Optimization of multibody systems

Citation for published version (APA):

Document status and date:
Published: 01/01/1993

Document Version:
Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:
• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.
Optimization of multibody systems

The SLP-method linked to the Samcef-Mecano multibody software

E.J.R.W. Thijsen

WFW report 93.151
Optimization of multibody systems

The SLP-method linked to the Samcef-Mecano multibody software

Graduation report

October 1993

Author: Edwin Thijsse

Professor: Dick van Campen

Coaches: Bert Schoofs
          Pascal Etman

Eindhoven University of Technology

Department of Mechanical Engineering

Computational and Experimental Mechanics Group
Summary

Optimization is still rather unknown in the field of multibody analysis. For structural analysis in general it is already a technique that can be very useful as a designing tool. One of the optimization techniques commonly used in structural optimization is the Sequential Linear Programming (SLP) method with move limits. In this report a design optimization tool that makes use of the SLP-method is developed for the multibody analysis code Mecano of the Samcef Finite Element software. So, on the one hand this report deals with multibody analysis and on the other hand with the numerical optimization process.

The multibody module Mecano is programmed according to the FE concept, which is a rather unconventional approach for multibody analyses. The FE concept uses a global coordinate frame like the Cartesian coordinate method. A characteristic of the FE concept is that the constraints of the multibody system contribute to the equations of motion in the same way as a separate element. Mecano treats finite rotations by making use of a formalism called the rotation vector. This formalism uses a minimal set of three parameters for describing the finite rotations and also has an easy geometric interpretation. Finally, the equations of motion are solved with the implicit Newmark method with use of the Newton-Raphson iteration procedure.

The SLP-method with move limits is an easy and often used concept in approximate optimization. The solution of the optimization problem is obtained by generating a sequence of limited optimization problems that deal with the original problem by means of linear approximation models. Application of this structural optimization method to time dependent systems leads to some modifications of the optimization problem, because the time dependency requires special treatment of the objective function and the constraints. Five methods for dealing with this time dependent behaviour in optimization methods are tested and compared with each other. It appears that all methods give comparable results with respect to the obtained optimum.

With use of the above described techniques, an optimization program for multibody systems has been developed. This program, called Mecopt, has been tested on three multibody optimization problems and two of them show optimal solutions that are comparable with the literature. The third optimization problem also obtained an optimal solution, but due to the change of the original problem formulation the solution could not be compared with the results from the literature anymore. In general may be concluded that the SLP-method with move limits is useful in the field of multibody systems.
Notation and symbols

Notational conventions:

\[ a \] scalar
\[ A \] scalar
\[ \mathbf{a} \] vector
\[ \mathbf{a} \] column matrix
\[ \mathbf{A} \] column matrix
\[ \mathbf{\dot{A}} \] skew-symmetric matrix of \( \mathbf{A} \)
\[ \mathbf{A} \] matrix
\[ \mathbf{A}^{-1} \] inverse of matrix \( \mathbf{A} \)
\[ \mathbf{A}^T \] transpose of matrix \( \mathbf{A} \)
\[ \mathbf{a}^T \] transpose of column matrix \( \mathbf{a} \)
\[ \dot{a} \] partial time derivative of \( a \)
\[ |\mathbf{a}| \] norm of column matrix \( \mathbf{a} \)
\[ a_n \] the subscript \( n \) indicates that \( a \) is considered at time \( t_n \) (unless indicated otherwise), where \( t_n = n\Delta t \).

List of symbols chapter 2:

- \( R \) rotation operator
- \( \psi \) rotational vector
- \( \Phi \) holonomic constraints
- \( \Sigma \) non-holonomic constraints
- \( \mathbf{q} \) generalized coordinates
- \( \mathbf{M} \) mass matrix
- \( \mathbf{K} \) tangent stiffness matrix
- \( \mathbf{C}^t \) tangent damping matrix
- \( \mathbf{C}^{int} \) internal forces vector
- \( \mathbf{C}^{ext} \) external forces vector
- \( \lambda \) Lagrange multipliers
- \( t \) time
- \( k \) scale factor
- \( p \) penalty factor
List of symbols chapter 3:

- $\mathbf{x}$: design variables
- $x_0$: starting values for the design variables
- $x^l$: lower limits on the design variables
- $x^u$: upper limits on the design variables
- $n$: number of design variables
- $m_l$: value of the move limit
- $\delta x^l, \delta x^u$: the absolute move limits of the design variables
- $F$: objective function
- $g$: constraints
- $n_c$: number of constraints
- $u$: response variables
- $t$: time
- $t_f$: ending time of the analysis
- $n_t$: number of time points in the time interval
## Contents

### Summary

4

### Notation and symbols

5

### Contents

7

1. Introduction

9

2. The multibody package Samcef-Mecano
   2.1. Introduction
   10
   2.2. The Mecano theory
      2.2.1. Representation of finite rotations in three-dimensional kinematics by Mecano
      11
      2.2.2. The Finite Element concept used by Samcef-Mecano
      13
      2.2.3. Modelling of joints
      14
      2.2.4. Time integration of the equations of motion
      16
   2.3. Example
   17
   2.4. Discussion
   20

3. Optimization of multibody systems
   3.1. Introduction
   21
   3.2. The optimization process
      3.2.1. The standard formulation
      21
      3.2.2. The Finite Element Method and optimization
      23
      3.2.3. Sequential Linear Programming with move limits
      23
   3.3. Specific features of multibody optimization
      3.3.1. Constraints in transient analysis
      26
      3.3.2. The objective function in transient analysis
      28
   3.4. The optimization process in Mecopt
   29

4. Experimental problems
   4.1. Introduction
   31
   4.2. Impact-absorber
      4.2.1. Analysis with time point constraints
      33
      4.2.2. Analysis with an integral constraint
      33
      4.2.3. Analysis with a sum constraint
      33
      4.2.4. Analysis with critical time point constraints
      34
      4.2.5. Analysis with special treatment of the objective function
      34
   4.3. Discussion of the various analysis methods
   35
   4.4. Landing gear mechanism
   37
   4.5. Four-bar mechanism
   39
   4.6. Discussion
   41
5. Conclusions and Recommendations

References

Appendix A: Different formalisms to describe finite rotations
Appendix B: Implementation of constraints in the finite element approach
Appendix C: Explicit and implicit time integration schemes
Appendix D: Overview of the Samcef modules
Appendix E: Accuracy of the first-order forward difference scheme
Appendix F: Results of the three optimization problems

  F.1. The impact-absorber with time point constraints
  F.2. The impact-absorber with an integral constraint
  F.3. The impact-absorber with a sum constraint
  F.4. The impact-absorber with critical time point constraints
  F.5. The impact-absorber with special treatment of the objective function
  F.6. The landing gear mechanism
  F.7. The four-bar mechanism
Chapter 1
Introduction

Techniques for analyzing multibody systems have been available from the beginning of the computer revolution. Therefore, a diversity of analysis packages for these systems exist. Especially during the last ten years the number of commercially available multibody analysis packages has significantly increased. However, only few packages have some capabilities to modify and improve the multibody system design. Such design tools have already been implemented in structural analysis packages, for example Samcef, Nastran and Ideas. A design tool consists of an optimization module, which is capable of modifying the system to the requirements and objectives of the user. In terms of multibody systems one can think of optimizing a landing gear to a minimum of forces that occur during landing, or of modelling a flexible robot mechanism to a minimum weight. Due to the increasing speed of computers, the field of optimization has made more progress the last few years. However, this is mainly a fact for structural optimization and not for the optimization of multibody systems.

A favorite technique within structural optimization are the approximation concepts. These concepts are used as the interface between the optimization module and the Finite Element (FE) package that computes the behaviour of the FE system. Normally, the optimization module iterates towards the optimal solution with in between making some evaluations of the FE system. If these evaluations of the FE system are carried out by approximation models instead of the FE package, a reduction of the computational effort can be achieved. A commonly used method in structural optimization is the Sequential Linear Programming (SLP) method with move limits, which is based on linear approximation models. Until now, this approximation method is mainly used in structural analysis and hardly in multibody analysis. The goal of this project is to investigate the possibilities of applying the SLP-method with move limits to multibody systems.

The multibody analysis program that has been used is the module Mecano of the FE package Samcef. Because of the introduction of the program within the department of mechanical engineering first the features of the program were studied. The information gained by self-study and from a two week during instruction at the university of Liège are bundled in a report (Thijssen, 1993a). Due to the fact that Mecano makes use of an unconventional approach for multibody analysis, it is relevant to investigate the theory that is used by Mecano. A summary of this theory is given in chapter 2 of this report. Chapter 3 describes the theory concerning the SLP-method and concentrates on the optimization of multibody systems. This manifests itself in section 3.3 and 3.4 in which the time dependency of multibody analyses is discussed. Section 3.5 describes the global construction of the developed optimization program Mecopt. This program is validated on three examples, which are presented in chapter 4. In that chapter, also the various ways of dealing with the time dependent optimization problem are tested. Finally, chapter 5 reports the conclusions of this study and recommendations for further research.
Chapter 2
The multibody package Samcef-Mecano

2.1. Introduction

Multibody systems are mechanical models that are compound of a finite number of coupled bodies (Sauren, 1990). The individual bodies are called links and the connection between two links is called a joint. These joints impose constraints on the system. The links can be rigid elements or flexible elements, like a spring, damper or a beam.

Multibody systems are characterized by the large displacements and rotations that occur during motion. This behaviour introduces geometric non-linearities to the system, which are difficult to deal with. Theories for systematically describing the behaviour of multibody systems were introduced in the late fifties. Lack of these theories was that they could not be solved analytically. Just in the sixties, when the computer hardware developed a lot, numerical approaches could be used to obtain a solution of the equations of motion of non-linear systems. Nowadays, many software packages are available that are able to simulate the behaviour of multibody systems. In spite of this still no general agreement exists on the most appropriate numerical approach to treat multibody systems.

The module Mecano of the finite element package Samcef can deal with multibody systems. The development of Samcef started in 1965 at the university of Liège, but nowadays it is developed, maintained and marketed by the company SAMTECH S.A. The multibody module Mecano was commercialized in 1989.

2.2. The Mecano theory

Mecano is based on the theory that is summarized in this section. The theory is comprehensively described in (Cardona, 1989), (Cardona et al., 1991), (Gérardin, 1990) and (Samtech, 1991).

Most of the existing multibody programs make use of one of the following main systems of coordinates:
- The method of Lagrangian (relative) coordinates. This method describes the position of a body relative to the position of a previously described body. This approach has the merit of keeping to a minimum number of generalized coordinates and constraints, and gives direct access to the displacements at the joints. However, it suffers from drawbacks such as complex topology description and high degree of non-linearity.
• The method of *Cartesian (global) coordinates*. This method describes the position of the separate bodies with regard to the global reference frame. This approach has specific attractive features such as generality, lower degree of non-linearity than the previous approach and easy topological description. Coupling is simply expressed as a set of constraints at joints. As a result, the number of kinematic unknowns is significantly higher than with the Lagrangian coordinate approach.

Mecano makes use of an approach called the *Finite Element method*. This approach may be regarded as a particular case of the method of Cartesian coordinates. An essential difference is the way in which the kinematics of flexible motion is described. When dealing with flexible bodies in the standard Cartesian approach, then the global motion is decomposed into a rigid-body motion on which a small deformation is superimposed. The finite element approach used by Mecano directly introduces the flexibility effects by the beam equations. According to this, the traditional approaches are used in such a sense that the total motion, rigid body motion and elastic deformation, is directly referred to the inertial frame.

A main problem in multibody kinematics and dynamics is the choice of an appropriate formalism to describe the kinematics of finite rotations. This difficulty arises from the fact that finite rotations are not additive quantities. Several formalisms are available from the literature, see appendix A. The formalism called the *rotation vector* is chosen as the "best" of these techniques and has been adopted in Mecano. The combination of this approach and the finite element method forms the main structure of the multibody module Mecano.

### 2.2.1. Representation of finite rotations in three-dimensional kinematics by Mecano

There is a difference in the description of the orientation of a body in two-dimensional and three-dimensional kinematics. The two-dimensional situation needs just one parameter to describe the orientation of a body in the plane, while in the three-dimensional situation it is necessary to describe the orientation of the body by three independent parameters. Therefore, a well-defined formalism is needed to represent this three-dimensional orientation. Before introducing the implemented formalism some basics of three-dimensional kinematics will be discussed.

The following vectors are defined in the three-dimensional space:
- \( \mathbf{x} \) = position vector of a point before rotation
- \( \mathbf{x}' \) = position vector of the point after rotation

The Cartesian components of these vectors are

\[
\mathbf{x} = [x_1, x_2, x_3]^T \quad \text{and} \quad \mathbf{x}' = [x_1', x_2', x_3']^T
\]

A pure rotation of vector \( \mathbf{x} \) into vector \( \mathbf{x}' \) can be described as a linear transformation:

\[
\mathbf{x}' = R \mathbf{x}
\]  \hspace{1cm} (2.1)

in which \( \mathbf{x} \) and \( \mathbf{x}' \) are the columns containing the components of \( \mathbf{x} \) and \( \mathbf{x}' \), respectively. The rotation operator \( R \) is defined by 9 components \( r_{ij} \) (\( i, j = 1,2,3 \)). The condition of preserving the
length of the vector during the spherical motion implies

$$x'^T x' = x^T x \quad \rightarrow \quad R^T R = I \quad (2.2)$$

Satisfying this orthonormality property means that there are 6 constraints between the components $r_{ij}$, which must be satisfied. These restrictions lead to 3 (9 components - 6 constraints) independent parameters for defining the rotation operator. Therefore, an arbitrary finite rotation can be expressed by means of the rotation operator, which depends on three parameters:

$$R = R(\alpha_1, \alpha_2, \alpha_3) \quad (2.3)$$

A choice of these three parameters $\alpha_i$ can be made according to several formalisms that will later be discussed.

Another technique for describing the linear transformation of equation (2.1) was introduced by Euler. The definition of Euler’s theorem on finite rotations is as follows: "A pure rotation of a vector about his origin can be represented by a unique rotation $\phi$ about a specified direction $\vec{n}$" (figure 2.1). Therefore, the rotation operator can be defined as

$$R = R(\vec{n}, \phi) \quad (2.4)$$

in which $\vec{n}$ is the column matrix containing the components of $\vec{n}$. According to this theorem it is necessary to use four parameters for describing the linear frame transformation. However, those four parameters are constrained by a relation that requires the vector $\vec{n}$ to be a unit vector.

The most commonly known choices for the rotation parameters as defined in equations (2.3) and (2.4) are described in appendix A. This section only presents the rotation vector, because that formalism is implemented in the Mecano module. Two advantages of this formalism are the minimum set of parameters and the simple geometric interpretation. The rotation vector is defined as

$$\psi = \vec{n} \psi \quad (2.5)$$

It consists of three components $\psi_1, \psi_2$ and $\psi_3$, which represent a rotation $\psi$ about the rotation direction $\vec{n}$. The rotation vector is obtained by scaling the vector of the rotation direction by the rotation angle. The simplified rotation operator of this formalism is derived in appendix A and is given by

$$R = \exp \vec{\psi} \quad (2.6)$$
Chapter 2: The multibody package Samcef-Mecano

with the exponential-function as

\[ \exp \mathbf{\psi} = 1 + \mathbf{\psi} + \frac{1}{2!} \mathbf{\psi}^2 + \ldots. \]  

(2.7)

The superscript \( ^\sim \) represents the following skew-symmetric matrix corresponding to the vector \( \mathbf{\psi} \):

\[
\mathbf{\psi}^\sim = \begin{bmatrix}
0 & -\psi_3 & \psi_2 \\
\psi_3 & 0 & -\psi_1 \\
-\psi_2 & \psi_1 & 0
\end{bmatrix}
\]

(2.8)

Equation (2.6) defines the direct relation between the rotation operator and the rotation parameters. The inverse relation cannot be written as an explicit expression like (2.6), but is described by an algorithm. The algorithm for deriving the rotation parameters from the rotation operator is complex and is described by Cardona (1989, 38-40).

Non-linear problems are mostly formulated in a sequential form, which is known as the incremental formulation. In that case the final solution is obtained by solving a sequence of partial problems. This means that the problem is solved by looking at the system of the last computed time step as if it is an initial system. Dealing with this formulation means that the reference rotation vector is computed as far as time step \( n \). On the other hand the incremental rotation vector is computed as the rotation vector describing the rotation between time step \( n \) and \( n+1 \). Therefore, the total rotation operator is computed as follows

\[ \mathbf{R} = \mathbf{R}_{\text{ref}} \exp(\mathbf{\psi}_{\text{ref}}) \exp(\mathbf{\psi}_{\text{inc}}) \]

(2.9)

The subscript \( \text{ref} \) expresses the reference position at time step \( n \) and the subscript \( \text{inc} \) expresses the fact that the solution is taken corresponding to the rotation between two consecutive time steps.

2.2.2. The Finite Element concept used by Samcef-Mecano

In Finite Element analyses the generalized coordinates are usually taken as the ordinary generalized displacements in transition and rotation. However, in three-dimensional kinematics of multibody systems it is necessary to formulate the generalized coordinates in the context of a certain formalism. As mentioned before in section 2.2.1, the formalism used by Samcef-Mecano is the rotation vector. The ordinary generalized coordinates are denoted as \( \mathbf{q} \) and the generalized coordinates in the context of the rotation vector are denoted as \( \mathbf{q}^* \). In terms of the incremental formulation stated in the previous subsection the transition can be described as follows

\[ \delta \mathbf{q} = \mathcal{Q} \delta \mathbf{q}^* \quad \text{with} \quad \mathcal{Q} = \begin{bmatrix}
1 & 0 \\
0 & \mathbf{T}(\mathbf{\psi})
\end{bmatrix}
\]

(2.10)

in which \( \delta \mathbf{q} \) and \( \delta \mathbf{q}^* \) represent the increment in the generalized coordinates. The matrix \( \mathbf{T} \) is a
transformation operator only dependent of the rotation vector (Cardona, 1989, p. 53):

\[
T(\psi) = \frac{\sin \psi}{\psi} \mathbf{I} + \left(1 - \frac{\sin \psi}{\psi}\right) \mathbf{n} \cdot \mathbf{n} \cdot \mathbf{r} - \frac{1}{2} \left(\frac{\sin \left(\frac{\psi}{2}\right)}{\left(\frac{\psi}{2}\right)}\right)^2 \psi
\] (2.11)

It is clear that if \( \psi \rightarrow 0 \Rightarrow T(\psi) \rightarrow \mathbf{I} \), as expected.

If the finite element methodology is used, it is necessary to evaluate the mass matrix \( \mathbf{M} \), the tangent damping matrix \( \mathbf{C}' \) and the tangent stiffness matrix \( \mathbf{K}' \) of the system. The prefix tangent refers to the time dependency of those matrices. This means that they have to be re-evaluated every time step. These three matrices are derived for each element separately, which leads to the element system matrices. The element mass matrices are directly derived from the elements. This in contrast with the element tangent stiffness and tangent damping matrices, which are derived from the internal forces vector. This vector is denoted by \( \mathbf{G}^{\text{int}} \) and contains the forces that appear within the elements. The matrices are simply the derivatives with respect to the generalized displacements \( \mathbf{q}^* \) and the generalized velocities \( \dot{\mathbf{q}}^* \), respectively:

\[
\mathbf{K}_i = \frac{\partial \mathbf{G}^{\text{int}}}{\partial \mathbf{q}^*} \quad \mathbf{C}_i = \frac{\partial \mathbf{G}^{\text{int}}}{\partial \dot{\mathbf{q}}^*}
\] (2.12)

This formulation is required for the linearization of the equations of motion, which is discussed in a forthcoming section. So, an element can be seen as internal forces on the one hand and as elementary tangent stiffness and tangent damping matrices on the other hand.

To keep the separate elements of the system together, constraints are imposed on the system by means of joints between the different elements. The implementation of the constraints in the total system of equations will be explained in the next subsection.

The total system of equations that is obtained after assembly is a system of differential algebraic equations (DAE). This system of DAE is solved by an implicit integration scheme with use of the Newton-Raphson iteration procedure, see section 2.2.4.

2.2.3. Modelling of joints

The various elements in multibody systems are linked by joints. These joints introduce several constraints to the system, which are formulated in terms of the generalized coordinates. The constraints can be classified into two main groups:

- The holonomic constraints are formulated as implicit functions of the generalized coordinates and eventually of time:

\[
\Phi(\mathbf{q}^*; t) = 0
\] (2.13)

Holonomic constraints may be separated into explicitly time dependent, moving or rheonomic constraints and not explicitly time dependent, stationary or scleronomic constraints. These
constraints pose a restriction on the number of degrees of freedom and therefore on the set of possible configurations of the system.

- The *non-holonomic constraints* are formulated as implicit functions of generalized coordinates, generalized velocities and time. They may be classified into bilateral and unilateral constraints:

  Bilateral constraints are formulated as an equality relationship:
  \[ \sigma(q^*, \dot{q}^*, t) = 0 \]  
  and unilateral constraints as an inequality relationship:
  \[ \sigma(q^*, \dot{q}^*, t) \geq 0 \]

Constraints of the non-holonomic class cannot generally be integrated in time and therefore, must be treated appropriately. Such differential constraints pose a restriction on the behaviour of the system, but not on the set of possible configurations.

The constraints are implemented in the finite element approach by adding them to the total system of equations by way of an augmented Lagrangian approach. This approach introduces a contribution to the system of equations with a scale factor \( k \) and a penalty factor \( p \). The contribution of the constraints to the system is equal to that of the elements, which means that for each joint the internal forces vector is computed and from this vector the tangent stiffness and tangent damping matrices are derived. The contributions of the holonomic and non-holonomic constraints are derived in appendix B (Cardona et al., 1991). In case of a holonomic constraint the internal forces vector can be written as

\[
G_{int} = \begin{bmatrix} B(p \dot{\phi} - k \lambda) \\ k \dot{\phi} \end{bmatrix}
\]

with \( \lambda \) the Lagrange multipliers, which are introduced by the augmented Lagrangian method. The tangent stiffness matrix can be derived from the internal forces vector as

\[
K_i = \frac{\partial G_{int}}{\partial q^*} = \begin{bmatrix} pBB^T & -kB \\ -kB^T & 0 \end{bmatrix}
\]

with: \( q^* = \begin{bmatrix} q^* \\ \lambda \end{bmatrix} \)

In case of a non-holonomic constraint, the internal forces vector can be written as

\[
G_{int} = \begin{bmatrix} B_{nh}(pq - k\lambda) \\ k \dot{q} \end{bmatrix}
\]

with: \( B_{nh} = \frac{\partial \sigma_i}{\partial \dot{q}_{ki}} \)

in which the subscript \( \text{nh} \) refers to the non-holonomic case. The tangent damping matrix can be
derived from the internal forces vector in a way similar to the tangent stiffness matrix:

\[ C_i = \frac{\partial G^{\text{int}}}{\partial \dot{q}_i} = \begin{bmatrix} p B_{ab} B^T_{ab} & -kB_{ab} \\ -kB^T_{ab} & 0 \end{bmatrix} \quad \text{with:} \quad \dot{q} = \begin{bmatrix} \dot{q}^* \\ \dot{\lambda} \end{bmatrix} \tag{2.19} \]

The physical meaning of the Lagrange multipliers \( \dot{\lambda} \) is that they provide a measure of the forces required to close the corresponding constraints. The scale factor and the penalty factor should be chosen to balance the extended equations of motion adequately. They are computed as a value that is related to the stiffness of the system.

### 2.2.4. Time integration of the equations of motion

The equations of motion of a multibody system, which are discretized by the finite element method take the general form:

\[ M \ddot{q} + G^{\text{int}} = G^{\text{ext}} \tag{2.20} \]

In which \( q \) are the generalized coordinates as defined in section 2.1.2. by \( q^* \). This set of equations is known as differential algebraic equations (DAE). The notation is simplified by including in the internal forces vector the contribution of:

- kinematic constraints
- elastic and plastic forces
- damping forces
- friction forces

Since the mass matrix can become singular it is necessary to adopt an implicit method of solution, instead of an explicit method, for time integration of the equations (2.20). The difference between explicit and implicit methods of solution is explained in appendix C.

The implicit time integration scheme that is used to solve the DAE (2.20) is the one-step method of Newmark. This method consists of calculating the displacements, velocities and accelerations at time \( t_{n+1} \) in terms of the same quantities at time \( t_n \). This is achieved by using a Taylor series expansion limited to second order:

\[ \ddot{q}_{n+1} = \ddot{q}_n + (1-\gamma)h \dddot{q}_n + \gamma h \dddot{q}_{n+1} \tag{2.21} \]

\[ q_{n+1} = q_n + \dot{q}_n + (\frac{1}{2}-\beta)h^2 \dddot{q}_n + \beta h^2 \dddot{q}_{n+1} \tag{2.22} \]

In these equations, \( \beta \) and \( \gamma \) are two free parameters of the method. Usual values given to these parameters to obtain an implicit scheme are:

\[ \beta = \frac{1}{4} \quad \text{and} \quad \gamma = \frac{1}{2} \] These values correspond to the assumption of average constant acceleration over the time step. This set of values provides the Newmark unconditional scheme with maximum accuracy.
\[ \beta = \frac{1}{6} \text{ and } \gamma = \frac{1}{2} \] These values correspond to the assumption of linear acceleration over the time step.

The aim of the implicit Newmark method is to solve the system of equations at time \( t_{n+1} \):

\[ M \ddot{q}_{n+1} + G_{n+1}^{\text{int}} = \frac{G_{n+1}^{\text{ext}}}{2} \] \hspace{1cm} (2.23)

The acceleration term in this equation is expressed in terms of \( q_n \), \( \dot{q}_n \), \( \ddot{q}_n \) and \( q_{n+1} \) by equation (2.22). Transformation of this equation and integration in equation (2.23) results in

\[ \frac{M}{\beta h^2} q_{n+1} + G_{n+1}^{\text{int}} = \frac{G_{n+1}^{\text{ext}}}{2} + \frac{M}{\beta h^2} (q_n + h \dot{q}_n + (\frac{1}{2} - \beta) h^2 \ddot{q}_n) \] \hspace{1cm} (2.24)

This equation has a known right hand-side and is non-linear in \( q_{n+1} \) through the term of internal forces. The system can be solved iteratively using the Newton-Raphson method. This method consists of calculating an improved approximation \( \Delta q \) from an initial approximation \( q_{n+1} \). When integrating in equation (2.24) the equations (2.21) and (2.22) with use of the improved approximation, the improvement \( \Delta q \) can be solved from the following linearized system

\[ \left( \frac{M}{\beta h^2} + \frac{\gamma C}{\beta h} + K \right) \Delta q = R_{n+1} \] \hspace{1cm} (2.25)

with the tangent stiffness and tangent damping matrices according to equation (2.12) and the residual vector as

\[ R_{n+1} = G_{n+1}^{\text{ext}} + \frac{M}{\beta h^2} (q_n + h \dot{q}_n + (\frac{1}{2} - \beta) h^2 \ddot{q}_n) - \frac{G_{n+1}^{\text{int}}(q_{n+1})}{2} - \frac{M}{\beta h^2} q_{n+1} \] \hspace{1cm} (2.26)

in which the internal forces vector is an estimate and the external forces vector is known at time \( t_{n+1} \). The matrix between brackets in equation (2.25) is called the iteration matrix \( S \), which is time dependent and therefore needs to be re-evaluated every time step. Re-evaluating this matrix combined with inversion is the most cost-intensive operation in the implicit time integration scheme. Therefore, Mecano has the feature of not making these steps every time step, but for example every two time steps. This strategy has important consequences on required computational effort and still remains good convergence for the algorithm.

2.3. Example

Mecano is one of the diverse modules of the finite element software package Samcef. In brief, Samcef can perform mechanical and thermal calculations as well as solve certain problems occurring in fluid-structures. It can also perform static, dynamic, linear and non-linear analyses. The material behaviour can be linear elastic, isotropic, anisotropic or non-linear. A complete overview of the different modules is presented in appendix D. The module Mecano is for the analysis of multibody systems. The program has a library of standard elements and also has the
Chapter 2: The multibody package Samcef-Mecano

possibility of implementing user written elements. Mecano can perform an incremental static, kinematic and dynamic analysis. It is also possible to perform a modal analysis at any time. For a detailed explanation of the use of the module Mecano one is referred to another report of this author (Thijssen, 1993a) and to the manual (Samtech, 1991).

An example of a dynamic analysis performed by Samcef-Mecano and some results are presented. For further details on the implementation of the problem in the form of an input file one is referred to Thijssen (1993a, 32-33). A two-dimensional construction of which the behaviour is easy to predict is shown in figure 2.2. It consists of a wheel rolling with its contact point (node 5) on the y-axis when it moves parallel to that axis. The spring connects the centre of the wheel (node 2) with the z-axis (node 1), which represents a wall. A rigid body is connected to the wheel by means of a revolute joint (node 2-3). The other end of the rigid body slides along the z-axis (node 7). Finally, a friction element in the contact point of the rigid body to the wall introduces dissipation in the construction.

The system is assembled under the constraint of the following initial condition. At time zero the centre of the wheel is moved 0.5 towards the z-axis (original position is 4.0). Then the construction is released and a dynamic analysis is performed. During the analysis the construction will oscillate until a static equilibrium is reached.

Figure 2.3 shows the y-position of the wheel centre as a function of time. The construction performs approximately 1.25 oscillations until it reaches a static equilibrium. Figure 2.4 shows the normal and friction force in the contact point between the rigid body and the wall. The normal force remains positive, which means that the rigid body remains in contact with the wall. On the other hand, the friction force changes sign each time the wheel center reaches a maximum/minimum in its y-position. This occurs at time 1.8, 4.3 and 6.2 s. During the oscillation the friction force is 0.075 times the normal force. In the static equilibrium this coefficient is about 0.015, which means that the friction force is computed according to the equilibrium in the construction and is not directly related to the normal force.
Figure 2.2. The mechanism with a friction- and a wheel element.

Figure 2.3. The y-component of the wheel centre.
2.4. Discussion

From the theory discussed in the preceding part of this chapter it can be concluded that the Finite Element approach greatly differs from the multibody simulation origin. That is why the company SAMTECH presents Mecano as a module for analyzing flexible mechanisms instead of a multibody analysis program.

The example presented in the previous section also has been analyzed with the integrated multibody/finite element program Madymo (Neijssen and Heuvelmans, 1992). The results obtained by Madymo are comparable with those presented in this report. The only difference is the size of the friction force in the final static configuration. Madymo computes the friction force as the normal force multiplied by the friction coefficient. However, Mecano computes that force in line with the static equilibrium, which of course leads to a more correct value.

With respect to the CPU-time consumption it may be mentioned that analyses performed with the Finite Element approach are much more time consuming than those with classical multibody approaches (Doan, 1993).

From a practical point of view it may be concluded that Samcef-Mecano is rather easy to use, especially the post-processing of results. Further, it is good to mention that the manuals of the program are written in a way that self-teaching of the program is possible, except from some specific features as user defined elements.
Chapter 3
Optimization of multibody systems

3.1. Introduction

The development of numerical optimization methods was started in the late forties. Those methods were just used for analyzing problems, not for designing. The first optimization process used for designing a structural system was introduced by Schmit in 1960 (Schmit, 1960). In 1976 Schmit and Miura (1976) introduced the approximation concepts, which extended the range of problems that could be solved by the optimization process. Nowadays, these approximation concepts are commonly used methods in structural optimization. The main challenge in structural optimization is to combine the optimization process with an existing package for numerical analysis and decrease the computational costs required for the optimization part.

The first part of this chapter presents the standard optimization formulation, followed by the Sequential Linear Programming (SLP) method with move limits. This commonly used approximation method is modified for the application to time dependent systems. Finally, a program is presented, which links the SLP-method to the multibody program Samcef-Mecano discussed in the previous chapter.

3.2. The optimization process

3.2.1. The standard formulation

In the design stage of a structural system, the optimization process can be used to modify the system design. The result of this optimization process is the modification of some characteristic parameters in the total set of parameters of the structural system. These characteristic parameters are called the design variables of the system. They define the design point of the optimization problem and are denoted as

\[ \mathbf{x}^T = [x_1 \ x_2 \ \ldots \ x_n] \]  

(3.1)

with \( n \) being the number of design variables. In structural optimization, the design variables are parameters like: cross-sectional dimensions or member sizes, parameters controlling the geometry of the structure, material properties, etc.

When optimizing a structural system, a measure of effectiveness has to be minimized. This measure is called the objective function and often depends implicitly on the value of the design
variables:

\[ F = F(x) \quad (3.2) \]

The objective function in structural optimization is usually defined in terms of weight, vibration frequencies, buckling loads or any combination of these.

A very important issue of the optimization problem are the constraints, which the optimization process must satisfy. First, the so called side-constraints can be distinguished:

\[ x_i^1 \leq x_i \leq x_i^n \quad i = 1, \ldots, n \quad (3.3) \]

with \( x_i^1 \) and \( x_i^n \) as the lower and upper bounds of design variable \( x_i \), respectively. These side-constraints define the design space of the structural system. Second, there are constraints that are often implicit functions of the design variables, they are usually denoted as the behaviour constraints or simply the constraints of the optimization problem:

\[ g_j(x) \leq 0 \quad j = 1, \ldots, nc \quad (3.4) \]

Examples of constraints in structural optimization are: displacements, stresses or vibration frequencies. The total set of constraints limits the feasible part of the design space. According to definition (3.4) the constraints are called active if they have a value of zero. A design point is called feasible when the constraints are satisfied, otherwise the design point is called infeasible. The definitions of design variables, objective function and constraints are summarized in the following standard formulation of the optimization problem:

minimize: \( F(x) \)

such that: \( g_j(x) \leq 0 \quad j = 1, \ldots, nc \)

\[ x_i^1 \leq x_i \leq x_i^n \quad i = 1, \ldots, n \]

(3.5)

Maximizing an objective function can be done by minimizing its negative. Similarly, a constraint defined as

\[ g_j(x) \geq 0 \]

(3.6)

can be transformed to the standard formulation by taking its negative.

After defining the formulation of the optimization problem there is a need for a numerical solution process. The most commonly used algorithms are based on the iterative procedure

\[ x^L = x^{L-1} + \alpha^L S^L \quad L \geq 1 \quad (3.7) \]

Starting from the initial design \( x^{L-1} \), a more optimal solution \( x^L \) is searched for in the search direction \( S^L \) with step size \( \alpha^L \). This solution process has to take into account the bounds that are imposed on the design space by the constraints. Several techniques exist for finding the optimal search direction and step size. Some of them are directly available via a standard programming library, for example NAG-library or Matlab-toolboxes.
Chapter 3: Optimization of multibody systems

3.2.2. The Finite Element Method and optimization

The behaviour of a complex structural system is usually determined with the aid of a Finite Element (FE) package. With the common FE packages it is not possible to design automatically a more optimal system. Design this more optimal system is the task of an optimization program. Combination of these two separate programs leads to a design tool for structural systems. The coupling of the analysis program and the optimization program will lead to several problems. First, there is the problem of evaluating the objective function and constraints. The standard optimization routines expect the evaluation of objective function and constraints in a subroutine form. In general this is not possible or it requires additional programming. The second problem is the amount of the computational effort. The standard optimization routines require relatively many evaluations of the objective function, constraints or their derivatives. When they have to be evaluated the FE package has to make at least one complete analysis, which is of course very cost-intensive.

3.2.3. Sequential Linear Programming with move limits

The problems stated in the previous section result from the need of coupling two different numerical concepts and can be solved by introducing an approximation concept. This concept generates approximation models for the objective function and constraints. Generally three types of approximation concepts can be distinguished according to their ranges of applicability. A local approximation concept is only valid near a certain design point. A global approximation is valid in nearly the whole design space. Finally, a mid-range concept attempts to give global qualities to local approximations. This report only deals with a local approximation concept. The others are explained by Barthelemy and Haftka (1993).

The simplest local approximation concept is based on the linear Taylor series expansion. Due to this linearization the approximation models are just valid in a small area of the design space around the design point. This requires the introduction of move limits, which create a local design space in which the approximations are valid. This method leads to the following standard formulation, linearized about a design $x_0$:

$$\text{minimize: } F(x_0) + \sum_{i=1}^{n} (x_i - x_{0i}) \left( \frac{\partial F}{\partial x_i} \right)_{x_0}$$

subject to:

$$g_i(x_0) + \sum_{i=1}^{n} (x_i - x_{0i}) \left( \frac{\partial g_i}{\partial x_i} \right)_{x_0} \leq 0 \quad j = 1, \ldots, nc$$

$$x_{0i} - \delta x_i^l \leq x_i \leq x_{0i} + \delta x_i^u \quad i = 1, \ldots, n$$

(3.8)

The move limits are denoted as the side-constraints in this formulation, with $\delta x_i^l$ and $\delta x_i^u$ being the lower and upper bounds in the change on design variable $x_i$. Because of the linearity of the problem, the solution can be obtained by using a simplex algorithm from a numerical library. The obtained solution is far from optimal, which is due to the linearization and to the existence of move limits. Therefore, the design $x_0$ is replaced by the more optimal design and the linear
optimization process is repeated. This means that the problem (3.5) is replaced by a sequence of linearized problems of the form (3.8). The method described here is known as the Sequential Linear Programming (SLP) method with move limits. Figure 3.1 shows the characteristic steps of this method. The sensitivity analysis is necessary for computing the derivatives of the objective function and constraints.

Figure 3.1. All characteristic steps of the SLP-method.

The approximation models can also be generated by using a quadratic Taylor series expansion, which leads to better approximations. Computation of the second order derivatives is very expensive, so mostly it appears to be inefficient to use the quadratic form. However, if the linear approximation is not exact enough and the computational costs of the quadratic form are too high, then a solution can be found by means of intervening variables. These variables can be described as a function of the original design variables, for example the reciproke value, which can lead to a more linear behavior of the approximated functions. For more information about these intervening design variables is referred to Barthelemy and Haftka (1993), Etman (1993,5-8) and Haftka and Gürdal (1992,211-213).

The following example has been included for a better understanding of the SLP-method. The objective function of a certain one-dimensional optimization problem is represented by the solid line in figure 3.2. In a starting design point $x_0$ the approximation model of the objective function, denoted by the dashed line, is introduced with a move limit. From this approximation model the optimization software computes a new solution $x_1$, which has an approximated objective function value that is lower than the real objective function value for $x_0$. The approximated value of the objective function in the new design point is $F_1^{\text{app}}$, while the exact value is $F_1$. This process will continue until the design point $x_d$ is reached. In that point, new approximation models are generated and a new solution $x_g$ is computed. From this new solution $x_g$, again the design point $x_d$ will be computed. This implies an oscillation of the solution until the move limit is decreased and the solution $x_{\text{loc}}$ is obtained. So, convergence is caused by means of decreasing the move limit. Another remark is placed on the optimum that has been found. It can be noticed that a local optimum has been found instead of the global optimum $x_d$. However, if the problem starts from the design point $x_0$, it is clear that the global optimum will be found. So the optimum that will be found depends on the starting design. One must always be aware of this drawback when using a local approximation method.

A very important issue in SLP is the choice of the move limits, because without a proper choice of the move limits the process may never converge. In general, the value of the move limits should decrease as the optimum is approached. One reason for these reduced move limits is that the accuracy of the approximation should be higher when the optimum has been approached.
Figure 3.2. Example of a one-dimensional optimization problem solved by the SLP-method with move limits.

However, shrinking the move limits too early in the optimization process will slow down the convergence. The need to reduce move limits is indicated if the final design of a cycle proves, upon exact analysis, to be inferior to the initial design of that cycle, or provides no gain in the objective function $F$. The move limit can also be chosen according to the error in the approximation. If this error appears to be too large, the new obtained design can be rejected and the process will be started again from the same solution with a reduced move limit. The move limit can also be increased if the error in the previous cycle appeared to be very small. The size of the move limits can be chosen in various ways:

- **absolute:**
  \[
  x_{a} - ml \leq x_{i} \leq x_{a} + ml
  \]
  If the value of the move limit is the same for each design variable, it is necessary that the design variables are of the same order.

- **relative to the bounds of the design variable:**
  \[
  x_{a} - \delta x_{i} \leq x_{i} \leq x_{a} + \delta x_{i} \quad \text{with: } \delta x_{i} = (x_{i}^{u} - x_{i}^{l}) * ml
  \]
  The values of the design variables do not need to be of the same order. A disadvantage is that the size of the move limit can become very large compared to the size of the design variable when the lower and the upper bounds are of a different order.

- **relative to the design variable:**
  a) The design space remains symmetric
  \[
  x_{a} - \delta x_{i} \leq x_{i} \leq x_{a} + \delta x_{i} \quad \text{with: } \delta x_{i} = |x_{a} * ml|
  \]
  b) The design space is asymmetric
  \[
  x_{a} * ml \leq x_{i} \leq x_{a} / ml
  \]
  The values of the design variables do not need to be of the same order. Problems occur at both variants if the design variable approaches zero (the design variable may get zero or change sign).

According to Haftka and Gürdal (1992,231), the most popular choice is variant a) of the third formulation. Proper choices for starting values are in a range of 10 to 30%. If the move limit
Chapter 3: Optimization of multibody systems

needs to be decreased, a good choice is 10 to 50% of the previous value.

Another phenomenon in SLP is the infeasible design. If SLP is started with an infeasible design, it is not necessary that the value of the objective function will be decreased in the new design point, because the new design point is searched in the direction that minimizes the violations of the constraints. This means that the objective function may deteriorate.

A disadvantage of the SLP-method is the need of approximating every function of the optimization model as a linear function. This is a drawback, because it may occur that one or more model functions are explicit non-linear functions of the design variables. To approximate these explicit functions means a loss of information on those functions. Losing this information can be excluded by generalizing the SLP-method. This can be done by using a Non-linear optimization algorithm, which can deal with non-linear approximation models (Haftka and Gürdal, 1992, 236), instead of a linear simplex algorithm.

3.3. Specific features of multibody optimization

The difference between structural optimization and multibody optimization manifests itself in the diversity of design variables, constraints and objective function. Below, a list of examples of these variables and functions is given.

Design variables: beam cross-sections, spring or damper characteristics, characteristics of a friction or a tire element, node coordinates, etc.

Constraints: stresses in beams, reaction forces in nodes, spring or damper forces, friction forces, displacements of nodes, mechanism constraints (ensure that a mechanism will not lock), etc.

The objective function: weight of elements, reaction forces in nodes, spring or damper forces, friction forces, displacements or accelerations of nodes, dissipation energies, etc.

In a problem of steady-state response, the objective function and constraints depend only on the design variables and the response variables. However, in a problem of transient response these functions can also depend on time. Whenever the objective function and constraints depend on that extra parameter time, special treatment of these functions is required.

3.3.1. Constraints in transient analysis

Because of simplicity of notation there is assumed to be only one constraint. A constraint that is time dependent may be written as

\[ g(u, x, t) \leq 0 \quad 0 \leq t \leq t_f \]  \hspace{1cm} (3.9)

in which \( u \) are the response functions and \( t_f \) is the ending time of the analysis. The constraint (3.9) must hold over the entire time interval \([0, t_f]\). This is carried out by a discretization of the
Chapter 3: Optimization of multibody systems

Time interval in \( n_t \) points and replacing constraint (3.9) by a total of \( n_t \) time point constraints:

\[
g_i = g(u, \mathbf{x}, t_i) \leq 0 \quad i = 1, \ldots, n_t
\]  

(3.10)

The distribution of the constraints over the time interval has to be chosen dense enough in order to avoid the violation of constraint (3.9) between time points. A disadvantage of this type of constraint definition is the increasing number of constraints if the number of time points is high. A high number of constraints means that the computational effort necessary for the optimization process is also high, which is not very favourable. The constraint definition according to equation (3.10) is a standard method when dealing with time dependent constraints. Some ways of eliminating the time dependence of constraints and so decreasing the number of constraints are presented below.

The first way of removing the time dependency is to introduce an equivalent integrated constraint, which averages the severity of the constraint over the time interval. The first example of such an equivalent constraint is the integral constraint (Haftka and Gürdal, 1992):

\[
\bar{g}(u, x) = \left[ \frac{1}{t_f - t_0} \int_{t_0}^{t_f} g(u, x, t) dt \right]^{1/2}
\]  

(3.11)

in which \( \langle g \rangle \) denotes \( \max(g, 0) \). This constraint is violated if any of the constraints (3.10) is violated. If the constraint is active, which indicates that at least one time point constraint is active, its value is equal to zero. Its value is also zero if none of the time point constraints is violated. This denotes that the constraint is zero in the entire feasible area of the design space, which means that there is no indication if the constraint is almost critical. So, this type of equivalent constraint looses a lot of information of the time point constraints. An equivalent constraint that is not zero in the feasible area is the sum constraint (Haftka and Gürdal, 1992):

\[
\bar{g}(u, x) = g_{\max} + \frac{1}{\rho} \ln \left( \sum_{i=1}^{n_t} e^{\rho (g_i - g_{\min})} \right)
\]  

(3.12)

in which \( g_{\max} \) denotes the maximum value of all time point constraints. For extreme values of the constraints \( g_i \), the bounds on \( \bar{g} \) are

\[
g_{\max} \leq \bar{g} \leq g_{\max} + \frac{\ln(n_t)}{\rho}
\]  

(3.13)

These bounds indicate that \( \bar{g} \) is always more critical than \( g \). The larger the parameter \( \rho \), the smaller the interval of \( \bar{g} \).

The disadvantage of both equivalent constraints (3.11) and (3.12) is the possibility that a new constraint seems to be less critical, while the opposite is true. Figure 3.3 denotes that the constraint of the new design is less critical over almost the complete time interval, which can lead to a less critical equivalent constraint. However, the most critical time point constraint has become more critical, so the maximum constraint value has moved closer to the boundary.
One has to be aware of the savings obtained by the introduction of an equivalent constraint. The savings are only gathered in the optimization effort and in the computation of the derivatives and not in the evaluation of the time point constraints, while they still have to be evaluated every time point. However, the equivalent constraints require some computational effort for the evaluation of the integral or the sum.

A constraint which avoids the problem of the equivalent constraints is the critical time point constraint (Haftka and Gürdal, 1992, 292):

\[ g(u, x, t_m) \leq 0 \quad i = 1, 2, \ldots \]  

(3.14)

This class defines only a constraint at time points where the original constraint has a local maximum (figure 3.3). An inconvenience can arise when the number of local maxima increases after some optimization cycles.

### 3.3.2. The objective function in transient analysis

If the objective function is defined as a max-value operation, it can be treated in a special way. The max-value operation is defined as

\[ F = \max_{t \in [0, T]} f(u, x, t) \]  

(3.15)

When the maximum value of \( f \) arises at another time point during change of the design variables, it is possible that this definition introduces a discontinuity in the derivative of the objective function with respect to the design variables. A transformation of the max-value operation into another operation is carried out by introducing an extra design variable \( x_{n+1} \), which is put into an inequality constraint of the form (Haug and Arora, 1979, 332) (Haftka and Gürdal, 1992, 48)

\[ f(u, x, t_i) - x_{n+1} \leq 0 \quad i = 1, \ldots, n \]  

(3.16)
While the constraints (3.16) have to be satisfied, the objective function can be replaced by

\[ F = x_{n+1} \]  

(3.17)

The additional constraint is of the form (3.10) and can also be handled as explained in section 3.3.1. Dealing with the max-value objective function this way, ensures a perfectly smooth objective function.

3.4. The optimization process in Mecopt

The program Mecopt is an optimization program that is linked to the multibody module Mecano of the Finite Element package Samcef. The name of the program is an abbreviation of MECanO OPTimization. This section contains a short explanation of the program and chapter 4 contains three problems that are treated by Mecopt.

The main problem in writing an optimization program is the combination of a new program (written in FORTRAN 77) and an existing FE program. Making use of the programming capabilities of the unix operating system appeared to be the simplest solution of this problem. A unix programmed file is called a *unix script-file*. The script-file of the program Mecopt is called 'mecopt' and is the interface between the optimization programs and the FE module Mecano. Its structure is presented in figure 3.4. The optimization software consists of four main programs. *Initialize* takes the correct starting design from an input file and makes some checks on errors that can occur. The name of the program *variation* already implies that it handles the variation of the design variables, that is the sensitivity analysis. Finally, the programs *optimization1* and *optimization2* control the optimization part. The coordination between these programs and the script-file is achieved by using dummy files. For example, if the program *initialize* detects an error in the input, then the program creates an empty file named 'error.dat'. That file is detected by the script-file, which terminates itself. A more detailed explanation and a manual of Mecopt are presented in (Thijssen,1993b).

The optimization technique used by Mecopt is the SLP-method described in section 3.2.3. This method was chosen instead of a non-linear method, because the first assumption was to proceed from linear approximation models. The derivatives of the objective function and constraints with respect to the design variables are obtained by using the first-order forward
difference scheme. For example, the derivative of the objective function:

\[
\frac{\partial F(x, \Delta x_i)}{\partial x_i} = \frac{\Delta F(x, \Delta x_i)}{\Delta x_i} = \frac{F(x_i + \Delta x_i) - F(x_i)}{\Delta x_i}
\]  

(3.18)

Use of this method requires \(n+1\) evaluations of the objective function and constraints. This means \(n+1\) FE computations, which are very time consuming. In spite of this the method is chosen, because it is the easiest way for computing the derivatives. Of course the user has to be aware of the size of the variation in the forward difference scheme. A good method for choosing that parameter is explained in appendix E. At last, the linearized optimization problem defined by equation (3.8), is solved by a standard subroutine for linear programming problems from the NAG-fortran-library, namely E04MBF.

Other techniques for computing the derivatives are semi-analytical methods. Such a method determines the derivatives, proceeding from the stiffness and damping matrices of the FE system. An advantage of these methods is that they greatly decrease the computational costs. Characteristic for multibody systems is the time dependency of the mentioned matrices because of the non-linearity of the system (section 2.2.2). This means that the matrices change each time step of the multibody analysis. Due to this time dependent behaviour the applicability of the semi-analytical methods requires additional research.

This chapter described the optimization process and particularly the SLP-method with move limits. Also has been explained how to modify this method has to be modified for application to time dependent problems. Finally, the program Mecopt was presented in which the SLP-method has been linked to the Samcef-Mecano multibody program presented in chapter 2. In the next chapter Mecopt will be tested on three different optimization problems.
Chapter 4
Experimental problems

4.1. Introduction

This chapter presents three examples in order to illustrate the functioning of the program Mecopt that was introduced in the previous chapter. The examples are discussed quite extensively to give a good impression of the behaviour of the program. The first example concerns the performance of an impact-absorber (Afimwala and Mayne, 1974). For this example the various solution methods for the time dependent optimization problem are tested. The second example is a problem in which an improved mechanism for a landing gear has to be designed (Hansen, 1992). And finally, the third example deals with flexible elements in a four-bar mechanism (Soehoni and Haug, 1982). The results of each optimization analysis are tabled and presented graphically in appendix F.

4.2. Impact-absorber

The configuration of the impact absorber is shown in figure 4.1. A mass $M$ is attached to a platform by a linear spring (stiffness coefficient $k$) and a linear damper (damping coefficient $b$). The impact-absorber is used for the following test. The displacement of mass $M$ is in the positive $y$-direction. When the platform contacts the ground the position of the mass is zero and its velocity is 5 m/s. From that point in time the behaviour of the mass as a function of time is regarded. It is assumed that the contact between the platform and the ground is remained during the transient response. The global response of the system is that the position of the mass will increase to a maximum and then decrease to zero with eventual oscillations around its zero position.

The purpose of an impact-absorber is to absorb the kinetic energy of a moving system. In this simple problem the energy is dissipated by the linear damper. The behaviour of the impact-absorber is determined by the characteristics of the spring and damper. The impact-absorber performs better if the maximum acceleration that occurs during the response is minimized, which means that the forces acting on the mass are minimized. The deflection of the mass from its zero position must be lower than 1.0 m during the response, because it is not allowed that the mass hits
the platform. With these motives as main requirements of the impact-absorber the optimization problem according to the standard formulation given in section 3.2.1. can be defined as:

\[
\text{minimize: } \max_{t \in [0,12]} |y(b,k,t)| \\
\text{such that: } |y(b,k,t)| - 1 \leq 0 \quad 0 \leq b \leq 1 \\
0 \leq k \leq 1
\] (4.1)

The time interval is discretized in 501 time points. However, in order to reduce the number of time point constraints, these constraints just have to be satisfied every 10 time points, which means every 0.24 seconds.

An exact representation of the objective function and the constraints in the design space can be obtained by solving the problem analytically (figure 4.2). The procedure for gaining this figure is skipped, because it is beyond the scope of this test. The figure shows the design space of the problem in which the objective function is displayed by means of contour lines and the constraint by means of the solid line parallel to the dashed-dotted line. The area at the side of the solid line represents the feasible area. The constraint in the figure is displayed according to the max-value operation as described by equation (3.15) in case of the objective function. A global impression of the optimization problem can be obtained from this figure. For example the optimum derived from figure 4.2 is approximately (0.50,0.35) with a value of the objective function equal to approximately 0.52.

![The impact-absorber](image)

**Figure 4.2.** Exact solution of the impact-absorber problem.

The next five subsections describe various solution methods for the impact-absorber problem. The standard formulation of each problem is equal to relation (4.1) and the starting point is (1.0,0.3), which is conform to an objective function value of 0.980. The move limit is chosen
relative (symmetric) to the design variables with a maximum of 30%. In the first few cycles of the optimization process a maximum relative error in the approximations of the objective function and the constraints of 20% is acceptable. The optimization process is stopped when convergence is obtained for the second time. The convergence rate is reached if the relative decrease of the objective function in two successive cycles is less than 0.005, which means less than 0.5% progress in objective.

4.2.1. Analysis with time point constraints

The first solution procedure for the impact-absorber problem uses time point constraints as defined by relation (3.10). The constraint of the standard formulation is replaced by 51 equally distributed time point constraints. This method of constraint formulation is the most commonly known and the simplest.

The development of the optimization process is tabulated and graphically presented in appendix F.1. The optimum obtained by Mecopt after 6 optimization cycles is (0.490, 0.362) with the value of the objective function equal to 0.526. This optimum has been obtained by reducing the move limits until the solution converged.

4.2.2. Analysis with an integral constraint

The second solution procedure makes use of the integral constraint formulation of relation (3.11). This means that the 51 time point constraints are replaced by just one constraint.

The progress of the optimization process is presented in appendix F.2. The optimum (0.478, 0.373) has been gained after 8 cycles with the value of the objective function equal to 0.526. The disadvantage of the uncontrolled constraint approach (section 3.3.1.) manifests itself in an unusual development of the design, which can be seen in appendix F.2.

4.2.3. Analysis with a sum constraint

This analysis uses the sum constraint formulation as presented in relation (3.12). For this type of constraint a free parameter \( p \) must be assigned a value. The value of \( p \) can be chosen according to the absolute error of the sum constraint, see relation (3.13). The influence of this parameter is displayed graphically in figure 4.3. The figure shows a great decrease of the error if \( p \) increases. For the impact-absorber two values for \( p \) are considered, namely 10 and 100. If \( p \) is 10 the absolute error is lower than \( \ln(51)/10 = 0.393 \) and if \( p \) is 100 the absolute error is lower than \( \ln(51)/100 = 0.039 \).

The results of both cases are presented in appendix F.3. The optimum of case 1 is (0.563, 0.545) and of case 2 (0.507, 0.351), with the values for the objective functions 0.631 after 6 cycles and 0.531 after 6 cycles, respectively. These results show the big influence of the parameter \( p \). When \( p \) is small a large error in the constraint appears, which has great influence on the
obtained optimum. The comparison of the lower-left plots of both cases in appendix F.3 shows the difference in interpretation of the constraint value in the starting point. For $\rho$ is 10 the constraint has a value of more than -0.1 in which the error is large, while for $\rho$ is 100 the value is less than -0.25 in which the error is small. This second value can be compared with the one obtained with time point constraints.

**4.2.4. Analysis with critical time point constraints**

The last analysis with special treatment of the constraint deals with critical time points. This method just takes those time points as constraint of the optimization process that are a local maximum in time of the original constraint.

The results are presented in appendix F.4. The optimum (0.484,0.369) has been found after 7 cycles with an objective function value of 0.527. From the results in appendix F.4 it can be concluded that the number of constraints increases after 2 optimization cycles.

**4.2.5. Analysis with special treatment of the objective function**

The analyses of sections 4.2.1 up to and including 4.2.4 deal with the objective function as a max-value operation. It is better to treat the function in a special way as presented in section 3.3.2. This special treatment introduces 501 extra time point constraints, which means a total of 552 time point constraints for this problem. The number of extra constraints is 501, because in the previous methods the objective function has been derived from an analysis in which the time interval was discretized in 501 points. The starting value of the extra design variable can be chosen in three different ways: feasible, active or infeasible. Three starting values corresponding to the three possible starting configurations are: 1.0, 0.980 and 0.1. However, if the special treatment of the objective function is reliable then the choice of the starting value may not have any influence on the computed optimum.

The results of the three cases are presented in appendix F.5. The achieved optimum is for all cases exactly the same, namely (0.484,0.368,0.526) with the value of the objective function equal to 0.526. That all three solutions are exactly the same is due to the fact that the extra constraints introduce a vertex in the design space in which the problem finds its optimum.

A remarkable fact can be seen from the results obtained by starting from an infeasible design. The change in the third design variable from the first to the second cycle is larger than the move limit allows. The reason for this is, when starting from an infeasible design the optimization routine tries to find a feasible design by minimizing the constraint violations over all constraints including the side-constraints, which can lead to a violation of the move limits.
4.3. Discussion of the various analysis methods

To enable a proper comparison of the various analysis methods of the previous subsections, the main results are summarized in Table 4.1. The first column of the table gives the method, the second the optimum design point, the third the optimum objective function value, the fourth contains the real constraint value of the most active time point, the fifth contains the number of optimization cycles necessary to obtain convergence and the last column contains the symbol corresponding to Figure 4.4. In this figure the optimum design points of each method are plotted in a part of the design space.

Table 4.1. The main results of the various solution methods

<table>
<thead>
<tr>
<th>Method</th>
<th>$x_{opt}$</th>
<th>$F_{opt}$</th>
<th>$\max(g_i)$</th>
<th>cycles</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>starting values</td>
<td>1.000</td>
<td>0.300</td>
<td>0.980</td>
<td>0.721</td>
<td></td>
</tr>
<tr>
<td>time points</td>
<td>0.490</td>
<td>0.362</td>
<td>0.526</td>
<td>0.99994</td>
<td>6</td>
</tr>
<tr>
<td>integral</td>
<td>0.478</td>
<td>0.373</td>
<td>0.526</td>
<td>1.00145</td>
<td>8</td>
</tr>
<tr>
<td>sum ($p = 10$)</td>
<td>0.563</td>
<td>0.545</td>
<td>0.631</td>
<td>0.837955</td>
<td>6</td>
</tr>
<tr>
<td>sum ($p = 100$)</td>
<td>0.507</td>
<td>0.351</td>
<td>0.531</td>
<td>0.994450</td>
<td>6</td>
</tr>
<tr>
<td>critical time points</td>
<td>0.484</td>
<td>0.369</td>
<td>0.527</td>
<td>1.000270</td>
<td>7</td>
</tr>
<tr>
<td>special obj.func.</td>
<td>0.484</td>
<td>0.368</td>
<td>0.526</td>
<td>1.000000</td>
<td>6</td>
</tr>
<tr>
<td>from Afimiwala</td>
<td>0.491</td>
<td>0.354</td>
<td>0.521</td>
<td>unknown</td>
<td></td>
</tr>
</tbody>
</table>

The four solution methods used for the constraints all show approximately the same results, with the exception of the sum constraint. The value of the objective function of that method is slightly higher than the ones obtained by the other methods and also the final real constraint value is not equal to 1.0. These differences are due to the previously mentioned influence of the parameter $p$ of this method. For relative high values of that parameter, the deviation to the real optimum approaches zero. This can be concluded from the table above, as well as from the theory in section 3.3.1. The absolute error of the constraint in the optimum is equal to $1 - 0.838 = 0.162$ for $p = 10$ and $1 - 0.994 = 0.006$ for $p = 100$. These errors are about 50% and 15% of the maximum error that can occur for these values of $p$ (see section 4.2.3.). This means that the maximum error value is a rather extreme maximum in this optimization problem.

An important drawback of the integral constraint is the already mentioned uncontrolled approach of the active constraint. A consequence of this phenomenon is that the constraint can be greatly violated during the optimization process and then turns smoothly to the active value. One can imagine that if the design is in the feasible area again (not in this example) during some optimization cycles, it will again jump into the infeasible area like the first time. This phenomenon is explained by Figure 4.5.
It is remarkable that the design path and the constraint behaviour of the sum constraint and the time point constraints are practically the same (compare appendix F.1 and F.3 case 2). This similarity implies that the sum constraint is a good alternative for the time point constraints.

The alternative constraint formulations have been introduced for decreasing the computational effort necessary for the optimization algorithm. However, these costs are extremely low compared with the costs necessary for the multibody analyses. This means that the alternative formulations do not have to be introduced in order to reduce the total computation time.

The method that treats the objective function in a special way also gets comparable results. An advantage of this method is that there is no need for decreasing the move limit when reaching the optimum, which is due to the extra constraints that are introduced with this method. These extra constraints introduce a vertex in the feasible design area in which the optimum design will be found. However, the existence of a vertex is not a general result when using this method, while besides the introduction of extra constraints also an extra design variable is introduced, which enlarges the dimension of the problem.

When comparing the five solution methods with the results obtained by Afsimiwala and Mayne (1974) (see table 4.1), it may be concluded that all methods have an optimum objective
function value that is on the one hand comparable and on the other hand higher than the one obtained by Asimiwal. Probably this slight difference is due to the convergence criterium used in the optimization program.

Finally, the behaviour of the system before and after the optimization process is displayed in figure 4.6. Both plots of this figure display the acceleration and the position of the mass during the motion. The upper plot represents the initial design while the lower plot represents the optimal design. From these results it may be concluded that the maximum acceleration of the mass decreased a lot and that the maximum deflection of the mass is exactly 1.0 for the optimum design. Further, it can be denoted that the maximum acceleration moved to another time point.

Figure 4.6. The behaviour of the initial (upper plot) and the optimal (lower plot) system. The solid line represents the acceleration and the dashed line the position of the mass.

4.4. Landing gear mechanism

The example discussed in this section is derived from the original problem described by Hansen (1992). A landing gear of an airplane is represented by the 2D model in figure 4.7. Before landing of the airplane, the landing gear has to be guided from the upper position \( C_1 \) to the lower position \( C_2 \). During this motion the landing gear must rotate 70° (1.222 radians) in order to position for a good landing.
The initial mechanism (Hansen, 1992) is shown in figure 4.8 in the starting configuration. The dashed landing gear in this figure represents the position of it after motion of crank 2-4 by an angle \( \theta \). From this figure it may be concluded that the initial mechanism is far from satisfying the requirements. The design variables of the model are the \( x \)- and \( y \)-coordinates of nodes 2, 4, 6 and 9. This means a total of 8 design variables. The point \( P \) in the initial design corresponds to point \( C_1 \), so the coordinates of point \( P \) are not used as design variables. This is a difference from the original problem of Hansen, who takes the coordinates of this point also as design variables. However, in this report the choice has been made to position point \( P \) directly on the desired position \( C_1 \). This is done for saving a lot of computation time in the sensitivity analysis. Another difference from the original problem is the choice of the driving angle \( \theta \) of the mechanism. Hansen also takes this parameter as a design variable, but since this is not possible with Mecopt it is taken as the optimum value gained by Hansen, namely 98° (1.710 radians).

The optimization problem is defined as follows: design the mechanism in such a way that the deviation of point \( P \) from the point \( C_2 \) at the end of the motion is minimized. The side-constraints on the design variables of nodes 2 and 9 are chosen corresponding to the area of attachment of the fuselage. The only constraint in the optimization problem is the equality constraint that requires the rotation of point \( P \) to be equal to 70° after total motion of the mechanism. However, this constraint is not treated as an ordinary equality constraint, but as part of the objective function by means of a penalty method. In case of an ordinary equality constraint the optimization process always tries to satisfy this constraint by violating the move limits, which leads to an uncontrolled optimization process. The advantage of the penalty method compared to the standard method of treating the constraints is that the optimization process is capable of gradually moving to a feasible design, which makes the optimization problem more flexible. The composed objective function that proceeds from this constraint approach is constructed according to the least-squares method. The definition of the optimization problem in terms of the standard formulation is
as follows

\[
\text{minimize: } (\Delta p)^2 + \alpha (\Delta \phi)^2 \\
\text{such that: } \begin{align*}
x_2 &\leq 100 \\
y_2 &\geq 100 \\
x_9 &\geq 180 \\
y_9 &\geq 100
\end{align*}
\] (4.2)

in which $\Delta p$ is the deviation of point $P$ (after motion) from point $C_2$ and $\Delta \phi$ the deviation of the total rotation from 1.222 radians. The penalty parameter $\alpha$ is chosen according to the values of $(\Delta p)^2$ and $(\Delta \phi)^2$ for the initial design. For that design holds: $(\Delta p)^2 = 2026$ and $(\Delta \phi)^2 = 1.551$. The value of $\alpha$ is chosen to be 5000, which means that the contribution of $(\Delta \phi)^2$ is higher than the contribution of $(\Delta p)^2$. Of course, this choice of the penalty parameter is questionable, but after a trial and error approach 5000 appeared to be a proper value for this problem. The move limit is taken absolute, which is allowed for this problem since the design variables are all of the same order. The relative error in the approximation of the objective function must stay below the limit of 20% and during the optimization process this maximum decreases to a value of about 5%.

The results obtained by Mecopt are presented in appendix F.6. The coordinates of the point $P$ at the end of the motion are $(158.82, 80.59)$ and the rotation is $70.29^\circ$ (1.227 radians), which means that the objective function is 1.86. In figure 4.9 the optimal mechanism is shown in the upper and the lower position together with the landing gear. When this optimal design is compared with the one obtained by Hansen it appears that the designs are slightly different from each other. However, the global view of both mechanisms is the same and they both almost satisfy the requirements. Probably, the difference is due to the simplification of the optimization problem stated in this report.

4.5. Four-bar mechanism

This problem is derived from the original problem given by Sohoni and Haug (1982). The 2D four-bar mechanism given in figure 4.10 is constructed out of three flexible beams of circular cross-section and one rigid element representing the ground. The mechanism is driven by a constant angular velocity on the first beam. It is desired to design the mechanism for a minimum mass, while requiring that the bending stresses in the beams do not exceed the allowable stress limit. The cross-sectional areas of the three flexible beams are taken as the design variables. The
maximum bending stress in a beam is given as (Sohoni and Haug, 1982)

\[ \sigma_i = \frac{4\pi}{A_i} \left| M_i \right| \quad i = 1, 2, 3 \]  

in which \( i \) is the beam number, \( M_i \) the maximum bending moment on beam \( i \) and \( \sigma_i \) the maximum bending stress in beam \( i \).

A problem due to Mecano occurs when dealing with this formulation, while Mecano can only compute the loads on the beam in the end nodes of the element. However, the maximum bending moment in the beams can not appear in these nodes, because these are hinge joints and hinge joints can not support moments. Due to this problem each beam is divided into 5 separate beams (figure 4.11), which make it possible to consider the bending moments in the intermediate nodes. The next problem was that Mecano appeared not to be able to analyse this modified system, at least not within the experiences of the author at that moment. The prescribed problems require a modification of the optimization problem. First, the maximum bending stresses in the beams are taken as the highest in the two end nodes. Since these stresses are zero, linear torsion dampers (damping coefficient is 2.0) are introduced in nodes 4 and 7 (figure 4.10). These modifications create an optimization problem that is no longer physical valid, but in spite of this it is still useful to validate the program Mecopt.

![Figure 4.10. The configuration of the four-bar mechanism.](image)

![Figure 4.11. Each beam divided into five separate beams.](image)

The standard formulation of the prescribed optimization problem can be formulated as follows

\[
\begin{align*}
\text{minimize: } & \sum_{i=1}^{3} \rho l_i A_i \\
\text{such that: } & \sigma_i(A_i; t) - \bar{\sigma} \leq 0 \quad i = 1, 2, 3 \\
& 0.4 \leq t \leq 1.0 \quad s \\
& A_i \geq 0
\end{align*}
\]

in which the density \( \rho \) is equal to 2757.24 kg/m\(^3\) and the stress limit \( \bar{\sigma} \) is equal to 1.3790E+7 N/m\(^2\). The modulus of elasticity is taken as 6.8948E+10 N/m\(^2\). The time interval considered is discretized in 151 time points and starts at 0.4 s in order to eliminate numerical oscillations in the beginning of the solution. The discretization leads to a total of 453 (3 beams \( \times \) 151 constraints) time point constraints.

The results of the optimization problem obtained by Mecopt are presented in appendix F.7. As already mentioned these results can not be compared to the ones obtained by Sohoni and Haug,
because the problem definition has been modified. The results show a good convergence in the objective function as well as in the design variables. The optimum design (0.00498, 0.00184, 0.00086) is obtained after 11 cycles. During the optimization process the objective function, representing the mass of the four-bar mechanism, has decreased from 546 kg to 10.6 kg. The final constraint violations in the optimum are almost zero, which means that none of the bending moments at the hinge joints is exceeded.

4.6. Discussion

The three optimization problems stated in the previous sections all showed good convergence in the sense that an optimum was obtained within a respectable number of iterations. However, an automatic move limit strategy was adapted to the optimization problem of the landing gear mechanism, because of the excessive computation time necessary for the sensitivity analysis. The optimization process of the impact-absorber and the landing gear mechanism show results that are comparable with the results presented in the literature. However, in case of the four-bar mechanism it appeared to be necessary to modify the optimization problem, which resulted in a problem that could not be compared with the original problem from the literature anymore. From the three presented optimization problems it may be concluded that the SLP-method with move limits is functioning well as an optimization method in the field of multibody systems.
Chapter 5
Conclusions and Recommendations

The objective of the study presented in the preceding part of this report was to investigate the possibilities of applying the SLP-method with move limits to multibody systems. The study has led to the following conclusions and recommendations:

Conclusions:

- The application of the SLP-method with move limits to the multibody program Mecano succeeded. Three optimization problems were tested and two of them show optimal solutions that are comparable with the literature. The third optimization problem also obtained an optimal solution, but due to the change of the original problem formulation the solution could not be compared with the results from the literature anymore. In general may be concluded that the SLP-method with move limits is useful in the field of multibody systems.

- The five methods for special treatment of the time dependent optimization problem all obtained comparable results, with respect to the obtained optimum design. The special treatment of the objective function in case of a max-value operation shows a good behaviour. This formulation introduces extra constraints in the optimization problem, which on the one hand lead to an increase of the computational effort and on the other hand can create a vertex in the design space. The methods for treating time dependent constraints also show good results. The time point constraint formulation is considered as the standard method, the others are meant for decreasing the number of constraints and so decreasing the computational costs. However, using the alternative constraint formulations is not necessary for the reduction of computation time, because the optimization costs are very low compared to the costs of the multibody analyses. However, if one wants to use one of the methods it is recommended to use the sum constraint formulation, because this formulation shows results that are very similar with the standard method of the time point constraints.
Chapter 5: Conclusions and Recommendations

Recommendations:

- The use of a simplex algorithm for computing the solution of the optimization problem has one big disadvantage. The functions (objective function and constraints) that are known as an explicit function of the design variables also have to be linearized, which means a loss of important information. This problem can be overcome by implementing an SQP-algorithm instead of a simplex algorithm.

- The computation of the derivatives of the objective function and constraints with respect to the design variables are now computed with a first-order forward difference scheme. This method requires n extra analyses to compute the derivatives, which is rather computational expensive. So it is recommended to search for a method for computing the derivatives that is less expensive.

  Commonly used methods for computing the derivatives in structural optimization are semi-analytical methods, which obtain the derivatives directly from the stiffness- and damping matrices. Maybe some of these techniques can be applied.

- The considerable lengths of the computations make the existence of a move limit strategy very valuable. A rather simple move limit strategy is used in this report, but this one is still far from optimal. So, it is recommended to develop a proper and general move limit strategy.
References


Doan, B., University of Liège, personal communication, 1993.


Appendix A: Different formalisms to describe finite rotations

This appendix is a summary of a study of Cardona (1989,11-66) on the various formalisms to describe finite rotations that are presented in the literature. The formalisms are described concisely for keeping a good overview of the differences between them. Before introducing a formalism for describing the rotation parameters, first the requirements on a specific formalism are considered. A formalism adapted to the needs of three-dimensional kinematics should meet all or most of the following requirements:

- minimal number of parameters
- absence of singularities in the interval \([0,2\pi]\) (or \([-\pi,\pi]\))
- computational simplicity (algebraic nature)
- easy geometric interpretation

For a given set of parameters to describe the finite rotations, the following relationships are required:

- direct and inverse relations between rotation parameters and rotation operator
- angular velocities and accelerations
- differential relationships (variations)
- incremental relationships

This appendix just considers the four requirements and the first relationship (just the direct relation), because these are the most important for comprehension of the formalisms. However, this may not imply that the others are unimportant. For each formalism just the direct relation between the rotation operator and the rotation parameters is derived. Further, the singularities of each formalism are mentioned.

Taking into account the mentioned requirements, there are several formalisms to describe finite rotations. All these formalisms can be derived from just three different approaches: the geometrical approach, the matrix approach and the algebraic approach (subdivided in quaternion algebra and matrix algebra). Most of the formalisms can be found by two or three different approaches, because the approaches only differ in the way they look at the problem of finite rotation. That is why in this report just the formalisms are mentioned, in spite of the approach from which they are derived. The formalisms that are gathered with the three approaches are the following:

- Euler angles
- Bryant angles
- a unique rotation about an arbitrary axis
- Rodrigues parameters
- Euler parameters
- conformal rotation vector (CRV)
- linear parameters
- rotation vector (implemented in the Mecano module)
Appendix A: Different formalisms to describe finite rotations

First the plane rotation operator will be defined, because it is required for the derivation of the rotation operator of the first three formalisms. The simplest rotation operation is a rotation about a coordinate axis. Figure A.1 shows the case in which a rotation $\phi$ is performed about the $z$-axis.

![Figure A.1. A simple rotation in the plane Oxy.](image)

After the rotation a column $\mathbf{x}$ with components $(x_1, x_2, x_3)$ will be transformed into the rotated column $\mathbf{x}'$. This rotated column can be written in terms of the components of $\mathbf{x}$ and the rotation angle $\phi$:

$$
\begin{align*}
    x'_1 &= x_1 \cos \phi - x_2 \sin \phi \\
    x'_2 &= x_1 \sin \phi + x_2 \cos \phi \\
    x'_3 &= x_3
\end{align*}
$$

(A.1)

In matrix form this rotation is given as

$$
\mathbf{x}' = \mathbf{R} \mathbf{x}
$$

(A.2)

with the plane rotation operator

$$
\mathbf{R}^\phi = \begin{bmatrix} \cos \phi & -\sin \phi & 0 \\ \sin \phi & \cos \phi & 0 \\ 0 & 0 & 1 \end{bmatrix}
$$

(A.3)

Similar plane rotations about the $x$ and $y$-axis generate the rotation operators

$$
\mathbf{R}^\theta = \begin{bmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{bmatrix}
$$

(A.4)
and

\[
R^\psi(\psi) = \begin{bmatrix}
1 & 0 & 0 \\
0 & \cos \psi & -\sin \psi \\
0 & \sin \psi & \cos \psi
\end{bmatrix}
\]  

\[\text{Equation A.5}\]

**Euler angles**

The formalism of Euler angles is a succession of three elementary rotations about three different axes (figure A.2).

- a rotation \(\phi\) about \(OZ\); rotation operator \(R^\phi(\phi)\)
- a rotation \(\theta\) about \(OX\); rotation operator \(R^\theta(\theta)\)
- a rotation \(\psi\) about \(OZ\); rotation operator \(R^\psi(\psi)\)

![Figure A.2. Description of finite rotations in terms of Euler angles.](image)

The frame transformation is similar to (A.2). In the case of Euler angles the rotation operator is a composition of the three elementary rotation operators.

\[
R = R^\phi(\phi) R^\theta(\theta) R^\psi(\psi)
\]  

\[\text{Equation A.6}\]

The elementary rotation operators are similar with (A.3) and (A.5). Therefore, the explicit direct relation is

\[
R = \begin{bmatrix}
\cos \phi \cos \psi - \sin \phi \cos \theta \sin \psi & -\cos \phi \sin \psi - \sin \phi \cos \theta \cos \psi & \sin \phi \sin \theta \\
\sin \phi \cos \psi + \cos \phi \cos \theta \sin \psi & -\sin \phi \sin \psi + \cos \phi \cos \theta \cos \psi & -\cos \phi \sin \theta \\
\sin \theta \sin \psi & \sin \theta \cos \psi & \cos \theta
\end{bmatrix}
\]  

\[\text{Equation A.7}\]

Singularities occur if \(\theta = 0\) or \(\pi\). In these cases both rotations about the "z-axis" are collinear, which leads to just one single rotation \((\phi \pm \psi)\) about the z-axis.
Appendix A: Different formalisms to describe finite rotations

Bryant angles

The formalism of Bryant angles is also a succession of three elementary rotations about three different axes (figure A.3):

- a rotation $\psi$ about $Oz$; rotation operator $R^z(\psi)$
- a rotation $\theta$ about $Oy$; rotation operator $R^y(\theta)$
- a rotation $\phi$ about $Oz$; rotation operator $R^z(\phi)$

Figure A.3. Description of finite rotations in terms of Bryant angles.

According to a frame transformation similar to (A.2), the explicit direct relation is obtained with (A.3), (A.4) and (A.5):

$$
R = \begin{bmatrix}
\cos \theta \cos \psi & \sin \theta \sin \phi \cos \psi - \sin \psi \cos \phi & \sin \theta \cos \phi \cos \psi + \sin \psi \sin \phi \\
\sin \phi \cos \theta & \sin \theta \sin \phi \sin \psi + \cos \psi \cos \phi & \sin \theta \cos \phi \sin \psi - \sin \phi \cos \psi \\
-\sin \theta & \sin \psi \cos \theta & \cos \phi \cos \theta
\end{bmatrix}
$$

Singularities occur if $\theta = \pm \frac{\pi}{2}$. In those cases both rotations about the "z-axis" are collinear, which leads to just one single rotation $(\theta \pm \psi)$ about the z-axis.

All formalisms that follow rely on Euler's theorem on finite rotations. The first formalism starts with this theorem and the following are derived from that formalism.

Finite rotation as a unique rotation about an arbitrary axis

This formalism makes direct use of Euler's theorem on finite rotations. This theorem states that any finite rotation can be expressed as a unique rotation $\phi$ about an appropriate axes $\mathbf{n}$. According to that theorem a finite rotation may be described by the four rotation parameters

$$
\mathbf{n}_x, \mathbf{n}_y, \mathbf{n}_z, \phi
$$

which are linked by one constraint (figure A.4):

$$
\| \mathbf{n} \| = \sqrt{n_x^2 + n_y^2 + n_z^2} = 1 \quad \phi \in [0, \pi]
$$
Figure A.4. A unique rotation about an arbitrary axis.

The rotation operator can be obtained by making use of three phases:

a) Use 2 elementary rotations (figure A.5)

\[ R_x(-\alpha) \quad \text{and} \quad R_y(+\beta) \]  

combined in one rotation matrix

\[ C = R_x(-\alpha) R_y(+\beta) \]  

with the effect of bringing the n-axis in coincidence with the Ox-axis.

b) Perform a rotation \( \phi \) about the Ox axis (is the virtual n-axis).

c) Bring the n-axis into its original position by the inverse transformation \( C^{-1} = C^T \).

Figure A.5. The n-axis transformed in the direction of the Ox-axis.

These three phases result in the following rotation operator

\[ R_x(\phi) = C R_y(\phi) C^T \]  

with the matrix \( C \) resulting from the plane rotation operators similar to (A.3) and (A.4):

\[ C = \begin{bmatrix} \cos\alpha \cos\beta & \sin\alpha & \cos\alpha \sin\beta \\ -\sin\alpha \cos\beta & \cos\alpha & -\sin\alpha \sin\beta \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \]
Further relations to simplify the matrix $\mathbb{C}$ can be derived from a geometrical point of view (figure A.5):

\[
\begin{align*}
\sin \beta &= n_z \\
\cos \beta &= \sqrt{n_x^2 + n_y^2} \\
\sin \alpha &= \frac{n_y}{\sqrt{n_x^2 + n_y^2}} \\
\cos \alpha &= \frac{n_x}{\sqrt{n_x^2 + n_y^2}}
\end{align*}
\]  \hspace{1cm} (A.15)

Combination of relations (A.13) with (A.14), (A.15) and (A.5) results in the direct relation:

\[
R = \begin{bmatrix}
n_x^2(1 - \cos \phi) + \cos \phi & n_xn_z(1 - \cos \phi) - n_z \sin \phi & n_xn_y(1 - \cos \phi) + n_y \sin \phi \\
n_zn_x(1 - \cos \phi) + n_x \sin \phi & n_y^2(1 - \cos \phi) + \cos \phi & n_yn_z(1 - \cos \phi) - n_x \sin \phi \\
n_zn_y(1 - \cos \phi) - n_y \sin \phi & n_yn_z(1 - \cos \phi) + n_x \sin \phi & n_z^2(1 - \cos \phi) + \cos \phi
\end{bmatrix}
\]  \hspace{1cm} (A.16)

Singularities do not occur with this formalism. An alternative derivation of this formalism is given by Sauren (1990,1.11-1.13).

The next formalism makes use of the Cayley form of an orthogonal matrix, which is explained in the following intermezzo.

---

**INTERMEZZO:**

In the beginning of this appendix is mentioned that a pure rotation can be denoted as

\[
\mathbf{x}' = R \mathbf{x}
\]  \hspace{1cm} (A.17)

The rotation operator $R$ is an orthogonal matrix, which states that the length of the column vector during the spherical motion must be preserved:

\[
\mathbf{x}'^T \mathbf{x}' - \mathbf{x}^T \mathbf{x} = (\mathbf{x}' + \mathbf{y})^T (\mathbf{x}' - \mathbf{x}) = 0
\]  \hspace{1cm} (A.18)

Introducing the columns

\[
\mathbf{f} = \mathbf{x}' - \mathbf{x} = (R - I) \mathbf{x} \\
\mathbf{g} = \mathbf{x}' + \mathbf{x} = (R + I) \mathbf{x}
\]  \hspace{1cm} (A.19)

and combining them with (A.18) leads to

\[
\mathbf{g}^T \mathbf{f} = \mathbf{g}^T \mathbf{B} \mathbf{g} = 0
\]  \hspace{1cm} (A.20)
in which $\mathbf{B}$ is the skew-symmetric matrix

$$
\mathbf{B} = \mathbf{\hat{b}} = (\mathbf{R} - \mathbf{I})(\mathbf{R} + \mathbf{I})^{-1} = 
\begin{bmatrix}
0 & -b_3 & b_2 \\
-b_3 & 0 & -b_1 \\
b_2 & b_1 & 0 \\
\end{bmatrix}
$$

(A.21)

Transforming (A.21) leads to an expression for the rotation operator in terms of the column $\mathbf{b}$ (Cardona, 1989, 32):

$$
\mathbf{R} = \mathbf{I} + \frac{2}{1 + \|\mathbf{b}\|^2}(\mathbf{b} + \mathbf{b}^T)
$$

(A.22)

The column $\mathbf{b}$ consists of three independent parameters $b_1, b_2, b_3$ that describe the finite rotation in three-dimensional kinematics.

**Rodrigues parameters**

The set of three independent parameters $b_1, b_2, b_3$ is usually called the Rodrigues parameters. The direct relation is of course defined by relation (A.22). The geometrical meaning of these parameters can be achieved after comparing relation (A.22) with relation (A.16) (Cardona, 1989, 33):

$$
b_1 = n_x \tan \frac{\phi}{2} \\
b_2 = n_y \tan \frac{\phi}{2} \\
b_3 = n_z \tan \frac{\phi}{2}
$$

(A.23)

A singularity occurs if the rotation angle $\phi$ becomes $\pm \pi$.

**Euler parameters**

The Euler parameters $e_0$ and $e_i$ are introduced by an operation on the Rodrigues parameters:

$$
b_i = \frac{e_i}{e_0} \quad i = 1, 2, 3
$$

(A.24)

with $e_0$ defined such that the following constraint is observed:

$$
e_0^2 + e_1^2 + e_2^2 + e_3^2 = 1
$$

(A.25)
Appendix A: Different formalisms to describe finite rotations

The geometrical meaning of the Euler parameters may be derived by combining relations (A.23), (A.24) and (A.25):

\[ e_0 = \cos \frac{\theta}{2}, \quad e_1 = n_x \sin \frac{\phi}{2}, \quad e_2 = n_y \sin \frac{\phi}{2}, \quad e_3 = n_z \sin \frac{\phi}{2} \tag{A.26} \]

A simple form of the direct relation can be achieved by combining relation (A.26) with relation (A.16):

\[ R = (2c_0^2 - 1)I + 2(ee^T + e_0\bar{e}) \tag{A.27} \]

with \( \bar{e} \) defined conform to \( \bar{b} \) in relation (A.21).

**Conformal rotation vector (CRV)**

The parameters of the Conformal rotation vector (CRV) are obtained by a conformal transformation on the Euler parameters:

\[ c_i = \frac{4c_i}{1 + e_0} \quad i = 0,1,2,3 \tag{A.28} \]

which produces a column \( \mathbf{c} \) containing a set of three independent parameters:

\[ c_1 = 4n_x \tan \frac{\phi}{4}, \quad c_2 = 4n_y \tan \frac{\phi}{4}, \quad c_3 = 4n_z \tan \frac{\phi}{4} \tag{A.29} \]

with the fourth parameter given by

\[ c_0 = \frac{1}{8}[16 - cc^T] = \frac{1}{8}[16 - ee^T] \tag{A.30} \]

The CRV is in fact defined by the three independent parameters. The fourth parameter is just a dummy-parameter. The derivation of the direct relation is a very complex one (Cardona, 1989, 35) eventually leading to the following expression

\[ R = F^2 \tag{A.31} \]

with the matrix \( F \) defined as

\[ F = \frac{1}{(4 - c_0)}[c_0I + \frac{1}{2}ee^T + \bar{e}] \tag{A.32} \]
The rotation vector

First, the Rodrigues parameters (A.23) are introduced in relation (A.22), which leads to the following form of the rotation operator:

$$R = 1 + \frac{2}{1 + \tan^2 \frac{\phi}{2}} \left( \tan \phi \frac{\mathbf{n}}{2} + \tan^2 \frac{\phi}{2} \mathbf{n} \mathbf{n} \right)$$

(A.33)

Second, the rotation vector is defined as

$$\mathbf{\Phi} = \mathbf{n} \phi$$

(A.34)

It consists of the rotation components $\phi_x, \phi_y$ and $\phi_z$ about the rotation direction $\mathbf{n}$. With this definition the rotation operator can be written in the form

$$R = 1 + \frac{\sin \phi}{\phi} \mathbf{\Phi} + \frac{1}{2} \left( \frac{\sin^2 \phi}{\phi^2} \mathbf{\Phi} \mathbf{\Phi} \right)$$

(A.35)

with $\mathbf{\Phi}$ defined conform to $\mathbf{n}$ in relation (A.21). After a complex derivation (Cardona, 1989, 36) with some algebraic steps and the use of power series the following simple direct relation arises:

$$R = \exp \mathbf{\Phi}$$

(A.36)

in which the exponential-function is defined as

$$\exp \mathbf{\Phi} = 1 + \mathbf{\Phi} + \frac{1}{2!} \mathbf{\Phi}^2 + \ldots.$$  

(A.37)

This formalism has no singularities.

Linear parameters

Starting from relation (A.35) the following expression arises after some algebraic steps:

$$R = \cos^2 \phi I + \frac{\sin^2 \phi}{1 + \cos \phi} \frac{\mathbf{\Phi} \mathbf{\Phi}^T}{\phi^2} + \sin \phi \frac{\mathbf{\Phi}}{\phi}$$

(A.38)

The linear parameters are defined as

$$s_0 = \cos \phi, \quad s_1 = n_x \sin \phi, \quad s_2 = n_y \sin \phi, \quad s_3 = n_z \sin \phi$$

(A.39)

The four parameters are constrained by

$$s_0^2 + s_1^2 + s_2^2 + s_3^2 = 1$$

(A.40)
Appendix A: Different formalisms to describe finite rotations

With these definitions the direct relation can be written as

\[ \mathbf{R} = \mathbf{s}_0 \mathbf{1} + \frac{1}{1 + \mathbf{s}_0} \mathbf{s} \mathbf{s}^T + \hat{\mathbf{s}} \]  

(A.41)

with \( \mathbf{s} = [s_1, s_2, s_3]^T \) and \( \hat{\mathbf{s}} \) again defined as the skew-symmetric matrix.

A singularity occurs at \( \phi = \pm \pi \).

When evaluating the (dis)advantages of the different formalisms, a good overview arises. With relation to the points of singularities, just the interval \(-\pi, \pi\) is considered:

- The Euler and Bryant angles are only applicable for specific problems. They have a point of singularity in the interval \(-\pi, \pi\).
- Rodrigues and linear parameters have a point of singularity at \( \pm \pi \).

The remaining formalisms (Euler parameters, CRV and rotation vector) have the advantage of having no point of singularity in the significant interval.

- A benefit of the CRV and rotation vector is the minimal set of parameters (three).
- An advantage of the Euler parameters and the CRV is the possibility of using quaternion algebra (Cardona, 1989, 41-46). Using this algebraic method leads to a more simple formulation of the formalisms.
- The rotation vector has the privilege of an easy geometric interpretation.
Appendix B: Implementation of constraints in the finite element approach

This appendix describes the implementation of the holonomic and non-holonomic constraints in the finite element approach for multibody systems. Before adding the constraints to the dynamic problem first the unconstrained dynamic problem is recalled, which can be formulated by means of the Lagrange equations:

\[ \frac{d}{dt} \left( \frac{\partial L}{\partial \dot{q}_k} \right) - \frac{\partial L}{\partial q_k} = Q_k \quad k = 1, \ldots, n \]  

(B.1)

with \( L \) the Lagrangian of the system defined as the difference between the kinetic and the potential energy of the system and with \( Q_k \) the generalized non-conservative forces that are the forces that cannot be derived from a potential.

The constraints are added to the unconstrained problem by means of an augmented Lagrangian approach. When employing this augmented approach, the constraint forces take a contribution from a scaling factor and an additional penalty factor. The approach also introduces a column matrix with Lagrange multipliers, denoted by \( \lambda \). Of course the holonomic and non-holonomic constraints must be treated in a different manner.

* The holonomic constraints are defined as

\[ \phi(q, t) = 0 \]  

(B.2)

These constraints are added to the Lagrangian of the system in order to obtain the augmented functional

\[ L^* = L + k \lambda^T \phi - \frac{1}{2} p \phi^T \phi \]  

(B.3)

with \( k \) the scale factor and \( p \) the penalty factor. The extended equations of motion can be obtained in the form

\[ \frac{d}{dt} \left( \frac{\partial L^*}{\partial \dot{q}_k^*} \right) - \frac{\partial L^*}{\partial q_k^*} = Q_k^* + (k \lambda_i - p \phi_i^*) \frac{\partial \phi_i}{\partial q_k^*} \quad k = 1, \ldots, n \quad i = 1, \ldots, m \]  

(B.4)

with the constraints scaled by factor \( k \). Comparison of these equations of motion with relation (B.1) implies the definition of the generalized constraint forces \( Q_k^* \) that require the system to satisfy the imposed constraints:

\[ Q_k^* = (k \lambda_i - p \phi_i^*) \frac{\partial \phi_i}{\partial q_k^*} \]  

(B.5)

These constraint forces have to be implemented in the finite element context. Therefore, each joint is treated as a separate matrix entity, which adds relations between its local degrees of freedom to the dynamic system. These relations correspond to the internal forces vector and the tangent stiffness matrix according to the finite element terminology. The internal forces vector is
obtained by comparing relation (B.4) with (B.1) and can be written as

\[ G_{\text{int}} = \begin{bmatrix} B(p\phi - k\dot{\lambda}) \\ k\phi \end{bmatrix} \quad \text{with: } B_{ii} = \frac{\partial\phi_i}{\partial\dot{q}_i^*} \]  

(B.6)

And the tangent stiffness matrix derived from the internal forces vector:

\[ K_i = \frac{\partial G_{\text{int}}}{\partial q_i^*} = \begin{bmatrix} pBB^T & -kB^T \\ -kB & 0 \end{bmatrix} \quad \text{with: } q_i^* = \begin{bmatrix} q_i^* \\ \dot{\lambda} \end{bmatrix} \]  

(B.7)

• The non-holonomic constraints considered here are just of the bilateral type:

\[ g(q,\dot{q},t) = 0 \]  

(B.8)

They can be treated in similar way as the holonomic constraints. The effect of these constraints on the system behaviour is taken into account by adding to the Lagrangian a dissipation function according to the augmented Lagrangian approach.

\[ D = \frac{1}{2}p\dot{\sigma}\dot{\sigma} - k\dot{\lambda}\dot{\sigma} \]  

(B.9)

The extended motion equations can be obtained in the form

\[ \frac{d}{dt} \left( \frac{\partial L}{\partial \dot{q}_k} \right) - \frac{\partial L}{\partial q_k} = Q_k + (k\dot{\lambda}_i - p\sigma_i) \frac{\partial \sigma_i}{\partial \dot{q}_k} \quad k = 1, \ldots, n \quad i = 1, \ldots, n_c \]  

with the generalized constraint forces defined similar to the holonomic case:

\[ Q_k^* = (k\dot{\lambda}_i - p\sigma_i) \frac{\partial \sigma_i}{\partial \dot{q}_k} \]  

(B.11)

According to the finite element terminology, the internal forces vector is written as

\[ G_{\text{int}} = \begin{bmatrix} B_{\text{sh}}(p\sigma - k\dot{\lambda}) \\ k\sigma \end{bmatrix} \quad \text{with: } B_{\text{sh}ki} = \frac{\partial \sigma_i}{\partial \dot{q}_k^*} \]  

(B.12)

And the tangent damping matrix derived from the internal forces vector in a way similar to the tangent stiffness matrix:

\[ C_i = \frac{\partial G_{\text{int}}}{\partial \dot{q}_i} = \begin{bmatrix} pB_{\text{sh}B}^T & -kB_{\text{sh}} \\ -kB_{\text{sh}}^T & 0 \end{bmatrix} \quad \text{with: } \dot{q}_i^* = \begin{bmatrix} \dot{q}_i^* \\ \dot{\lambda} \end{bmatrix} \]  

(B.13)

The relations (B.6), (B.7), (B.12) and (B.13) are the relations that define the contribution of the constraints to the dynamical problem in the context of the finite element approach.
Appendix C: Explicit and implicit time integration schemes

Time integration schemes are used to solve non-linear differential equations (DAE), which can be represented in the following form

\[ M\ddot{q} + \dot{G}^{\text{int}}(q,\dot{q},t) = G^{\text{ext}}(q, t) \]  \hspace{1cm} (C.1)

with \( M \) the constant mass matrix of the system, \( q \) a column matrix containing the displacements, \( \dot{G}^{\text{int}} \) a column matrix containing the internal forces, which are a function of the displacements, the velocities and time, and \( G^{\text{ext}} \) the column matrix containing the external forces, which are a function of the displacements and time.

Basically, there are two types of integration methods for solving the equations (C.1), namely explicit and implicit methods. A method is called explicit if the displacements and the velocities just depend on quantities of a previous point in time. On the other hand an implicit method also needs quantities of the new point in time to calculate the displacements and velocities. Summarized in formula form it holds

explicit: \[ q_{n+1} = f(q_n, \dot{q}_n, \ddot{q}_n) \] \hspace{1cm} (C.2)
\[ q_{n+1} = g(q_n, \dot{q}_n, \ddot{q}_n) \]

implicit: \[ q_{n+1} = f(q_n, \dot{q}_n, \ddot{q}_n, \dddot{q}_n) \] \hspace{1cm} (C.3)
\[ q_{n+1} = g(q_n, \dot{q}_n, \ddot{q}_n, \dddot{q}_n, \dddot{q}_n) \]

There is a main difference in the stability of the two methods. The explicit method is only stable if the time integration step is lower than a particular value, which depends on the method. This means

\[ \Delta t \leq \Delta t_{\text{max}} \]  \hspace{1cm} (C.4)

On the contrary the implicit method is unconditionally stable with respect to the size of the time step. The size just depends on the desired accuracy.

An example of explicit integration methods is the central-difference method (Bruijs,1990,25)

\[ \dot{q}_{n+1/2} = \dot{q}_{n-1/2} + \Delta t \ddot{q}_n \] \hspace{1cm} (C.5)
\[ q_{n+1} = q_n + \Delta t \dot{q}_{n+1/2} \] \hspace{1cm} (C.6)

According to the explicit method, the positions and velocities are calculated from quantities at previous time only. The equations (C.1) are considered at time \( n \Delta t \)

\[ M\ddot{q}_n + \dot{G}^{\text{int}}_n = G^{\text{ext}}_n \] \hspace{1cm} (C.7)
The positions are known up to and including time $n \Delta t$ and the velocities up to and including time $(n-1/2) \Delta t$. Next, relation (C.7) may be used to calculate the accelerations at time $n \Delta t$. Relations (C.5) and (C.6) may be used to calculate the positions and velocities at time $(n+1) \Delta t$ and $(n+1/2) \Delta t$, respectively. Then relation (C.1) is considered at time $(n+1) \Delta t$ and used to calculate the acceleration at that time, etc. For starting the computation, the positions and velocities must be known at time zero. The velocities at time $(-1/2) \Delta t$ have to be known, there are a few possible methods for computing them:

\[
\ddot{q}_{-1/2} = \ddot{q}_0 - \frac{1}{2} \Delta t \ddot{q}_0
\]  

or just

\[
\ddot{q}_{-1/2} = \ddot{q}_0
\]  

An example of implicit integration methods is the Newmark method that is explained in section 2.2.4 of this report.
Appendix D: Overview of the Samcef modules

The finite element package Samcef consists of an extensive list of modules, which all have their own applications. The modules can be divided in groups by way of their function in the package. The functions of the modules can be classified by:

- Pre-processing
- Analysis
- Optimization of shape and dimension
- Post-processing and display of results

### Pre-processing

BACON Graphic interactive pre- and post-processing module. Modelling, meshing, generation of data, display of data are possible operations.

### Analysis

#### Linear Static Analyses

- **ASEF** Linear static analysis of an elastic or thermal linear problem;
- **FOURIER** Linear static analysis of a problem developed by Fourier series;
- **STABI** Modal analysis of bifurcation of elastic equilibrium;
- **HELIOS** Linear static analysis of a cyclic periodic structure.

#### Non-linear Static Analyses

- **SAMNL** Non-linear static analysis;
- **STABI** Incremental stability;
- **VISCO** Cyclic visco-plasticity.

#### Linear Dynamic Analyses

- **DYNAM** Modal analysis of an elastic problem;
- **REPDYN** Dynamic response of a linear elastic problem with damping either by modal superimposition, or direct integration; Spectral response.

#### Non-linear Dynamic Analyses

- **NLDYN** Non-linear dynamic analysis.
Appendix D: Overview of the Samcef modules

- Specific Dynamic Analyses
  - ROTOR and ROTORT
  - MECANO
  - CABLE

  Stability analysis, calculation of critical speeds and stationary or transient response of structures containing rotating parts;
  Program for the analysis of the kinematics and dynamics of 3-D flexible mechanisms;
  Analysis of the non-linear behaviour of cable and beam networks in static and dynamic modes subjected to various loadings: short-circuit, electromagnetic effect, wind etc.

- Linear and Non-linear Thermal Analyses
  - THERNL
  - AMARYLLIS

  Stationary and transient analysis of non-linear thermal problems (heat transfer);
  Analysis of thermal response of 2-D and axisymmetrical media with physical characteristics which vary through phase changes or chemical reactions (ablation phenomenon).

- Electrical Analysis
  - ISABEL

  Calculation of electrical field, incompressible fluid flow.

- Design Utility
  - OPTI

  Sizing and shape optimization.

- Post-Processing and Display of Results
  - BACON
  - POSTFAC

  Graphic interactive pre- and post-processing module;
  Ensures a range of post-processors using output from storage files.
Appendix E: Accuracy of the first-order forward difference scheme

The first-order forward difference scheme is defined as

\[
\frac{\partial f(x_i)}{\partial x_i}(x_{i_0}) = \frac{f(x_i + \Delta x_i) - f(x_i)}{\Delta x_i}
\]  
(E.1)

with in this report \(\Delta x_i\) computed relative to the corresponding design variable:

\[
\Delta x_i = \alpha x_i
\]  
(E.2)

The function \(f\) in (E.1) represents the objective function or a constraint. In the following of this appendix the subscript \(i\) is omitted, which means that the number of design variables is one. Using the first-order forward difference scheme for computing the approximate derivatives introduces two sources of errors: truncation and condition errors (Haftka and Gürdal, 1992, 256) (Iott, 1985, 2-3).

The truncation error results from the neglected terms in the Taylor series expansion of the varied design. The Taylor series expansion of \(f(x + \Delta x)\) can be written as

\[
f(x + \Delta x) = f(x) + \Delta x \frac{df}{dx}(x) + \frac{(\Delta x)^2}{2} \frac{d^2f}{dx^2}(x + \zeta \Delta x) \quad 0 \leq \zeta \leq 1
\]  
(E.3)

From this equation it follows that the size of the truncation error is given by

\[
e_T(\Delta x) = \frac{\Delta x}{2} \frac{d^2f}{dx^2}(x + \zeta \Delta x) \quad 0 \leq \zeta \leq 1
\]  
(E.4)

Determining the value of the error from this relation is handled further on.

The condition error exists because of the difference in the numerical evaluation of the function and its precise value. One contribution is the round-off error in calculating \(df/dx\) from the original and perturbed values of \(f\). Normally, this contribution is very small, because most computers are very accurate. In the case of Mecopt there is a chance that this contribution is not very small, since Mecopt gets these values from Mecano in just six significant digits. The condition error can be estimated as

\[
e_C(\Delta x) = \frac{2}{\Delta x} \varepsilon_f
\]  
(E.5)

in which \(\varepsilon_f\) is the absolute error in the function \(f\).

Computation of the total error is not as simple as it seems, because determining the truncation error is very difficult. That is why the following experimental method is used. The derivative of the function \(f\), representing the objective function or a constraint, will be computed for a range of step sizes. When plotting the values of the derivatives as a function of the step size, a value for the step size can be chosen.

The method described here is carried out for the impact-absorber problem of section 4.2. The derivatives of the objective function and five equally distributed time point constraints are considered. The derivatives with respect to both design variables are computed for values of \(\alpha\) in a range of \(10^{-5}\) to \(10^{-1}\). This procedure is carried out for four designs, namely: \(x = (1.0,1.0), (0.5,0.5), (1.0,0.5)\) and \((0.5,1.0)\). The results of these experiments are presented in figure E.1. The
values of the derivatives are all scaled to the value for \( \alpha \) is \( 10^{-5} \). The plots show that most of the derivatives remain constant for \( \alpha \)-values of \( 10^{-4} \) to \( 10^{-2} \). It is obvious that the best \( \alpha \)-value for the shock-absorber problem is about \( 10^{-3} \).

Figure E.1. The derivatives of the objective function and constraints with respect to both design variables as a function of the parameter \( \alpha \).
Appendix F: Results of three optimization examples

This appendix contains the results of all examples that are treated in chapter 4. Every analysis is presented by a table containing several characteristic results of the optimization process and by some plots that present the same results graphically.

The 1st column in the table contains the number of the optimization cycle. The 2nd column contains the value of the move limit that is used in that cycle. A negative move limit indicates that one or more previous designs were rejected (see section 3.2.3.). The 3th column displays a flag that is 1 if there is at least one move limit in one design direction active and which is 0 if no move limit is active. The 4th and 5th column contain the objective function value and the approximated objective function value, respectively. Columns 6 up to and including 9 display the behaviour of the constraints during the optimization process. Due to the use of NAG-routine E04MBF, Mecopt always requires a lower and an upper bound for each constraint. However, for the display of the constraint behaviour in columns 6-9 these two-sided constraints are transformed into two one-sided constraints. Second, all constraints are scaled to a bound one, also the approximated constraints are treated this way. Then, the relative error in the approximations can be computed as

$$\max(Gerr) = \max \left( \frac{g_i - \bar{g}_i}{\bar{g}_i} \right) \quad i = 1, \ldots, nc$$

in which $g_i$ is the constraint value, $\bar{g}_i$ the approximated constraint value and nc the number of constraints. The maximum error of all one-sided constraints is displayed in column 9. Column 8 displays the number of the constraint to which the value in column 9 belongs. A negative constraint number denotes the lower bound and a positive number the upper bound of the two-sided constraint. Finally, the one-sided real constraint values are scaled to a bound zero. Now, they represent a constraint defined according to the standard formulation of equation (3.5). The maximum constraint value of these scaled constraints is presented in column 7, with column 6 defined similar with column 8. The value of column 7 is a measure for the relative deviation to the bound for the most critical constraint. The line(s) above the table display the active constraints of the linearized model in the optimal design.
Appendix F: Results of three optimization problems

F.1. The impact-absorber with time point constraints

The approximated constraint 10 is active at its upper bound

<table>
<thead>
<tr>
<th>L</th>
<th>ML</th>
<th>MAct</th>
<th>F</th>
<th>Fben</th>
<th>no max(G)</th>
<th>no max(Gerr)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.000E+00</td>
<td>0</td>
<td>.980E+00</td>
<td>.000E+00</td>
<td>9  -.279E+00</td>
<td>0  .000E+00</td>
</tr>
<tr>
<td>2</td>
<td>.300E+00</td>
<td>1</td>
<td>.693E+00</td>
<td>.696E+00</td>
<td>11 -.407E-01</td>
<td>21 .177E+00</td>
</tr>
<tr>
<td>3</td>
<td>.300E+00</td>
<td>1</td>
<td>.577E+00</td>
<td>.577E+00</td>
<td>10 -.261E-02</td>
<td>21 .240E-01</td>
</tr>
<tr>
<td>4</td>
<td>.300E+00</td>
<td>1</td>
<td>.527E+00</td>
<td>.504E+00</td>
<td>10 -.620E-03</td>
<td>18 .785E-02</td>
</tr>
<tr>
<td>5</td>
<td>-.500E-01</td>
<td>1</td>
<td>.526E+00</td>
<td>.525E+00</td>
<td>10 -.400E-04</td>
<td>18 .381E-03</td>
</tr>
<tr>
<td>6</td>
<td>-.300E-01</td>
<td>1</td>
<td>.526E+00</td>
<td>.526E+00</td>
<td>10 -.600E-04</td>
<td>18 .204E-03</td>
</tr>
</tbody>
</table>

The optimum design is obtained by decreasing the move limit when the new design provides an increase in objective function. From the values of the 7th and 9th column it can be concluded that the constraints are approximated quite exact in the optimum. From the line above the table it can be concluded that one linearized time point constraint is active in the optimum.
Appendix F: Results of three optimization problems

F.2. The impact-absorber with an integral constraint

From the 7th column can be concluded that, during the first two optimization cycles there is no warning that the design will get infeasible in the third cycle. This drawback (see section 3.3.1.) of the integral constraint leads to a large constraint violation in the third cycle. This behaviour results in an unusual development of the design, which is displayed in the upper-left plot.
F.3. The impact-absorber with a sum constraint

case 1: \( \rho = 10 \)

<table>
<thead>
<tr>
<th>L</th>
<th>ML</th>
<th>MLaet</th>
<th>F</th>
<th>Fben</th>
<th>no max(G)</th>
<th>no max(Gerr)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.000E+00</td>
<td>0</td>
<td>.980E+00</td>
<td>.000E+00</td>
<td>1 -.753E-01</td>
<td>0 .000E+00</td>
</tr>
<tr>
<td>2</td>
<td>.300E+00</td>
<td>1</td>
<td>.701E+00</td>
<td>.703E+00</td>
<td>1 .192E-01</td>
<td>1 .188E-01</td>
</tr>
<tr>
<td>3</td>
<td>.300E+00</td>
<td>1</td>
<td>.632E+00</td>
<td>.613E+00</td>
<td>1 .509E-02</td>
<td>1 .506E-02</td>
</tr>
<tr>
<td>4</td>
<td>-.100E+00</td>
<td>1</td>
<td>.630E+00</td>
<td>.623E+00</td>
<td>1 .678E-03</td>
<td>1 .677E-03</td>
</tr>
<tr>
<td>5</td>
<td>-.200E-01</td>
<td>1</td>
<td>.631E+00</td>
<td>.630E+00</td>
<td>1 .326E-04</td>
<td>1 .326E-04</td>
</tr>
<tr>
<td>6</td>
<td>.200E-01</td>
<td>1</td>
<td>.631E+00</td>
<td>.630E+00</td>
<td>1 .277E-04</td>
<td>1 .277E-04</td>
</tr>
</tbody>
</table>

![Design space diagram](image1)

![Design variable diagram](image2)

![Sum constraint diagram](image3)

![Objective function diagram](image4)
Appendix F: Results of three optimization problems

case 2: \( p = 100 \)

The approximated constraint \( L \) is active at its upper bound

<table>
<thead>
<tr>
<th>( L )</th>
<th>( ML )</th>
<th>( ML_{\text{act}} )</th>
<th>( F )</th>
<th>( F_{\text{ben}} )</th>
<th>no max((G))</th>
<th>no max((G_{\text{err}}))</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( 0.000E+00 )</td>
<td>( 0.980E+00 )</td>
<td>( 0.000E+00 )</td>
<td>( 1.270E+00 )</td>
<td>( 0.000E+00 )</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>( 3.000E+00 )</td>
<td>( 0.693E+00 )</td>
<td>( 0.696E+00 )</td>
<td>( 1.322E+00 )</td>
<td>( 1.647E-01 )</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>( 3.000E+00 )</td>
<td>( 0.585E+00 )</td>
<td>( 0.586E+00 )</td>
<td>( 1.371E-02 )</td>
<td>( 1.370E-02 )</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>( 3.000E+00 )</td>
<td>( 0.528E+00 )</td>
<td>( 0.527E+00 )</td>
<td>( 1.381E-02 )</td>
<td>( 1.379E-02 )</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>( -1.000E+00 )</td>
<td>( 0.531E+00 )</td>
<td>( 0.526E+00 )</td>
<td>( 1.595E-03 )</td>
<td>( 1.594E-03 )</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>( 1.000E+00 )</td>
<td>( 0.531E+00 )</td>
<td>( 0.525E+00 )</td>
<td>( 1.644E-03 )</td>
<td>( 1.644E-03 )</td>
<td></td>
</tr>
</tbody>
</table>

\[
\begin{array}{c|c|c|c|c|c|c}
\text{L} & \text{x(1)} & \text{x(2)} & \text{x1(\text{+})} & \text{x2(\text{o})} & \text{max}(G) & \text{max}(G_{\text{err}}) \\
\hline
1 & 1.000E+01 & 3.000E+00 & 0.000E+00 & 0.000E+00 & 0.000E+00 & 0.000E+00 \\
2 & 7.000E+00 & 2.100E+00 & 0.000E+00 & 0.000E+00 & 0.000E+00 & 0.000E+00 \\
3 & 5.945E+00 & 2.730E+00 & 0.000E+00 & 0.000E+00 & 0.000E+00 & 0.000E+00 \\
4 & 4.997E+00 & 3.549E+00 & 0.000E+00 & 0.000E+00 & 0.000E+00 & 0.000E+00 \\
5 & 4.693E+00 & 3.904E+00 & 0.000E+00 & 0.000E+00 & 0.000E+00 & 0.000E+00 \\
6 & 5.071E+00 & 3.514E+00 & 0.000E+00 & 0.000E+00 & 0.000E+00 & 0.000E+00 \\
\end{array}
\]
Appendix F: Results of three optimization problems

F.4. The impact-absorber with critical time point constraints

The approximated constraint \( l \) is active at its upper bound

<table>
<thead>
<tr>
<th>L</th>
<th>ML</th>
<th>MLact</th>
<th>F</th>
<th>Fhen</th>
<th>no</th>
<th>max(G)</th>
<th>no max(Gerr)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4.00E+00</td>
<td>0</td>
<td>0.98E+00</td>
<td>0.00E+00</td>
<td>1</td>
<td>-4.27E+00</td>
<td>0.00E+00</td>
</tr>
<tr>
<td>2</td>
<td>3.30E+00</td>
<td>1</td>
<td>0.69E+00</td>
<td>0.69E+00</td>
<td>1</td>
<td>-4.07E+01</td>
<td>6.62E-01</td>
</tr>
<tr>
<td>3</td>
<td>3.00E+00</td>
<td>1</td>
<td>0.57E+00</td>
<td>0.57E+00</td>
<td>1</td>
<td>-4.13E-02</td>
<td>1.12E+00</td>
</tr>
<tr>
<td>4</td>
<td>3.00E+00</td>
<td>1</td>
<td>0.52E+00</td>
<td>0.52E+00</td>
<td>1</td>
<td>-2.78E-02</td>
<td>2.77E-02</td>
</tr>
<tr>
<td>5</td>
<td>-1.00E+00</td>
<td>1</td>
<td>0.52E+00</td>
<td>0.52E+00</td>
<td>1</td>
<td>-3.20E-03</td>
<td>3.20E-03</td>
</tr>
<tr>
<td>6</td>
<td>.50E+00</td>
<td>1</td>
<td>0.52E+00</td>
<td>0.52E+00</td>
<td>1</td>
<td>-1.17E-02</td>
<td>1.17E-02</td>
</tr>
<tr>
<td>7</td>
<td>.50E-01</td>
<td>1</td>
<td>0.52E+00</td>
<td>0.52E+00</td>
<td>1</td>
<td>-2.70E-03</td>
<td>2.70E-03</td>
</tr>
</tbody>
</table>

In the third cycle of the optimization process the maximum error in the approximation of the second constraint is about 11.2%. This high value is not due to a bad approximation, but to the increase of the number of constraints during the optimization process. In cycles 1 and 2 there is just one constraint, while in cycle 3 a second critical time point constraint appears that is not approximated before. So, in fact the error value of that constraint has no meaning.
Appendix F: Results of three optimization problems

F.5. The impact-absorber with special treatment of the objective function

case 1: feasible starting point, \( x_0(3) = 1.0 \)

The approximated constraint 10 is active at its upper bound
The approximated constraint 77 is active at its upper bound
The approximated constraint 78 is active at its upper bound

<table>
<thead>
<tr>
<th>L</th>
<th>ML</th>
<th>MLact</th>
<th>F</th>
<th>Fben</th>
<th>no max(G)</th>
<th>no max(Gerr)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.000E+00</td>
<td>0</td>
<td>.100E+01</td>
<td>.000E+00</td>
<td>54</td>
<td>-.198E-01</td>
</tr>
<tr>
<td>2</td>
<td>.300E+00</td>
<td>1</td>
<td>.700E+00</td>
<td>.700E+00</td>
<td>54</td>
<td>-.293E-02</td>
</tr>
<tr>
<td>3</td>
<td>.300E+00</td>
<td>1</td>
<td>.577E+00</td>
<td>.577E+00</td>
<td>10</td>
<td>.311E-02</td>
</tr>
<tr>
<td>4</td>
<td>.300E+00</td>
<td>1</td>
<td>.529E+00</td>
<td>.529E+00</td>
<td>73</td>
<td>-.658E-03</td>
</tr>
<tr>
<td>5</td>
<td>.300E+00</td>
<td>0</td>
<td>.526E+00</td>
<td>.526E+00</td>
<td>10</td>
<td>.300E-04</td>
</tr>
<tr>
<td>6</td>
<td>.300E+00</td>
<td>0</td>
<td>.526E+00</td>
<td>.526E+00</td>
<td>78</td>
<td>.000E+00</td>
</tr>
</tbody>
</table>

It can be seen from the third column that during the last two optimization cycles the move limit is no longer active, which indicates that the optimum has been found in a vertex of the constraints. This can also be concluded from the move limit in column 2, since it appears to be unnecessary to decrease the move limit during the optimization process. The lines above the table give the linearized time point constraints which create the vertex.
case 2: active starting point, $x_0(3) = 0.980$

The approximated constraint 10 is active at its upper bound
The approximated constraint 77 is active at its upper bound
The approximated constraint 78 is active at its upper bound

<table>
<thead>
<tr>
<th>L</th>
<th>M̅L</th>
<th>M̄Lact</th>
<th>F</th>
<th>F̄ben</th>
<th>no max(G)</th>
<th>no max(Gerr)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.000E+00</td>
<td>0</td>
<td>.980E+00</td>
<td>.000E+00</td>
<td>54</td>
<td>.000E+00</td>
</tr>
<tr>
<td>2</td>
<td>.300E+00</td>
<td>1</td>
<td>.696E+00</td>
<td>.696E+00</td>
<td>54</td>
<td>.301E-02</td>
</tr>
<tr>
<td>3</td>
<td>.300E+00</td>
<td>1</td>
<td>.577E+00</td>
<td>.577E+00</td>
<td>10</td>
<td>.261E-02</td>
</tr>
<tr>
<td>4</td>
<td>.300E+00</td>
<td>1</td>
<td>.528E+00</td>
<td>.528E+00</td>
<td>10</td>
<td>.620E-03</td>
</tr>
<tr>
<td>5</td>
<td>.300E+00</td>
<td>0</td>
<td>.526E+00</td>
<td>.526E+00</td>
<td>10</td>
<td>.300E-04</td>
</tr>
<tr>
<td>6</td>
<td>.300E+00</td>
<td>0</td>
<td>.526E+00</td>
<td>.526E+00</td>
<td>10</td>
<td>.000E+00</td>
</tr>
</tbody>
</table>

L X(1) X(2) X(3) 1.1000E+01 .3000E+00 .9800E+00
2 .7000E+00 .2100E+00 .6961E+00
3 .5759E+00 .2730E+00 .5773E+00
4 .4962E+00 .3549E+00 .5276E+00
5 .4841E+00 .3677E+00 .5262E+00
6 .4841E+00 .3677E+00 .5262E+00

- $x_1(\cdot)$, $x_2(\cdot)$, $x_3(\cdot)$ optimizes design variable
- 51 time point constraints
- $x_3$ = objective function
Appendix F: Results of three optimization problems

Case 3: infeasible starting point, $x_0(3) = 0.1$

The approximated constraint 10 is active at its upper bound
The approximated constraint 77 is active at its upper bound
The approximated constraint 78 is active at its upper bound

<table>
<thead>
<tr>
<th>L</th>
<th>ML</th>
<th>MLact</th>
<th>F</th>
<th>Fben</th>
<th>no max(G)</th>
<th>no max(Gerr)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.000E+00</td>
<td>0</td>
<td>.100E+00</td>
<td>.000E+00</td>
<td>54</td>
<td>.880E+00</td>
</tr>
<tr>
<td>2</td>
<td>.300E+00</td>
<td>1</td>
<td>.691E+00</td>
<td>.566E+00</td>
<td>54</td>
<td>.195E-02</td>
</tr>
<tr>
<td>3</td>
<td>.300E+00</td>
<td>1</td>
<td>.577E+00</td>
<td>.577E+00</td>
<td>10</td>
<td>.261E-02</td>
</tr>
<tr>
<td>4</td>
<td>.300E+00</td>
<td>1</td>
<td>.528E+00</td>
<td>.528E+00</td>
<td>10</td>
<td>.620E-03</td>
</tr>
<tr>
<td>5</td>
<td>.300E+00</td>
<td>0</td>
<td>.526E+00</td>
<td>.526E+00</td>
<td>10</td>
<td>.300E-04</td>
</tr>
<tr>
<td>6</td>
<td>.300E+00</td>
<td>0</td>
<td>.526E+00</td>
<td>.526E+00</td>
<td>10</td>
<td>.000E+00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>L</th>
<th>X(1)</th>
<th>X(2)</th>
<th>X(3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.1000E+01</td>
<td>.3000E+00</td>
<td>.1000E+00</td>
</tr>
<tr>
<td>2</td>
<td>.7000E+00</td>
<td>.2100E+00</td>
<td>.6911E+00</td>
</tr>
<tr>
<td>3</td>
<td>.5759E+00</td>
<td>.2730E+00</td>
<td>.5773E+00</td>
</tr>
<tr>
<td>4</td>
<td>.4962E+00</td>
<td>.3549E+00</td>
<td>.5276E+00</td>
</tr>
<tr>
<td>5</td>
<td>.4841E+00</td>
<td>.3677E+00</td>
<td>.5262E+00</td>
</tr>
<tr>
<td>6</td>
<td>.4841E+00</td>
<td>.3677E+00</td>
<td>.5262E+00</td>
</tr>
</tbody>
</table>

![Graphs showing design space, design variable, 51 time point constraints, and real function vs approximated function.](image)
F.6. The landing gear mechanism

The results obtained by Mecopt are presented below. The design variables are scaled with a factor 0.01 within the optimization process, which can be noticed from the table. When looking at the column of the objective function, one can see that no convergence is obtained with respect to this function. In spite of this, the changes (in fact no changes) in the design variables imply that convergence is already obtained. A measure for the change in the design variables is the value of the move limit, while it is absolute and not relative. From those values one can see that the value of the move limit in the 23rd optimization cycle is relative small compared to the size of the design variables. This phenomenon of convergence in the design variables and not yet in the objective function is due to the fact that a very small change in the design variables has still influence on the objective function, that is to say on the behaviour of the mechanism. However, it is reasonable to assume that the small changes in design that are obtained after the 23rd optimization cycle are not of any importance for the designer. So it is assumed that convergence is obtained after 23 optimization cycles. The small changes in the design variables can also be seen in the first plot on the next page.

The value of the move limit in each iteration cycle is computed by a semi-automatic algorithm. This algorithm is used because of the excessively high computation time that is necessary for gaining an optimal solution for this problem, which is due to the 8 extra analyses necessary for the sensitivity analysis. It is useful to mention that the algorithm is not yet very convenient, but it can be improved.
### Appendix F: Results of three optimization problems

<table>
<thead>
<tr>
<th>L</th>
<th>X(1)</th>
<th>X(2)</th>
<th>X(3)</th>
<th>X(4)</th>
<th>X(5)</th>
<th>X(6)</th>
<th>X(7)</th>
<th>X(8)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.1000E+01</td>
<td>.1100E+01</td>
<td>.1300E+01</td>
<td>.1100E+01</td>
<td>.1600E+01</td>
<td>.1550E+01</td>
<td>.1800E+01</td>
<td>.1000E+01</td>
</tr>
<tr>
<td>2</td>
<td>.1000E+01</td>
<td>.1066E+01</td>
<td>.1266E+01</td>
<td>.1134E+01</td>
<td>.1634E+01</td>
<td>.1584E+01</td>
<td>.1800E+01</td>
<td>.1000E+01</td>
</tr>
<tr>
<td>3</td>
<td>.1000E+01</td>
<td>.1033E+01</td>
<td>.1233E+01</td>
<td>.1168E+01</td>
<td>.1668E+01</td>
<td>.1618E+01</td>
<td>.1800E+01</td>
<td>.1000E+01</td>
</tr>
<tr>
<td>4</td>
<td>.1000E+01</td>
<td>.1000E+01</td>
<td>.1199E+01</td>
<td>.1201E+01</td>
<td>.1701E+01</td>
<td>.1651E+01</td>
<td>.1800E+01</td>
<td>.1000E+01</td>
</tr>
<tr>
<td>5</td>
<td>.1000E+01</td>
<td>.1000E+01</td>
<td>.1249E+01</td>
<td>.1252E+01</td>
<td>.1752E+01</td>
<td>.1702E+01</td>
<td>.1800E+01</td>
<td>.1000E+01</td>
</tr>
<tr>
<td>6</td>
<td>.1000E+01</td>
<td>.1000E+01</td>
<td>.1278E+01</td>
<td>.1280E+01</td>
<td>.1780E+01</td>
<td>.1730E+01</td>
<td>.1800E+01</td>
<td>.1000E+01</td>
</tr>
<tr>
<td>7</td>
<td>.1000E+01</td>
<td>.1000E+01</td>
<td>.1256E+01</td>
<td>.1302E+01</td>
<td>.1802E+01</td>
<td>.1752E+01</td>
<td>.1800E+01</td>
<td>.1000E+01</td>
</tr>
<tr>
<td>8</td>
<td>.1000E+01</td>
<td>.1000E+01</td>
<td>.1273E+01</td>
<td>.1318E+01</td>
<td>.1780E+01</td>
<td>.1736E+01</td>
<td>.1800E+01</td>
<td>.1000E+01</td>
</tr>
<tr>
<td>9</td>
<td>.1000E+01</td>
<td>.1000E+01</td>
<td>.1285E+01</td>
<td>.1330E+01</td>
<td>.1774E+01</td>
<td>.1748E+01</td>
<td>.1800E+01</td>
<td>.1000E+01</td>
</tr>
<tr>
<td>10</td>
<td>.1000E+01</td>
<td>.1000E+01</td>
<td>.1276E+01</td>
<td>.1339E+01</td>
<td>.1765E+01</td>
<td>.1739E+01</td>
<td>.1800E+01</td>
<td>.1000E+01</td>
</tr>
<tr>
<td>11</td>
<td>.1000E+01</td>
<td>.1000E+01</td>
<td>.1285E+01</td>
<td>.1346E+01</td>
<td>.1756E+01</td>
<td>.1748E+01</td>
<td>.1800E+01</td>
<td>.1000E+01</td>
</tr>
<tr>
<td>12</td>
<td>.1000E+01</td>
<td>.1000E+01</td>
<td>.1285E+01</td>
<td>.1357E+01</td>
<td>.1765E+01</td>
<td>.1739E+01</td>
<td>.1800E+01</td>
<td>.1000E+01</td>
</tr>
<tr>
<td>13</td>
<td>.1000E+01</td>
<td>.1000E+01</td>
<td>.1285E+01</td>
<td>.1366E+01</td>
<td>.1774E+01</td>
<td>.1748E+01</td>
<td>.1800E+01</td>
<td>.1000E+01</td>
</tr>
<tr>
<td>14</td>
<td>.1000E+01</td>
<td>.1000E+01</td>
<td>.1278E+01</td>
<td>.1373E+01</td>
<td>.1780E+01</td>
<td>.1754E+01</td>
<td>.1802E+01</td>
<td>.1025E+01</td>
</tr>
<tr>
<td>15</td>
<td>.9957E+00</td>
<td>.1000E+01</td>
<td>.1273E+01</td>
<td>.1377E+01</td>
<td>.1785E+01</td>
<td>.1759E+01</td>
<td>.1825E+01</td>
<td>.1029E+01</td>
</tr>
<tr>
<td>16</td>
<td>.9989E+00</td>
<td>.1000E+01</td>
<td>.1279E+01</td>
<td>.1380E+01</td>
<td>.1788E+01</td>
<td>.1762E+01</td>
<td>.1832E+01</td>
<td>.1032E+01</td>
</tr>
<tr>
<td>17</td>
<td>.9977E+00</td>
<td>.1000E+01</td>
<td>.1280E+01</td>
<td>.1379E+01</td>
<td>.1789E+01</td>
<td>.1763E+01</td>
<td>.1833E+01</td>
<td>.1033E+01</td>
</tr>
<tr>
<td>18</td>
<td>.9952E+00</td>
<td>.1000E+01</td>
<td>.1283E+01</td>
<td>.1381E+01</td>
<td>.1792E+01</td>
<td>.1766E+01</td>
<td>.1836E+01</td>
<td>.1036E+01</td>
</tr>
<tr>
<td>19</td>
<td>.9968E+00</td>
<td>.1000E+01</td>
<td>.1281E+01</td>
<td>.1383E+01</td>
<td>.1793E+01</td>
<td>.1767E+01</td>
<td>.1838E+01</td>
<td>.1038E+01</td>
</tr>
<tr>
<td>20</td>
<td>.9943E+00</td>
<td>.1000E+01</td>
<td>.1283E+01</td>
<td>.1385E+01</td>
<td>.1796E+01</td>
<td>.1770E+01</td>
<td>.1840E+01</td>
<td>.1040E+01</td>
</tr>
<tr>
<td>21</td>
<td>.9956E+00</td>
<td>.1000E+01</td>
<td>.1282E+01</td>
<td>.1387E+01</td>
<td>.1797E+01</td>
<td>.1771E+01</td>
<td>.1841E+01</td>
<td>.1041E+01</td>
</tr>
<tr>
<td>22</td>
<td>.9952E+00</td>
<td>.1000E+01</td>
<td>.1282E+01</td>
<td>.1386E+01</td>
<td>.1797E+01</td>
<td>.1771E+01</td>
<td>.1842E+01</td>
<td>.1042E+01</td>
</tr>
<tr>
<td>23</td>
<td>.9956E+00</td>
<td>.1000E+01</td>
<td>.1282E+01</td>
<td>.1387E+01</td>
<td>.1798E+01</td>
<td>.1772E+01</td>
<td>.1842E+01</td>
<td>.1042E+01</td>
</tr>
</tbody>
</table>
Appendix F: Results of three optimization problems

- Graph showing design variables over optimization cycles.
- Graphs of objective functions for real (+) and approximated (o) scenarios.
- Graph of move limit over optimization cycles.
F.7. The four-bar mechanism

The results obtained by Mecopt are presented below. Convergence can be denoted in the plot of the objective function as well as from the logarithmic plot of the design variables. From the third column can be denoted that in the optimum design the move limit is no longer active, which means that the optimum is found in a vertex of at least three constraints. The three active linearized time point constraints in the optimum are given above the table. Since from each beam 151 time point constraints are derived, it can be concluded from these numbers that of each beam one linearized time point constraint is active.

<table>
<thead>
<tr>
<th>L</th>
<th>Ml</th>
<th>Mlact</th>
<th>F</th>
<th>Fben</th>
<th>no max(G)</th>
<th>no max(Gerr)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.000E+00</td>
<td>0</td>
<td>.546E+03</td>
<td>.000E+00</td>
<td>139</td>
<td>.234E-01</td>
</tr>
<tr>
<td>2</td>
<td>-.600E+00</td>
<td>1</td>
<td>.236E+03</td>
<td>.236E+03</td>
<td>139</td>
<td>-.139E+00</td>
</tr>
<tr>
<td>3</td>
<td>-.750E+00</td>
<td>1</td>
<td>.653E+02</td>
<td>.653E+02</td>
<td>139</td>
<td>-.702E+00</td>
</tr>
<tr>
<td>4</td>
<td>-.650E+00</td>
<td>1</td>
<td>.251E+02</td>
<td>.251E+02</td>
<td>139</td>
<td>-.891E-01</td>
</tr>
<tr>
<td>5</td>
<td>-.250E+00</td>
<td>1</td>
<td>.108E+02</td>
<td>.108E+02</td>
<td>139</td>
<td>-.792E-02</td>
</tr>
<tr>
<td>6</td>
<td>.250E+00</td>
<td>1</td>
<td>.145E+02</td>
<td>.145E+02</td>
<td>139</td>
<td>-.183E-01</td>
</tr>
<tr>
<td>7</td>
<td>-.300E+00</td>
<td>1</td>
<td>.121E+02</td>
<td>.121E+02</td>
<td>292</td>
<td>.122E-01</td>
</tr>
<tr>
<td>8</td>
<td>.300E+00</td>
<td>1</td>
<td>.106E+02</td>
<td>.106E+02</td>
<td>330</td>
<td>.329E-01</td>
</tr>
<tr>
<td>9</td>
<td>.300E+00</td>
<td>0</td>
<td>.106E+02</td>
<td>.106E+02</td>
<td>330</td>
<td>.462E-03</td>
</tr>
<tr>
<td>10</td>
<td>.300E+00</td>
<td>0</td>
<td>.106E+02</td>
<td>.106E+02</td>
<td>292</td>
<td>.905E-06</td>
</tr>
<tr>
<td>11</td>
<td>.300E+00</td>
<td>0</td>
<td>.106E+02</td>
<td>.106E+02</td>
<td>292</td>
<td>.905E-06</td>
</tr>
</tbody>
</table>
Appendix F: Results of three optimization problems

\[ x_1(+) x_2(\circ) x_3(\circ) \]

![Graphs showing design variable, objective function, move limit, and maximum constraint value over optimization cycles.](image)

- Design variable
- Objective function
- Move limit
- Maximum constraint value