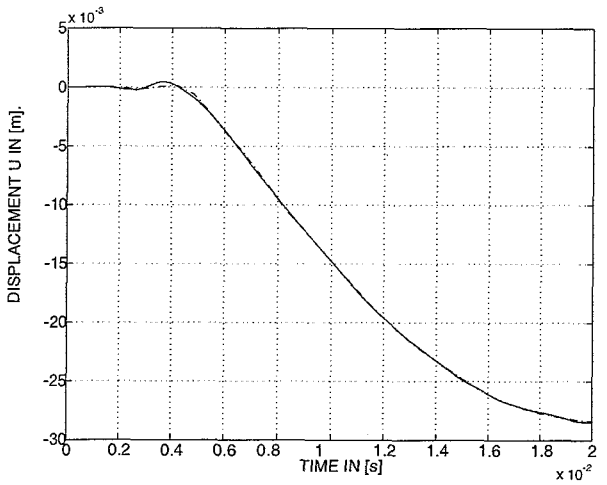


Figure 4.1 **a**: *Displacement u in x-direction*

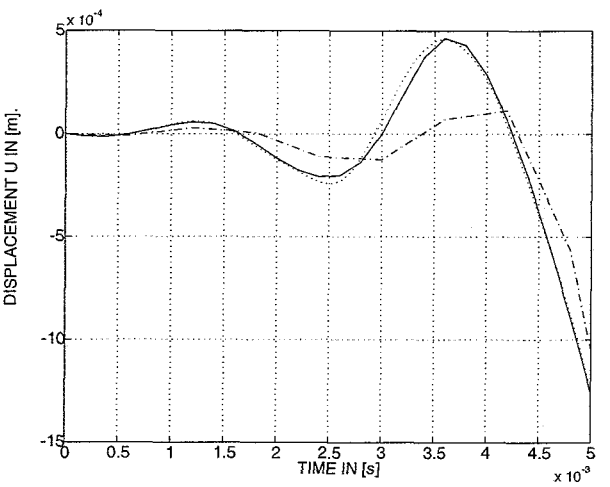


Figure 4.1 **b**: *Displacement u in x-direction close-up*

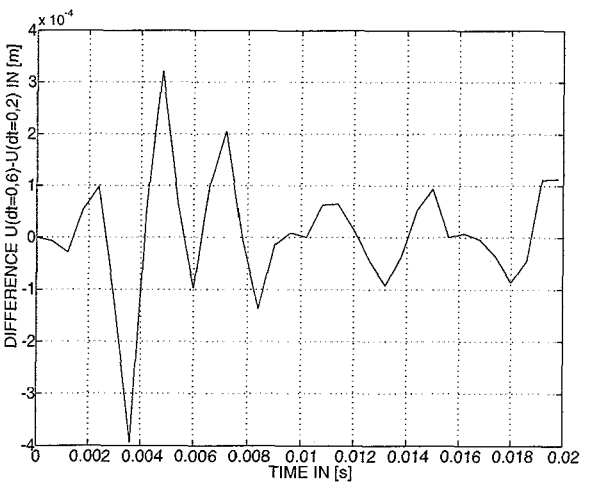


Figure 4.1 **c**: *u(dt = 0,6) - u(dt = 0,2)*

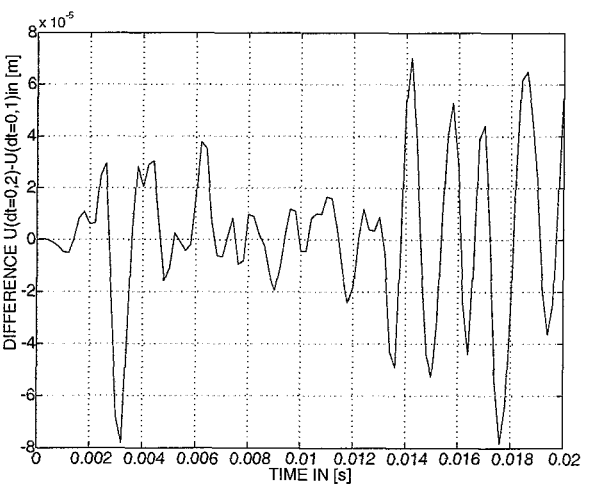


Figure 4.1 **d**: *u(dt = 0,2) - u(dt = 0,1)*

Figure 4.1: ***a*** *and* ***b****: Displacements of one node for dt=0,6(- . -), dt=0,2(—) and dt=0,1 ms (. . .)*
c *and* ***d****: Differences between displacements calculated with different time step sizes, dt.*

From the Figs 4.1a and 4.1b one can conclude that the oscillation disappears after a certain point in time. To investigate this in Fig 4.1c the difference between the displacements u, calculated with $dt = 0,6\ ms$ and with $dt = 0,2\ ms$, is shown. For the time steps when the oscillation is visible in figure a and b, the difference $u(dt = 0,6) - u(dt = 0,2)$ has order $10^{-4}\ m$. After that the difference decreases but still stays of order $10^{-4}\ m$. Differences that did not decrease have also been found at other nodes investigated. The difference is the result of the inaccuracy of the result with the large time step compared to the result with the smaller time step. As can be seen in Fig. 4.1c, the difference for points in time beyond then 8 ms also seems to fluctuate around the value $u = 0$ in a relatively smooth manner. Because the result with the large time step size does not describe the oscillations very well and because in the result with the smaller time step the oscillations are clearly visible it can be concluded that when in the difference of these results a fluctuation is found it will be caused by the oscillation in the result with the smaller time step size. This means that there also has to be an oscillation in the displacement calculated with the small time step after $0,4\ ms$ when the oscillation seems to have disappeared in the figures 4.1 a and b.

The displacements for $dt = 0,2\ ms$ and $dt = 0,1\ ms$, which are shown in Fig 4.1a and 4.1b, do not differ much. The differences $u(dt=0,2) - u(dt=0,1)$ in Fig 4.1d, are of order $10^{-6}\ m$ and have a random shape. This means that the accuracy of the calculated displacements is independent of the oscillations that occur in the calculated displacements. The time step $dt = 0,2\ ms$ then is suitable to describe the oscillations.

The fact that $dt = 0,6\ ms$ doesn't describe the oscillation properly and $dt = 0,2\ ms$ does, indicates that the oscillations behave as if they were the result of wave propagation because $dt_{max} = 0,21\ ms$.

Fig 4.1b shows clearly that the displacement u converges to a certain solution that contains the oscillation when smaller time step sizes are used. This means that the oscillation in the graph is not a result of the finite accuracy of the integration method because then it would be expected that the oscillations would become less. The period times of the oscillations decrease for smaller time steps. This is to be expected because of the used Newmark-β integration method. In the report of van Hoof (1994) is explained that the trapezoidal method ($\gamma = \frac{1}{2}, \beta = \frac{1}{4}$) introduces only period elongation and no amplitude decay. This means while for smaller time steps the result becomes more accurate, the period elongation as a result of the integration method becomes smaller.

Next the displacements in y-direction are considered. These are plotted in Fig 4.2.

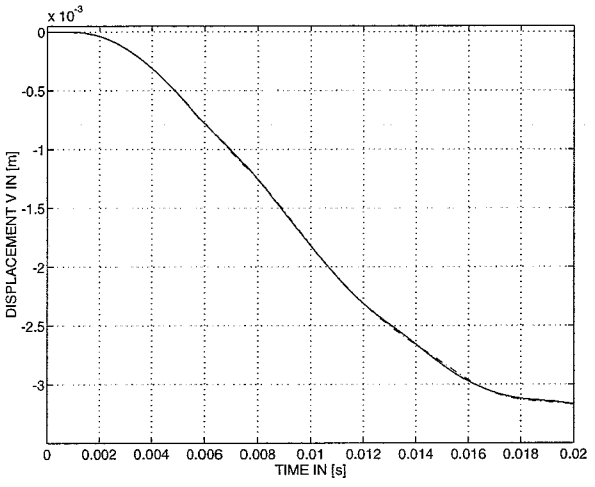


Figure 4.2**a**

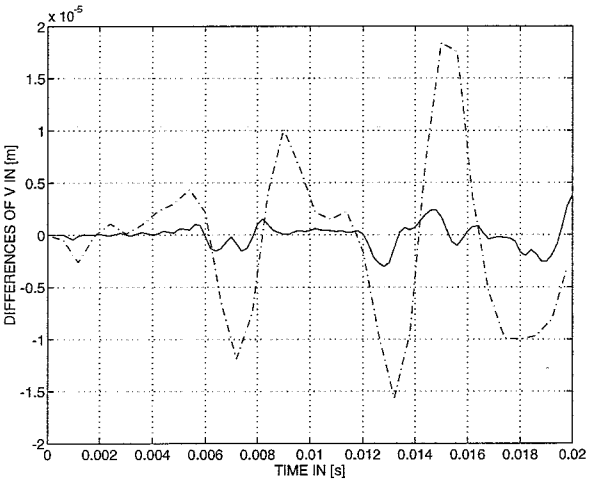


Figure 4.2**b**

*Figure 4.2**a**: Displacements v in y-direction for dt = 0,1 (. . .), dt = 0,2 (—) and dt = 0,6 ms (- . -)*
***b**: Differences [v(dt = 0,6 ms) - v(dt = 0,2 ms)] (- . -) and [v(dt = 0,2 ms) - v(dt = 0,1 ms)] (—)*

Fig 4.2a shows the displacements v in y-direction calculated with different time step sizes. Almost no differences between the displacements can be seen from this figure. That is why in Fig 4.2b the differences between the displacements is plotted. This figure shows that, as a result of taking $dt = 0,2$ instead of $dt = 0,6$, the displacements differ in a constant range of 10^{-5} m. The difference between the displacements calculated with $dt = 0,2$ ms and those calculated with $dt = 0,1$ ms is of order 10^{-6}. There is no influence visible of the oscillations that occur in the displacements u.

Resuming, one can say that the oscillations in the displacement u of the nodes is not the result of the finite numerical accuracy of the integration method. The oscillations start right after the experiment starts and do not seem to stop in the period of time simulated and can therefore not be the result of wave propagation in the actual model. The fact that these oscillations can only be properly described when $dt \leq dt_{max}$ indicates that they behave like being the result of wave propagation.

The displacements v do not oscillate and the improvements as a result of taking a smaller time step size are of constant order for the simulated period in time which was to be expected.

An exception of the behaviour of the displacements u and v as mentioned above are the displacements of the nodes at the border of the mesh. Neither the displacements u or v of these nodes show oscillations.

Until now the displacements in both x- and y-directions have been considered for one node.

In section 4.3 the results of several nodes are combined to make an effort to determine the wave velocities occurring during the simulation in the model.

4.3 Determination of wave velocities out of the numerical results

In this section an effort will be made to determine the velocities of wave propagation in the model. This will be done with the prescribed displacement described in chapter 2 with time step *dt=0,2*. The mesh used is the one that has been refined once. The nodes considered here are located on the line between the centre and the border of the mesh parallel with the *y*-axis and are displayed in Fig 4.3.

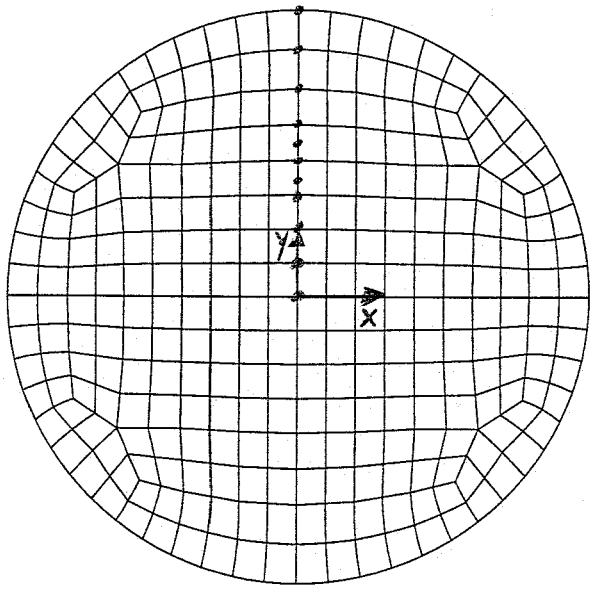


Figure 4.3: The nodes used to determine the velocity of wave propagation.

The basic thought of the approach used to determine the wave velocities is that a node cannot have any displacements until a wave has arrived there. This means that when a certain node starts to move, a wave has arrived at that node. If it is known where the wave propagation starts, it is possible to determine the distance the wave has travelled. This distance divided by the time necessary for the wave to arrive at that node, is the average wave propagation velocity with which the wave propagates until it reaches that node.

To apply the approach mentioned above the shape of the wave fronts and the way they travel through the model has to be known. That is why the assumption is made that the waves propagate as explained in section 3.4.1.

The determination of the wave velocities in the model is the same procedure for both types of wave. The waves move in radial direction from the border towards the centre of the model. This means that the distance the wave has to travel through the material to a certain node equals the radius minus the distance between that node and the centre. By dividing this distance by the time necessary for the wave to propagate towards the node, the average wave propagation velocity can be calculated. Written as a formula this becomes

$$\bar{c} = \frac{r - \sqrt{x_0^{\,2} + y_0^{\,2}}}{t_{start}} \qquad (4.2)$$

in which $\bar{c}$ is the average velocity, r represents the radius, x_0 and y_0 are the initial coordinates of the node considered on $t = 0\ ms$, and t_{start} is the point in time when the observed node starts to move.

As can be concluded from the second part of section 3.4.1, if one wants to determine the average velocity of the longitudinal wave, t_{start} has to be the moment in time when a displacement in y-direction starts at the observed node. The average velocity of the transverse wave can be determined by defining t_{start} to be the moment in time when displacements in x-direction occur at the node of interest.

First the velocity of the longitudinal wave is determined for each considered node. This is done by determining t_{start} from plots in which the displacements in y-direction versus the time for each node considered are printed. A difficulty that had to be solved is that there is no clear point in time when the displacement starts to differ from zero. This is why t_{start} is chosen as the moment in time when the absolute value of the displacement reaches $5 \cdot 10^{-7} m$. The velocities then were calculated using eq. (4.2). The calculated average velocity $\bar{c}$ for the nodes considered and the theoretical longitudinal wave propagation velocity c_l are plotted in Fig 4.4a versus their y-coordinate.

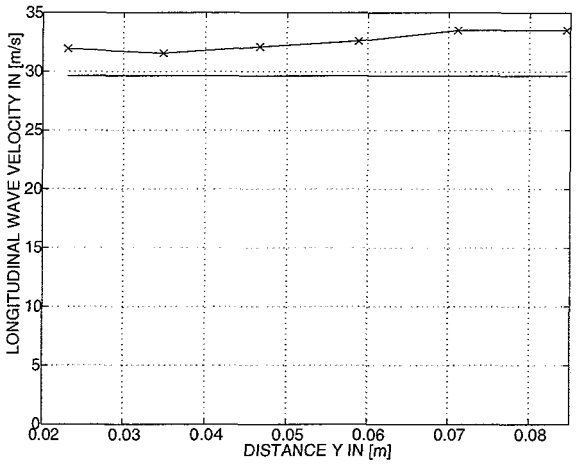


Figure **a**

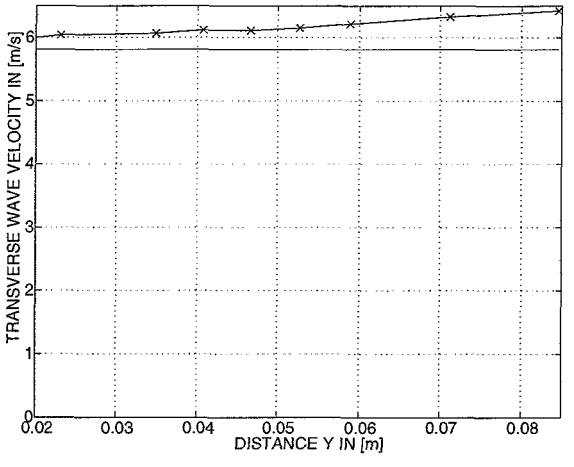


Figure **b**

Figure 4.4: ***a**: Calculated longitudinal wave velocity (-X-) and theoretical value (——)*
***b**: Calculated transverse wave velocity (-X-) and theoretical value (——)*

As can be seen the graphs do not start at $y = 0.0\ m$ because the displacements of the node in the centre of the mesh are that small that no reliable velocity could be determined.

The calculated values for the velocity of the longitudinal wave are all higher than the theoretical value of $29{,}637\ ^m/_s$. The average difference between the calculated values and the theoretical value for the velocity is $9{,}8\ \%$.

To calculate the velocity of the transverse wave propagation t_{start} is to be determined from the moment in time when the node considered gets a displacement in x-direction that doesn't equal zero. This is when the oscillation in the displacement u starts. The oscillation starts at

$t_{start} = 0\ ms$ as can be seen in Fig 4.1b of section 4.2. This is theoretically impossible so the oscillation cannot be the result of wave propagation as is discussed in chapter 3.
The approach to determine t_{start} for the transverse wave, used in this report, is based on the reasoning that in theory there cannot be any displacement at a node until a wave has arrived at that node. As mentioned before the oscillations are not the result of wave propagation so it is assumed that the 'real' displacement equals zero during the time that the oscillation occurs. When the transverse wave arrives at a node the displacement in x-direction is expected to get a negative value because of the prescribed counter-clock wise angular displacement at the border of the mesh. The intersection of the line $u = 0$ and the line of the displacements when the last oscillation is over, is assumed to be the arrival time of the transverse wave at the node. This is illustrated in Fig 4.5. The velocities calculated in this way are shown in fig 4.4b.

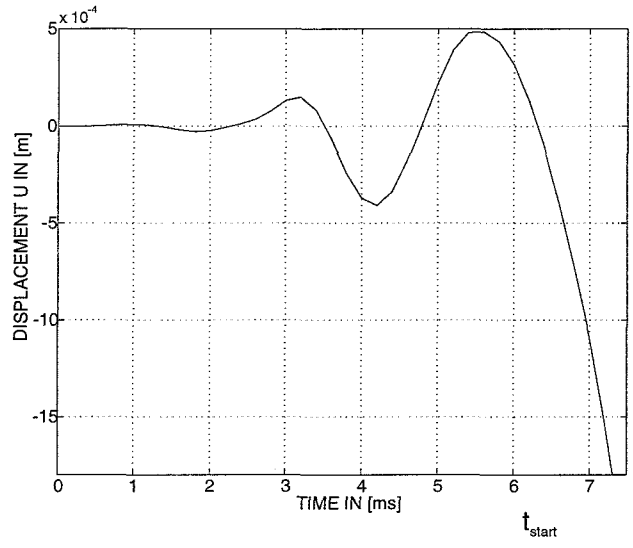


Figure 4.5: Displacement in x direction of the fourth node considered, counted from the border of the mesh and the displacement used to find t_{start} from this data.

The calculated velocities of transverse wave propagation are higher than the theoretical value of $5{,}8124\ m/s$. The average difference between the calculated and the theoretical velocities is $6{,}3\ \%$.
An explanation for the higher values found for the wave velocities for both the longitudinal and transverse wave is given by van Hoof (1994). He states that the velocity in the numerical results is always greater than the actual velocity when a consistent mass matrix is used. In section 2.4 it is noticed that the integration method used for this numerical experiment, makes use of a consistent mass matrix. An other explanation could be that it can be concluded from eq. (3.1) and (3.2) that the wave velocities are proportional to $\sqrt{E}$, which is the square root of the stiffness. When a mesh is modelled to coarse the numerical stiffness of the mesh will be too large and thus the wave velocities will be overestimated [Hoof, van (1994)]. This effect can be seen in the previous chapter in Fig. 3.4b.
From the results in this section it can be concluded that there is strong evidence that wave propagation in the model occurs as assumed but the oscillations in the displacement in x-

direction are definitely not the result of wave propagation. The waves involved are longitudinal waves and transverse waves. At first they both propagate in radial direction from the border of the mesh towards the centre. What happens after that has not been investigated. The longitudinal waves travel faster than the transverse waves. The velocities of wave propagation found are about *10 %* greater than the theoretical value for the longitudinal waves and about *6 %* larger for the transverse waves.

5. Conclusions and discussion

In this report a two-dimensional plane strain Finite Element Model of a simplified physical model of the mid-sagittal cross-section of the human head is considered. In the calculated tangential displacements, as a result of a rotational acceleration, oscillations occur. These oscillations are not present in the prescribed displacement of the nodes at the border of the mesh. To investigate if these oscillations could be the result of wave propagation in the model, the shape of the wave fronts that could propagate through the model had to be assumed and the wave velocities had to be calculated using wave propagation theory. It is explained that because the nodes investigated in this report are lying on the global y-axis the tangential direction can be approximated with the displacement u in global x-direction and the radial displacement can be approximated with the displacement v in global y-direction.
After investigation of the spatial discretization of the model the next observations can be made:

- The period times of the oscillations, occurring in the displacement u, are not the same for different meshes, they seem to be proportional to a property of the mesh, like for example the distance between two nodes or the distance between the integration points.
- The oscillation starts at $t = 0\ ms$, that is at the same time when the nodes at the border of the mesh start their prescribed displacements. From wave propagation theory it can be concluded that this is impossible for nodes not situated directly on the border of the model where the displacement is prescribed.
- When a more refined mesh is used it can be seen that the amplitude of the oscillation decreases but there is still an oscillation present.

From these observations it can be concluded that the oscillation in the radial direction cannot be a physical phenomenon that can occur in the real physical model.
To investigate if the oscillation could be a result of the numerical inaccuracy of the Newmark-β integration method used, the influence of temporal discretization on the calculated displacements has been investigated. The maximum allowable time step size with which it is possible to maintain the wave front is determined. The three time steps used to investigate the numerical accuracy of the integration method are chosen in such a way that the largest time step size is larger than the maximum allowable time step. The two other time step sizes are smaller than this maximum allowable time step.
When comparing the displacements per node calculated with the three time steps one can see the following results,

- The oscillation in the results with the largest time step is hardly visible. When using the second time step size that is smaller than the maximum allowable time step, the oscillation occurs clearly. This indicates that the oscillation is a result of wave propagation.

- It is also seen that the oscillation starts when the simulation starts and doesn't disappear after a certain moment in time although it seems so when looking at the displacements u
- The curves for the three time step sizes in decreasing order, converge to a solution in which the oscillation is present. From this it can be concluded that the oscillation cannot be the result of the finite numerical accuracy of the Newmark-β integration method used, because this method becomes more accurate when smaller time step sizes are used.
- The calculated period times of the oscillation get shorter when calculated with smaller time step sizes. This is because the period elongation introduced by the integration method becomes smaller as a result of the increasing accuracy.

A property of the oscillation that occurs for every time step size is that the period times which occur are not of equal size. The period times seem to become longer when the simulation time proceeds. This indicates that that oscillation is not an eigenmode of the mesh used because then the frequency in the oscillation had to be an eigenfrequency occurring in the mesh. Also a growth of the amplitude of the oscillation can be seen for every time step size.
These two properties cannot be explained and need further investigation.

The radial displacements which coincide with the displacements v in y direction as is explained in section 3.4.1 do not show any oscillation at all. The changes in calculated displacements as a result of mesh refinement behave as can be expected. The improvement of the accuracy by using the mesh that was refined once instead of the coarse one is about 10^{-6} m. The improvement as a result of the second mesh refinement is about 10^{-7} m. This indicates that no mesh refinement is necessary.
The changes as a result of a smaller time step size are also as can be expected. The difference between the results of the smallest time step sizes are about 10^{-6} m. The difference between the results of the largest and the second time step are about 10^{-5} m.

To check if the assumed shape of the wave fronts is correct the wave propagation velocities of the transverse and the longitudinal wave propagation have been determined from the calculated displacements. The calculated values for the speed of the longitudinal wave are determined from the radial nodal displacements and are about *9,8 %* higher then the theoretical value. The transverse wave velocities are calculated from the tangential nodal displacements, with the assumption that when the oscillation occurs the 'real' displacement equals zero, and are about *6,3 %* higher than the theoretical value.
The first two explanations applying for the differences of both calculated velocities are taken from van Hoof (1994). They are:

- The numerical values for the velocities of wave propagation are always higher than the actual velocity when a consistent mass matrix is used. Also the use of an explicit integration method will have the same effect. Fortunately the used Newmark-β integration method is an implicit method and should compensate for the influence of the consistent mass matrix used.

- The wave velocities are proportional to $\sqrt{E}$, which is the square root of the stiffness. When a mesh is modelled to coarse the numerical stiffness of the mesh will be too large and as a result of eq. (3.1) and eq. (3.2) the wave velocities will be overestimated.
- Also the moment in time when a wave reaches the considered node is rather arbitrarily chosen.

Concluding one can say that waves seem to propagate in the model as assumed. This means that both longitudinal and transverse waves propagate in radial direction from the border of the mesh towards the centre with velocities that can be determined within the 10 % range. What happens after the waves reach the centre of the model is not investigated.

The accuracy of the calculated wave propagation velocities in the model can be improved by using mesh refinement. A drawback of this method is that numerical expenses will increase.

The use of smaller time steps doesn't seem to give any obvious improved results but is always recommended because of the higher accuracy of the Newmark-β integration method.

Although the calculated wave propagation velocities in this model seem to correspond with the assumed shape and travel-direction of the wave fronts as shown in section 3.4.1 nothing really proves that this assumption is right. When longitudinal waves depart from nodes not lying on the radial axis considered, they could cause displacements at the observed nodes that will be explained as being a result of a transverse wave departing from the outer node on the radial line considered. This could be further investigated by considering for example the plane strain beam. The mesh of this model should contain the same elements with the same element properties as used in the model in this report. The nodes of one side of the beam have to follow a prescribed displacement parallel to this side in lets call it the y-direction. Theoretically only a transverse wave should be propagating in the mesh in x-direction. A second experiment could be to prescribe the same tangential displacement as used in this report for the nodes at the left edge of the beam and look if there are any agreements with the wave propagation found here.

Because the model used has a circular shape and its boundary is tangentially exited around the centre of the model, it would be easy if the displacements could be presented in cylinder coordinates. Especially when using the assumed shapes of the wave fronts. This advantage disappears when the model will be rotated around an excentrical positioned axis of rotation which will be necessary in future projects because it describes the movement of the head in a more realistic way.

Until now no explanation for the oscillation occurring in the tangential displacements in the model is found. A possible explanation could be found in the report of van Hoof (1994). In that report is shown that in the calculated values of a strain also an oscillation occurs before the actual wave front. This oscillation disappears when mesh refinement is used. There are several differences with the model in this report. First van Hoof uses a mesh with elements of equal size with a constant angle of propagation. A result of that is that the numerical

dispersion[1] as a result of spatial discretization is minimal. Second the elements used by van Hoof are linear axi-symmetric quadrilaterals while the elements used here are plane strain isoparametric distorted quadrilateral elements. And third the model of van Hoof is exposed to a force on one edge during a very short period of time. This will result in a one-dimensional wave while the loading applied here takes much longer and introduces longitudinal and transverse waves. In spite of these differences a mesh refinement is something that can be very useful to investigate especially because the amplitude of the oscillation decreases when a more refined mesh is used.

There is no obvious explanation for the oscillation that occurs in tangential direction during the simulation described in this report. One of the reasons why no explanation was not found is the absence of data from physical experiments. It is strongly advised to model future numerical models closer to a physical model of which data is available so that comparison with experimental results is possible.

[1] Numerical dispersion is when not every mode (or frequency component) of a wave propagates with the same velocity as a result of temporal or spatial discretization.

6. References

[Hoof, van (1994)]
Hoof, J.F.A.M. van : *One- and Two-Dimensional Wave Propagation in Solids.* graduate report WFW 94-055; Department of Fundamental Mechanics; Eindhoven University of technology ; Eindhoven ; 1994.

[Kessels and Peerlings (1993)]
Kessels, P.H.L. and R.H.J. Peerlings : *Een eenvoudig vlakke-rekmodel van het hoofd.* Report WFW 93.179; Department of Fundamental Mechanics; Eindhoven University of technology ; Eindhoven ; 1993.

[Lee (1990)]
Lee, E.-S. : *A large-strain, transient-dynamic analysis of head injury problems by the finite element method.* PhD. Dissertation; Georgia Institute of Technology; 1990.

[Pape (1994)]
Pape K. : *Kinematics of Cranial Contents under Inertial Loading: Analysis of experiments with physical models of parasagittal cranial cross-sections.* Report R030 ; Department of Injury Prevention ; Chalmers university of technology ; Göteborg; Sweden ; 1994.

[Ruan (1991)]
Ruan, J.S., T. Khalil and A. I. King : *Human head dynamic response to side impact by finite element modelling.* J. Biomechanical Engineering; Vol. 20; No. 9; 1987; pp. 276-283.

[Sauren and Claessens (1993)]
Sauren, A.A.H.J. and M.H.A. Claessens : *Finite Element Modelling of Head Impact: The second Decade.* Proceedings of Int. IRCOBI Conf. on the Biomech. of Impacts ; 1993 ; pp. 241-254.

Appendix A: A user subroutine for describing a rotation around an non-central positioned rotation axis

The F.E.M package MARC offers the possibility of forcing displacements on selected nodes by using the user subroutine *forcdt.f*. This routine will be called on every increment during a transient analysis for each selected node. It then calculates the incremental displacements of that node.
The user subroutine used in this report is based on a user subroutine written by Bas Michielsen. It offers the possibility to prescribe angular displacements on selected nodes in a 2-dimensional geometry defined in the FEM program MARC. The angular displacement in the user subroutine is prescribed for every node in such way that the selected nodes rotate as if they were part of a rigid body that rotates around an axis that is not positioned in the centre of the geometry. In this section first a mathematical explanation about how the program works will be given and then a listing of the FORTRAN code will be shown.

A.1 Mathematical explanation of the user subroutine

In figure A.1 the geometry of the model is represented by a circle. The nodes with prescribed displacements in this report are also positioned on a circular border but that is not necessary for this routine to work.

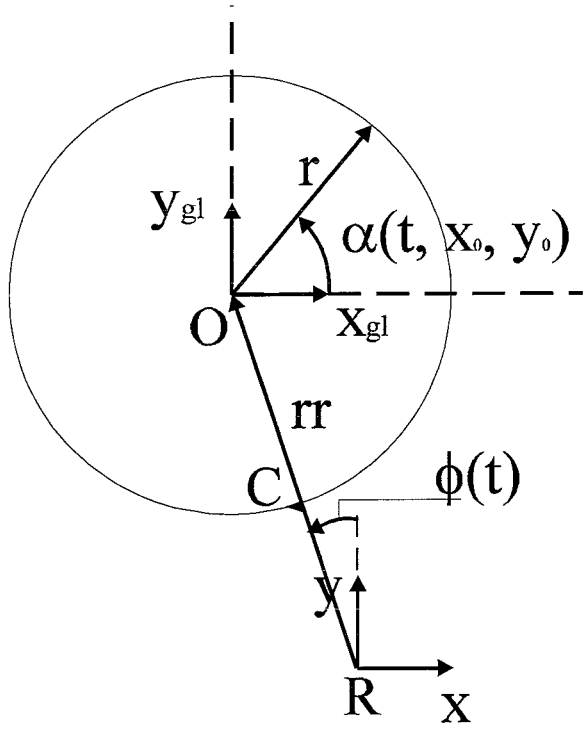


Figure A.1: Definition of the coordinate systems and angles used in the user subroutine.

The model rotates around point R with a prescribed angle $\varphi(t)$. This point lies always underneath the point O at the start of a simulation while $\varphi(t = 0) = 0$ *rad.* The point O represents the centroid of the geometry and coincides with the origin of the coordinate system used by MARC. The position of the origin of this coordinate system, denoted with x_{gl}-y_{gl}, is now attached to the model. The direction of the coordinate axes stays the same during a simulation and coincides with the direction of the coordinate system x-y that has origin R. The distance between O and R is the variable *rr*. The value of this variable is user defined of it has value *0* the prescribed rotation will be around O.
The joint between the line R-C and the model is defined to be rigid this means that the model rotates around O with angle $\varphi(t)$.
The user subroutine calculates incremental displacements. This means that the absolute coordinates are not of interest. Because the directions of respectively the x and the y axis in both coordinate systems are the same it is allowed to use the values of the displacements calculated in the user defined coordinate system, x-y, and use them in the coordinate system x_{gl}-y_{gl} used by MARC.
The way in which the user subroutine uses this information to calculate the displacements in x and y direction is explained next.
The routine is called for every node with prescribed displacements for every time increment First the routine calculates the difference between the node considered, and origin O of the coordinate system used by MARC. Because in this case the nodes with prescribed displacements are positioned in a circle around O this difference represents the radius of this circle. Radius r can be calculated as follows from the coordinates used by MARC at $t = 0$,

$$r = \sqrt{x_0^{\,2} + y_0^{\,2}} \qquad \text{(A.1)}$$

Then the angle $\alpha_0(x_0, y_0)$ is calculated. This is the angle between the positive x-axis of the coordinate system x_{gl}-y_{gl} and the line between O and the node considered, on $t = 0$. It is independent of the moment in time.

$$\alpha_0(x_0, y_0) = arctan\left(\frac{y_0}{x_0}\right) \qquad \text{(A.2)}$$

The prescribed angle $\varphi(t)$ is fully determined by the time and independent of the coordinates of the selected node. It can be inserted in the user subroutine as a polynomial of degree n and will be calculated for the considered moment in time.

$$\varphi(t) = c_0 + c_1 \cdot t + c_2 \cdot t^2 + \ldots + c_n \cdot t^n \qquad \text{(A.3)}$$

The polynomial coefficients can have any real value. There is one exception, c_0 has to be equal to zero because on $t = 0$ the value of angle $\varphi(t)$ has to equal zero. In the user subroutine the coefficients of this polynomial are read from a file called *rot.dat*.
From eq.(A.2) and eq.(A.3) angle $\alpha(t, x_0, y_0)$ can be determined using:

$$\alpha(t, x_0, y_0) = \alpha_0(x_0, y_0) + \varphi(t) \tag{A.4}$$

This is the angle of an observed node with original coordinates (x_0 , y_0) at a moment in time t, with the positive global x-axis of the global coordinate system used by MARC as shown if Fig. A.1.
After this the user subroutine calculates the coordinates relative to the coordinate system x-y for every specified node at a moment in time. For these coordinates yields

$$\begin{aligned} x(t) &= r \cdot cos(\alpha(t, x_0, y_0)) - rr \cdot sin(\varphi(t)) \\ y(t) &= r \cdot sin(\alpha(t, x_0, y_0)) + rr \cdot cos(\varphi(t)) \end{aligned} \tag{A.5}$$

The incremental displacements that MARC uses then are calculated in both x and y direction using

$$\begin{aligned} du &= x(t + dt) - x(t) \\ dv &= y(t + dt) - y(t) \end{aligned} \tag{A.6}$$

and returned to the mean program.

A.2 Listing of the user subroutine

This section contains a listing of the user subroutine. The comment typed with **bold** characters is additional comment and refers to the equations used in section A.1. The *italic* characters represent the basic changes that have been performed to the original program necessary to implement the rotation around a non central positioned rotation-axis.

```
c* * *dbrot.f. This program will allow you to prescribe an excentrical
c    rotation on a body which will behave as a rigid body.
c    needed: input file containing polynomial coefficients for rotation angle
c          rot.dat
c    N.B. in deze versie wordt ervan uit gegaan dat het draaipunt op t=0
c    op de y-as beneden het middelpunt van het model ligt (y=0,x=-...)
c* * * * * *
     subroutine forcdt(u,v,a,dp,du,time,dtime,ndeg,node,
     1ug,xord,ncrd,iacflg,inc,ipass)
c* * * * * *
c    discribes a polynomical translation and rotation.
c    changed version.
c
c    input of time dependent forcing functions and boundary
c    conditions.
c
c    u         total displacements at a node
c    v         velocity at a node
c    a         acceleration at a node
c    dp         load increments at a node
c    du         displacement increments at a node
c    time       time
c    dtime      time increment
c    ndeg       number of degrees of freedom per node
c    node       node number
c    ug         total displacements at node in global system
c    xord       original coordinates
c    ncrd       number of coordinates
c    iacflg     acceleration flag - set to 1 if accelerations given
c    inc        increment number
c
c* * * * * *
     implicit real*8 (a-h,o-z)                                    dp
     dimension u(ndeg),v(ndeg),a(ndeg),dp(ndeg),du(ndeg)
     dimension ug(1),xord(ndeg), uittrans(2), uitdraai(2)
c
c
     open (unit=42, file='node3.dat')
c definitie standaardvariabele
     pi=4*atan(1.00)
c
c
```

```
ccccccccccccccccccc
c
      r=sqrt(xord(1)*xord(1)+xord(2)*xord(2)+xord(3)*xord(3)) c Eq. A.1
c
      call rotatie(du,time,dtime,ndeg,xord,inc,node,uitdraai)
c
      du(1)=uitdraai(1)                    c du from Eq. A.6
      du(2)=uitdraai(2)                    c dv from Eq. A.6
c
      return
      end



      subroutine rotatie(du,time,dtime,ndeg,xord,inc,node,
     1uitdraai)
      implicit real*8 (a-h,o-z)                          dp
      dimension du(ndeg), xord(ndeg), uitdraai(2)
c
c  definitie standaardvariabelen r en alfa0
c  r = f(xord(1,2,3) en alfa0 = f(xord(1,2))
c
      pi=4*atan(1.00)
      r=sqrt(xord(1)*xord(1)+xord(2)*xord(2)+xord(3)*xord(3))
c                                                  Eq. A.1
c  definitie van beginhoek van knooppunt (alfa0), ofwel de hoek v.d.
c  ligging v.h. knooppunt.                         Eq. A.2
c  1e kwadrant
      if ((xord(1).gt.0).and.(xord(2).ge.0)) then
        alfa0=atan(xord(2)/xord(1))
      end if
c  2e kwadrant
      if ((xord(1).le.0).and.(xord(2).ge.0)) then
        alfa0=pi-atan(abs(xord(2)/xord(1)))
      end if
c  3e kwadrant
      if ((xord(1).lt.0).and.(xord(2).lt.0)) then
        alfa0=pi+atan(xord(2)/xord(1))
      end if
c  4e kwadrant
      if ((xord(1).ge.0).and.(xord(2).lt.0)) then
        alfa0=2*pi-atan(abs(xord(2)/xord(1)))
      end if
c
c  berekening van huidige hoek en hoek in vorige increment
c  phi is de hoek van het hele model, alfa die v.e. knooppunt.
c
      phi2=hoek(time+dtime)      c φ(t + dt)  Eq.A.3
      phi1=hoek(time)            c φ(tt)      Eq.A.3
      alfa2=phi2+alfa0           c α(t + dt)            Eq. A.4
      alfa1=phi1+alfa0           c α(t)       Eq. A.4

c
c  berekening van de benodigde verplaatsingen
c  .....!! rr ZELF INVULLEN !!!!!.....
      rr = 0.05
      xeen=r*cos(alfa1)-rr*sin(phi1)       c x(t+dt)  Eq. A.5
      xtwee=r*cos(alfa2)-rr*sin(phi2)      c x(t)     Eq. A.5
      yeen=r*sin(alfa1)+rr*cos(phi1)       c y(t+dt)  Eq. A.5
      ytwee=r*sin(alfa2)+rr*cos(phi2)      c y(t)     Eq. A.5
      uitdraai(1)=xtwee-xeen               c du       Eq. A.6
      uitdraai(2)=ytwee-yeen               c dv       Eq. A.6
c
      if (node.eq.3) then                            c          output
of some
        write(42,1000) time+dtime,phi2     c          variables
      end if
1000  format(2e13.5)
c
      return
      end

ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c ***** COEFF(graad+1) ZELF INVULLEN !!!!! *****

      function hoek(time)          c calculates φ(time)  using Eq.A.4
      implicit real*8 (a-h,o-z)                                  dp
      dimension coeff(10)
      integer graad
      open(unit=8,file='rot.dat',status='old')
c
c ***** GRAAD ZELF AANPASSEN !!!!! *****
c
      graad=9                      c          degree n
      do 12 i=1,graad+1
12    read (unit=8,*) coeff(i)

      hoek=0.00
      do 15 i=1,graad+1
        hoek=hoek+coeff(i)*time**(graad+1-i)
15    continue
      close(8)
      return
      end
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
```

Appendix B: The input file and its user subroutines.

B.1 The MARC input file.

This is an input file for the coarse mesh. It is given to show the options used, and therefore are not all of the nodal coordinates are given. Also the connectivity block is shortened.

```
$ grove mesh 25 incr dt = 0.6 ms 9de graads polyn.
title    ngeom grof
sizing     1000000  72  245  490
elements   27
dynamic     2   0   0   0
all points
dist loads   1  72
setname     2
end
$.....
solver
   0   0   0
optimize    9
connectivity

   1  27  78  72   3   5 116 117   4 118
   2  27   7  77  78   5 119 120 118   6
   3  27  77  73  72  78 121 122 116 120
.
. etc.
.
  71  27 105 106  92  93 237 244 215 245
  72  27  62 105  93  61 240 245 217 185
coordinates
   3 245
   1 9.80000-2 0.00000+0 0.00000+0
   2 9.73856-2 1.09354-2 0.00000+0
.
. etc.
.
  68 1.70304-8 7.11563-2 0.00000+0
.
.
 244-9.88189-3-4.64201-2 0.00000+0
 245-9.86495-3-2.30386-2 0.00000+0
define   node    set     rand_nodes
     1    to    56
isotropic

   1
1.00000+5 4.80000-1 1.00000+3 0.00000+0 0.00000+0
0.00000+0
     1    to    72
fixed disp

0.00000+0 0.00000+0
   1   2
rand_nodes
no print
post
   0  16  17   0   0  19  20   0   1   0   6
udump
 , , ,
forcdt
rand_nodes
end option
control
99999,10,0,0,0,1,0,0,1,0
0.1,0.0,0.0,0.0,0.0
$.....
$....start of loadcase rot.f met 9de graads polynoom 25 incr,
$    dt =0.6 ms
dynamic change
 6.0e-4,1.5e-2,25,0,0,0,0.5,0.25
continue
$....end of loadcase
$.....
```

B.2 The user subroutines.

Two user subroutines are used. The first one is *forcdt.f* to prescribe the incremental displacements for the desired nodes at the border of the mesh. In the MARC input file these are the nodes called *'rand_nodes'*. In this file the original version of B. Michielsen is taken (only some comment is altered and a unused part of the routine to prescribe translations is removed). This version does only allow rotations around the origin of the coordinate system used by MARC. The coefficients of the polynomial are read from a file called *rot.dat*. The second one is *impd.f*. It is called in the main MARC program using the command *udump*. This routine offers the possibility print coordinates, incremental displacements, total displacements etc. in a file. This file can be used in for instance MATLAB to perform some post processing. They are put together in one file.

```
c
c     this file contains both rot.f and impd.f.
c
      subroutine forcdt(u,v,a,dp,du,time,dtime,ndeg,node,
     1ug,xord,ncrd,iacflg,inc,ipass)
c* * * * * *
c     discribes a polynomical translation and rotation.
c     by B. Michielsen.
c
c     input of time dependent forcing functions and boundary
c     conditions.
c
c     u         total displacements at a node
c     v         velocity at a node
c     a         acceleration at a node
c     dp        load increments at a node
c     du        displacement increments at a node
c     time      time
c     dtime     time increment
c     ndeg      number of degrees of freedom per node
c     node      node number
c     ug        total displacements at node in global system
c     xord      original coordinates
c     ncrd      number of coordinates
c     iacflg    acceleration flag - set to 1 if accelerations
given
c     inc       increment number
c
c* * * * * *
      implicit real*8 (a-h,o-z)                                dp
      dimension u(ndeg),v(ndeg),a(ndeg),dp(ndeg),du(ndeg)
      dimension ug(1),xord(ndeg), uittrans(2), uitdraai(2)
c
c
      open (unit=45, file='node15.dat')
c definitie standaardvariabele
      pi=4*atan(1.00)
c
ccccccccccccccccccc
      r=sqrt(xord(1)*xord(1)+xord(2)*xord(2)+xord(3)*xord(3))
c
      call rotatie(du,time,dtime,ndeg,xord,inc,node,uitdraai)
c
      du(1)=uittrans(1)+uitdraai(1)
      du(2)=uittrans(2)+uitdraai(2)
c
      return
      end

      subroutine rotatie(du,time,dtime,ndeg,xord,inc,node,
     1uitdraai)
      implicit real*8 (a-h,o-z)                                dp
      dimension du(ndeg), xord(ndeg), uitdraai(2)
c
c definitie standaardvariabelen
      pi=4*atan(1.00)
      r=sqrt(xord(1)*xord(1)+xord(2)*xord(2)+xord(3)*xord(3))
c
c definitie van beginhoek van knooppunt
c 1e kwadrant
      if ((xord(1).gt.0).and.(xord(2).ge.0)) then
        alfa0=atan(xord(2)/xord(1))
      end if
c 2e kwadrant
      if ((xord(1).le.0).and.(xord(2).ge.0)) then
        alfa0=pi-atan(abs(xord(2)/xord(1)))
      end if
c 3e kwadrant
      if ((xord(1).lt.0).and.(xord(2).lt.0)) then
        alfa0=pi+atan(xord(2)/xord(1))
      end if
c 4e kwadrant
      if ((xord(1).ge.0).and.(xord(2).lt.0)) then
        alfa0=2*pi-atan(abs(xord(2)/xord(1)))
      end if
c
c berekening van huidige hoek en hoek in vorige increment
      alfa2=hoek(time+dtime,inc,node)+alfa0
      alfa1=hoek(time,inc-1,node)+alfa0
c
```

```
c berekening van de benodigde verplaatsingen
      uitdraai(1)=r*cos(alfa2)-r*cos(alfa1)
      uitdraai(2)=r*sin(alfa2)-r*sin(alfa1)
c
      if (node.eq.15) then
        write(45,1000) phi2,alfa2,uitdraai(1),uitdraai(2)
      end if
1000  format(4e16.8)
c
      return
      end

ccccccccccccccccccccccccccccccccccccccccccccccccccccc
ccccccc

      function sgn(x)
      implicit real*8 (a-h,o-z)                              dp
      if (x.eq.0.00) then
        sgn=0.00
      else
        sgn=x/abs(x)
      end if
      return
      end

ccccccccccccccccccccccccccccccccccccccccccccccccccccc
ccccccc
c ***** COEFF(graad+1) ZELF INVULLEN !!!!! *****

      function hoek(time,inc,node)
      implicit real*8 (a-h,o-z)                              dp
      dimension coeff(10)
      integer graad
      open(unit=8,file='rot.dat',status='old')
c
c ***** GRAAD ZELF AANPASSEN !!!!! *****
c
      graad=9
      do 12 i=1,graad+1
12    read (unit=8,*) coeff(i)

      hoek=0.00
      do 15 i=1,graad+1
        hoek=hoek+coeff(i)*time**(graad+1-i)
15    continue
      close(8)
      return
      end
ccccccccccccccccccccccccccccccccccccccccccccccccccccc
cccccccc
      subroutine impd(n,dd,td,xord,f,v,a,ndeg,ncrd)
c* * * * * *
c     user subroutine for output of displacements.
c     n          node number
c     dd         array of incremental displacements of this
node
c     td (ndeg,numnp)       array of total displacements of this
node
c     xord (ncrd,numnp)     array of coordinates of this node
c     ndeg       number of degrees of freedom per node
c
c* * * * * *
      implicit real*8 (a-h,o-z)                              dp
      dimension dd(ndeg),td(ndeg),xord(ncrd)
     1 ,f(ndeg),v(1),a(1)
      open (unit=46, file='marcnod61.dat')
      open (unit=47, file='marcnod149.dat')
      open (unit=48, file='marcnod68.dat')
      open (unit=49, file='marcnod145.dat')
      open (unit=50, file='marcnod151.dat')
      open (unit=51, file='marcnod66.dat')
      open (unit=52, file='marcnod147.dat')
      open (unit=53, file='marcnod15.dat')
      if (n.eq.61) then
        write(46,2000) n, xord(1), xord(2), td(1), td(2), dd(1),
dd(2)
      end if
      if (n.eq.149) then
        write(47,2000) n, xord(1), xord(2), td(1), td(2),dd(1),
dd(2)
      end if
      if (n.eq.68) then
        write(48,2000) n, xord(1), xord(2), td(1), td(2), dd(1),
dd(2)
      end if
      if (n.eq.145) then
        write(49,2000) n, xord(1), xord(2), td(1), td(2), dd(1),
dd(2)
      end if
      if (n.eq.151) then
        write(50,2000) n, xord(1), xord(2), td(1), td(2), dd(1),
dd(2)
      end if
      if (n.eq.66) then
        write(51,2000) n, xord(1), xord(2), td(1), td(2), dd(1),
dd(2)
      end if
      if (n.eq.147) then
        write(52,2000) n, xord(1), xord(2), td(1), td(2), dd(1),
dd(2)
      end if
      if (n.eq.15) then
        write(53,2000) n, xord(1), xord(2), td(1), td(2), dd(1),
dd(2)
      end if
2000  format(i3,6e16.8)
      return
      end
```

The input file *rot.dat* containing the polynomial coefficients (c_0 is the lowest value) of the 9^{th} degree polynomial used here is shown next.

-1.3806248e+011	c	⇒	c_9
7.9010989e+010	c	⇒	c_8
-1.8963205e+010	c	⇒	c_7
2.4677976e+009	c	⇒	c_6
-1.8730325e+008	c	⇒	c_5
8.2175458e+006	c	⇒	c_4
-1.8660178e+005	c	⇒	c_3
1.2406819e+003	c	⇒	c_2
2.2268127e+001	c	⇒	c_1
0.0000000e+000	c	⇒	c_0