6 Kinematic optimization

6.1 Introduction

In the previous chapter the kinematic analysis of mechanisms has been explained using a numerical approach based on the FEM. The motion of a given mechanism (the transfer functions) can be calculated in a generalized way. The result is numerical: a list with values for the various motion quantities. Higher order transfer functions and time derivatives (velocities, accelerations) are included in the method.
Optimization of the mechanism can be done when the motion of the mechanism is given and the dimensions of the elements are to be calculated. In the design process of mechanism, see fig. 1.3.1, this activity belongs typically to the kinematic dimension synthesis. Kinematic optimization is a synthesis method for mechanisms. When carried out with a numerical approach the motion is typically to be specified as a list of values for certain motion quantities. They can be considered as (discrete) conditions for the calculation of the kinematic parameters.
Compared with other synthesis methods one property can be mentioned as characteristic for optimization, see fig. 6.1.1. The number of motion conditions should be greater than or equal to the number of variables (kinematic parameters) to optimize. Usually an approximated solution for the motion problem can be found.

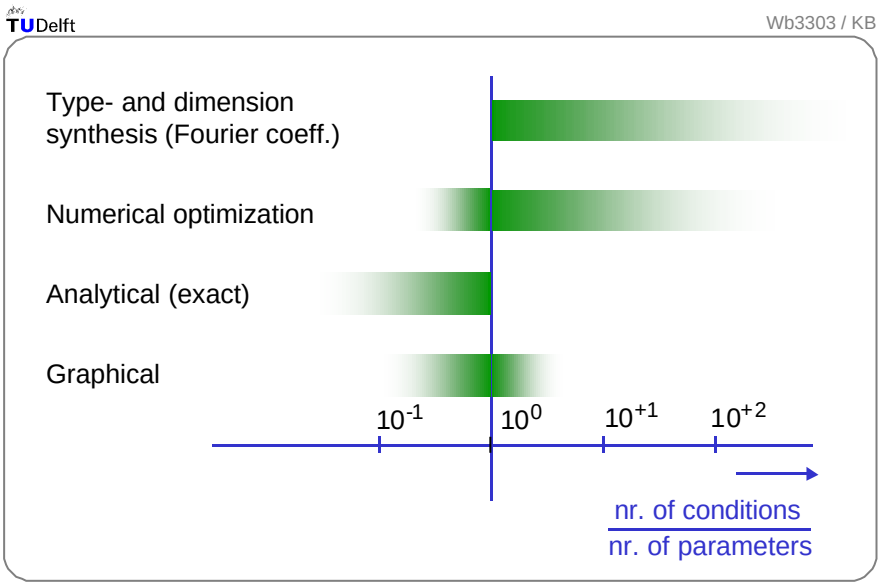

Fig. 6.1.1 Comparison of various synthesis methods

A basic calculation scheme is depicted in fig. 6.1.2, showing that optimization is actually improvement of a start solution. This means that the mechanism type and an initial configuration (initial values v_j of the variables to optimize) must be given. The actual motion quantities M_i, as determined by the actual values of the variables, and corresponding to the desired motion values G_i, can be contained in a least squares error function to be minimized. Weight factors w_i for the individual motion errors can be applied. They offer the user some influence on the behaviour of the error function F, while they are needed also to combine motion conditions with different physical meaning.

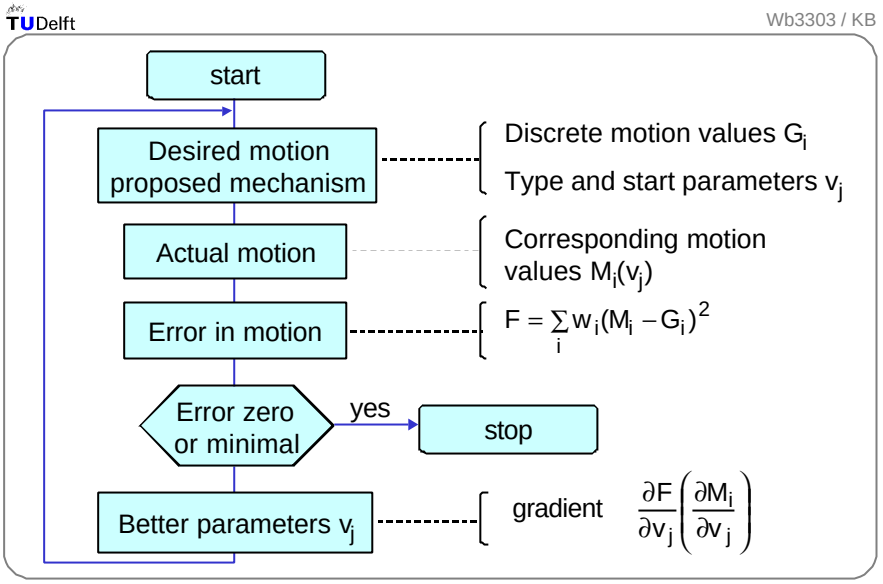

Fig. 6.1.2 Kinematic optimisation, basic calculation scheme

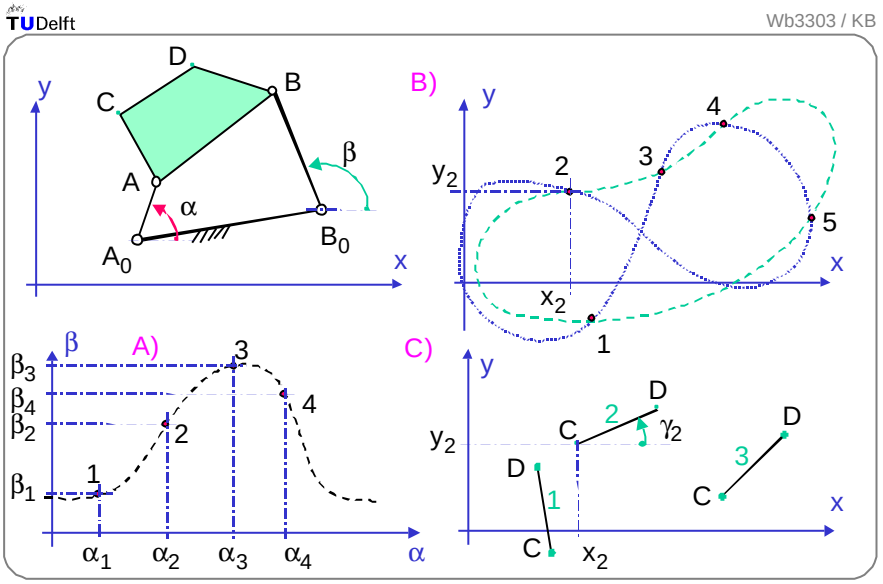

Fig. 6.1.3 Typical discrete motion conditions: A) function- B) path- C) plane motion - generation

The strategy to obtain better parameters is subject of extensive mathematical research [6.1]. Roughly optimization methods can be divided into gradient methods and non-gradient methods. The first group uses the gradient of the error function, which is by definition the vector of partial derivatives with respect to the parameters. This group offers usually superior results. In case that the derivatives are not available, an algorithm of the second group must be chosen.

Typical optimization problems with discrete motion conditions are depicted in fig. 6.1.3.

A) The transfer function $\beta(\alpha)$ of a given mechanism type must pass a certain number of prescribed points $\beta_i(\alpha_i)$. This type of problem is called a function generation problem.

B) The point C of the coupler plane must pass some points in the xy co-ordinate system of the fixed plane. Two types of motion definition can be distinguished.
   - ⇒ Just the points (x_i,y_i) are prescribed (path generation). Several solutions may exist regarding the sequence of passing the points, as indicated in the figure 6.1.3 upper right.
   - ⇒ The sequence of passing the points is included. This sequence can be expressed with crank angle α (timed path generation). The values (x_i,y_i,α_i) are to be specified, which eliminates by the way the sequence problem.

C) The coupler plane, represented by a point and orientation (or by two points CD as in fig. 6.1.3), has prescribed positions. This problem is called a plane motion generation problem. A timed version of this problem also exists: the sequence of the plane positions is then also specified.

The individual motion conditions are, in optimization terminology, called the *residual functions.* As mentioned before, they can be of any type or physical dimension. It will be clear that motion conditions of different type can be combined. Non-uniform conditions of motion can be used to specify a desired motion. This offers a great variety of motion specifications for the user. In fact any motion condition that can be expressed with transfer functions can be used. In combination with a general method for kinematic analysis like the FEM method any mechanism type can be optimized. Optimization must be regarded therefor as a very powerful tool in mechanism synthesis, as can be summarized in fig. 6.1.4.

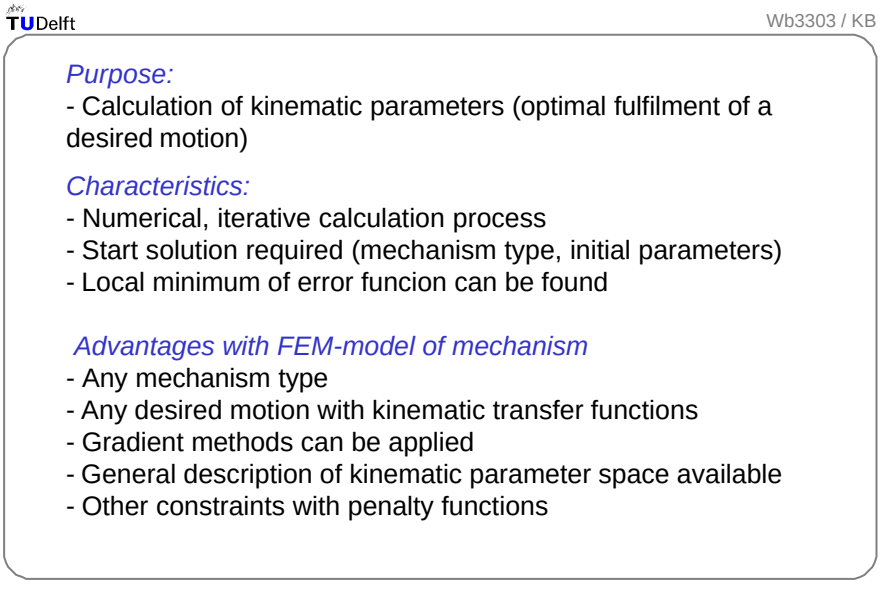

Fig. 6.1.4 Characteristics of FEM-optimisation

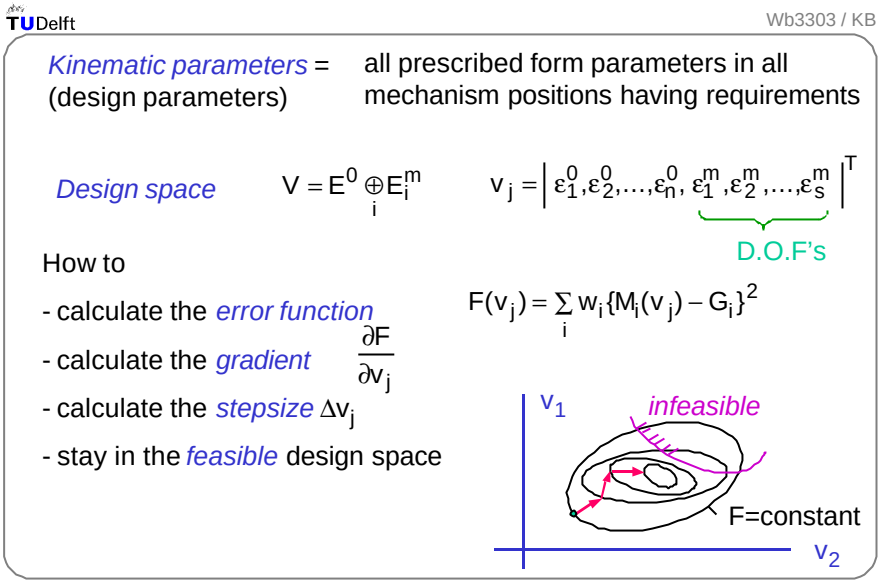

Fig. 6.2.1 Required calculations in FEM-optimisation

The basic theory of gradient optimization methods will be given attention in chapter 6.2. Feasibility of the parameter space and other constraint problems are the subjects of chapter 6.3. Practical use of the optimization method and examples using RUNMEC can be found in chapter 6.4. A complete case study of an optimization problem is presented in appendix 6A of this chapter.

6.2 Gradient method and FEM

Motion conditions can be prescribed in one or more positions of a mechanism. In kinematics these positions are identified by the values of the degrees of freedom. In the FEM these positions concern the values of the form parameters representing the degrees of freedom in the kinematic model. Taking into account that the kinematic parameters consist of prescribed form parameters ("each link can be modelled with one element"), the set of variables v_j that comes into account as optimization variable consists of:

$$v_j = \left| \varepsilon_1^o, \varepsilon_2^o, \varepsilon_3^o, \cdots, \varepsilon_n^o, \varepsilon_1^m, \varepsilon_2^m, \cdots, \varepsilon_s^m \right|^T \tag{6.1}$$

This is the set of prescribed form parameters (superscript o) and the values of all inputs in all positions (s positions). It depends on the motion problem whether they can be used all or partly. For instance: in the timed path generation problem the crank angles are chosen as fixed values. They must then be excluded from the list of variables to be optimized.
Further reduction of the list is accomplished when a designer makes explicitly the choice to keep a certain kinematic parameter unaltered. The vector space

$$V = E^o \bigoplus_i E_i^m \tag{6.2}$$

or a selection of it is to be recognized as the *design space* of parameters to optimize.
In this design space the error function can be expressed typically as

$$F(v_j) = \sum_i w_i \{M_i(v_j) - G_i\}^2 \tag{6.3}$$

The gradient, which is by definition the vector of partial derivatives with respect to the design parameters, can be derived as:

$$\frac{\partial F}{\partial v_j} = 2 \sum_i w_i (M_i - G_i) \frac{\partial M_i}{\partial v_j} \tag{6.4}$$

The derivatives of the motion quantity M, corresponding to a certain motion requirement G, are thus required in a gradient method of optimization.

Suppose now that the motion requirement concerns a *transfer function of order zero*. The motion conditions of fig. 6.1.3 belong to this group. In this case the partial derivatives of M concern just a partial derivative of some co-ordinate with respect to some form parameter. This is a coefficient of the inverse of matrix D^c. Such coefficients are required to calculate anyhow in kinematics. They are thus already available.

Suppose that the motion command concerns a *first order transfer function*. In this case the gradient requires the calculation of second order partial derivatives of the co-ordinate to be considered. They are also available in the FEM-theory, see fig. 5.3.7 and eq.(5.28). When the motion command M can be expressed with first order transfer functions (for instance the tangent to a path, see fig. 3.2.1) the derivatives of M can still be constructed with the second order derivatives of the co-ordinates.

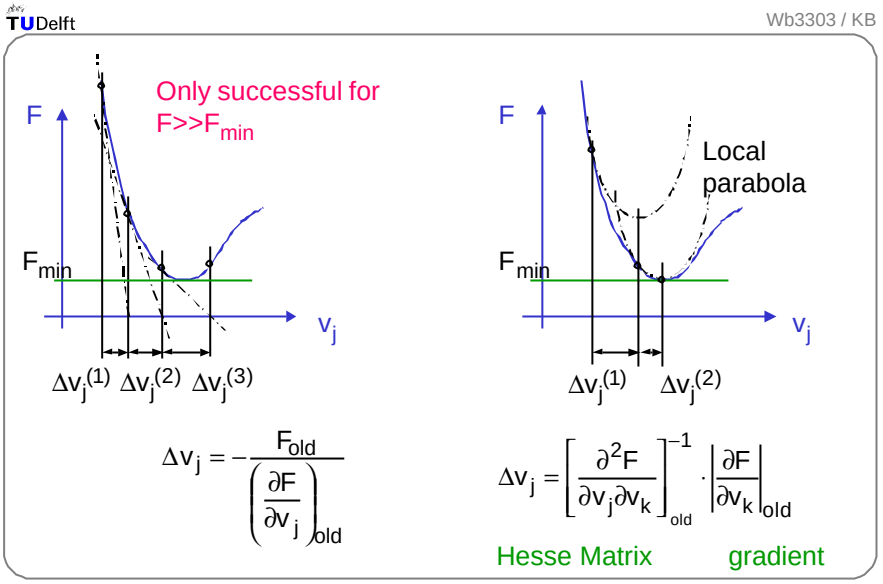

Fig. 6.2.2 Calculation of minimum point (Kuhn-Tucker point) First or second order extrapolation

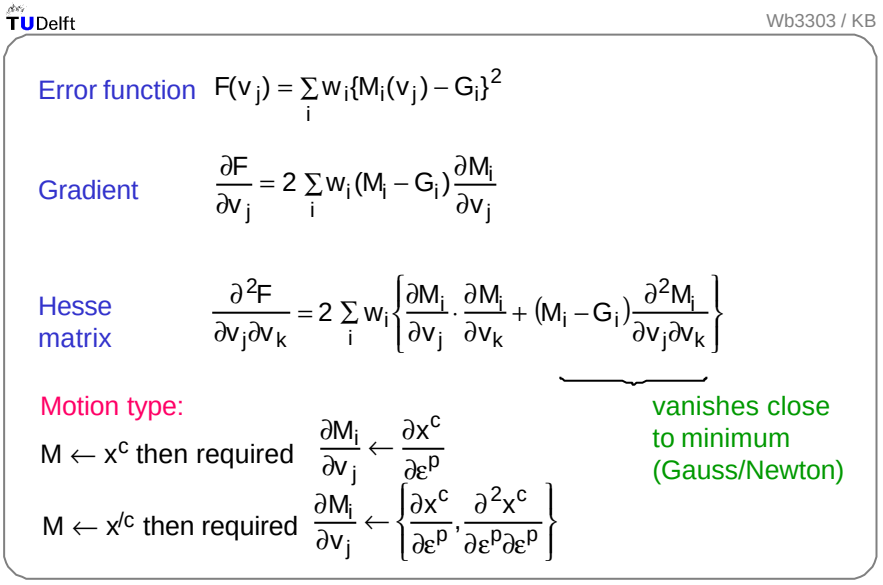

Fig. 6.2.3 Derivatives required for Gauss/Newton optimisation method

In case that the motion requirement concerns or uses a *second order transfer function* the calculation of the gradient needs anyhow the third order derivatives of the co-ordinates. In the present state of the theory they are not available.

> *Conclusion: for motion requirements up to order one, gradient methods can be used very advantageously in kinematic optimization with the FEM-approach. The partial derivatives needed to calculate the gradient are available in the theory.*

The gradient plays a role in the improvement of the error function. Practically this means that a better point in the (feasible) design space is to be found, see fig. 6.2.1. A proper step in the design space (a vector with increments Δv_j) must be applied to estimate the minimum point.
Problems concerning the feasibility will be given attention in chapter 6.3. Here the step to find the minimum point in an unconstrained design space will be considered.
It will be clear that in a minimum point the gradient is a zero vector (but the reverse is not true: a point in the design space having a zero gradient, called a *Kuhn/Tucker point*, is not always a minimum!). A sum of squares has however anyhow a minimum when F=0.
A search in pure gradient direction can improve the error function by the following calculation scheme, see fig. 6.2.2 (left part) for a one-dimensional impression. The new point can be estimated with

$$F_{new} = F_{old} + \left(\frac{\partial F}{\partial v_j}\right)_{old} (v_{j,new} - v_{j,old}) = F_{old} + \left(\frac{\partial F}{\partial v_j}\right)_{old} \Delta v_j \tag{6.5}$$

It can be tried to find a minimum $F_{new} = 0$. The step Δv_j is then

$$\Delta v_j = -\frac{F_{old}}{\left(\frac{\partial F}{\partial v_j}\right)_{old}} \tag{6.6}$$

This improvement (6.6) can only be used when none of the gradient coefficients is (nearly) zero, that means only far from the minimum point. Near the minimum it is a better idea to make a so-called line search in the gradient direction. Usually some trial steps in gradient direction and polynomial interpolation provide sufficient information to obtain a better point.

To find a Kuhn/Tucker point (gradient zero) a comparable estimation should be done with the gradient itself:

$$\left(\frac{\partial F}{\partial v_j}\right)_{new} = \left(\frac{\partial F}{\partial v_j}\right)_{old} + \left(\frac{\partial^2 F}{\partial v_j \partial v_k}\right)_{old} (v_{k,new} - v_{k,old}) \tag{6.7}$$

A zero gradient in the new point can be estimated then, see fig. 6.2.2 (right part) with

$$\Delta v_j = \left[\frac{\partial^2 F}{\partial v_j \partial v_k}\right]^{-1}_{old} \cdot \left|\frac{\partial F}{\partial v_k}\right|_{old} \tag{6.8}$$

The matrix with second order derivatives of F (usually called H, the *Hesse matrix*) is then required. This matrix must be inverted.
In the least squares error function the Hesse matrix is defined by differentiation of eq.(6.3):

$$\frac{\partial^2 F}{\partial v_j \partial v_k} = 2 \sum_i w_i \left\{\frac{\partial M_i}{\partial v_j} \cdot \frac{\partial M_i}{\partial v_k} + (M_i - G_i)\frac{\partial^2 M_i}{\partial v_j \partial v_k}\right\} \tag{6.9}$$

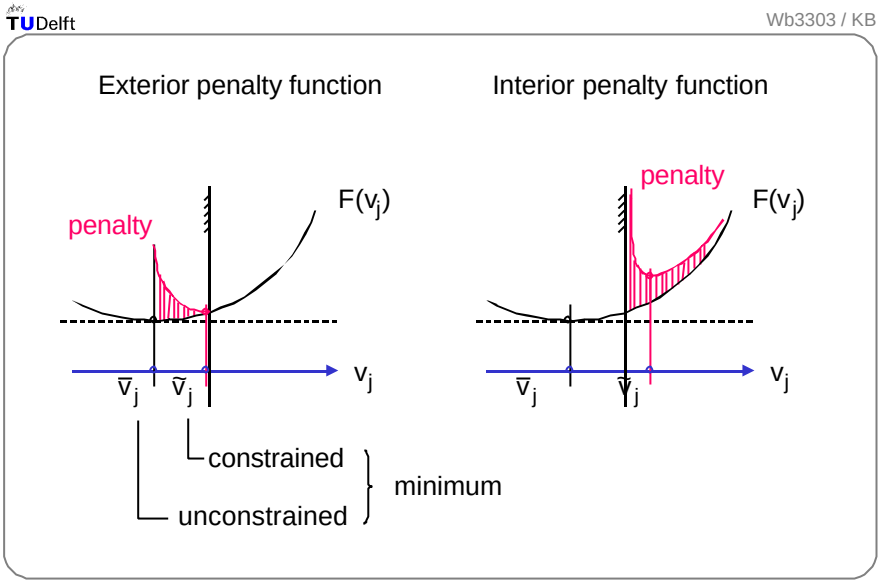

Fig. 6.3.1 Types of penalty functions

Now the second order derivation of the motion quantity M is required. In the FEM approach this means that only motion requirements of order zero (positions of the mechanism) can be considered.
Note that the step in the design space acc. (6.9) is not in the pure gradient direction. It anticipates for the second order behaviour of the error function to find the minimum. This method, applying the Hesse matrix, is called in literature the *Newton method* of optimization. It is generally recognized as a better method then the previously mentioned line search in gradient direction [6.1].
When a minimum point is close to zero the terms $(M_i\text{-}G_i)$ all become small compared with the other terms in (6.9). In that case the Hesse matrix can be approximated with:

$$\frac{\partial^2 F}{\partial v_j \partial v_k} = 2 \sum_i w_i \left\{ \frac{\partial M_i}{\partial v_j} \cdot \frac{\partial M_i}{\partial v_k} \right\} \tag{6.10}$$

Now only first derivatives of the required motion conditions are needed. In that case motion quantities (transfer functions) up to order one can be used. This approximation method should be taken as a compromise between quality of the optimization algorithm and functionality of the synthesis of mechanisms. The method using (6.10) as the Hesse matrix is called the Gauss/Newton method. All methods using some approximation of the Hesse matrix (including the one described here) are called Quasi-Newton methods. Figure 6.2.3 presents an overview of the theory described so far.

The Hesse matrix, obtained either exactly or by approximation, needs to be inverted in (6.8) and must therefor be regular (it is by definition a square matrix). In a Kuhn/Tucker point the Hesse matrix contains the information whether the function is convex (a real minimum) or not. In the optimization literature it can be found that the function is convex when all Eigenvalues of the Hesse matrix are positive. The Hesse matrix is called then positive-definite (when one or more Eigenvalues are zero the matrix is called semi positive-definite).This property of the Hesse matrix makes the (quasi-) Newton methods superior to the pure gradient methods, especially near the minimum point.
In practical problems it may easily occur that the Hesse matrix is (nearly) singular. In most optimization strategies a trick is proposed to avoid calculation problems. A well-known trick is to supply extra terms to the main diagonal of the Hesse matrix (that means a matrix αI, the unity matrix with scale factor α, is added). In this strategy the factor α should be chosen as less as possible. When $\alpha=0$ cannot be applied, the minimum is said to be weak. Uncertainty about the minimum point remains then.

6.3 Constraints

6.3.1 Conversion to an unconstrained optimization problem (penalty functions)

Constraints are inequalities that limit the feasible space of design parameters (optimization variables) directly or indirectly. Typical kinds of constraints are:

- Direct limits (upper and/or lower bound) of a design parameter. For instance: the length of a binary link may not be zero or negative, or a designer specifies explicitly limits for the length of a link.
- A boundary for a motion quantity. For instance: the point P of a coupler plane must stay in a certain region of the xy-plane.
- The overall feasibility of the kinematic parameter space. In most cases it is not possible to give one link any arbitrary length without assembly problems.

Such constraints can be converted towards an unconstrained optimization problem with the help of penalty functions, see fig. 6.3.1.

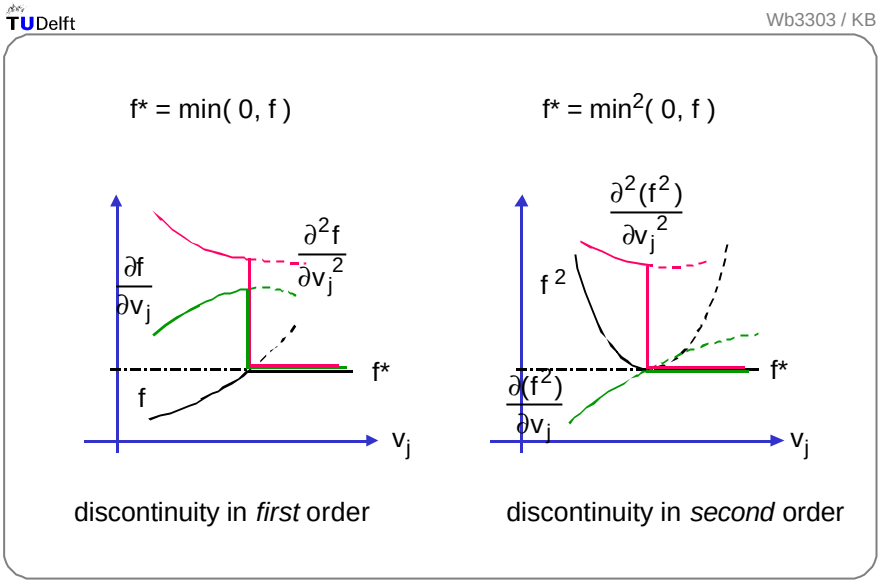

Fig. 6.3.2 *The min^2 filter for inequalities of motion*

The idea is to increase the error function when the constraint is violated. A distinction can be made with respect to feasibility of the violation. When a small violation can be accepted, the exterior type can be applied. The penalty will be added as soon as the violation has begun. In case that the parameter space outside the boundary is not feasible, the interior type must be applied: the closer the distance to the boundary, the more penalty. The third constraint category mentioned above belongs certainly to this penalty function type.

For each constraint, which is in its essence an inequality of the type

$$f(v_j)>0 \tag{6.11}$$

a residual function of the type

$$f^*(v_j)=\min(0,f) \tag{6.12}$$

could be taken. The error function will be extended, compare (6.1):

$$F(v_j) = \sum_i w_i \{M_i(v_j) - G_i\}^2 + \sum_k r_k \{f_k^*(v_j)\}^2 \tag{6.13}$$

in which r is a chosen penalty factor (a kind of weight factor).
In gradient methods it is also required to have the derivatives of the residual (penalty) function f* available. In general however such a derivative would be a discontinuous function, see fig. 6.3.2 (left part), and so would be the total error function. In such a situation the general optimization theory, both the gradient method and the Newton method, will certainly have difficulties to find or recognize a minimum point in the parameter space.
A better idea of defining a penalty function is with the $\min^2$ function:

$$f^*(v_j)=\min^2(0,f) \tag{6.14}$$

Now at least the first derivatives are continuous, see fig. 6.3.2 (right part). It is possible then to use all gradient methods and the method of Gauss/Newton.

6.3.2 Range of a parameter

Any design parameter v_k can be given a range using penalty functions. When a user specifies explicitly a minimum value v_{min} and a maximum value v_{max}, with the intention to keep

$$v_{min} < v_k < v_{max} \tag{6.14}$$

a (residual) penalty function like

$$f(v_k) = r_k \cdot \{\min^2(0, v - v_{min})_k + \min^2(0, v_{max} - v)_k\} \tag{6.15}$$

would be appropriate.
For calculation of the gradient it is required to have the first derivatives of f, with respect to all design parameters v. It will be clear that then also the first derivatives of v_k are required.

These derivatives are available: they concern either a 1 or a zero. The $\min^2$ filtering takes care of the continuity of the derivatives.

6.3.3 Range of a motion requirement

A discrete motion command can be regarded as an equality constraint: the value M_i should be equal to a desired value G_i. In some cases a designer might prefer an allowed range, with the intention to keep

$$G_{min} < M_i < G_{max} \tag{6.16}$$

This can be achieved with an external penalty function like

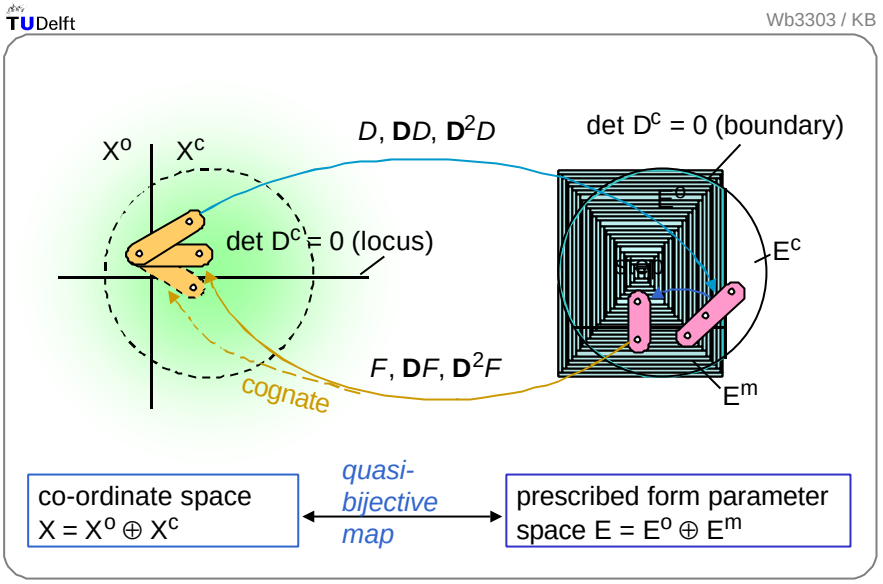

Fig. 6.3.3 Det D^c =0 as the boundary in the space of form parameters

$$f(v_k) = w_i \cdot \{\min^2(0, M(v_k) - G_{min})_i + \min^2(0, G_{max} - M(v_k))_i\} \qquad (6.17)$$

Note that, when the region is zero, this residual function is actually the same as a normal equality constraint.
Concerning derivatives of this penalty function: finally the derivatives of the motion quantities M are required. They are thus available in the same way as for normal equality constraints. Here again the $\min^2$ filtering takes care of the continuity of these derivatives.
A typical application of this type of constraint is the following. Suppose a coupler point should stay within a certain rectangle in the global xy-plane, for a certain mechanism position (DOF-value). This can be expressed with two conditions of the type (6.16/6.17): one for the x-range and one for the y-range.

6.3.4 Feasible parameter space

To make a step in the parameter space (design space) is always risky. How to avoid an attempt to step in the infeasible space, because in this part of the space the error function is undefined.
A practical point of view is that virtually all user-defined constraints can adequately be specified with exterior penalty functions. They are allowed to be violated and should be penalized then sufficiently. By restricting the user-defined constraints to the exterior penalty type, there will be no feasibility problems.

The overall kinematic parameter space may however not be violated. This is physically impossible, and the error function is not defined either. The question now is, whether such an illegal step can be made within the algorithm of optimization and FEM method.
Consider now a mechanism in a given position *and try to make a step outside the kinematic parameter space*. The mechanism position is given by its co-ordinate vector x, which is a point in the co-ordinate space X, see fig. 6.3.3. By proper set-up of the elements, co-ordinates can be varied without limits, so the co-ordinate space has no limits. With the distinction of fixed and moving co-ordinates the x-vector concerns then a double point in the co-ordinate space, as drawn in fig. 6.3.3. The vector x has an image vector (map D) in the space of prescribed form parameters E^p. This point (or double point regarding the distinction in fixed and driving form parameters) is always in the feasible space of the mechanism parameters, because of proper definition of the continuity equations of the elements.
It will be clear that the space E^p of prescribed form parameters is bounded. Suppose now that a step outside the feasible space is attempted, which means that a too big step for one or more prescribed parameters is required. Due to the iterative procedure to arrive at the new mechanism position, see fig. 5.3.6, always a feasible point (in space X) will be predicted (map F). This point on its turn has an image point in the feasible part of space E^p, and so on. In other words: the infeasible point will never be reached, the iteration process will fail due to lack of convergence. Probably the result lies on or near the boundary of the parameter space, but never crosses the boundary.
The boundary itself can, as a generalization of the limit position (chapter 5.5), be expressed as:

Det D^c = 0 is the boundary of the kinematic parameter space of a mechanism

Further it can be detected that the image of the boundary in the co-ordinate space is a subspace (one dimension less). Passing this subspace means that det D^c changes sign. In general two different assembly situations of the mechanism involve different signs of the determinant. It will be clear then that two or more points in X can have the same image point in E^p. It depends on the initial point in X to which new mechanism a step in E^p will lead. Due to the iteration process a (quasi)-bijective map between both spaces exists.

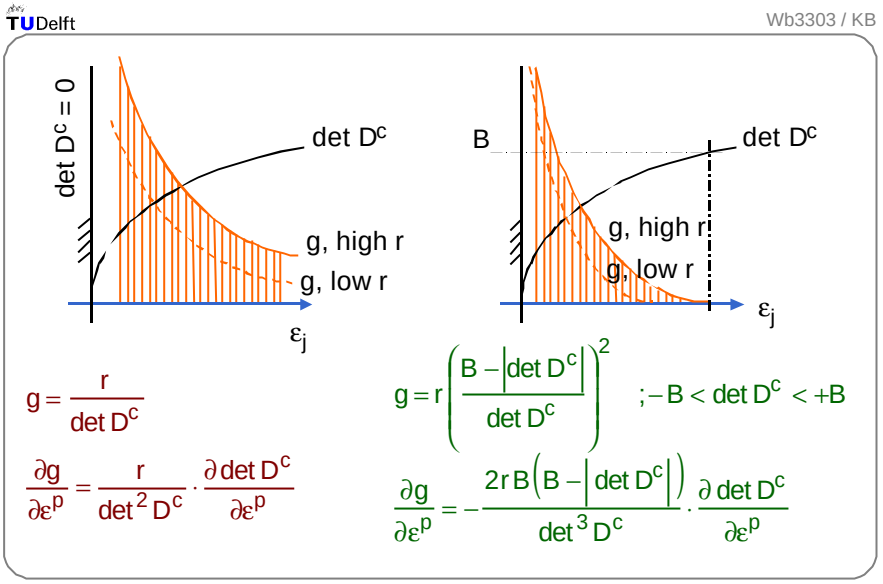

Fig. 6.3.4 (Interior) penalty functions for boundary det D^c=0 approach

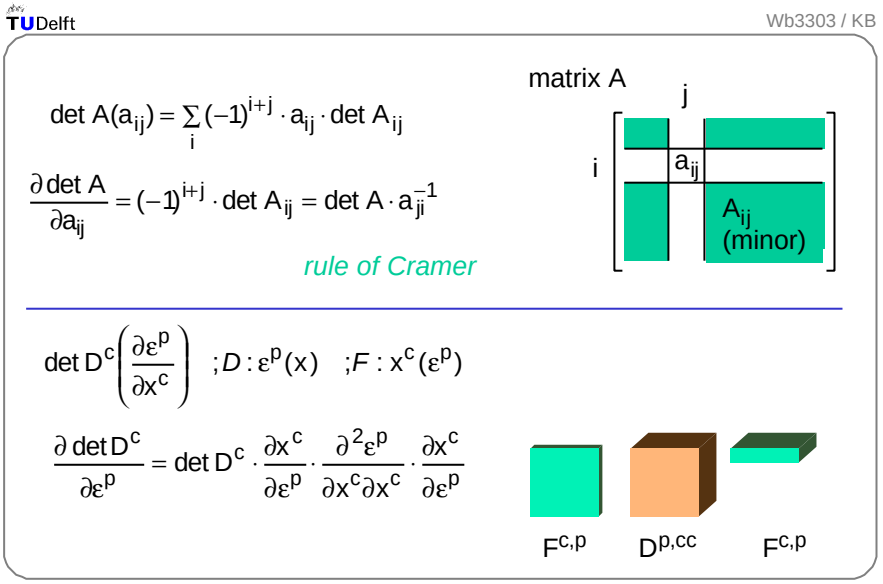

Fig.6.3.5 Derivatives of matrix D^c

To apply det $D^c = 0$ as a constraint (of the interior penalty type) it must be recognized that the total space of design parameters v_j is usually more extended than the kinematic parameter space, see eq.(6.1). The optimization problem concerns usually more than one mechanism position, and all positions need the boundary protection. It is required then to apply the constraint for each of the mechanism positions individually.

A penalty function for boundary protection can be constructed in several ways. A simple one could be the inverse of the determinant, see fig. 6.3.4 (left part). A disadvantage of this type is that the residual function g never will become zero, the same is true for the total error function as constructed with eq. (6.13). The application in the Gauss/Newton method is then unfortunate.
A more appropriate penalty function is depicted in fig. 6.3.4 (right part).

$$g = r\left(\frac{B - \left|\det D^c\right|}{\det D^c}\right)^2 \quad ; \quad \left|\det D^c\right| < B \tag{6.18}$$

Here a minimum value B (barrier) for the determinant is used, for higher determinant values no penalty is counted. Such a function needs also the $\min^2$ filtering for continuity of the derivatives of the penalty function.
In a gradient method of optimization, applying constraints as penalty functions, the derivatives of the penalty functions are also required. For the boundary of the kinematic parameter space it is finally required that the derivatives of det D^c, with respect to all (prescribed) form parameters, must be available. How they can be obtained is depicted in fig. 6.3.5. The determinant of a square matrix A is taken as a function of its elements a_{ij} (i and j count the rows and columns):

$$\det A(a_{ij}) = \sum_i (-1)^{i+j} \cdot a_{ij} \cdot \det A_{ij} \tag{6.19}$$

in which det Aij is the determinant of the minor matrix (row i and the column j removed). After differentiation with respect to matrix element a_{ij} and applying Cramer's rule it follows that

$$\frac{\partial \det A}{\partial a_{ij}} = (-1)^{i+j} \cdot \det A_{ij} = \det A \cdot a_{ji}^{-1} \tag{6.20}$$

This theory can be applied to the derivatives of the D^c matrix. The matrix coefficients have been identified as functions of the co-ordinates, see for instance eq.(5.5), and need to be differentiated thus firstly with respect to the co-ordinates and secondly with respect to the form parameters (chain rule). Not that both maps $X \rightarrow E^p$ and $E^p \rightarrow X$ exist, as has been explained in fig. 6.3.3. The final result of differentiation can be expressed as

$$\frac{\partial \det D^c}{\partial \varepsilon^p} = \det D^c \cdot \frac{\partial x^c}{\partial \varepsilon^p} \cdot \frac{\partial^2 \varepsilon^p}{\partial x^c \partial x^c} \cdot \frac{\partial x^c}{\partial \varepsilon^p} \tag{6.21}$$

The second order continuity functions are thus required. The calculation goes partially the same as in the calculation of the second order transfer function (5.28), which is required for derivatives of first order motion commands.
Note that the equations (6.20) and (6.21) are only valid for a nonzero determinant. The gradient of the boundary can thus be calculated near the boundary, but not on the boundary. A well-chosen penalty factor r in eq.(6.18) should avoid calculation problems.

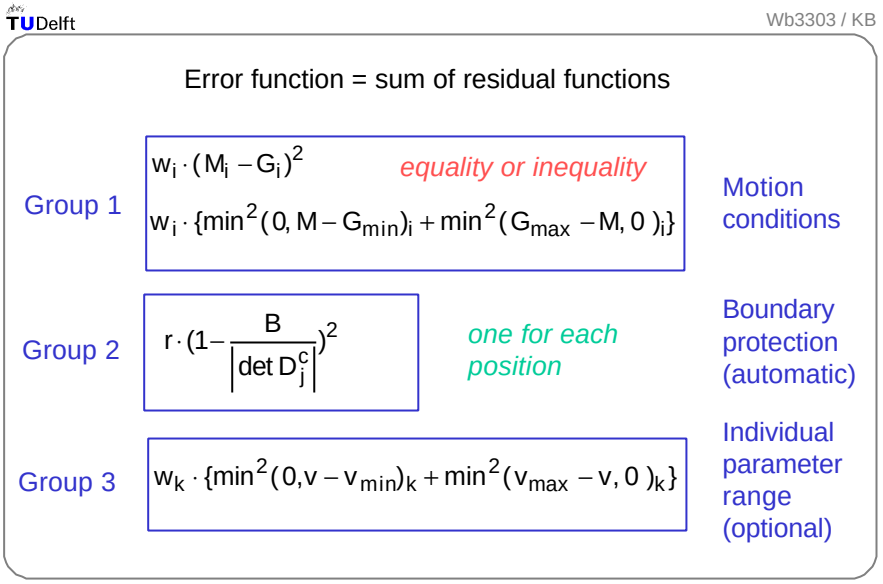

Fig. 6.3.6 *Composition of least squares error function with residual functions (summary)*

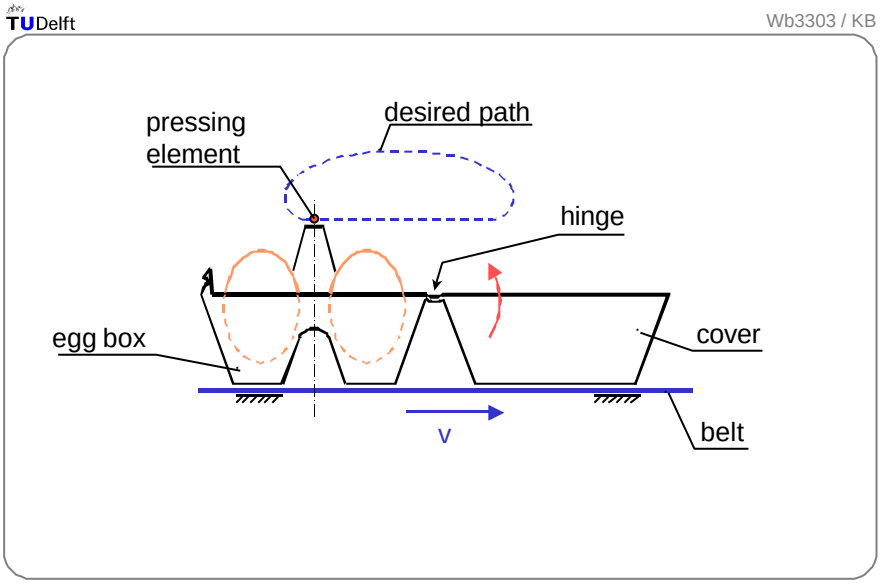

Fig. 6.4.1 *Egg box closing machine*

It can be concluded that automatic protection of the kinematic parameter space (concerning a mechanism in one position) and the space of optimization parameters (the mechanism in all positions) is possible. The user has two factors available to control the protection process:

- A value B for the penalty barrier of the determinant. It depends on the mechanism model in which range "good" values lie. Probably a user will initially start with a relatively high value (about the lowest determinant value in the initial configuration). When the residual penalty shows to be nonzero (the barrier is active) a lower barrier could be tried. In case of convergence errors a higher B-value can be needed.
- A penalty factor r as a weight factor for the penalty. It depends on the values of other motion requirements in which range the value of r is effective. Normally it is wise to start with normal or low values (typically r=1), and apply higher values in case of calculation problems like lack of convergence.

6.3.5 Combined motion requirements and constraints

In the previous chapters the basic types of motion requirements and constraints have been introduced (overview in fig. 6.3.6). In some situations a user may need a combination of those values. Typical examples are described in the following.

- A motion M requirement that concerns more than one mechanism position. For instance: the stroke of a slider motion $s(\alpha)$ is required to have a certain value. The stroke is the difference between a maximum and a minimum value, which occurs naturally at different mechanism positions called here α_{max} and α_{min}. The stroke requirement can be described with three conditions:

  $\Rightarrow$ Demand $s'=0$ in the position α_{max}

  $\Rightarrow$ Demand $s'=0$ in the position α_{min}.

  $\Rightarrow$ Demand $s(\alpha_{max})-s(\alpha_{min})$ to have the desired stroke

  The last requirement concerns two different mechanism positions. As the calculation of actual mechanism motion can only be done in one position at a time, such a combination needs intermediate storage. When all motion conditions will be given a unique storage place (*destination number for motion conditions*), the user can compose such combined motion conditions. The sign of the destination number could indicate addition or subtraction. The combination affects also the desired value G (the same treatment as M) and the weight factors (the highest value of the two). Inequality conditions of the type (6.17) can be included in this combination method. The values α_{max} and α_{min} are design parameters, which can be subject of optimization.
- Constraints on parameters of the type (6.15) can be subject of a combined requirement. Consider for instance the sequence problem of a coupler point as depicted in fig. 6.1.3 (B). With (inequality) constraints of the type

  $\Rightarrow \quad \alpha_i < \alpha_{i+1}$

  the sequence of passing the desired points can be expressed. Such a condition concerns two different design parameters α_i and α_{i+1}. The combining can be organized with destination numbers (separated *destination numbers for parameter conditions*) and use of the sign for addition and subtraction. The combining is to be applied to the range dimensions (the same treatment as the design parameter) and the penalty factor (the highest value will be applied).

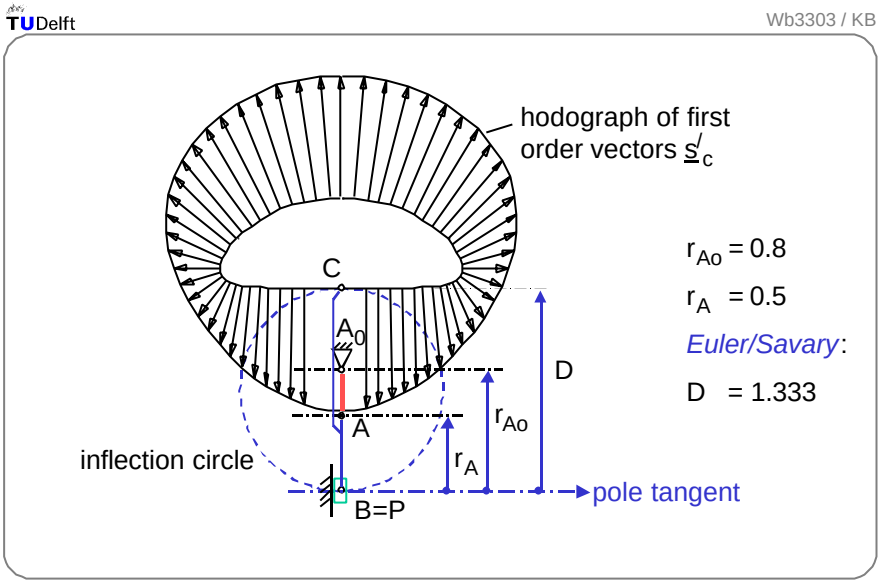

Fig. 6.4.2 Initial configuration of the crank-slider mechanism A_0AB with coupler point C

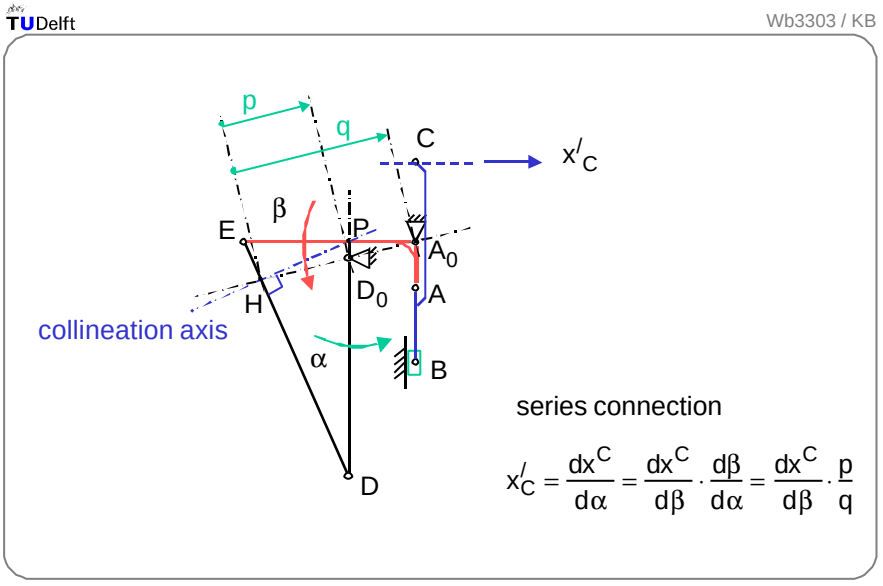

Fig. 6.4.3 Initial configuration of the double crank mechanism D_0DEA_0

- Exact equality of two design parameters. This is useful for instance to specify that two links have the same length (in some symmetrical mechanisms). Another example can be found for a mechanism, having degree of freedom two, where one of the inputs serves for adjustment of motion. Assuming that the motion requirement concerns more than one position of the first input, the second input must be kept constant precisely (but can still be a design parameter). In the definition of the design parameters however, each input value in each mechanism position is taken as a design parameter. All second input values must then be precisely the same, which needs the explicitly specification of equality constraints. It can be organized by making design parameters dependent on others ("append").

6.4 Modelling optimization problems (example)

6.4.1 The initial mechanism

The motion problem of the example concerns the closing of the cover of an egg box, see fig. 6.4.1. Cardboard boxes filled with eggs are placed in line at equal distances on a belt running at constant speed. The cover motion itself is not the subject of this example. Here the attention is given to the problem of avoiding the lift of the box, due to resistance in the hinge (deformation of the cardboard), while closing the cover. A pressing element moving along with the box is required during the closing of the cover. The pressing element could be a small roll, represented by a point, that moves partially along a straight line (but with the speed of the belt), and returns quickly by some return path. A path like drawn in the figure 6.4.1 could be generated by a link mechanism with a crank coupled to the driving shaft of the belt. The link mechanism can for instance be a symmetrical slider-crank mechanism A_0AB with coupler point C, see fig. 6.4.2. For given link dimensions A_0A and AB the point C that generates (instantaneously) a straight line can easily be calculated with the Euler/Savary equation (4.5), see chapter 4.4. The kinematic analysis shows that the path is sufficiently straight, but that the velocity (first order vector of point C) is insufficiently constant.
The velocity can be modified, without change of the path, by driving crank A_0A non-uniformly, that means by series connection. A simple mechanism, like the double crank mechanism, should be preferred here, see fig. 6.4.3. The output link of the double crank mechanism A_0E must be connected to the input link of the slider-crank, but an arbitrary angle AA_0E (assembly angle) can be chosen. As has been discussed in chapter 4.3, the first order transfer function of the double crank should be in a minimum, and this requires that the collineation axis is perpendicular to the coupler link ED. The velocity of point C will be reduced now by a factor p/q appearing directly from the relative pole distances of the mechanism (see also chapter 4.1 and fig. 4.1.3). Now there is enough information to draw a proper initial mechanism configuration that is suitable for optimization. Both the mechanism type and the initial dimensions have been chosen
Because many designers prefer mechanisms with revolute joints rather than slider joints, the mechanism could be modified in this way by replacing the slider by a relatively long rocker. Probably the initial mechanism solution will be a little worse, but it is hoped that the optimization will correct it sufficiently

6.4.2 The mechanism model.

To obtain the best possible optimization results, the mechanism should be modelled such that is suited best for the optimization purpose. Critical considerations are the following.

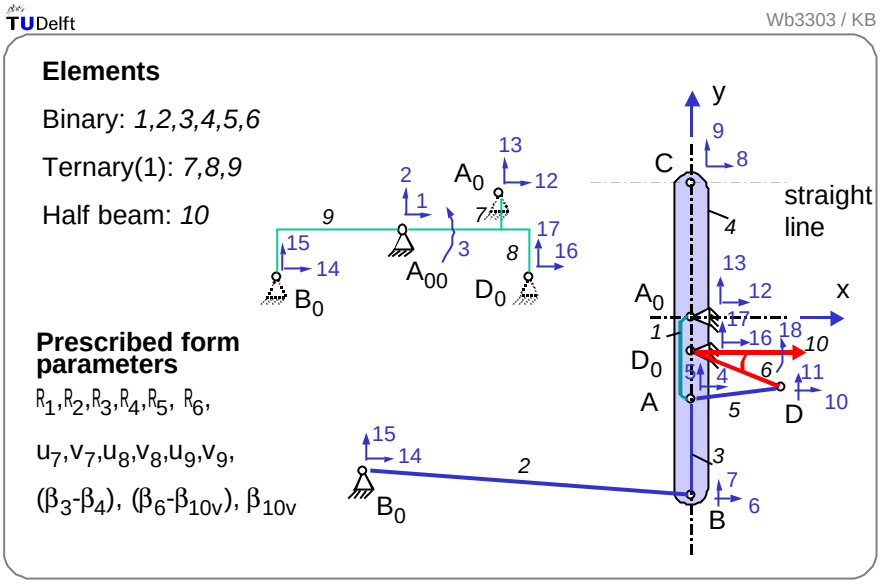

Fig. 6.4.4 *RUNMEC-model of the egg box closing mechanism*

- The more kinematic parameters, as meant in (6.1), the more possibilities to satisfy the desired motion. The user needs to recognize which variables can be effective to improve the final solution. In the example of 6.4.1 the positions of the fixed pivots certainly have influence on the path. *The co-ordinates of the fixed point should be modelled then as design parameters.* In a practical FEM-model of the mechanism a fixed point could be attached to the global co-ordinate system by two binary elements or one ternary element, with sufficiently form parameters as prescribed ones to keep that point at its place when the mechanism is in operation. Because the prescribed form parameters can be subject of optimization, the fixed point can be displaced by the optimization procedure.

- *Kinematic parameters that have no influence on the error function should be kept constant.* Such a parameter causes the error function to be not really convex (Hesse matrix semi-positive definite, the minimum would be a canyon instead of a point). The quality of the convergence could be reduced and the final result is not unique (the parameter considered gets a value by chance). The same problem arises when two or more parameters can be changed without influence on the error function. In the example of the egg-box such a dependency can be detected. It is possible to enlarge the double crank mechanism (all four bars at a time with the same factor) without changing the factor of non-uniformity p/q in fig. 6.3.4. At least one of the four link lengths must therefor be kept constant. But also the angle EA_0A can have any arbitrary value. This angle should be kept fixed (in the figure the angle is 90°)

- *Parameters should be allowed to vary as much as possible*. A coupler triangle should therefor preferably not be modelled with a triangle of binary elements. A model with two binary elements and a fixed angle is usually better. In a case like in fig. 6.4.2 this is obvious, since the three points A, B and C lie on a line. Probably the best choice for a coupler point is the use of a ternary element. The u- and v-parameter in this model allow the coupler point C to be displaced all over the coupler plane, even when coinciding with the pivots A or B of the coupler link.

- *Assembly angles may provide extra kinematic parameters*. The motion problem of the egg-box cover is typically a problem of “precision points specified for given crank angles”. But actually the crank angles need only to be equidistant. The first crank angle could have any value and need not to be prescribed. It would be correct to specify all subsequent crank angle distances by means of combined motion specification (as meant in 6.3.5), but this requires a long list of extra specification. A more practical idea is to introduce an assembly angle at the crank, providing a reference line in the crank plane, which is different from the crank orientation. The motion requirements (the precision points) can be specified with respect to the new crank angle. The assembly parameter (phase angle) can serve now as an (extra) optimization parameter, which will shift all crank angles. Now the first crank angle is immaterial.

- In general an assembly parameter can be used in the *input link and in the output link*, when discrete values for the angular position will be specified. Note that in the egg box example the output link concerns the coupler plane, but the angular position of this plane is not subject of motion specification. Therefore the coupler plane may not have an assembly angle subject to optimization (such a parameter would have no influence).

With the considerations as mentioned above an optimization model for the egg box mechanism can be proposed, see fig. 6.4.4. In chapter 6.4.4 it will be explained more in detail.

6.4.3 The motion specification

The desired motion is to be specified with discrete motion requirements, within the framework that the FEM offers. The following concerns important considerations.

- *The number of equality conditions*. The required path of the coupler point in the example has a part of major importance: the straight line part. Naturally a user would specify it with a lot of points lying on the ideal straight line. But how many points are required? Too many points might introduce calculation inefficiency instead of accuracy improvement. A rough guideline can be derived from the nature of optimization problems: the number of conditions should be greater then (or at least equal) the number of optimization parameters. The mechanism model of fig. 6.4.4 has 14 kinematic parameters (the crank angles are not optimizable). Each precision point needs two conditions to prescribe it. So at least seven points must be used to describe the coupler curve. Here the choice for ten points (nine for the straight line part, one for the return part) looks realistic.

- *The type of equality conditions*. Here the question is, which types of motion conditions specify the problem best? The straight line part of the egg box problem can probably be described more precisely by specifying the tangential directions, in addition to a number of points lying on a line. (Remind, tangential direction needs only first order derivatives, and can thus be made available in the Gauss/Newton optimization method). The user has the option to try this motion specification as well.

- *Use of weight factors*. In the example the return path is less important. This can be a reason to choose a low weight factor for the motion conditions concerning this point. The user has the option here to improve (intermediate) optimization results.

- *Use of inequality conditions*. Sometimes they appear to be necessary to avoid unwanted results. Typically they are inserted afterwards. In our egg box example however one condition can be recognized as necessary on beforehand. Compared with the initial mechanism configuration the slider (point B) has been replaced by a relatively long rocker. Now it can be expected that the optimization procedure will try to make the length of this rocker infinite. It looks a good idea then to add a region for this length.

- *Protection of the feasible parameter space.* The theory includes automatic protection, but the user is responsible for effective values of the penalty factor and the penalty distance. In practice the user must be aware of the "normal" values of the determinant of the matrix D^c of the mechanism model. A trial run is usually sufficient to obtain the det D^c values in all mechanism positions having motion conditions (the max. and min. values of the determinant will be normal output).

- *Awareness of alternative motion specification*. In the egg box example it was decided to count the points of the straight-line part from left to right, with increasing values of the corresponding crank angles. Actually this means a crank that rotates counter-clockwise. But a crank rotating clockwise (descending values of crank angles) could provide a solution as well. In this case however the specification of desired motion is symmetrical about the vertical axis. It can be expected that the clockwise specification of desired motion would finally yield a mechanism solution, which is symmetrical also to the original solution. But note that the mechanism solution does not have to produce a symmetrical curve. So there is still an alternative solution then! (The example in appendix A of this chapter presents another example where this point is present).

```
KIN OPT

TOPOLOGY
ELEM 1 BIN 12 13  4  5
     2 BIN 14 15  6  7
     3 BIN  4  5  6  7
     4 BIN  4  5  8  9
     5 BIN  4  5 10 11
     6 BIN 16 17 10 11
     7 TR1  1  2  3 12 13
     8 TR1  1  2  3 16 17
     9 TR1  1  2  3 14 15
    10 HBE 16 17 18

FORM 1 LENL  1  1  1.D0
     2 LENL  2  2  1.D0
     3 LENL  3  3  1.D0
     4 LENL  4  4  1.D0
     5 LENL  5  5  1.D0
     6 LENL  6  6  1.D0
     7 ANGB  3  7 +1.D0
     8 ANGB  4  7 -1.D0
     9 LENU  7  8  1.D0
    10 LENV  7  9  1.D0
    11 LENU  8 10  1.D0
    12 LENV  8 11  1.D0
    13 LENU  9 12  1.D0
    14 LENV  9 13  1.D0
    15 ANGB  6 14 +1.D0
    16 ANGB 10 14 -1.D0
    17 ANGB 10 15  1.D0
NRDOF 1

GEOMETRY
XFIX 1   0.0
     2   0.0
     3   0.0
XMOV 4   0.0
     5  -0.3
     6   0.0
     7  -0.8
     8   0.0
     9   0.53
     10  0.3
     11 -0.2
     12  0.0
     13  0.0
     14 -1.5
     15 -0.6
     16  0.0
     17 -0.12
     18  0.0

; all presc. form parameters can be optimizable, but 2 ranges are useful.
; RANGE info: dest.nr weight.factor lower.bound higher.bound
PARA 2 1.51327 RANGE 1 1.0 1.0 2.0  ; prevents length 2 to go to infinity
     5 0.31623 RANGE 2 1.0 0.0 0.5  ; user's choice (protects length 5)
```

Table 6.1

6.4.4 Optimization results with RUNMEC

Many of the modelling considerations have been applied in the FEM-model of the egg box closing mechanism, see fig. 6.4.4.

- The three points B, A and C of the coupler plane, which are in-line, have been modelled with two binary links (3 and 4) and a fixed angle.
- The new input link (6) has been given an assembly angle (new reference line of element 10).
- The fixed points A_0, B_0 and D_0 have been made optimizable by fixing them to the frame with prescribed (u,v)-parameters of ternary links.
- Choosing the four-bar mechanism in series (D_0DEA_0, see fig. 6.4.3) such that the links A_0E and A_0A are identical, causes a smaller model.

To specify which prescribed form parameters are optimizable, in RUNMEC the following idea has been adopted:

All prescribed form parameters ε^p are optimizable (without limits) unless they are marked by means of a PARA-statement. The PARA statement allows a form parameter

- Either to keep the specified value,
- Or to be subject of optimization between a lower and an upper bound (the weight factor is the penalty factor for exterior penalty). In such a case the keyword RANGE can be used on the PARA-statement, see table 6.1

To specify the motion commands of the mechanism, for the program RUNMEC two extra tables are required (extra with respect to the analysis model).

A block in the RUNMEC input file , preceded by the header word “OPTIMIZE” must contain the two tables.

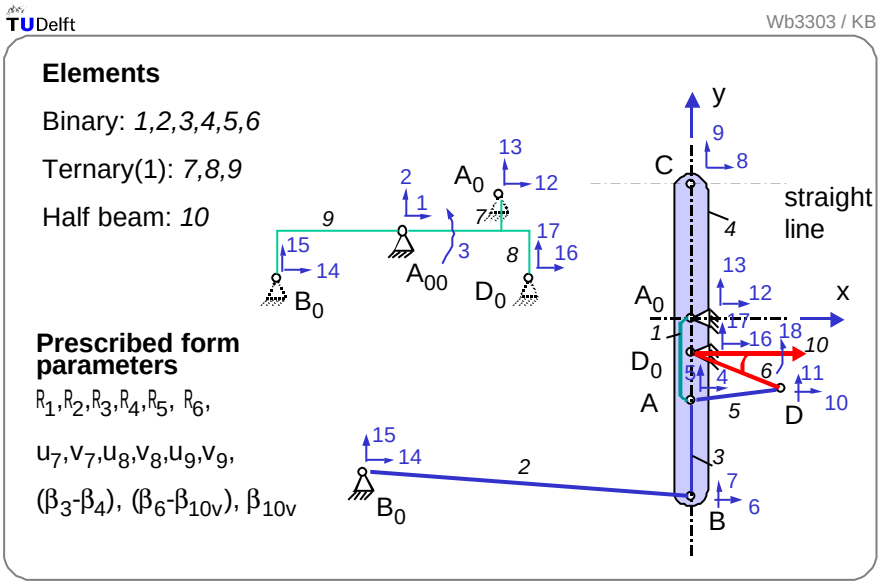

Fig. 6.4.4 RUNMEC-model of the egg box closing mechanism

```
OPTIMIZE
; 10 points for coupler point C are prescribed
;
; syntax of POSDEF command, describing the positions of the DOFs (crank angles):
; POSDEF  seq.nr, pos.nr, dof_nr, dof_value, status <YESOP/NOTOP>,
;                                          {dest.nr.K, Wf, Lowerb, Upperb.}

POSDEF 1 1 1  0.0       NOTOP  ; sequence of crank angles counter-clockwise
       2 2 1  0.3490658 NOTOP
       3 3 1  0.6981317 NOTOP
       4 4 1  1.0471976 NOTOP
       5 5 1  1.3962634 NOTOP
       6 6 1  3.02      YESOP  ; concerns the 'return point" in the path
       7 7 1  4.8869219 NOTOP
       8 8 1  5.2359878 NOTOP
       9 9 1  5.5850536 NOTOP
     10 10 1  5.9341195 NOTOP
```

Table 6.2

```
; the 10 points table (alternating an x- and a y-value at a line).
; Syntax of the MOTDEF command:
; MOTDEF seq.nr, pos.nr, dest.nr, goal_value, Wf, cond_type ,mot.def.type,
;                                                    {Par.1  ...  Par.4}

 MOTDEF 1  1  1  0.0        1 EQ 1 8
        2  1  2  0.533333   1 EQ 1 9
        3  2  3  0.16       1 EQ 1 8
        4  2  4  0.533333   1 EQ 1 9
        5  3  5  0.32       1 EQ 1 8
        6  3  6  0.533333   1 EQ 1 9
        7  4  7  0.48       1 EQ 1 8
        8  4  8  0.533333   1 EQ 1 9
        9  5  9  0.64       1 EQ 1 8
       10  5 10  0.55       1 EQ 1 9
       11  6 11  0.8        1 GT 1 9 ;  y-value > 0.8 (return point)
       12  6 12  1.2        1 LT 1 9 ;  y-value < 1.2 (return point)
       13  7 13 -0.64       1 EQ 1 8
       14  7 14  0.55       1 EQ 1 9
       15  8 15 -0.48       1 EQ 1 8
       16  8 16  0.533333   1 EQ 1 9
       17  9 17 -0.32       1 EQ 1 8
       18  9 18  0.533333   1 EQ 1 9
       19 10 19 -0.16       1 EQ 1 8
       20 10 20  0.533333   1 EQ 1 9
```

Table 6.3

The first block (lines with keyword POSDEF) defines all mechanism positions (DOF's) in which motion requirements can be specified, see table 6.2. By the status word (YESOP or NOTOP) it can be specified whether the crank angle value is to be considered as fixed or optimizable. The table has only effect for the optimization calculations (input motion for kinematic analysis is to be given with the MOVEMENT block). It will be used to generate the mechanism (the co-ordinate vector) in all required positions. Actually this is a special kind of kinematic analysis, which precedes the optimization calculations. In the present state of the program there is a noticeable difference with ordinary analysis: there is no need to take care of small steps for the DOF's. The program automatically creates intermediate values when necessary.
Note that the "return point" (number 6) has been taken with an optimizable crank angle. This increases the number of optimizable variables to 15.

The second block (lines with keyword MOTDEF) contains all required motion values, see table 6.3. They are related to the POSDEF table by means of the pos.nr item. The destination number (the third value) is part of a contiguous list of numbers, with which combinations of motion requirements can be made (as indicated in chapter 6.3.5). Here all motion requirements are single values, so each requirement has a unique destination number. The demanded values have here all weight factor 1 and are of the type "equality" (except for position 6, the point used to indicate the return part of the coupler curve). The item mot.def.type specifies the type of motion. The number 1 stands for a coefficient of the co-ordinate vector (a transfer function of order zero). The (maximally 4) values at the end give type-specific further information. In this mechanism model the co-ordinate numbers 8 and 9 specify the x-and y-value of the coupler point C. All transfer functions of order zero and one, concerning either co-ordinates or dependent form parameters, can be used in the present version. Further information can be obtained from the user manual of the program.

For the case of the egg-box mechanism some results of the program RUNMEC have been depicted in figures 6.4.5 and 6.4.6 (the program has been run with its default settings).
The final result of optimization is listed in the table 6.4, as has been produced by the RUNMEC printfile. The list of numbers v(1) –v(15) concerns the vector of optimization variables according (6.1). These values have, for this example, the meaning as indicated in fig. 6.4.4 (prescribed form parameters). The values of the start solution have been placed behind (in italics).

The TOMS 573 algorithm concerns the public domain software [6.4] that fits with the Gauss-Newton optimization theory.
The max. stepsize concerns the norm of the step Δv_j (general protection in euclidian space).
The boundary distance and weight factor concern the penalty function for the determinant, see r and B in eq.(6.18).
The maximum number of function evaluations, see eq.(6.13), has been set for time-out of calculation. In this case the calculations have been restarted 5 times, so in total 125 function evaluations were required with these settings.

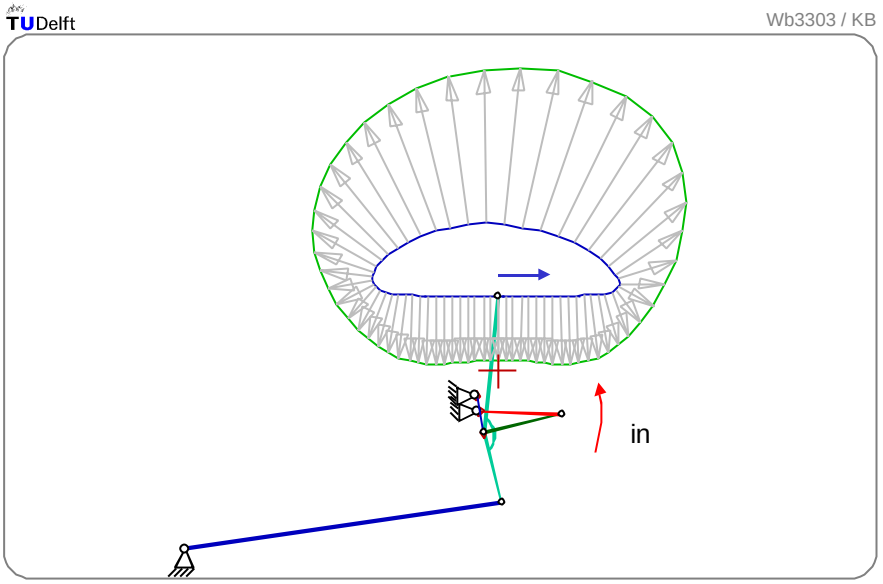

Fig. 6.4.5 Optimized egg-box closing mechanism

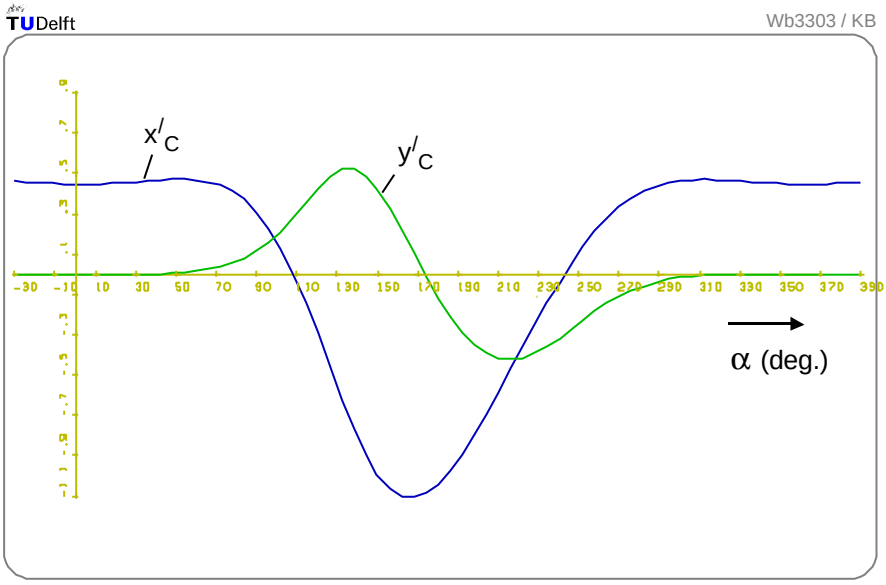

Fig. 6.4.6 First order path components of optimized mechanism

```
Settings: Algorithm:                  TOMS 573
          Maximum step size:              .01000
          Boundary distance:              .27657
          Boundary weight factor:       10.00000
          Max. nr. func. eval. :          25
          Max nr. euclidian steps:        25
          ABS minimum sum squares     .10000E-04
          Convergence tolerance ABS:  .10000E-09
          Convergence tolerance REL:  .10000E-09
          Min. gradient:              .10000E-09
 Diagnostic at termination: function evaluation limit reached
 Results: sum of squares:                 .0000644538
          minimum determinant:          -2.0015865839
          maximum determinant:          -1.4288209202
 Values of optimizable variables:
  V (  1) =    .2676987866147755  ( 0.30000)
  V (  2) =   2.0024072337383790  ( 1.51327)
  V (  3) =    .5017408219772163  ( 0.50000)
  V (  4) =    .9644958954251737  ( 0.83000)
  V (  5) =    .5023276923849926  ( 0.31623)
  V (  6) =    .5152587624691654  ( 0.31048)
  V (  7) =  -2.8281489070987630  (-3.14159)
  V (  8) =   -.1382735858784422  ( 0.00000)
  V (  9) =   -.1637913682929575  ( 0.00000)
  V ( 10) =   -.1211238475596558  ( 0.00000)
  V ( 11) =   -.2758761311166729  (-0.12000)
  V ( 12) =  -1.9571886166978480  (-1.50000)
  V ( 13) =  -1.2466972653049170  (-0.60000)
  V ( 14) =   -.0375806478010526  (-0.26060)
  V ( 15) =   3.0200000000000000  ( 3.02000)
 End optimization
```

Table 6.4

6.5 References

[1]Gill, P.E., W.Murray and M.Wright: Practical optimization. Academic Press Inc., London, 1981.

[2]Klein Breteler, A.J.: Kinematic optimization of mechanisms, a finite element approach. Thesis TU Delft, 1987

[3] Bergsma, B.: Optimaliseren van mechanismen (in Dutch). Afstudeerverslag TU Delft/WbMT, sectie Bedrijfsmechanismatie (Nr. 16.A.22), 1998.

[4]Dennis, J.E., D.M. Gay and R.E. Welsch: An adaptive nonlinear least-squares algorithm. ACM Transactions on Mathematical software. Vol.7, No.3, sept. 1981, pp. 348-368.