Crashworthiness design optimisation using multipoint approximations

Adriaens, J.M.T.A.

Award date:
1996

Crashworthiness Design Optimisation using Multipoint Approximations

J.M.T.A. Adriaens

WFW report 95.180

Master's thesis

J.M.T.A. Adriaens

Delft, December 1995

Examining-board:

Prof. dr. ir. J.S.H.M. Wismans (TNO/TUE)
Dr. ir. A.J.G. Schoofs (TUE)
Dr. ir. F. van Keulen (TUD)
Ir. L.F.P. Etman (TUE)
Ir. M.T.P. van Slagmaat (TNO)

Netherlands Organization for Applied Scientific Research (TNO),
Crash-Safety Research Centre,
Delft, The Netherlands

&

Eindhoven University of Technology (TUE),
Computational and Experimental Mechanics Group,
Eindhoven, The Netherlands

Abstract

In order to improve the crashworthiness performance of road vehicles, safety measures, like airbags and seat belts, are developed. The coupling of an optimisation program with a crash simulation package, like MADYMO, can aid the designer in his search for the optimum safety measures design, using as few time expensive crash analyses as possible.

A combined criterion of several injury parameters is formulated as the objective function, and the by law obliged maximum injury parameter values form the constraints. Injury parameters, and thus the objective function and constraints, show a noisy behaviour as a function of the design variables. Therefore, derivatives are unuseful and instead of an optimum point there is an optimum region where all points are considered equivalent.

Within the general concept of sequential approximate optimisation, the midrange multi-point-path concept is chosen. With a movelimit strategy a subregion of the design space is defined. Herein several design points are selected and analysed. Using only function values, linear approximation models for objective function and constraints are built. The resulting approximate optimisation problem is solved by a simplex algorithm. Until a stopping criterion is reached, new optimisation cycles are carried out.

After several tests, the first version of the developed method has been successfully applied to the optimisation of a full-scale off-set impact simulation, where six design variables were defined. Starting from an infeasible design, a strongly improved and feasible design resulted in only nine optimisation cycles. Possibly, the movelimit strategy and the plan points selection may be improved to yield a program that requires even less MADYMO analyses for the optimisation of the crashworthiness performance of road vehicles.

Contents

1 Introduction

Road accidents are an important cause of fatal and severe injuries. Crash-safety research centres have successfully developed new safety measures, like airbags, seat belts and childseats. Additionally, injury parameters have been formulated to quantitatively describe severity of injuries. Governments nowadays impose restrictions on maximum injury parameter values for the crashworthiness performance of road vehicles. They stimulate or oblige by law to apply newly developed safety measures.

Historically, crashworthiness design optimisation was mainly based on trial and error. For every design change, physical tests had to give the decisive answer with regard to the improvement. Such physical tests are cost expensive as well as time expensive. To overcome these disadvantages, crash simulation software has been developed in order to replace physical tests by numerical tests. An important precondition for this replacement is that the mathematical models of both vehicle and occupant sufficiently describe the real physical behaviour. By using computer simulation tools, crashworthiness design improvements can be systematically performed. Although much cheaper than physical tests numerical experiments are still computationally expensive, even for relatively simple crashworthiness simulations. So the number of different designs that can be analysed is limited. Optimisation programs can aid the designer in his search for the optimum design using as few time expensive crashworthiness analyses as possible.

In the last years several optimisation methods have been applied to crashworthiness design problems. Bennett & Park (1991) compared a sequential linear programming (SLP) method with a method proposed by Vanderplaats (1979). In the SLP method first order derivatives were obtained by using moderately large finite difference steps. Several crashworthiness design problems were optimised. Bosio & Lupker (1991) used Taguchi's design of experiments method to obtain a linear response surface model including interaction terms. A driver simulation with a seat belt and an airbag was optimised. Klink (1991) constructed global response surface models (linear and quadratic), using a design of experiments method. An objective function, build from several neck injury parameters, was formulated to optimise a childseat. De Jager (1993) compared an algorithm developed by Nelder & Mead (1964), a midrange multi-point-path method developed by Toropov (1989), and a midrange single-point-path method developed by Vanderplaats (1979). Several tests on analytical functions and

crashworthiness design problems, among which Klink's problem, were carried out. He concluded that Vanderplaats' method performs best.

This report describes the development, implementation and application of a method for crashworthiness design optimisation. The crash simulation package MADYMO, to which the optimisation program is applied, is developed at TNO Crash-Safety Research Centre (Delft, The Netherlands). In chapter 2, the sequential approximate optimisation concept and more specifically the multipoint approximation concept is chosen for application to crashworthiness design optimisation problems. The development and implementation of a multipoint sequential linear programming method will be considered in chapter 3. Tests on several analytical design optimisation problems, known from the literature, are discussed in chapter 4, as well as the successful application to several crashworthiness design optimisation problems, known from practice. The developed multipoint approximation method is able to automatically handle problems with noise on the objective function and constraints. This and other conclusions, together with some recommendations, are presented in chapter 5.

2 Choice for the optimisation concept

Crash simulation packages, and in fact structural analysis packages in general, are capable of analysing a user-supplied mechanical model, represented by a set of design variables. A design can be improved by manipulating the design variable values. It is necessary to formulate a criterion from which it can be concluded whether the design change is an improvement. In optimisation theory this criterion is called the objective function, which is a function of the design variables. The freedom of design is in general restricted by two types of constraints. The first type represents the restrictions with respect to the behaviour of the design (e.g. maximum value on stress in a truss). The behaviour is formulated in constraint functions, which are functions of the design variables (e.g. stress in a truss as a function of the cross section). The second type represents the lower and upper bound on the design variables. The design space is defined by these so-called design constraints.

A general formulation of a structural optimisation problem is to find the set of design variables $x \in R^n$, that will minimise the objective function:

$$F(x) \tag{2.1}$$

subject to the constraints:

$$g_j(x) \leq 0 \qquad j = 1,2,...,m \tag{2.2}$$

within the design space:

$$lb_i \leq x_i \leq ub_i \qquad i = 1,2,...,n \tag{2.3}$$

If objective function and constraints are explicit functions of the design variables, the optimisation problem can be efficiently solved by a standard mathematical programming algorithm. In structural analysis, however, the relation between output response and design variables is, in general, only known implicitly. Generally, these implicit relations are very complex, resulting in computationally expensive analyses. Solving design problems with these properties, using a standard mathematical programming algorithm, brings about two difficulties (Haftka & Gürdal, 1992). The first one is a programming problem. It is very difficult to transform a complex analysis

package into a subroutine called by the optimisation program. The second one might be the high number of computationally expensive analyses these standard mathematical programming algorithms generally require, in case of nonlinear relations between output responses and design variables.

There is an approach that solves both difficulties. In this approach a suitable approximation concept is used to interface the structural analysis software and the optimisation program (Etman et al., 1994). This popular approach is called sequential approximate optimisation.

Characteristic crashworthiness design optimisation problems, will be discussed in section 1. In section 2 all sub concepts within the sequential approximate optimisation concept will be briefly considered, whereafter their applicability in crashworthiness design optimisation will be decided about. Two popular multipoint approximation methods will be compared in section 3, whereafter it will be decided whether one of these two methods or a newly to develop multipoint method will be applied.

2.1 Crashworthiness design optimisation problems

Occupant responses in a crash event (e.g. accelerations, displacements and contact forces) are functions of time. It is very hard to judge injuries upon time signals. Therefore, time dependent occupant responses are replaced by injury parameter values, resulting from mathematical operations on the original time signals. Governments impose restrictions on maximum injury parameter values. These maximum values are the constraints in crashworthiness design optimisation. An overall improvement with respect to injury minimisation can be achieved by minimising a combined criterion of several injury parameters.

In the functional behaviour of injury parameters as a function of the design variables discontinuities might occur, caused by design changes which lead to the appearance of contact situations. This is typical for crash problems. Injury parameter values result from mathematical operations on (a part of) the original time signals. For each simulation, the injury parameter values will generally be found between two different time points (another part of the original time signal). This is the cause for a second kind of discontinuities, due to the definitions of the injury parameters. In MADYMO the equations of motion are solved numerically. MADYMO simulations using the finite element module require a numerical method with a fixed integration time step. The

value of this fixed time step determines the accuracy of the solution. A second time step that has to be specified in MADYMO is the output time step. The response values at the output time points are used to calculate the injury parameter values. From appendix A, where several injury parameters are discussed in greater detail, it is concluded that, due to several causes injury parameters show a noisy behaviour as a function of the design variables. Therefore, derivative information is unuseful, because this local information does not represent the global behaviour. Typical for optimisation problems which contain noise, is the fact that instead of an optimum point there is an optimum region, where all points are considered equal. The optimisation method to apply must be able to automatically handle problems, containing noise. Another aspect that has to be taken into account is that injury parameter values result from time expensive calculations. Therefore, the number of analyses in an optimisation process is limited.

Application of optimisation software in crashworthiness design is initially sought in optimising occupant - safety measures interactions. Safety measures design variables are physical quantities, like diameters and thicknesses. In many optimisation problems the functional behaviour of objective function and constraints can be linearised by introducing so-called intermediate response variables (Barthelemy & Haftka, 1993). This can be explained by a simple crashworthiness example. The vent area of an airbag system has an important influence on head injuries. If instead of this area ($A = \pi d^2/4$) its diameter d (the physical design variable), is defined as a design variable, an unnecessary square is introduced. Therefore, it is a more effective approach to approximate the occupant response as a function of A than as a function of d.

In the introduction of this chapter it is stated that sequential approximate optimisation is a popular approach to deal with the optimisation of complex structures, like a vehicle - occupant combination. Within this concept an optimisation method will be sought for. There are several requirements this method will have to meet. Firstly, derivatives may not be required. Secondly, the number of necessary computationally expensive crashworthiness analyses should be limited. Thirdly, robustness should guarantee an optimum, satisfying all constraints and close to the global or close to a "good" local optimum.

2.2 Approximate optimisation for crashworthiness design

The basic principle of sequential approximate optimisation is to generate explicit approximations of the objective function and constraints for a certain part of the design space, and to solve the optimum point for this approximate optimisation problem using a standard mathematical programming algorithm (Etman et al., 1994). If the search subregion does not cover the complete design space, a new cycle of approximation and optimisation can be started at the optimum. This process is repeated until an acceptable optimum is achieved. A global, local, and midrange concept can be distinguished. The last concept can be subdivided in a single-point-path (SPP) and a multi-point-path (MPP) concept. All concepts will be briefly considered, whereafter their applicability in crashworthiness design optimisation will be discussed.

Global

The idea is to generate explicit approximation models of objective function and constraints for the entire design space or a large part of it. A successful approach is to build analytical response surfaces using function values from design points, carefully selected by some design of experiments method. A comprehensive work on experimental design and structural optimisation is given by Schoofs (1987). The use of sensitivities is optional in global methods. Generally, objective function and constraints show a nonlinear behaviour as a function of the design variables. The relatively simple global approximations will then be quite coarse, generally resulting in an unsatisfying optimum design. However, this optimum can be used as a high quality start design for a next approximate optimisation in a subregion of the design space. Klink (1991) optimised several crashworthiness design problems, and concluded that building global approximation models is a non-automated process, which demands a great deal of understanding from the user.

Local

In local methods data of only a single design point is used. Therefore, local methods require at least function value and first order derivatives. The relatively simple approximations will only be valid for a small subregion of the design space. Within this region an optimum is located. The iterative process of approximation and optimisation is repeated until an acceptable optimum is achieved. The noisy functional behaviour

of objective function and constraints, resulting in unuseful derivatives, make the local concept unsuitable for crashworthiness design optimisation.

Midrange

Midrange methods attempt to combine the advantages of both local and global methods. The sequential character is the same as in local methods. In each optimisation cycle, approximation models are constructed using data from more than one design point, as in global methods. As already mentioned, a single-point-path (SPP) and a multi-point-path (MPP) concept can be distinguished.

- *Single-point-path*

In local methods all data from previous optima is discarded. If this data would be used to enhance the approximations, the search subregion could be enlarged. By doing so the neighbourhood of the optimum would be reached in fewer steps. All SPP methods, except for the method proposed by Vanderplaats (1979), use function values and first order derivatives to construct the explicit approximation models. Therefore, Vanderplaats' method is the only applicable SPP method in crashworthiness design optimisation.

- *Multi-point-path*

In the current search subregion some additional design points are analysed. This additional information is used to construct explicit approximation models, valid for a larger subregion than would be seen in local and SPP methods. If no derivatives are used, these approximations are relatively well capable of describing functions, which show a noisy behaviour. Therefore, it is concluded that MPP methods are applicable in crashworthiness design optimisation.

Summary

Based on the characteristic crashworthiness design optimisation problems, we conclude that midrange multi-point-path methods and Vanderplaats' midrange single-point-path method are applicable in crashworthiness design optimisation in practice.

2.3 Multipoint approximation concepts

Both midrange multi-point-path methods and Vanderplaats' midrange single-point-path method use function values of more than one design point to construct explicit approximation models for a subregion of the design space. Therefore, both methods will be referred to as multipoint methods. A popular midrange multi-point-path method is developed by Toropov et al. (1993). In this section, Toropov's and Vanderplaats' methods will be discussed, whereafter it will be decided whether one of these two methods or a newly to develop method will be applied.

Multipoint approximate optimisation as suggested by Toropov

The start up procedure consists of choosing a start design, a search direction and movelimit factors, which defines the first search subregion. Herein plan points are chosen in the same way as in determination of finite difference derivatives. Explicit approximation models of objective function and constraints are built, using function values of start and plan points, and, if close to the current search subregion, some previous analysed design points. The resulting approximate optimisation problem is solved by a standard mathematical programming algorithm. After analysing the optimum design point, the maximum approximation error and maximum constraint violation are calculated. If the optimum is accepted, but no convergence has occurred yet, a new search subregion is defined, new plan points are chosen, and so on.

Multipoint approximate optimisation as suggested by Vanderplaats

At any point in the optimisation process a second order approximation is constructed about the current best design by fitting a full second order approximation using a least squares method to determine the "best" second order fit for the data (Bennett & Park, 1991). The new optimum, calculated from the approximate problem, is added to the list of points and the process is continued until a stopping criterion is reached. In each search subregion this stopping criterion is based on the positions of last few optima. The process can be started with any number of points. If at any iteration, less than the number of points required for a full second order fit are available, a reduced approximation is constructed. Terms are added sequentially through the first order terms, the diagonal second order terms, and finally the off diagonal terms. As the design process converges only points close to the optimum are retained.

Discussion

In Toropov's method a large number of analyses is carried out in each optimisation cycle. However, for calculation of finite difference derivatives the same number of analyses is necessary.

In Vanderplaats' method the design points used to construct the approximation models will not be spread homogeneous over the search subregion. This is not favourable for the model accuracy. In the process of sequentially adding second order terms to the model, it is only known which term is important when a full quadratic model is available. Therefore, one can't speak of convergence during this process of adding terms to the approximation model.

De Jager (1993) compared Vanderplaats' with Toropov's method. For the latter only a simple movelimit strategy was implemented. Several tests were carried out on analytical functions and crashworthiness design problems, whereafter he concluded that Vanderplaats' method performs best. Closer examination of the results teaches us that the main disadvantage of Toropov's method is that many optimisation cycles are carried out in the last stage of the optimisation process. These cycles are of no use if the noisy functional behaviour of the injury parameters is taken into account. Considering this, there is only a small, negligible, difference between the two methods.

We believe that with a more advanced movelimit strategy Toropov's method will perform better than Vanderplaats' method. Another improvement, which will extensively be discussed in the next chapter, is application of the experimental design theory instead of large finite difference steps. To summarise: based on Toropov's method we will develop a new midrange multi-point-path method.

3 A multipoint sequential linear programming method

The multipoint approximate optimisation process is visualised in Figure 1. By specifying a start design, among other things, the first search subregion is defined. Herein several extra design points are selected in order to construct the approximation models. In section 1 the choice for linear approximation models will be made and the plan points selection will be discussed. The linear approximation models for objective function and constraints are built using function values of the start and plan points. The resulting approximate optimisation problem is solved by a simplex algorithm, which will be considered in section 2. In the movelimit strategy module it is decided whether convergence has occurred. If not, a new search subregion is defined, starting from the approximate optimum (section 3). In section 4 the analysis module will be considered and in section 5 the implementation will be discussed.

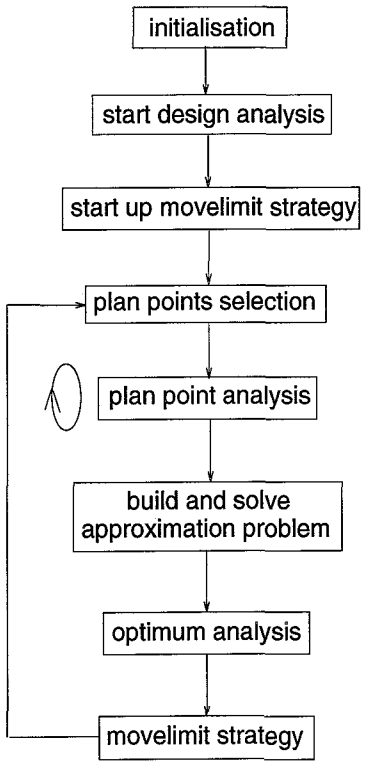


Figure 1: Block diagram of a multipoint approximate optimisation process.

3.1 Linear approximation models and plan points selection

In the literature it is concluded many times (e.g. Haftka & Gürdal, 1992, p.231 and Schittkowski et al., 1994) that sequential linear programming methods can be successfully applied to a wide range of optimisation problems. Bennett & Park (1991) and De Jager (1993) concluded that linear approximations, obtained by using large finite difference steps, have smoothing properties with respect to noise and local non-linearities. More generally speaking, the large finite difference steps can be considered as an experimental design for linear models: a non-optimal experimental design, which can be enhanced by application of the experimental design theory.

The experimental design theory has been developed for the planning of comprehensive physical experiments in order to reduce the number of required experiments, while preserving the amount of information which can be extracted from the experiments (Schoofs & Van Campen, 1991). This situation is very similar to that of structural optimisation, where the number of expensive numerical experiments has to be minimised (Schoofs, 1987). The general theory will be discussed, whereafter the D-optimal experimental design for linear approximation models will be briefly considered. At this point it should be noted that, for reasons of simplicity, we applied "finite difference experimental designs" in the first version of the implemented method.

Regression model

Constructing approximation models is finding functions which describe the response quantities as a function of the set of n design variables x:

$$y_j = y_j(x) \qquad j = 1,2,...,t \tag{3.1}$$

In the sequel we will consider only one response y_j and for brevity we omit the index j. To find the relation $y=y(x)$ we assume a mathematical model. Mostly a linear model will apply:

$$y = f^T(x)\, b + e = b_1 f_1(x) + ... + b_k f_k(x) + e \tag{3.2}$$

where the k model coefficients of column b are unknown. The functions $f_1(x),...,f_k(x)$ can be chosen linear and nonlinear. In most cases a polynomial is chosen. The variable e accounts for the stochastic and/or deterministic model error, inherent to every model assumption.

Model coefficients and response estimation

To estimate the k model coefficients, using function values of N design points ($N \geq k$), the following set of linear equations is defined:

$$y = X\hat{b} + e \tag{3.3}$$

where $y(N)$ is a vector of function values, $X(N,k)$ is a matrix of design variable values, $\hat{b}(k)$ is a vector of estimated model coefficients, and $e(N)$ is a vector of differences between real and model response in the design points used to construct the approximation models. Note that if N equals k, the model response is exact in these design points, and thus e is a zero-vector.

The model coefficients can be estimated by minimising the residual sum of squares:

$$RSS = e^T e = (y - X\hat{b})^T (y - X\hat{b}) \tag{3.4}$$

This gives:

$$\hat{b} = (X^T X)^{-1} X^T y \tag{3.5}$$

For each set of design variables values, x, the model response can be estimated by:

$$\hat{y} = f^T(x)\,\hat{b} \tag{3.6}$$

Accuracy of the estimates

A measure for the accuracy of $\hat{b}$ is the variance-covariance matrix:

$$V(\hat{b}) = E\left((\hat{b} - b)(\hat{b} - b)^T\right) = (X^T X)^{-1}\sigma^2 \tag{3.7}$$

where σ^2 is the variance of the response y. If unknown it can be estimated from:

$$\hat{\sigma}^2 = \frac{1}{N-k} RSS \tag{3.8}$$

For the response estimator $\hat{y}(x)$ the variance $V(\hat{y}(x))$ is used as a measure for its accuracy. From (3.6) and (3.7) follows:

$$V(\hat{y}(x)) = f^T(x)(X^T X)^{-1} f(x)\sigma^2 \tag{3.9}$$

Optimal experimental design

In the optimal experimental design theory, the accuracy of the estimated model coefficients and/or the variance of the response estimator are used as criteria to search for optimal experimental designs. In both accuracy measures the term $(X^TX)^{-1}$ occurs. For a particular approximation model (structure of X) an experimental design (filling of X) can be determined, which minimises this term and thus the variances. To judge the minimum of a matrix, a number of criteria been developed. The most important ones are: D-optimality (minimise $\det(X^TX)^{-1}$), G-optimality (minimise the maximum response variance), and V-optimality (minimise the average response variance).

Optimal experimental design for linear models

For linear approximations the number of model coefficients is equal to $1+n$ and the model function vector is $f^T(x)=[1\ x]^T$. Using a MATLAB program (appendix B) the D-optimal experimental design for three design variables is determined. The result is visualised in Figure 2. For the "finite difference experimental design" $\det(X^TX)^{-1} = 64$ and for the D-optimal experimental design 0.0039.

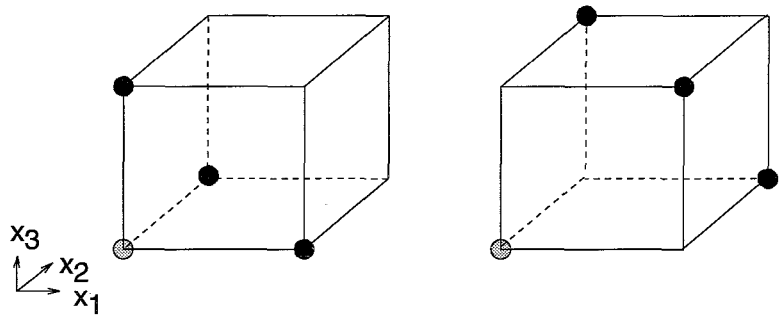


Figure 2: Finite difference and D-optimal experimental designs for three design variables.

3.2 Building and solving an approximate optimisation problem

In the midrange method suggested by Toropov et al. (1993) earlier analysed design points, close to the current search subregion, are added to the set of start and plan points, and weight factors are attributed to all points from this set. Coefficients of the approximation models of objective function and constraints are calculated by minimising the residual sum of squares, as considered in section 1. For reasons of

simplicity, we have chosen not to use already analysed design points and not to use weight factors in the first version of the implemented method. The number of start and plan points is therefore equal to the number of model coefficients and thus the model responses are exact in these design points (section 1). The resulting linear approximate optimisation problem is solved by a simplex algorithm, implemented by Press et al. (1992).

In solving coefficients from a system of equations, round off errors can have a significant effect on the solution accuracy. Scaling design variable values is an effective counter-measure to avoid these problems (Kessels, 1994). The following strategy scales lower and upper bound to 0 and 1, respectively:

$$x_{i,\text{scaled}} = \frac{x_i - slb_i}{sub_i - slb_i} \qquad i = 1,2,...,n \tag{3.10}$$

where slb_i and sub_i are the unscaled search subregion lower and upper bounds on the i-th design variable.

To avoid problems with a start search subregion which is completely in the infeasible region (no design that satisfies all constraints), an extra variable may be added to the set of design variables (Haftka & Gürdal, 1992). The model coefficients for this extra, so-called constraint relaxation variable should be chosen in such a way that an increasing value gives decreasing constraint values and a strongly increasing objective function value:

$$\hat{F} = f^T(x)\,\hat{b}_{obj} + b_{\text{relax},obj}\,x_{\text{relax}} \tag{3.11}$$

$$\hat{g}_j = f^T(x)\,\hat{b}_{g_j} + b_{\text{relax},g_j}\,x_{\text{relax}} \qquad j = 1,...,m \tag{3.12}$$

where

$$x_{\text{relax}} \geq 0\ , \quad b_{\text{relax},obj} > 0 \quad \text{and} \quad b_{\text{relax},g_j} < 0 \tag{3.13}$$

By doing so, the approximate optimisation problem will always have a solution within the search subregion, satisfying all approximate constraints. The next search subregion will be moved towards the feasible region.

3.3 Movelimit strategy

In case of linear approximations, the search subregions size is important for the convergence behaviour of the optimisation process (Etman et al., 1994). Large movelimits can cause the solution process to oscillate, while small movelimits slow down the convergence rate. For defining new search subregions, the movelimit strategy for multipoint methods consists of determining a search direction, determining movelimit factors, and calculating the new bounds.

Search direction

Contrary to local and midrange SPP methods, in midrange MPP methods the approximate optimum design of the current cycle is chosen as a point in a corner of the search subregion of the next cycle. From this point, the new search direction points to the centre of the new search subregion. This search direction is determined by the position of the optimum with respect to the centre of the current search subregion:

$$direc_i = \text{sign}\left(x_i - \frac{sub_i + slb_i}{2}\right) \qquad i = 1,2,...n \tag{3.14}$$

For two design variables, the "search direction strategy" is visualised in Figure 3.

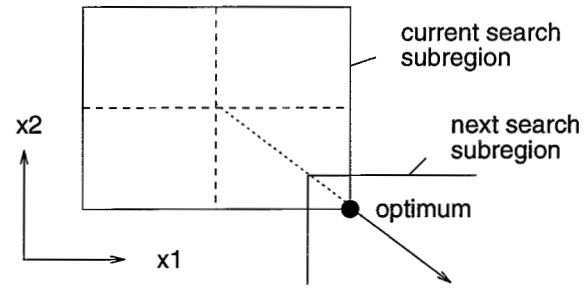


Figure 3: Example of the search direction strategy for two design variables.

Movelimit factors

For each optimum, approximate objective function and approximate constraint values can be compared with crash analysis values. The following relative errors give an idea

of the quality of the approximation models for the current search subregion:

$$e_F = \left| \frac{\tilde{F}(x_{opt}) - F(x_{opt})}{F(x_{opt})} \right| \tag{3.15}$$

$$e_{g_j} = \left| \frac{\tilde{g}_j(x_{opt}) - g_j(x_{opt})}{g_j(x_{opt})} \right| \qquad j = 1,2,...m \tag{3.16}$$

where a tilde denotes an approximate value and x_{opt} is the optimum design computed from the approximate optimisation problem.

The movelimit strategy as described and implemented by Etman et al. (1994) is used. It is presented in pseudo code in appendix C. An optimisation process is started with movelimits being 10 to 30 % of the design space. The movelimits are decreased or enlarged, depending on the calculated errors and on the convergence history. Herefore, eight user-defined levels on maximum approximation error and on maximum constraint violation were introduced:

- *errmax*: Maximum approximation error allowed during optimisation (40%)
- *errlrg*: Approximation error considered as large during optimisation (25%)
- *errsml*: Approximation error considered as small during optimisation (10%)
- *viomax*: Maximum constraint violation allowed during optimisation (10%)
- *violrg*: Constraint violation considered as large during optimisation (7.5%)
- *viosml*: Constraint violation considered as small during optimisation (5%)
- *objacc*: Desired accuracy objective function at final optimum design (0.1%)
- *vioacc*: Maximum constraint violation allowed, and desired accuracy of the constraints at final optimum design (0.1%)

During cycles with steady decrease of the objective function, the movelimit strategy tries to keep the errors between *errsml* and *errlrg*. Near the optimum, a higher accuracy is desired. The movelimits should be decreased whenever the convergence slows down or oscillations occur. An optimum design is rejected if the errors are too large (> *errmax*) or too high an infeasibility occurs (> *viomax*) starting from a feasible or nearly feasible design. Then a new cycle is started from the same starting design, with the same search direction, but with smaller movelimits.

Clearly, optimal settings for the accuracy criteria are problem dependent. However, for several smooth problems Etman et al. (1994) obtained good results with the settings

between brackets. In case of noisy functional behaviour a calculated response value is only an estimate for the "real" value (Figure 4). Therefore the criteria should be less strict. To get an idea of the bandwidth of the noise, several design points close to one another could be analysed. The relation between the bandwidth of the noise and the movelimit strategy criteria is discussed in appendix D.

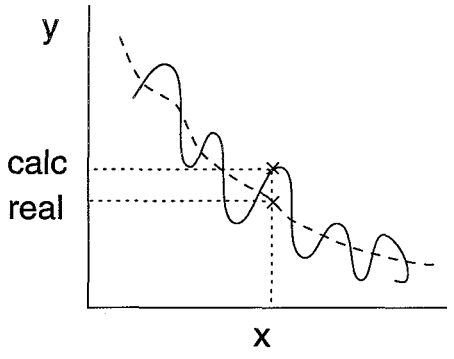


Figure 4: Calculated and "real" responses.

Calculate search subregion bounds

To give room for some extrapolation, start and plan points are not chosen in a vertex but a little (10%) inside the search subregion. If the search direction for the *i*-th design variable equals 1, search subregion bounds on that design variable are calculated by:

$$slb_i = x0_i - extrpf * mvlim_i \tag{3.17}$$

$$sub_i = x0_i + (1 - extrpf) * mvlim_i \tag{3.18}$$

and if the search direction equals -1:

$$slb_i = x0_i - (1 - extrpf) * mvlim_i \tag{3.19}$$

$$sub_i = x0_i + extrpf * mvlim_i \tag{3.20}$$

where *extrpf* is the user-defined extrapolation factor, $mvlim_i = mvlimf_i\ (ub_i - lb_i)$ is the movelimit on the *i*-th design variable. Parameter *mvlimf*(*i*) is the movelimit factor, and lb_i and ub_i are the design space bounds. An example is presented in Figure 5. If a calculated search subregion bound violates the design space bound, the search direction is reversed and new bounds are calculated.

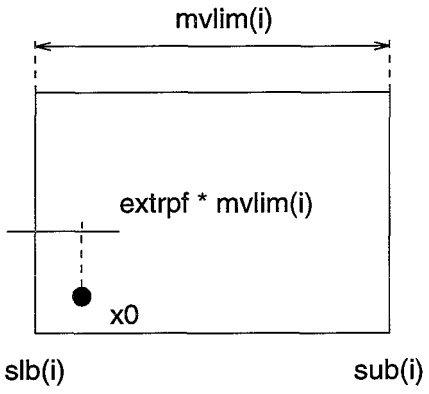


Figure 5: Example of calculating search subregion bounds for two design variables.

3.4 Analysis

The system to be analysed is specified in a MADYMO data deck. Such a data deck consists, besides the design variables, also of quantities that depend on the design variables. For example, the node coordinates of an airbag depend on the airbag diameter. The optimisation program has to be able to automatically generate a MADYMO data deck for each set of design variable values. After the actual MADYMO analysis, the results have to be translated into objective function and constraint values.

3.5 Implementation

The multipoint approximate optimisation process can be automated by a control program which sequentially carries out the different tasks of the block diagram of Figure 1. In this block diagram, the movelimit strategy is subdivided in two modules and there are three analysis modules, which are actually similar. In order to minimise the operating system and analysis tool dependent parts, the analysis module is taken as the centre of an optimisation cycle. The different tasks can be rearranged in such a way that before and after each analysis a task has to be performed (Figure 6). Block diagrams of the different modules are presented in appendix E.

Among other things, the start design and the accuracy settings for the movelimit strategy have to be specified in an input file for the optimisation program. To be able to automatically generate a MADYMO data deck (section 3.4) a parametrised data

deck has to be created. Instead of design variable values and values of the quantities which depend on the design variables, it has to consist of unique parameter codes. The dependence of each parameter on the design variables has to be specified in a user-defined subroutine. For each design point and for each optimisation cycle a database file is created. In a history file, the numbers of the next task, the last analysed design point, and the current cycle are stored.

In the before analysis 2 module the parametrised data deck is translated into a real data deck for the design point to analyse. The analysis results are translated into objective function and constraint values in the after analysis 2 module. After analysing an approximate optimum design, in the after analysis 1 module a new search subregion is defined. Within this region, plan points are selected in the before analysis 1 module. A file with the plan points numbers is created, whereafter these points are sequentially analysed. In the after analysis 1 module the approximate optimisation problem is built and solved, whereafter the resulting approximate optimum design is analysed. Until convergence has occurred new optimisation cycles are carried out.

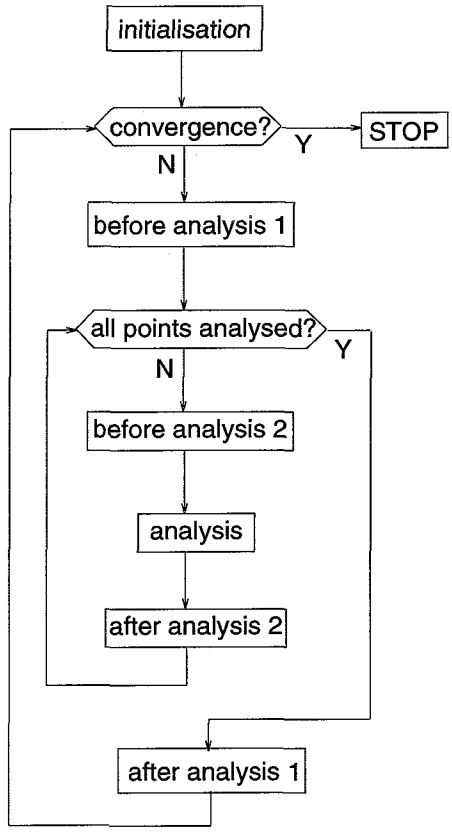


Figure 6: Block-diagram of the multipoint sequential linear programming method, with the analysis as the centre of an optimisation cycle.

4 Tests and applications

The multipoint sequential linear programming method is tested on three "two-design-variable" problems for which the optimum design is known in advance. The optimum of the analytical two-bar truss (section 1) is under-constrained. In other words, the optimum is defined by the curvature of a number of constraints, smaller than the number of design variables. The optimum of the analytical three-bar truss (section 2) is an example of a constrained optimum (number of active constraints equal to the number of design variables). For a certain crashworthiness design problem (section 3) a complete grid of design points has been analysed. The optimum of this problem is unconstrained (number of active constraints equal to zero). After these tests the program has been successfully applied to a crashworthiness design problem with six design variables, for which the optimum is not known in advance (section 4).

4.1 Two-bar truss

The two-bar truss (Figure 7) is a frequently used test problem (e.g. Toropov et al., 1993 and Kessels, 1994). The structure is loaded by an external force P, with P_x = 24.8 kN and P_y = 198.4 kN ($P_y = 8\ P_x$, $|P| = 200$ kN). Two design variables are defined: x_1 (cm^2) is the cross-section area of both bars, and x_2 (m) is half of the distance between nodes 1 and 2. The vertical coordinate of node 3 is fixed (1 m).

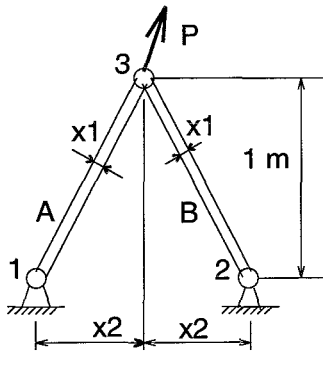


Figure 7: Two-bar truss.

The objective function is the weight of the structure and the constraints define stresses

in both bars, which must be less than 100 N/mm^2:

$$F = x_1\sqrt{1+x_2^2} \tag{4.1}$$

$$g_1 = 0.124\sqrt{1+x_2^2}\left(\frac{8}{x_1}+\frac{1}{x_1 x_2}\right) - 1 \le 0 \tag{4.2}$$

$$g_2 = 0.124\sqrt{1+x_2^2}\left(\frac{8}{x_1}-\frac{1}{x_1 x_2}\right) - 1 \le 0 \tag{4.3}$$

The lower and upper bounds on the design space are lb_1=0.2, lb_2=0.1, ub_1=4.0 and ub_2=1.6. A contour plot for this problem has been created by analysing a complete grid of design points (Figure 8). Note that the second constraint never becomes active, and that the optimum is defined by the curvature of the first constraint (under-constrained).

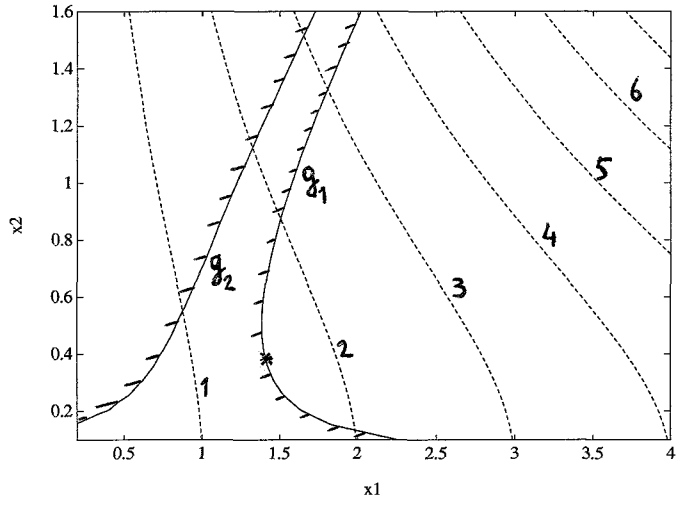


Figure 8: Contour plot two-bar truss: optimum at * (1.41 , 0.38) with F=1.51, constraint bounds (-), iso-objective function lines (--).

To simulate 10% noise, in x_1- and x_2-direction a high frequency sinus (amplitude 0.025) is added to the objective function and constraints. Both smooth and noisy problems have been optimised, starting from $x0$=(0.5 , 0.25) and $x0$=(2.5 , 1.0), and using start movelimits of 20%. For the smooth problem, Etman's accuracy settings (section 3.3) are used, and for the noisy problem the calculated settings of appendix D. Objective function values and maximum constraint violations of the initial and optimum designs are presented in Table 1, together with the numbers of optimisation cycles and analyses. More results of the optimisation processes are included in appendix F.

Two-bar truss	initial design		optimum design		num. cycl.	num. anal.
	F [kg]	viol. [%]	F [kg]	viol. [%]		
1-smooth	0.515	207	1.510	-5.327e-2	9	28
1-noisy	0.492	205	1.549	1.913	4	13
2-smooth	3.536	-36.9	1.509	1.198e-2	17	52
2-noisy	3.524	-38.1	1.621	2.368	4	13

Table 1: Two-bar truss optimisation results: problems 1 and 2 correspond with initial design $x0=(0.5 , 0.25)$ and $x0=(2.5 , 1.0)$, respectively.

For the smooth problems, the last stage of the convergence process elapses relatively slow (the difference between smooth 1 and 2 is considered accidental). In case of an under- or unconstrained optimum this is due to the inability of linear approximations to describe curvature, in combination with too large movelimits. For the problem with noise, however, the curvature of the first constraint does not define an optimum point but an optimum region. For the noisy problems convergence occurred before oscillating in this region. Therefore, it can be concluded that the relation between the bandwidth of the noise and the accuracy settings is well quantified.

4.2 Three-bar truss

The three-bar truss problem (Figure 9) is described by Etman & Van Houten (1994). This structure is loaded by an external force P, with $P_x = P_y$. The first design variable is the cross-section area of bars A and C. For reasons of symmetry these are chosen equal. The second design variable is the cross-section area of bar B.

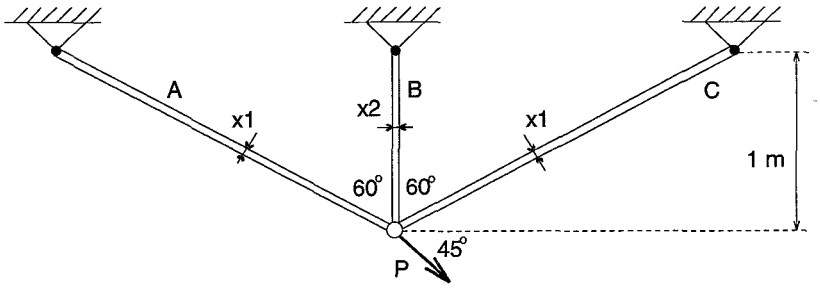


Figure 9: Three-bar truss.

Note that in bars A and B tension and in bar C compression will occur. The bounds on

tensile stresses are σ_0 and on compressive stresses 2/3 σ_0. If the normalised design variables are defined as:

$$x_1 = \frac{A_A \sigma_0}{P} \quad , \quad x_2 = \frac{A_B \sigma_0}{P} \tag{4.4}$$

the objective function and constraints can be formulated as:

$$F = 4x_1 + x_2 \tag{4.5}$$

$$g_1 = \frac{\sigma_A}{\sigma_0} - 1 = \frac{1}{2}\sqrt{2}\left(\sqrt{3}\frac{1}{3x_1} + \frac{1}{x_1 + 4x_2}\right) - 1 \leq 0 \tag{4.6}$$

$$g_2 = \frac{\sigma_B}{\sigma_0} - 1 = 2\sqrt{2}\frac{1}{x_1 + 4x_2} - 1 \leq 0 \tag{4.7}$$

$$g_3 = -\frac{\sigma_C}{\sigma_0} - 1 = -\frac{3}{4}\sqrt{2}\left(-\sqrt{3}\frac{1}{3x_1} + \frac{1}{x_1 + 4x_2}\right) - 1 \leq 0 \tag{4.8}$$

The design space bounds are lb_1=0.1, lb_2=0.1, ub_1=1.5 and ub_2=1.5. A contour plot for this problem is presented in Figure 10. Note that in the neighbourhood of the optimum point the bound on the first constraint is almost parallel to the iso-objective function lines. For the same problem with 10% noise added to the objective function and constraints, the region between the real optimum and the intersection of g_1 and g_3 can therefore be considered as the optimum region.

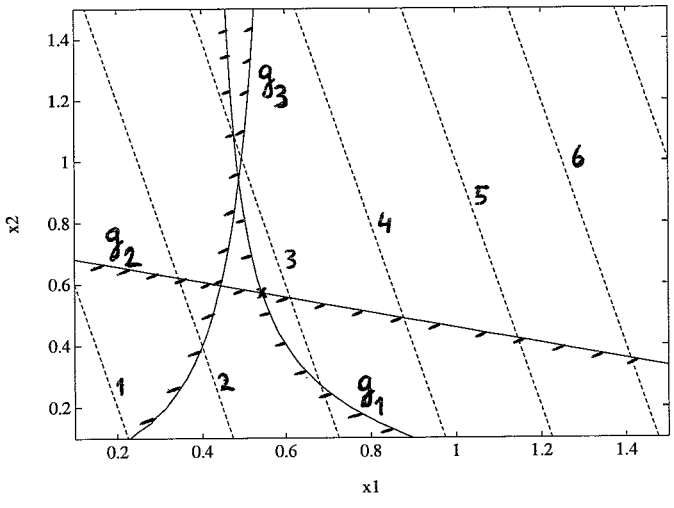


Figure 10: Contour plot three-bar truss: optimum at **x** (0.544 , 0.571) with F=2.747, constraint bounds (-), iso-objective function lines (--).

Both smooth and noisy problems have been optimised, starting from $x0$=(1.0 , 1.0) and $x0$=(0.2 , 0.2), using start movelimits of 20%. The same accuracy settings, as for the two-bar truss, are used. In table 2, objective function values and maximum constraint violations of the initial and optimum designs are presented, together with the numbers of cycles and analyses. More results are included in appendix G.

Three-bar truss	initial design		optimum design		num. cycl.	num. anal.
	F [kg]	viol. [%]	F [kg]	viol. [%]		
1-smooth	5.000	-43.4	2.748	1.039e-2	11	34
1-noisy	5.039	-39.5	2.911	-2.968	4	13
2-smooth	1.000	183	2.749	2.778e-2	5	16
2-noisy	0.978	181	2.848	-2.334	3	10

Table 2: Three-bar truss optimisation results: problems 1 and 2 correspond with initial designs $x0$=(1.0 , 1.0) and $x0$=(0.2 , 0.2), respectively.

For a constrained problem the convergence behaviour depends on the quality of the linear approximations for the optimum point. Generally, in multipoint approximations small movelimits are required in order to accurately approximate the optimums derivatives. Accidently, for the second start design, the derivatives are accurately approximated for relatively large movelimits. As expected, for the noisy problem, the optimisation processes are stopped before oscillating in the optimum region.

4.3 Full-scale frontal impact

In a confidential TNO report the crashworthiness performance optimisation of a full-scale frontal impact simulation (Figure 11) is described. The optimisation has been performed by analysing a complete grid of design points for two design variables of the airbag system. The resulting contour plot is presented in Figure 12. The first design variable is the vent diameter ($0.01 \leq x_1 \leq 0.065$ m) and the second is the airbag diameter ($0.4 \leq x_2 \leq 0.8$ m). By forcing the airbag gas through the vent, the occupants kinetic energy is dissipated.

A combined criterion of two head injury parameters is formulated as the objective function (F = HIC + head 3MS). Both HIC (Head Injury Criterion) and head 3MS (head 3 MilliSeconds criterion) values result from mathematical operations on the resultant

linear head acceleration. The by law obliged maximum injury parameter values form the constraints. In scaled form: g_1 = HIC/1000 - 1 ≤ 0 and g_2 = head 3MS/75g - 1 ≤ 0.

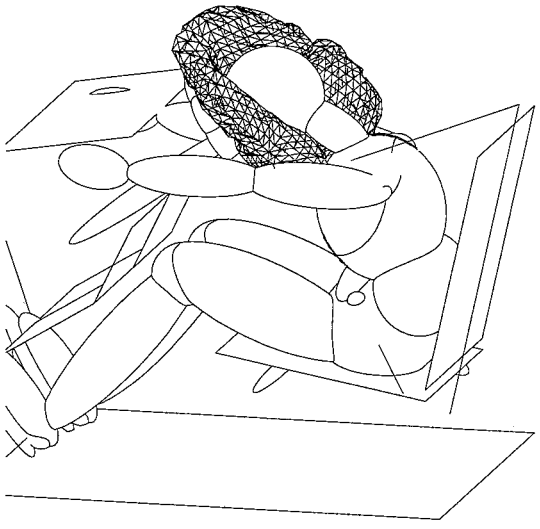


Figure 11: Full-scale frontal impact simulation model.

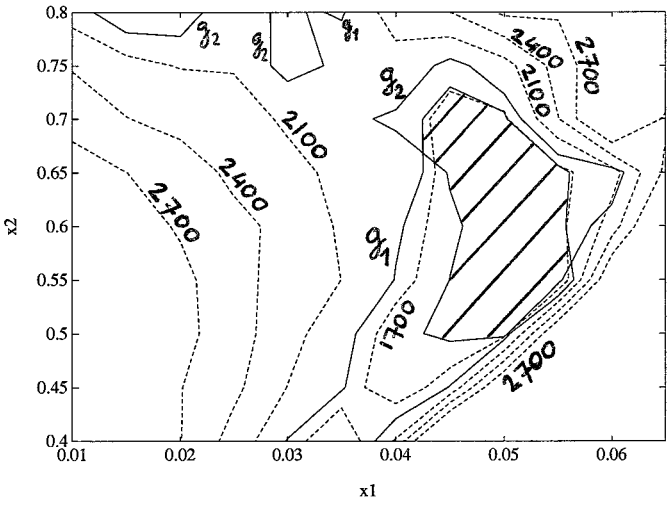


Figure 12: Contour plot airbag system 1: constraint bounds (-), iso-objective function lines (--), feasible region (///).

As mentioned in section 3.4, a MADYMO input deck may consist of quantities which depend on the design variables. In this problem the 1542 x- and y-coordinates of the airbag mesh depend on the airbag diameter.

The following accuracy settings have been used: errmax=0.5, errlrg=0.35, errsml=0.2, viomax=0.15, violrg=0.125, viosml=0.1, objacc=0.06 and vioacc=0.06. Starting from

$x0$=(2.5e-2 , 0.5) and $x0$=(5.5e-2 , 0.5) optimisations have been performed, using start movelimits of 20%. In two contour plots the optimisation processes are visualised (Figure 13). Objective function values and maximum constraint violations of the initial and optimum designs are presented in Table 3, together with the numbers of cycles and MADYMO analyses. In appendix H, among other things, animations for the second initial design and the accompanying optimum design are presented.

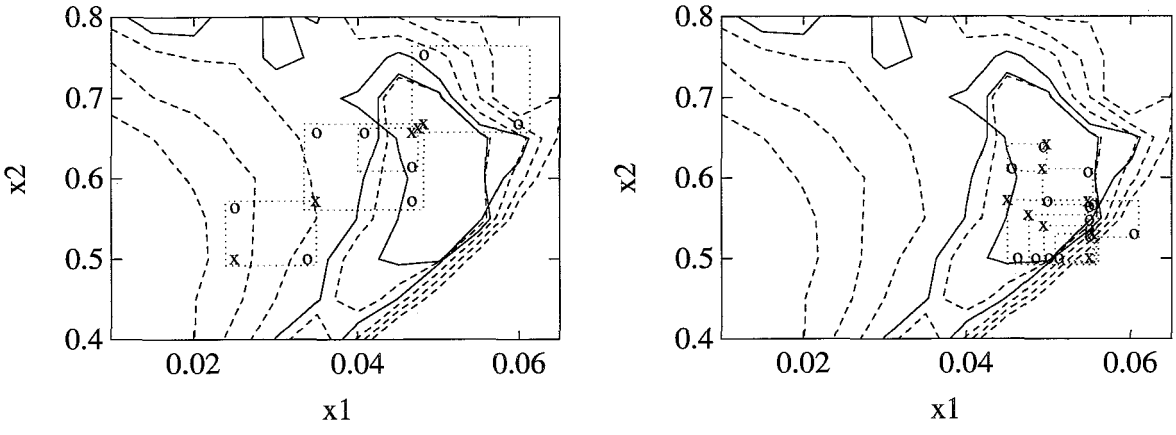


Figure 13: The different search subregions during the optimisation processes, started from $x0$=(0.025 , 0.5) (left) and $x0$=(0.055 , 0.5) (right), approximate optima (x), plan points (o).

Airbag system 1	initial design		optimum design		num. cycl.	num. anal.
	F [-]	viol. [%]	F [-]	viol. [%]		
run 1	2409	46.5	1334	-10.2	4	13
run 2	3454	134	1348	-7.92	7	22

Table 3: Airbag system 1 optimisation results: run 1 and 2 correspond with initial designs $x0$=(0.025 , 0.5) and $x0$=(0.055 , 0.5), respectively.

From the first start design, the approximate optimum converges in only four cycles to a design in the optimum region. In the neighbourhood of the second start design, the HIC and head 3MS behaviour is very noisy and even the global behaviour is very sensitive for design changes. As a consequence, the first three approximate optima have been rejected, whereafter the optimum region is reached in only four cycles. Both optimal designs easily satisfy the by law obliged maximum injury parameter values (maximum constraint violation in the order of -10%).

4.4 Full-scale off-set impact

In another confidential TNO report the investigation of a full-scale off-set impact simulation (Figure 14) is described. To optimise the crashworthiness performance, we defined six design variables. Because there is no idea about the position of the optimum, a large design space is defined:

- squared vent diameter (0.4e-3 $\leq x_1 \leq$ 4.9e-3 m^2)
- x-size of the airbag (0.2 $\leq x_2 \leq$ 0.45 m)
- y-size of the airbag (0.2 $\leq x_3 \leq$ 0.45 m)
- load limiter level (2.0e3 $\leq x_4 \leq$ 8.0e3 N)
- vent opening pressure (10.0e3 $\leq x_5 \leq$ 50e3 N/m^2)
- inflator gas mass (15.0 $\leq x_6 \leq$ 35.0 gram)

A device in the belt system assures that the belt forces do not exceed a certain value, the so-called load limiter level. The airbag is inflated with nitrogen. The larger the gas mass, the harder the airbag. The vent is opened for a certain pressure in the airbag.

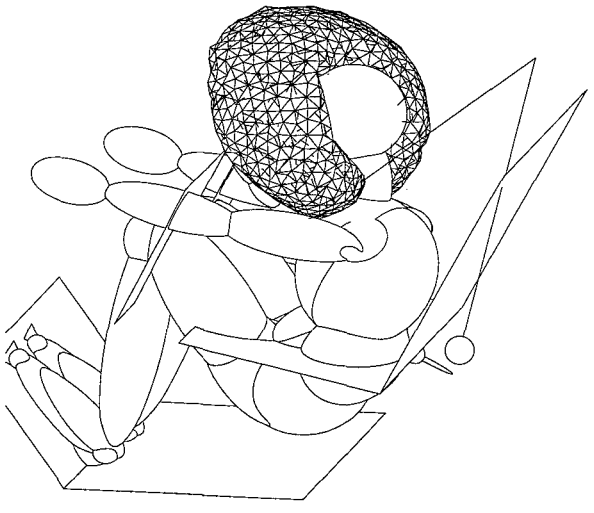


Figure 14: Full-scale off-set impact simulation model.

Four injury parameters strongly depend on the design variables: The chest 3MS (chest 3 MilliSeconds criterion) which results from a mathematical operation on the resultant chest acceleration signal, the chest deflection which is measured at the sternum, and the in section 4.3 already mentioned Head Injury Criterion (HIC) and head 3 Milli-

Seconds criterion (head 3MS). A combined criterion of these injury parameters is formulated as the objective function. The four injury parameters are considered of equal importance. Therefore, all parameters are weighed with their by law obliged maximum value:

$$F = \frac{\text{HIC}}{1000} + \frac{\text{head 3MS}}{75\text{g}} + \frac{\text{chest 3MS}}{60\text{g}} + \frac{\text{chest def.}}{3*0.0254} \tag{4.9}$$

Besides the four injury parameters, the distances head - steering wheel and chest - steering wheel are formulated as constraints. The bounds on the injury parameters are taken more strictly than the by law obliged maximum values:

$$g_1 = \frac{\text{HIC}}{483.3} - 1 \le 0 \qquad g_4 = \frac{\text{chest 3MS}}{346.2} - 1 \le 0$$

$$g_2 = \frac{\text{head 3MS}}{400} - 1 \le 0 \qquad g_5 = \frac{\text{chest def.}}{0.0367525} - 1 \le 0$$

$$g_3 = 1 - \frac{\text{head dis.}}{0.17} \le 0 \qquad g_6 = 1 - \frac{\text{chest dis.}}{0.155} \le 0$$

An optimisation is performed, starting with a large airbag ($x_2 = x_3 = 0.4$ m) and a large quantity of gas ($x_6 = 30.0$ gram). Because good results were obtained for the first crashworthiness problem, the same accuracy settings and start movelimits have been used: errmax=0.5, errlrg=0.35, errsml=0.2, viomax=0.15, violrg=0.125, viosml=0.1, objacc=0.06, vioacc=0.06, and start movelimits of 20%. The design variable values of the initial and optimum designs are presented in Table 4. The progress of the objective function and maximum constraint violation is visualised in Figure 15. Nine optimisation cycles (64 analyses) have been carried out. One approximate optimum was rejected and is therefore not plotted. In appendix I, among other things, animations for the initial design and the accompanying optimum design are presented.

	x1	x2	x3	x4	x5	x6
initial	2.290e-3	0.400	0.400	4520	26800	30.00
optimum	3.748e-3	0.328	0.345	2913	10000	35.00

Table 4: Airbag system 2: design variable values of the initial and optimum design.

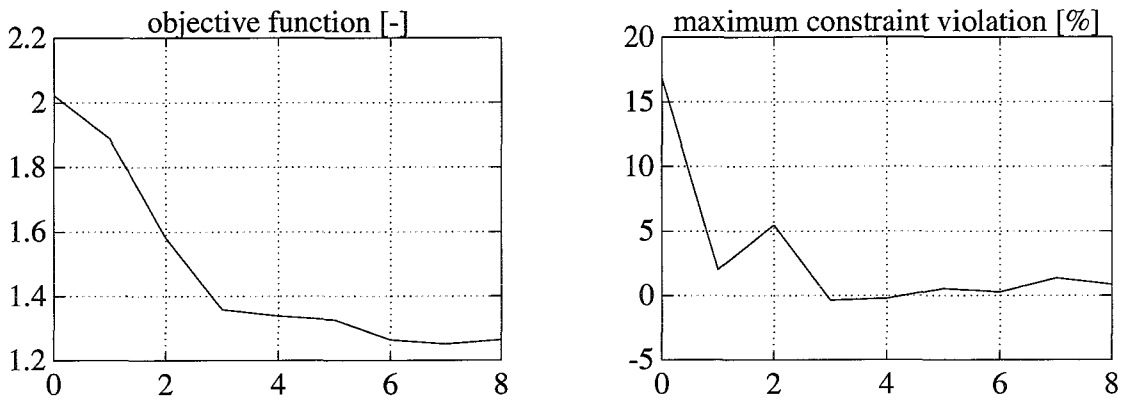


Figure 15: Airbag system 2: progress of the objective function value and maximum constraint violation during the optimisation process.

At the optimum design, g_6 is active (minimum distance chest - steering wheel zero), x_6 is at the design space upper bound (full airbag), and x_5 is at the design space lower bound (vent opens as quickly as possible). Therefore, it can be concluded that the deceleration of the occupant is optimally distributed.

The optimum safety measures design is very effective for the average male in the correct driving position. However, if this average male is out of the correct position, the airbag might possibly react to "powerful". The same problem occurs for small people, and for tall people the airbag might possibly not react powerful enough. The solution to this problem is multi-model optimisation.

5 Conclusions and recommendations

In crashworthiness design optimisation problems, the objective function and constraints often contain more than 10% noise. For such problems, the developed multipoint sequential linear programming method is well capable of converging in only a few optimisation cycles to an optimum, without oscillating in the bandwidth of the noise. For a full-scale off-set impact simulation with six design variables an optimum design, satisfying all by law obliged maximum injury parameter values, resulted after nine optimisation cycles.

The linear approximation models for objective function and constraints, obtained by using large finite difference steps, are well capable of describing the global functional behaviour in a subregion of the design space. These finite difference experimental designs for linear models can be enhanced by application of the experimental design theory (section 3.1). Toropov et al. (1993) add some of the earlier analysed design points to the set of start and finite difference points, and attribute weight factors to all points from this set, in order to improve the accuracy of the linear approximation models. The effectiveness of this approach in combination with enhanced experimental designs should be investigated, whereafter implementation can be considered. Instead of one by one calculation of the model coefficients, they should then be calculated from a system of algebraic equations.

The functional behaviour of objective function and constraints can be linearised by introducing inter-mediate response variables (section 2.1). If well defined by the user, movelimits can be enlarged, and as a result the optimum region will be reached in less opitmisation cycles.

The sensitivities of objective function and constraints for small design changes are important quantities of the optimum. Often a design with a higher objective function value but low design sensitivities is preferred to a design with a lower objective function value but high design sensitivities. After the optimisation process has converged design sensitivities can be determined, using close to the optimum located design points.

To be able to automatically calculate constraint approximation errors, bounds on "≤-constraints" have to be scaled to 1 and on "≥-constraints" to -1. In the first version ≤-constraints can be handled, but for ≥-constraints the subroutine where approximation

errors are calculated becomes problem dependent. It is recommended to implement the possibility to automatically calculate the approxiamtion errors for both types of constraints.

In sequential approximate optimisation, the movelimit strategy is decisive for the methods efficiency. The first version is very effective in finding an optimum for noisy problems. However, if an approximate optimum is rejected, new plan points, with smaller finite difference steps, are selected and analysed. This is not very efficient, because many analyses are performed to find a next approximate optimum (e.g. run 2 for the full-scale frontal impact simulation, section 4.3). Possibly, the efficiency can be improved with a flexible extrapolation factor. If a cycle is started with a large value, it can be decreased if the approximate optimum is rejected, whereafter a new approximate optimisation can be performed, using the same start and plan points. Note that this is only useful if the approximate optimum is located in the extrapolation region. Another possible improvement is to accept all approximate optima (Toropov et al., 1993). Both proposed solutions require further investigation.

The relation between the accuracy settings for the movelimit strategy and the bandwidth of the noise is well quantified, as indicated by the two- and three-bar truss results. If for a certain problem, however, no proper estimate for the bandwidth of the noise is available, it is advised to overestimate this bandwidth. Depending on the results one can decide to consider the optimisation process converged or to decrease the settings and to start a new optimisation process from the optimum.

For all optimisation problems good results were obtained with start movelimits, being 20% of the design space. However, it can be reasoned that in case of a larger quantity of noise, the global functional behaviour will be described more accurately for a larger subregion.

The solution to the problem that the optimum safety measures design is only optimally effective for the average male in the correct driving position (section 4.4) is multi-model optimisation. Constraint values for different dummies and dummy positions can be obtained by performing crashworthiness analyses for all combinations. As an objective function a combined criterion of injury parameters of all combinations can be formulated. If the number of constraints strongly increases it could be considered to apply a more advanced linear programming method.

References

Barthelemy, J.F.; Haftka, R.T. (1993) Approximation concepts for optimum structural design - a review. *Structural Optimization*, 5, 129-144.

Bennett, J.A.; Park, G.J. (1991) Automotive occupant dynamics optimization. *ASME Advances in Design Automation*, DE 32-1, 1-7.

Bosio, A.C.; Lupker, H.A. (1991) Design of experiments in occupant simulation. *SAE technical paper series*, 910891.

Etman, L.F.P.; Houten, M.H. van (1994) Optimisation using approximation concepts, (In Dutch). *Structural optimisation course*, Eindhoven University of Technology.

Etman, L.F.P.; Thijssen, E.J.R.W.; Schoofs, A.J.G.; Campen, D.H. van (1994) Optimization of multibody systems using sequential linear programming. *ASME Advances in Design Automation*, DE 69-2, 525-530.

Haftka, R.T.; Gürdal, Z. (1992) *Elements of structural optimization.* Kluwer, Dordrecht.

Jager, J.A. de (1993) Optimization of restraint system design using mid-range approximation concepts. *WFW report*, 93.183, Eindhoven University of Technology.

Kessels, P.H.L. (1994) Development and implementation of a mid-range approximation method. *WFW report*, 94.161, Eindhoven University of Technology.

Klink, M.B.M. (1991) Optimization of a child-seat with CADE and MADYMO, (In Dutch). *WFW report*, Eindhoven University of Technology.

Nelder, J.A.; Mead, R. (1964) A simplex method for function minimization. *Computer Journal*, 7, 308-313.

Press, W.H.; Terukolsky, S.A.; Vetterling, W.T. et al. (1992) *Numerical recipes in Fortran: the art of scientific computing.* Cambridge University.

Schittkowski, K.; Zillober, C.; Zotemantel, R. (1994) Numerical comparison of nonlinear programming algorithms for structural optimization. *Structural Optimization*, 7, 1-19.

Schoofs, A.J.G. (1987) *Experimental design and structural optimization.* Doctoral Dissertation, Eindhoven University of Technology.

Schoofs, A.J.G.; Campen, D.H. van (1991) Approximation methods in optimization using design and analysis of numerical experiments. *Optimization of structural systems and industrial applications*, 91, 183-194.

Toropov, V.V. (1989) Simulation approach to structural optimization. *Structural Optimization*, 1, 37-46.

Toropov, V.V.; Filatov, A.A.; Polynkin, A.A. (1993) Multiparameter structural optimization using FEM and multipoint approximations. *Structural optimization*, 6, 7-14.

Vanderplaats, G.N. (1979) Approximation concepts for numerical airfoil optimization. *NASA technical paper*, 1370.

Appendix A: Injury parameter calculations

The problems in injury parameter calculations will be explained for two of the most important head injury parameters for frontal crashes: the Head Injury Criterion (HIC) and the head 3 MilliSeconds criterion (head 3MS). The following cases will be successively discussed: 1) solving the equations of motion, 2) calculating the injury parameter values, 3) possible errors, due to the numerical solution method, 4) possible discontinuities, due to the definition and the problem itself, and 5) results and discussion.

Solve the equations of motion

For MADYMO simulations using the finite element module only a fourth order Runge-Kutta integration method with a fixed integration time step is available. This fixed time step is user defined.

After every integration time step the numerical solution will differ a little from the exact solution (local truncation error). These small errors are transmitted in the following integration steps, resulting in a global truncation error.

Calculate injury parameter values

The Head Injury Criterion (HIC) is given by:

$$HIC = \max_{T0 \le t_1 \le t_2 \le TE} \left[\frac{1}{t_2 - t_1} \int_{t_1}^{t_2} R(t)\,dt \right]^{2.5} (t_2 - t_1) \qquad (A.1)$$

where $T0$ is the starting time of the simulation, TE is the end time of the simulation, $R(t)$ is the resultant linear head acceleration in g's (measured at the head's centre of gravity) over the time interval $T0 \le t \le TE$, and t_1 and t_2 are the initial and final time point of the interval during which the HIC attains a maximum value. The maximum interval is 36 milliseconds.

The 3MS criterion is defined as the highest acceleration level with a duration of at least 3 milliseconds. It is computed like the HIC, but with a 3 ms time window.

Errors in the functional behaviour of injury parameters as a function of the design variables, caused by the numerical solution method

The global truncation error in the acceleration signal causes an error in the HIC and 3MS values. This error can be positive as well as negative. This may cause a noisy behaviour of the HIC and 3MS as a function of the design variables.

The size of the window for calculating an HIC-value is equal to a round number of times the output time step (*TSOUT*). This size will generally differ from the size with which the "real" HIC-value would be calculated. The calculated HIC-value will thus always be smaller than the "real" HIC-value. However, *TSOUT* is much smaller than the time window width. Its influence will therefore be small.

Discontinuities in the functional behaviour of injury parameters as a function of the design variables, caused by definition and problem itself

In HIC and 3MS behaviour as a function of the design variables two different kinds of discontinuities might be present. The first one is caused by the fact that after each simulation the HIC and 3MS value will generally be found between two different time points. The second one is caused by the fact that a (small) design change might result in a hard contact situation (e.g. occupant - steering wheel).

Generally, discontinuities of the first kind will appear. Discontinuities of the second kind are characteristic for crash simulations and cannot be avoided.

Results and discussion

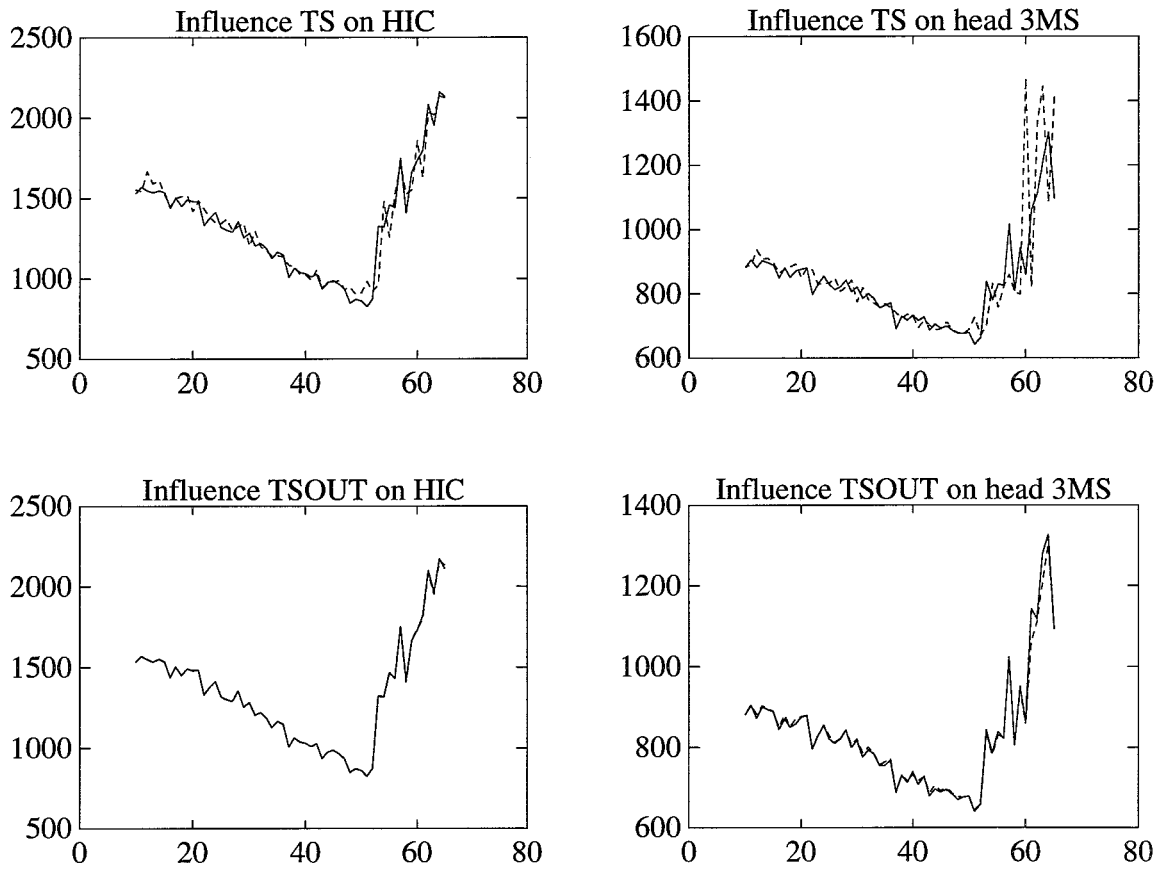
For the crashworthiness design problem described by Van Slagmaat (1995) the vent diameter of the airbag system is varied between 10 mm and 65 mm with steps of 1 mm. To show the importance of the influence of *TS* two series of simulations with the same *TSOUT* and different *TS* are compared. Vice versa for the influence of *TSOUT*.

- Influence of *TS*:
  1) *TS* = 0.2e-4, *TSOUT* = 2.0e-4.
  2) *TS* = 2.0e-4, *TSOUT* = 2.0e-4.

- Influence of *TSOUT*:
  3) *TS* = 0.2e-4, *TSOUT* = 0.2e-4.
  4) *TS* = 0.2e-4, *TSOUT* = 2.0e-4.

The results of the first and fourth series of simulations, which are the same, are obtained by a rerun of the injury parameter calculations, using the results of the third series of simulations.

The results are presented in the figures below, where on the x-axes the vent diameter in mm. In the upper left and upper right figure the solid lines are the results of the first series of simulations and the dashed lines of the second. In the lower left and lower right figure the solid lines are the results of the third and the dashed lines of the fourth series of simulations.



Obviously, the influence of *TSOUT* is negligible. As expected its influence on the head 3MS is largest, because here a smaller time window is used.

It is clear that *TS* has influence on both HIC and head 3MS values. However, the bandwidth of the noise does not differ so much for both levels, except for the head 3MS behaviour for large vent diameters. Therefore we conclude that the discontinuities caused by the fact that after each simulation the HIC and head 3MS values will generally be found between two different time points are also important.

Appendix B: DESOFEXP.M

```
% MATLAB program for determining the D-optimal experimental design
% for three design variables.
%
% J.M.T.A. Adriaens (March 26, 1995)

% number of design variables
ndv = 3;

% number of levels of variation
nlv = 2;

% linear approximation model:
% number of model coefficients = 1 + number of design variables
nmc = 1 + ndv;

x1min = -1; x1max = 1;
x2min = -1; x2max = 1;
x3min = -1; x3max = 1;

x1 = x1min:(x1max-x1min)/(nlv-1):x1max;
x2 = x2min:(x2max-x2min)/(nlv-1):x2max;
x3 = x3min:(x3max-x3min)/(nlv-1):x3max;

% full factorial
FF = [];
for k = 1:nlv
  for l = 1:nlv
    for m = 1:nlv
      FF = [FF ; 1 x1(k) x2(l) x3(m)];
    end
  end
end

% design of experiments (fraction of the full factorial)
X = [];
p = 1;
FM = 1e25;
R = [];
for k = 2:nlv^ndv-nmc+2
  for l = k+1:nlv^ndv-nmc+3
    for m = l+1:nlv^ndv-nmc+4
      X = [FF(1,:) ; FF(k,:) ; FF(l,:) ; FF(m,:)];
      MD = X'*X;
      if rank(MD) == 4
         F = det(inv(MD));
         if F < FM-eps
            XM = X;
            FM = F;
            R = [p F];
         elseif F >= FM-eps & F <= FM+eps
            R = [R ; p F];
         end
       end
       p = p+1;
    end
  end
end
```

Appendix C: Movelimit strategy

In pseudo code the applied movelimit strategy, developed by Etman et al. (1994).

```
Check whether the approximate optimum will be accepted.
if ( (error > errmax) or ((maxg > viomax) and (actrlx = no)) ) then
   do not accept optimum, start new cycle, use same starting design, same
   search direction, but smaller movelimit factors.
   f(i) = f(i) * 3/4
else
   accept optimum, check convergence.
   if ( ( (Fchg < objacc) and (maxg < vioacc) ) and
      ( (Faperr < objacc) and (gaperr < vioacc) ) ) then
      Convergence occurred, stop optimisation process.
   else
      No convergence, determine new movelimit factors.
      for i = 1:1:n
         if ( (xdrchg(i) < 0) and ( ( (Fdrchg < 0) or (fchg < (4.0*objacc)) ) and
            (actrlx = no) ) ) then
            f(i) = f(i) /2
         elseif ((Fopt > F0) and (actrlx = no)) then
            f(i) = f(i) * 3/4
         else
            if ( (actmvl(i) = yes) and (actbnd(i) = no) ) then
               if ( (error < errsml) and ((maxg < viosml) or (actrlx = yes)) ) then
                  f(i) = f(i) * 4/3
               elseif ( (error > errlrg) or ((maxg > violrg) and (actrlx = no)) ) then
                  f(i) = f(i) * 3/4
               end
            else
               if ( (error > errlrg) or ((maxg > violrg) and (actrlx = no)) ) then
                  f(i) = f(i) * 3/4
               end
            end
         end
      end
   end
end
```

abbreviations:

```
n          : number of design variables
actbnd(i)  : indicator defining whether a design space bound is active or not
actmvl(i)  : indicator defining whether a movelimit is active or not
actrlx     : indicator defining whether constraint relaxation is active or not
Faperr     : objective function approximation error at the approximate optimum
gaperr     : maximum constraint approximation error at the approximate optimum
error = max(Faperr,gaperr)
           : maximum approximation error
Fopt       : objective function value of the approximate optimum of the current cycle
F0         : objective function value of the start point of the current cycle
Fprv       : objective function value of the start point of the previous cycle
dFcur = Fopt-F0
           : objective function value change during current cycle
dFprv = F0-Fprv
           : objective function value change during previous cycle
Fdrchg = dFcur*dFprv
           : direction change in objective function value
Fchg = abs((Fopt-F0)/F0)
           : objective function value change
maxg       : maximum constraint violation at the approximate optimum
xopt       : design variable values of the approximate optimum of the current cycle
x0         : design variable values of the start point of the current cycle
xprv       : design variable values of the start point of the previous cycle
dxcur = xopt(i)-x0(i)
           : design variable change during current cycle
dxprv = x0(i)-xprv(i)
           : design variable change during previous cycle
xdrchg(i) = dxcur*dxprv
           : direction change in design variable values
f(i)       : on entry: movelimit factors of the current cycle
             on exit : movelimit factors of the next cycle
```

Appendix D: Noise and accuracy criteria

In case of noisy functional behaviour the calculated response y is an estimate for the "real" response y_{real}:

$$y - \frac{bwn}{2} \leq y_{real} \leq y + \frac{bwn}{2} \tag{D.1}$$

where *bwn* is the bandwidth of the noise. In this appendix we suppose that this bandwidth is a constant fraction of the response value (α = *bwn* / y_i). Half of the bandwidth is denoted as *hbwn*.

The first three criteria relate to the maximum approximation error, the second three to the maximum constraint violation, and the last two to the convergence. These three sets will be successively discussed.

Maximum approximation error criteria

A response aproximation error is calculated by:

$$ae_y = \left| \frac{\tilde{y} - y}{y} \right| \tag{D.2}$$

where the tilde denotes the approximate value. This calculated approximation error is an estimate for the real value:

$$\left| \frac{\tilde{y} - (y + hbwn)}{y + hbwn} \right| \leq ae_y \leq \left| \frac{\tilde{y} - (y - hbwn)}{y - hbwn} \right| \tag{D.3}$$

If the lower bound on the estimate is smaller than for example *errsml* it can be considered a small approximation error:

$$\left| \frac{\tilde{y} - (y + hbwn)}{y + hbwn} \right| < errsml \tag{D.4}$$

Filling in of *hbwn* / $y = \alpha/2$ and working out gives:

$$\left| \frac{\tilde{y} - y}{y} \right| < errsml + \frac{\alpha}{2}(1 + errsml) \tag{D.5}$$

Note that the left hand side is the calculated approximation error. The same goes for *errmax* and *errlrg*.

Maxium constraint violation criteria

Because constraint bounds are scaled to one, the calculated response value minus 1 is an estimate for the constraint violation. If the lower bound on the estimated response

minus 1 is smaller than for example *viosml* it can be considered a small constraint violation:

$$y - hbwn - 1 < viosml \tag{D.6}$$

Filling in of *hbwn* / $y = \alpha/2$ and working out gives:

$$y - 1 < \frac{1}{1-\alpha/2}\,(viosml + \alpha/2) \tag{D.7}$$

Note that the left hand side is the calculated constraint violation The same goes for *viomax* and *violrg*.

Convergence criteria

Convergence occurs if the relative change of the objective function is smaller than *objacc* in combination with a constraint violation smaller than *vioacc*, and the objective function approximation error smaller than *objacc*, and the maximum constraint approximation error smaller than *vioacc*.

In case of noise, two response values which only differ the bandwidth of the noise can be considered equivalent. Therefore, the criterion for the absolute change in the response value is:

$$|y_i - y_{i-1}| < bwn \tag{D.8}$$

Filling in of *bwn* / $y = \alpha$ and working out gives:

$$\left|\frac{y_i - y_{i-1}}{y_{i-1}}\right| < \alpha \tag{D.9}$$

Note that the left hand side is the calculated relative change of the response. The *objacc* criterion is also used for the objective function approximation error at the final optimum design. However, the latter is smaller for each α.

The setting of *vioacc* is also influenced by the noise. If we consider the maximum constraint approximation error the new setting would be:

$$vioacc + \frac{\alpha}{2}(1 + vioacc) \tag{D.10}$$

Considering the maximum constraint violation the new setting would be:

$$\frac{1}{1-\alpha/2}\,(vioacc + \alpha/2) \tag{D.11}$$

For each α the second setting is largest

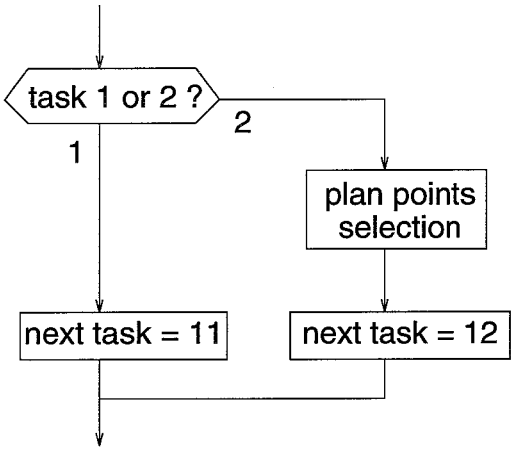
Summary

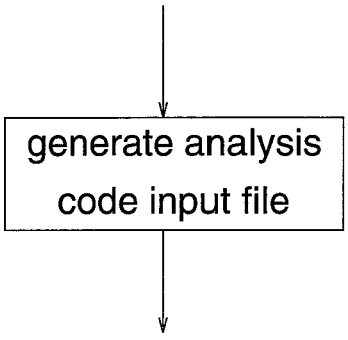
The settings in case of relative noise α, the "smooth" value and new value for α=0.1:

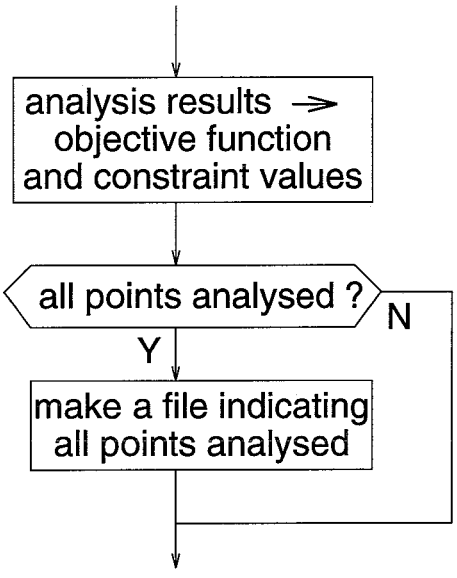
- *errmax* = *errmax* + α/2 (1+*errmax*) (0.4 → 0.47)
- *errlrg* = *errlrg* + α/2 (1+*errlrg*) (0.25 → 0.313)
- *errsml* = *errsml* + α/2 (1+*errsml*) (0.1 → 0.155)
- *viomax* = (*viomax*+α/2) / (1-α/2) (0.1 → 0.158)
- *violrg* = (*violrg*+α/2) / (1-α/2) (0.075 → 0.132)
- *viosml* = (*viosml*+α/2) / (1-α/2) (0.05 → 0.105)
- *objacc* = α (0.001 → 0.1)
- *vioacc* = (*vioacc*+α/2) / (1-α/2) (0.001 → 0.054)

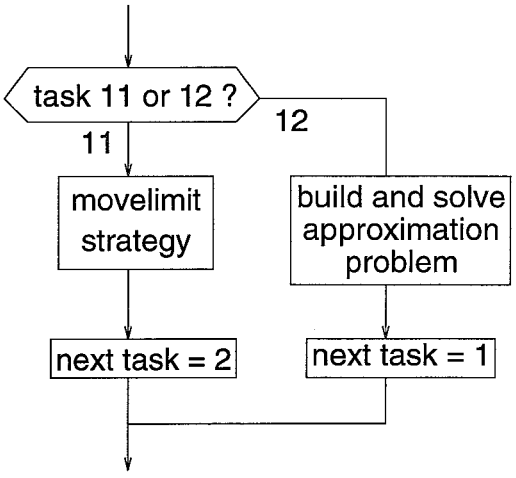
Appendix E: Program modules

In the upper left figure the before analysis 1, in the upper right the before analysis 2, in the lower left the after analysis 2, and in the lower right the after analysis 1 module.









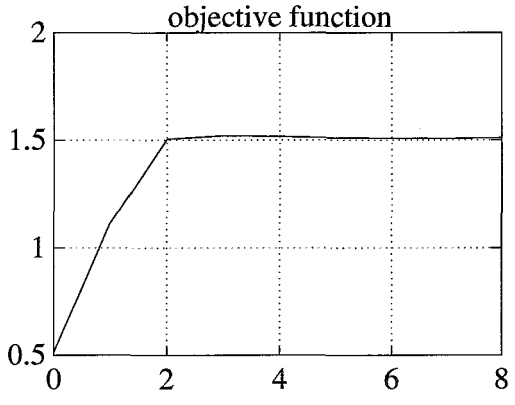
Appendix F: Two-bar truss results

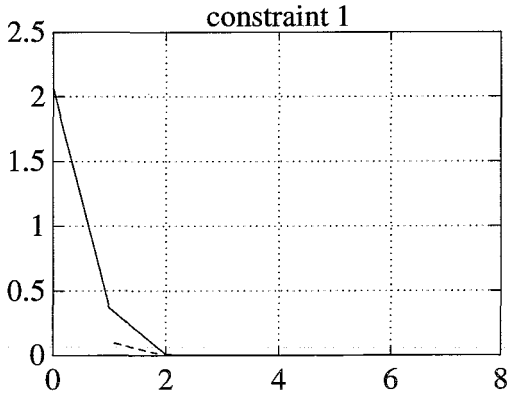
Two-bar truss, $x0$ = (0.5 , 0.25)

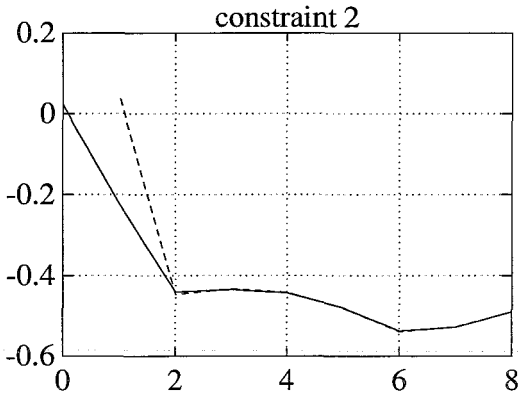
Table of objective function values, maximum constraint violations (%), maximum approximation errors (%), and design variable values for all optimisation cycles.

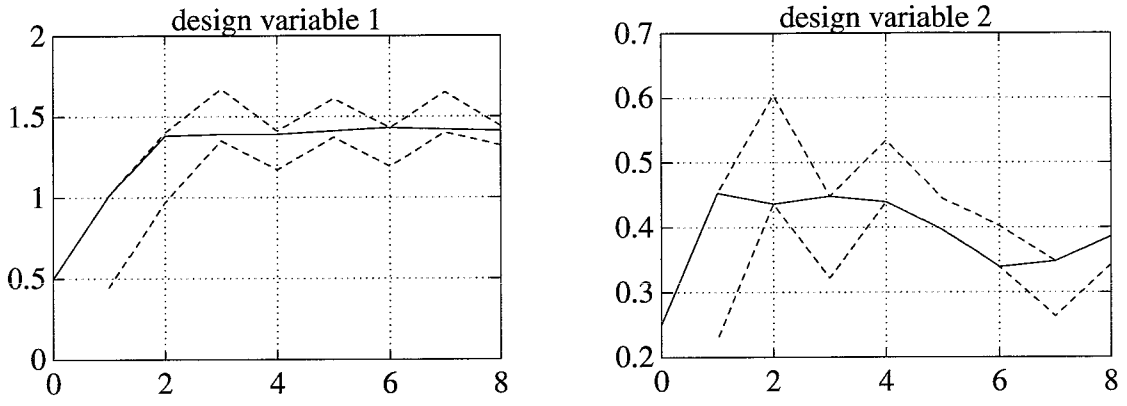
cyc	obj	viol	err	x1	x2
0	0.515	2.068e2		0.500	0.250
1		0.208e2	0.424e2		
2	1.112	0.372e2	0.346e2	1.013	0.453
3	1.505	9.413e-1	1.128	1.380	0.436
4	1.521	1.431e-1	3.800e-1	1.388	0.448
5	1.518	1.146e-1	1.144e-1	1.390	0.439
6	1.511	-9.536e-2	1.002e-1	1.405	0.396
7	1.509	3.388e-1	3.830e-1	1.429	0.339
8	1.508	2.258e-1	2.875e-1	1.425	0.348
9	1.510	-5.327e-2	5.329e-2	1.409	0.386

Plots of the progress of: 1) objective function, 2) exact (-) and approximate (--) constraints, and 3) optimum design variables (-) and search subregion lower and upper bounds (--). On the x-axes the optimisation cycle number.

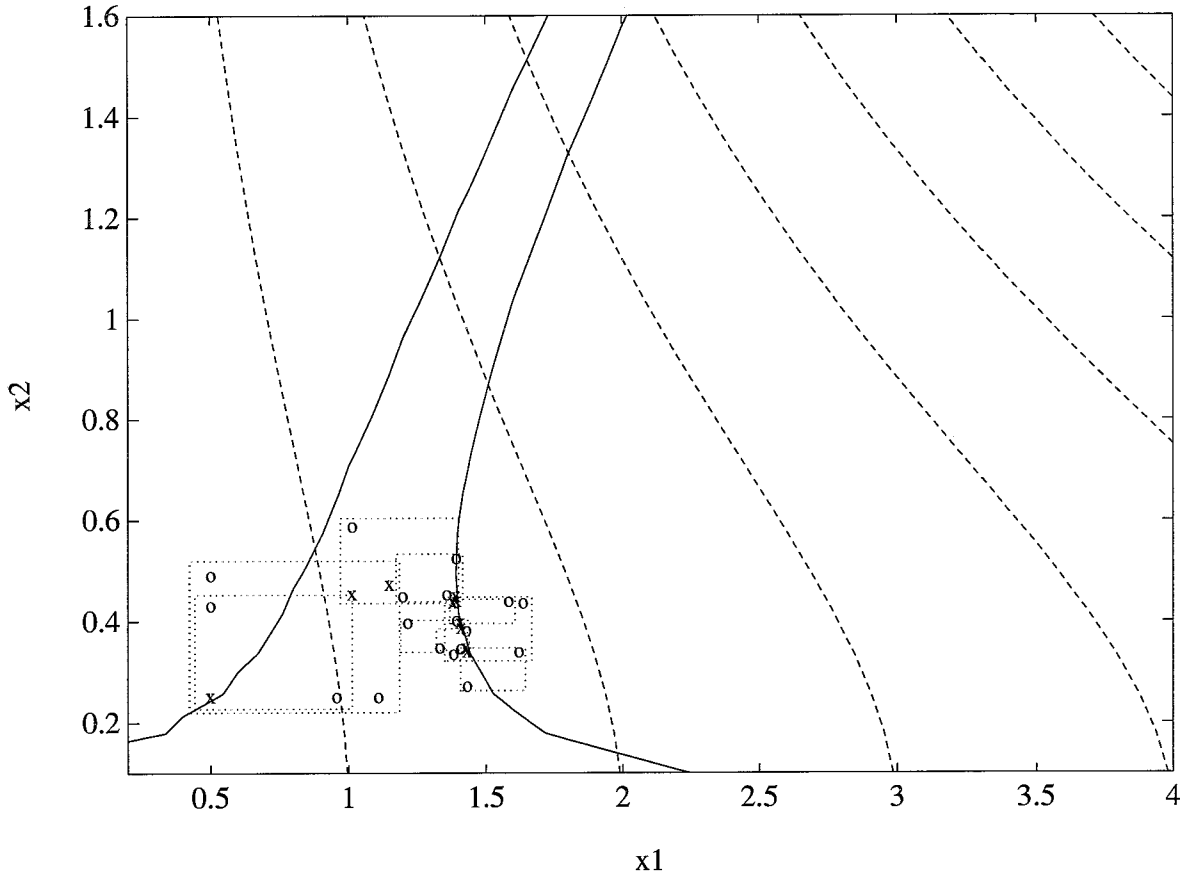






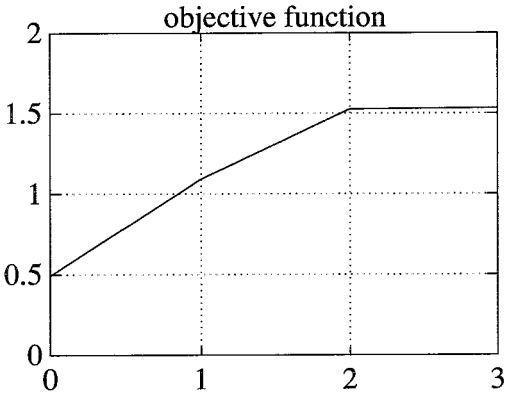


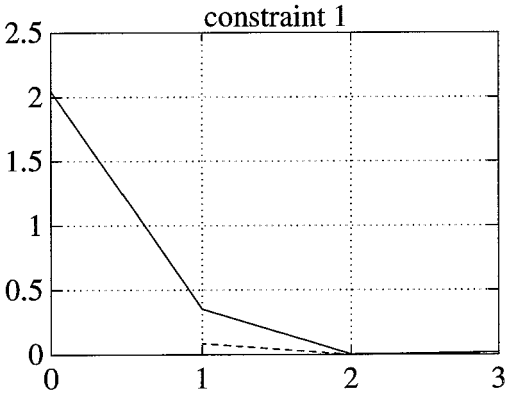
Plot of the different search subregions during the optimisation process.

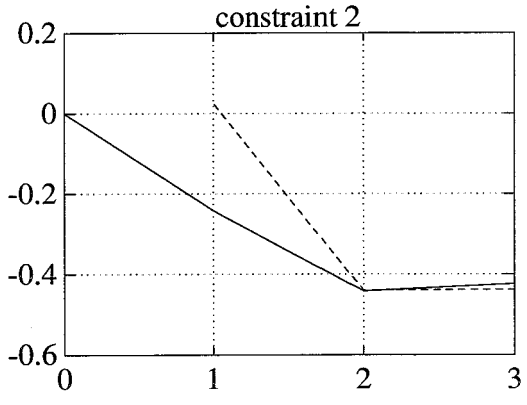


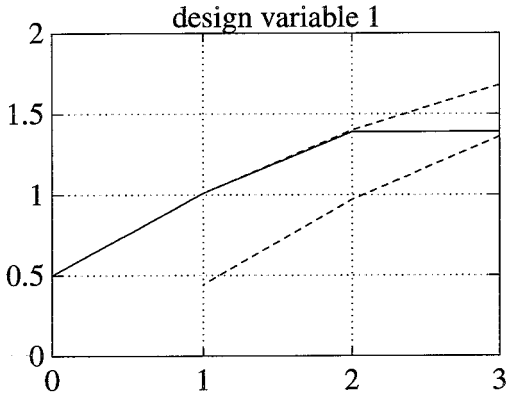
Two-bar truss with noise, x0 = (0.5 , 0.25)

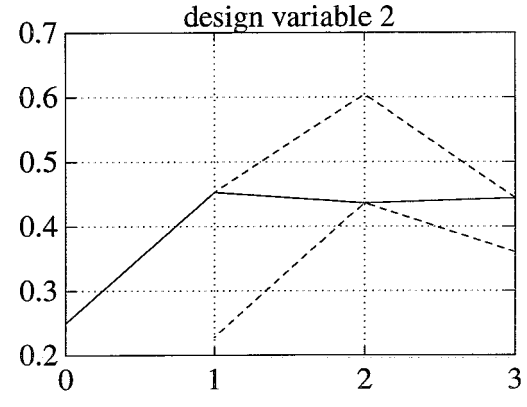
cyc	obj	viol	err	x1	x2
0	0.492	2.045e2		0.500	0.250
1		0.239e2	0.364e2		
2	1.092	0.352e2	0.354e2	1.013	0.453
3	1.526	3.369e-1	7.882e-1	1.394	0.436
4	1.533	1.632	2.505	1.388	0.444

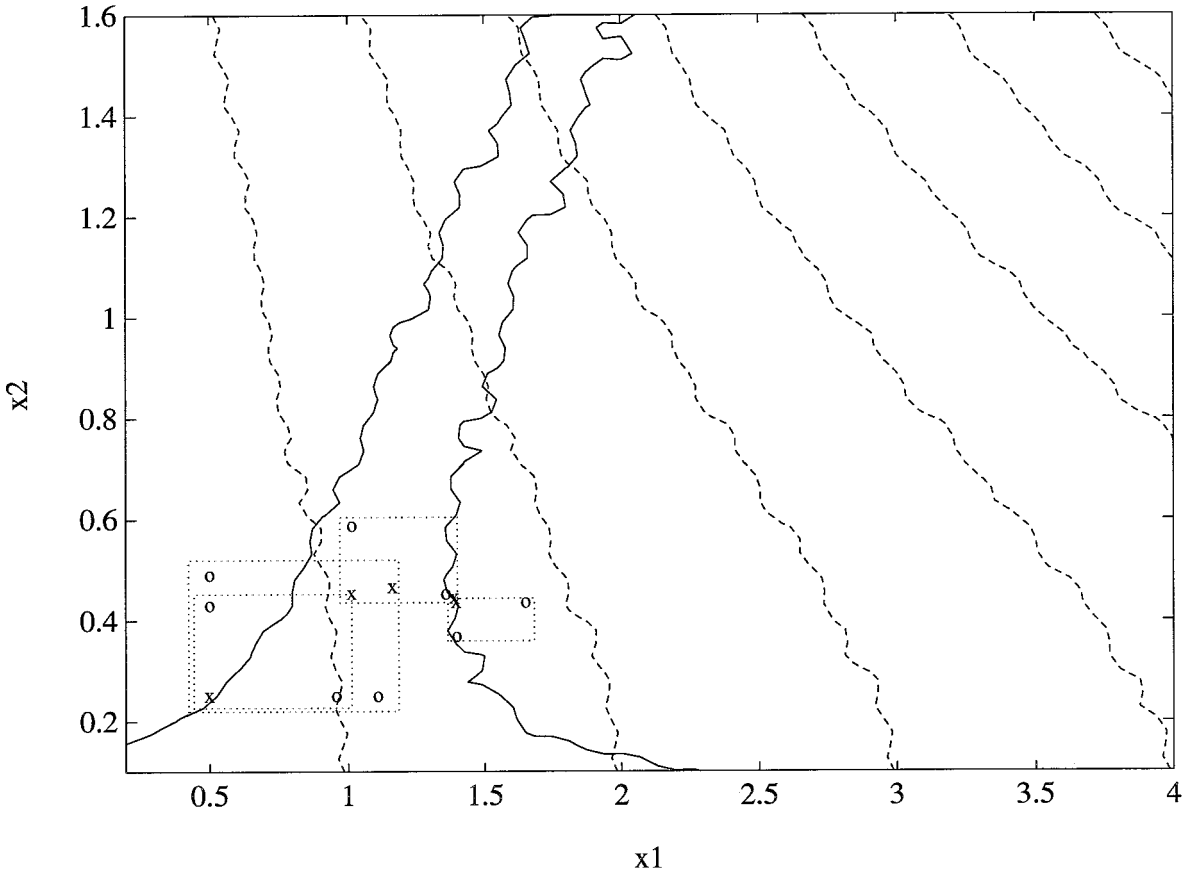






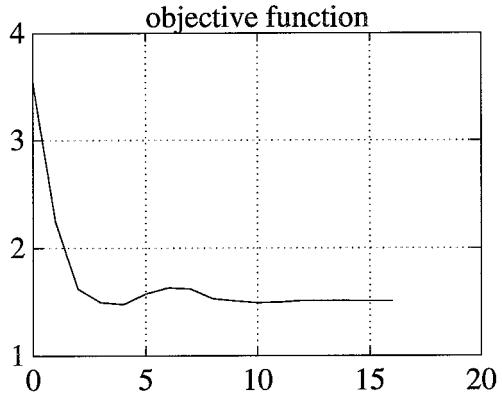


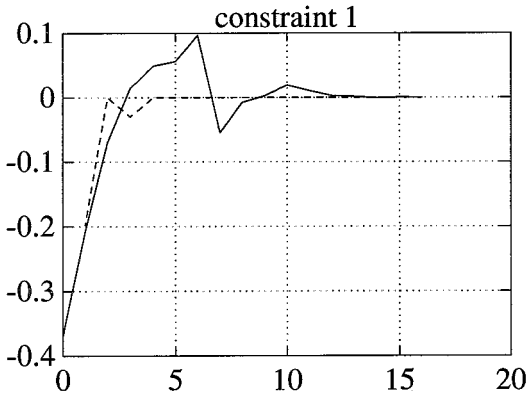


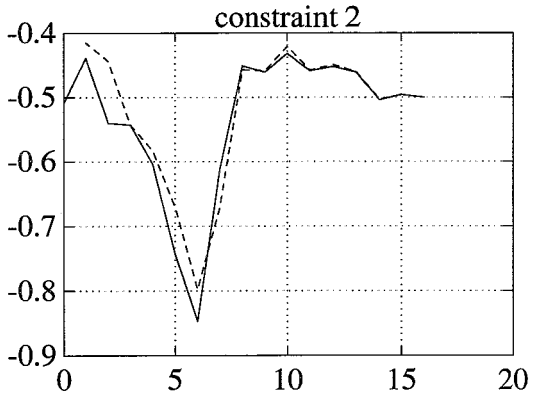

1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
x2
0.5
1
1.5
2
2.5
3
3.5
4
x1

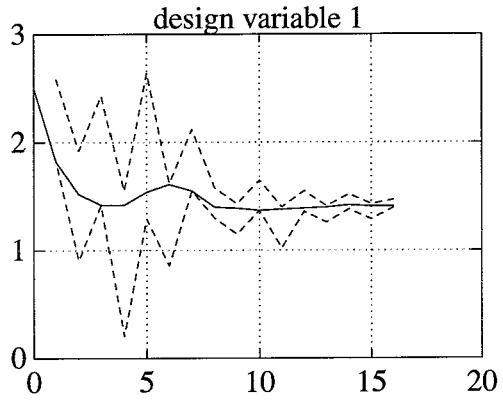
Two-bar truss, $x0$ = (2.5 , 1.0)

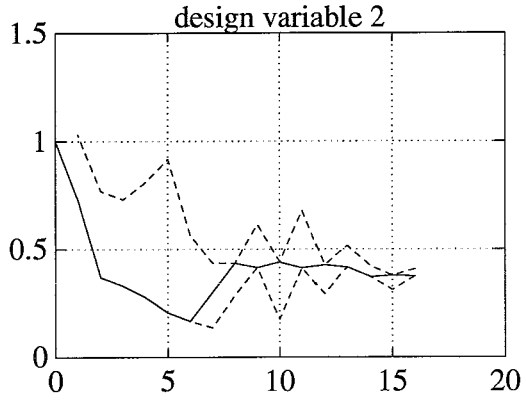
cyc	obj	viol	err	x1	x2
0	3.536	-0.369e2		2.500	1.000
1	2.248	-0.208e2	5.563	1.816	0.730
2	1.622	-6.973	0.208e2	1.521	0.370
3	1.495	1.445	4.381	1.420	0.330
4	1.478	4.910	5.170	1.424	0.277
5	1.575	5.597	0.282e2	1.542	0.206
6		0.147e2	0.733e2		
7	1.631	-9.669	0.319e2	1.631	0.166
8	1.621	-5.480	0.154e2	1.552	0.300
9	1.530	-7.625e-1	1.185	1.403	0.435
10	1.509	2.618e-1	5.164e-1	1.394	0.415
11	1.492	1.927	1.891	1.365	0.440
12	1.497	1.047	1.036	1.384	0.413
13	1.512	2.612e-1	4.537e-1	1.391	0.427
14	1.511	1.401e-1	1.399e-1	1.395	0.417
15	1.510	-6.510e-2	6.514e-2	1.415	0.372
16	1.508	5.945e-2	1.564e-1	1.410	0.378
17	1.509	1.198e-2	2.973e-2	1.412	0.375

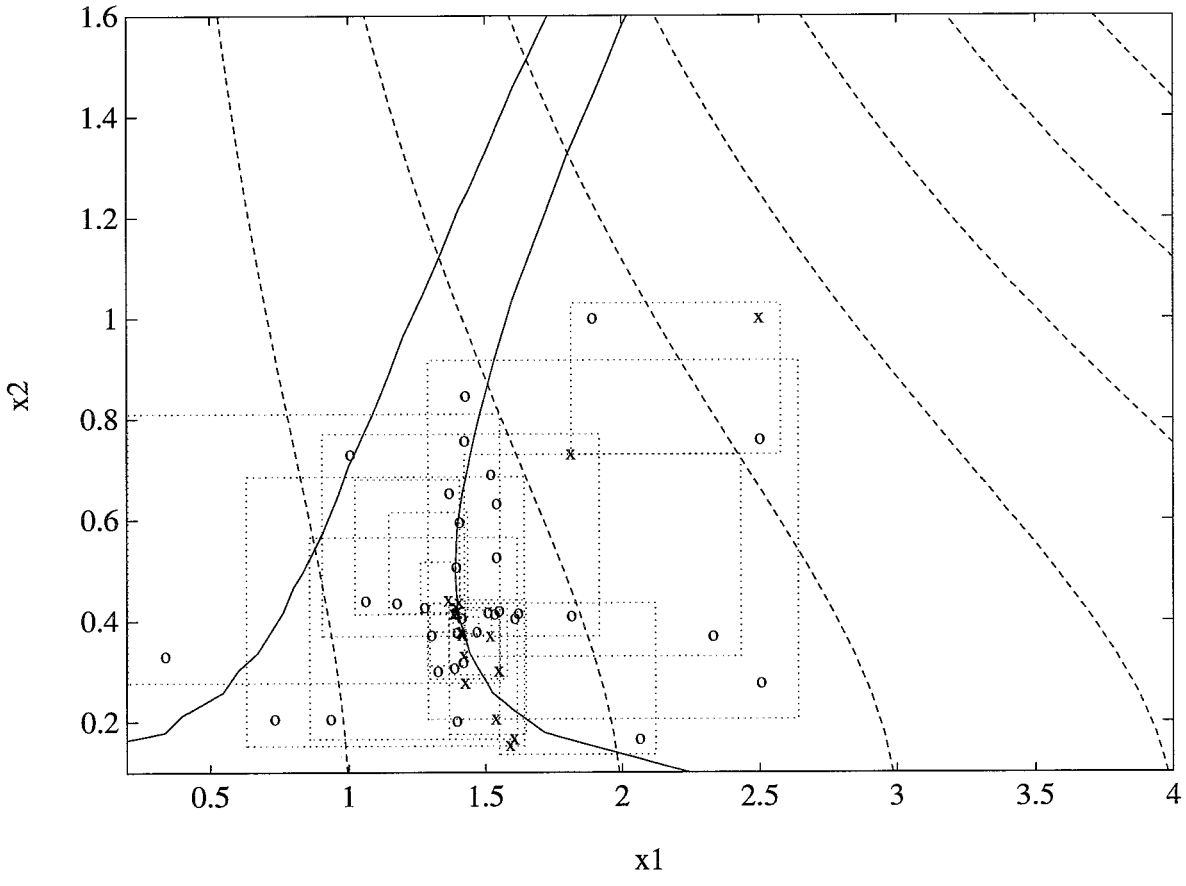






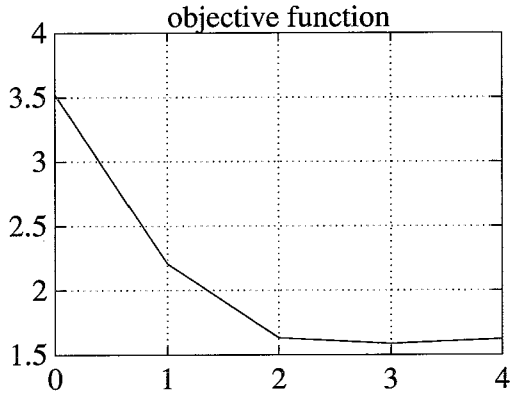

design variable 1
3
2
1
0
0
5
10
15
20

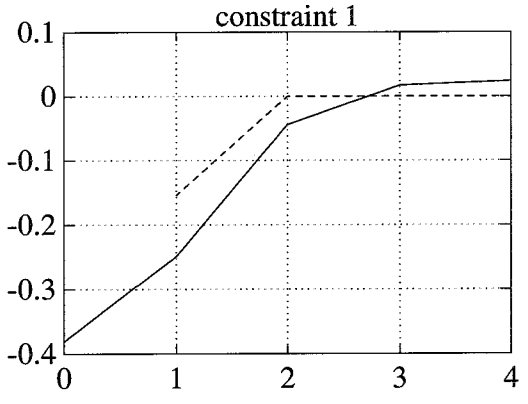

design variable 2
1.5
1
0.5
0
0
5
10
15
20

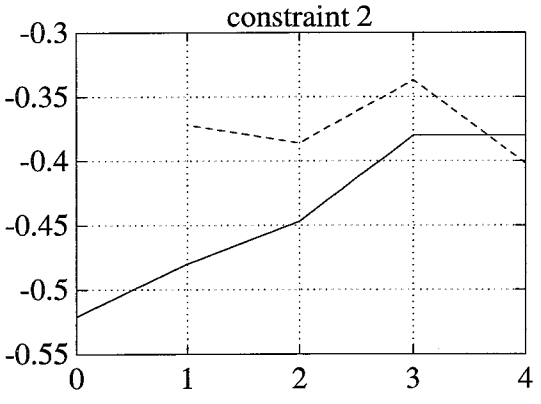

x2
x1
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0.5
1
1.5
2
2.5
3
3.5
4

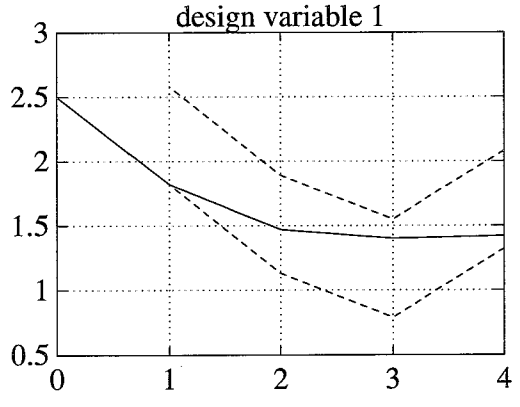
Two-bar truss with noise, x0 = (2.5 , 1.0)

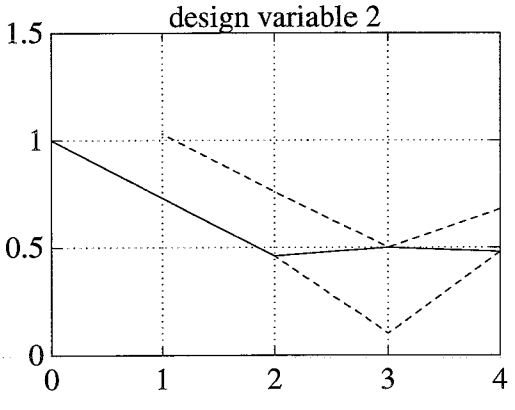
cyc	obj	viol	err	x1	x2
0	3.524	-0.381e2		2.500	1.000
1	2.207	-0.249e2	0.209e2	1.816	0.730
2	1.631	-4.408	0.111e2	1.471	0.460
3	1.586	1.714	6.987	1.397	0.500
4	1.621	2.368	3.516	1.420	0.480

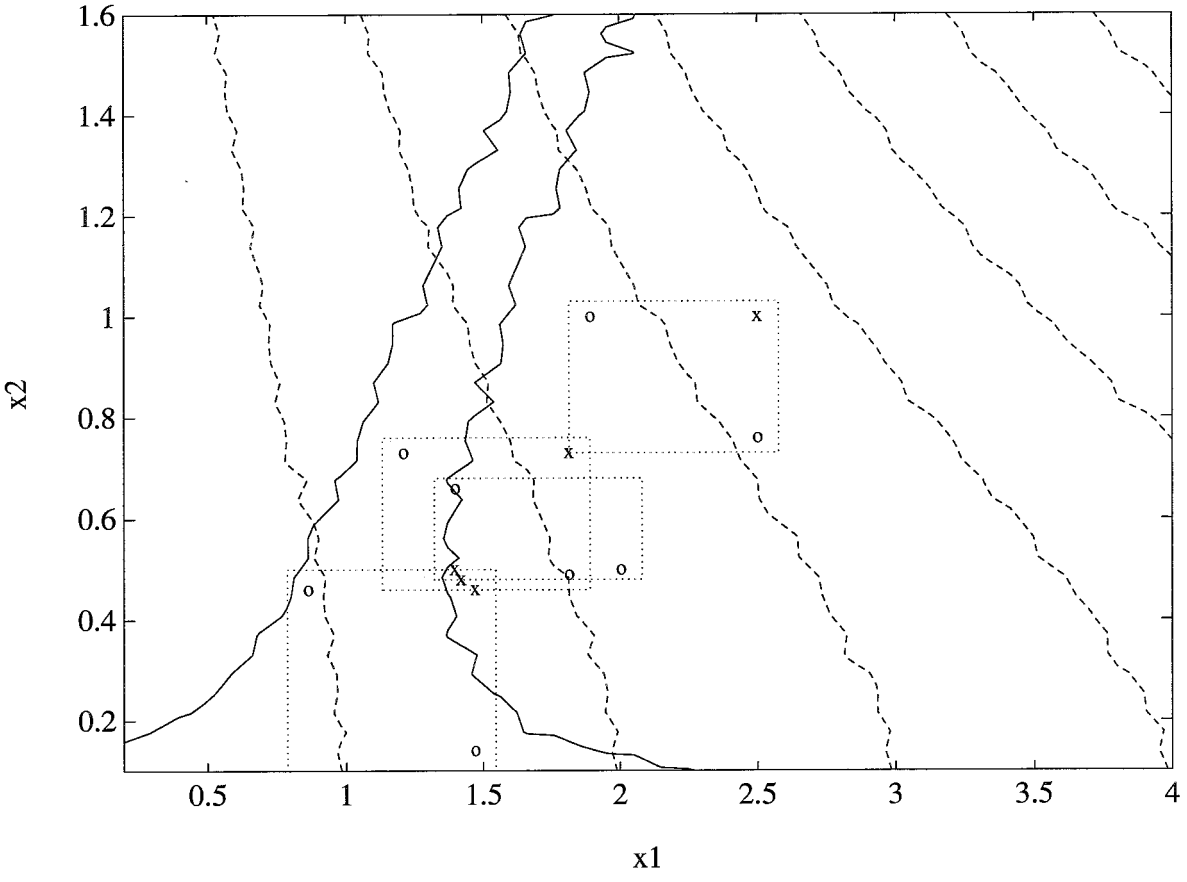











1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
x2
0.5
1
1.5
2
2.5
3
3.5
4
x1

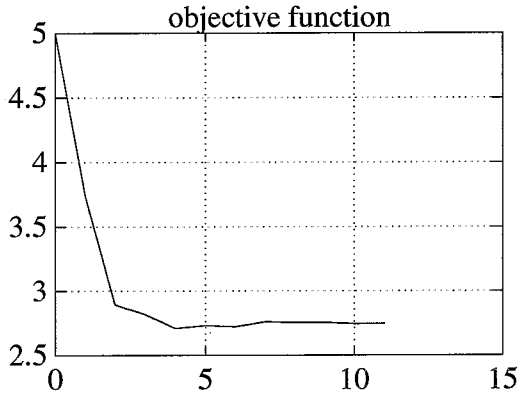
Appendix G: Three-bar truss results

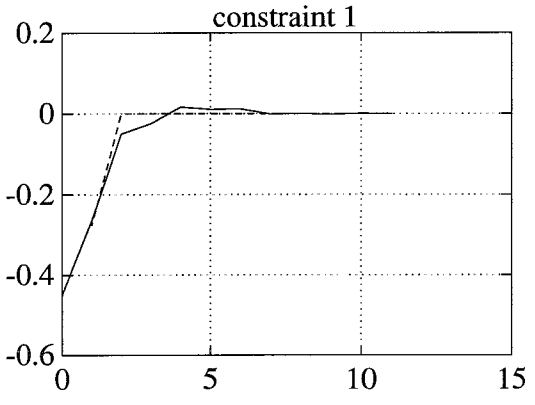
Three-bar truss, $x0$ = (1.0 , 1.0)

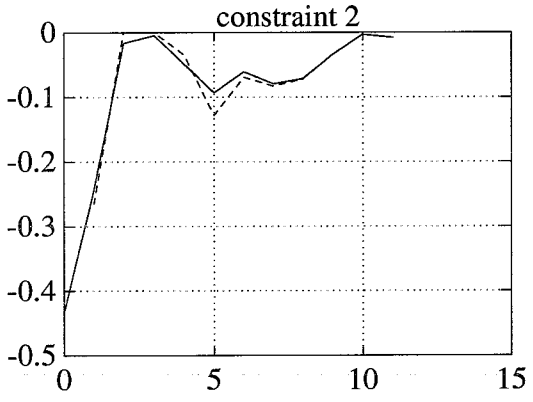
Table of objective function values, maximum constraint violations (%), maximum approximation errors (%), and design variable values for all optimisation cycles.

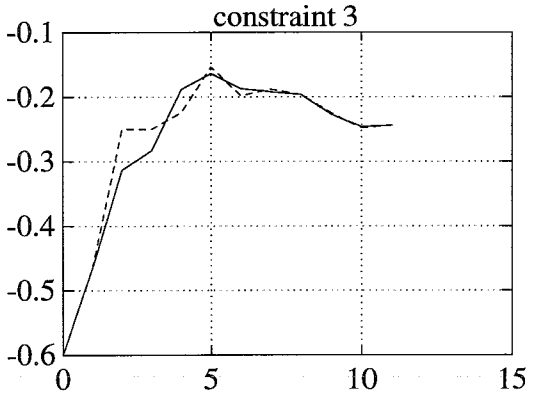
cyc	obj	viol	err	x1	x2
0	5.000	-0.434e2		1.000	1.000
1	3.740	-0.244e2	2.881	0.748	0.748
2	2.894	-1.650	9.134	0.580	0.574
3	2.816	-4.529e-1	4.562	0.562	0.570
4	2.709	1.666	4.397	0.524	0.612
5	2.732	1.096	3.795	0.521	0.650
6	2.723	1.189	1.407	0.525	0.622
7	2.759	-8.826e-2	4.750e-1	0.531	0.636
8	2.754	3.802e-2	7.000e-2	0.531	0.629
9	2.755	-1.904e-1	3.590e-1	0.540	0.597
10	2.746	9.574e-2	1.830e-1	0.543	0.573
11	2.748	1.039e-2	2.299e-2	0.543	0.577

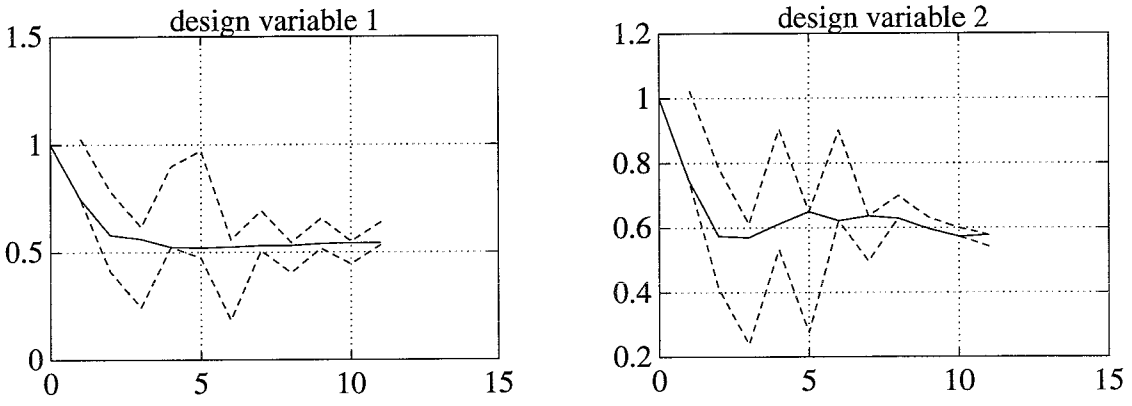
Plots of the progress of: 1) objective function, 2) exact (-) and approximate (--) constraints, and 3) optimum design variables (-) and subregion lower and upper bounds (--). On the x-axes the optimisation cycle number.



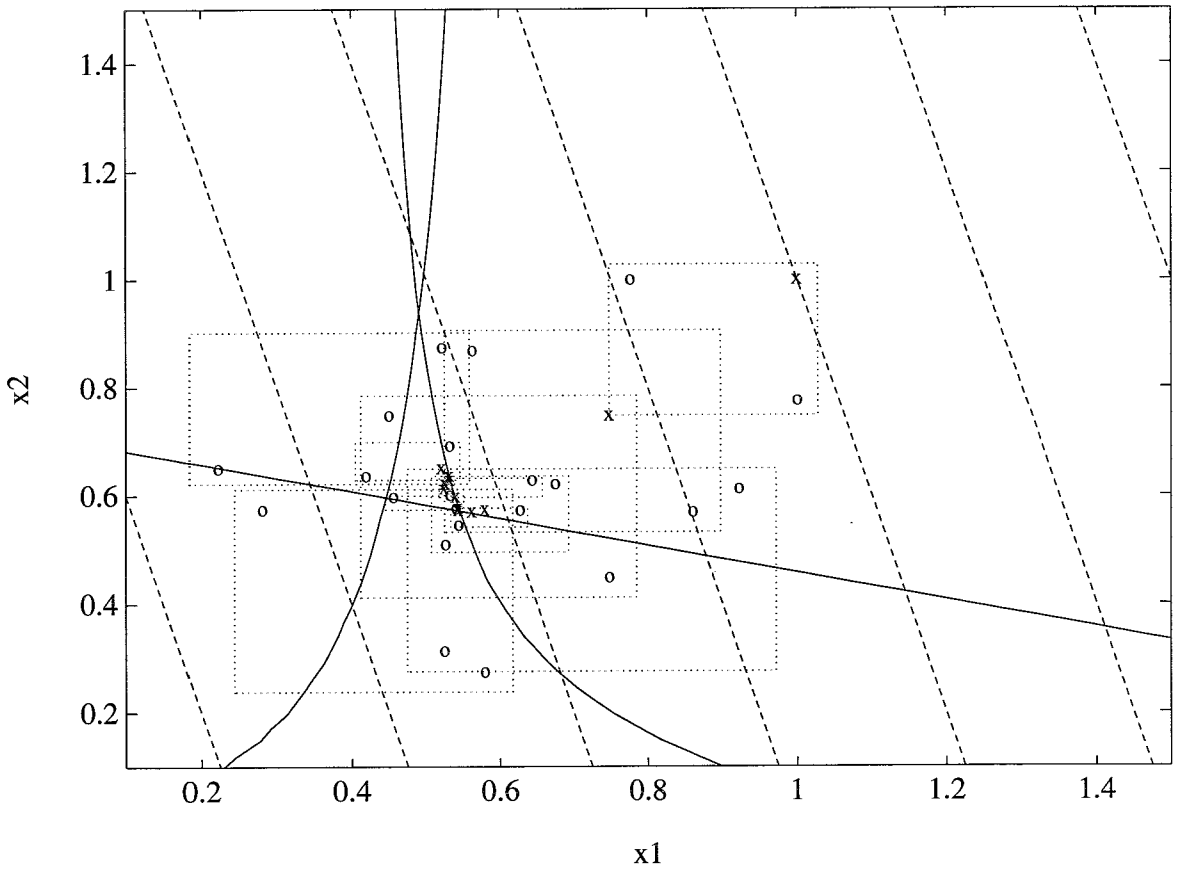






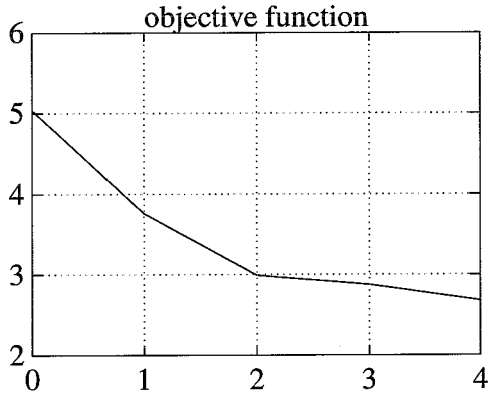


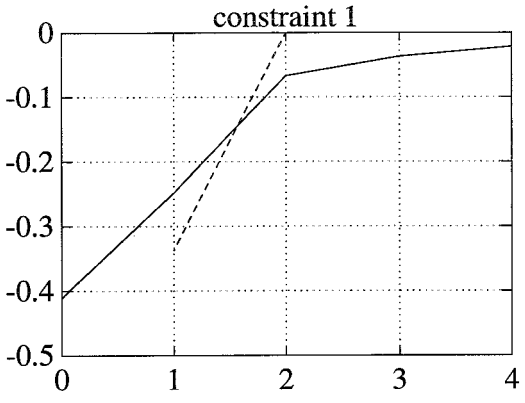
Plot of the different search subregions during the optimisation process.

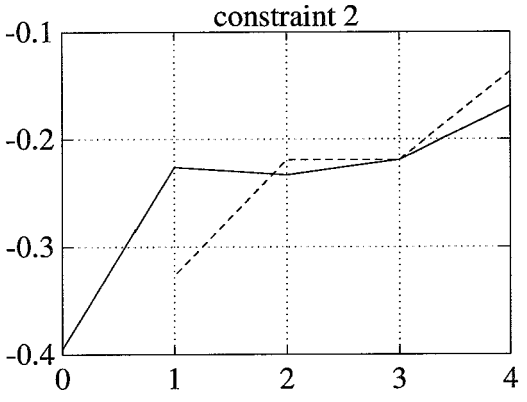


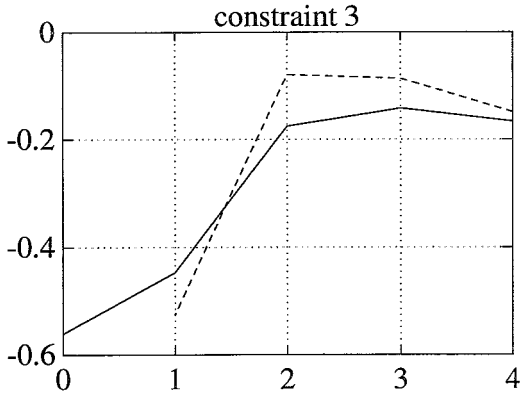
Three-bar truss with noise, $x0$ = (1.0 , 1.0)

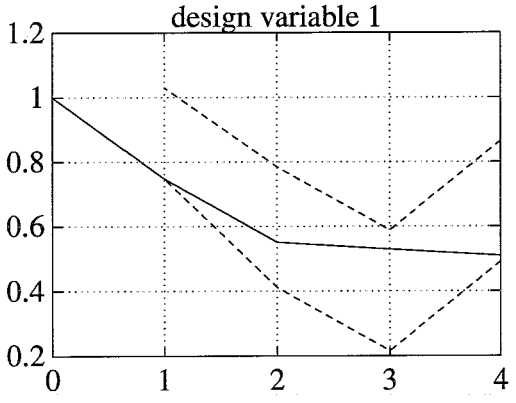
cyc	obj	viol	err	x1	x2
0	5.039	-0.395e2		1.000	1.000
1	3.758	-0.226e2	0.141e2	0.748	0.748
2	2.990	-6.685	0.116e2	0.551	0.785
3	2.875	-3.686	6.481	0.529	0.767
4	2.678	-2.161	3.754	0.509	0.683

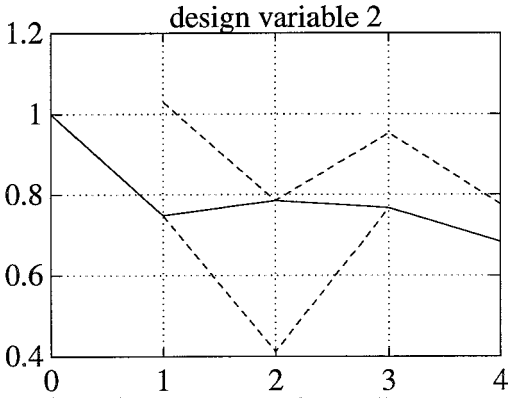


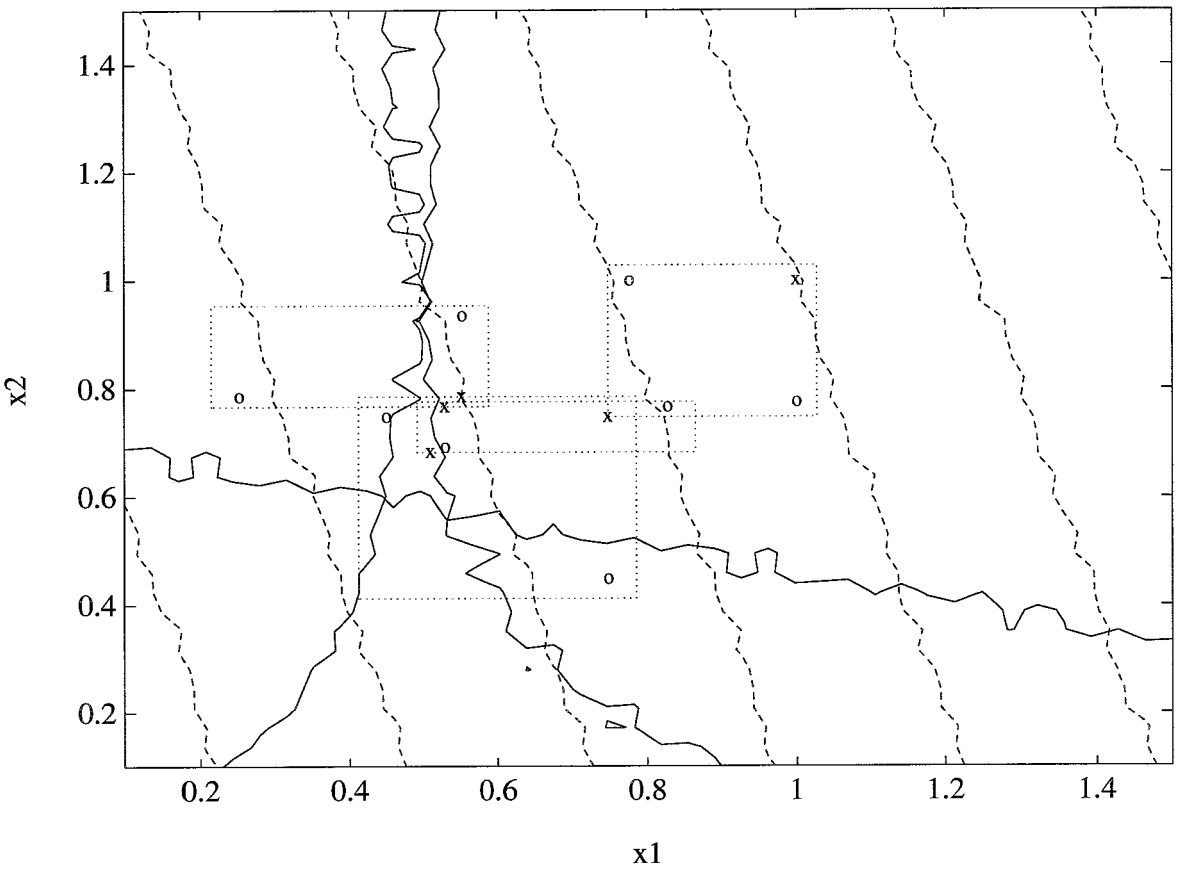






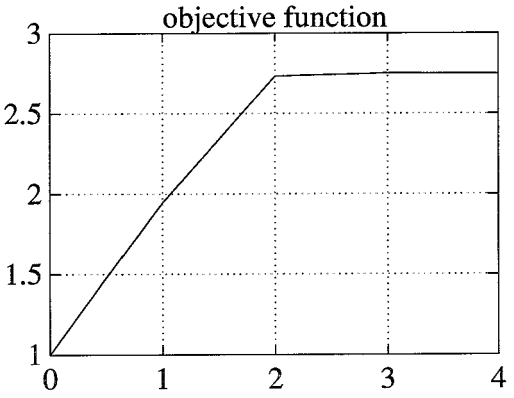


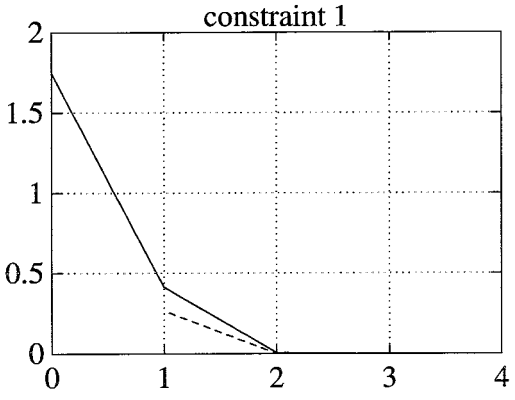


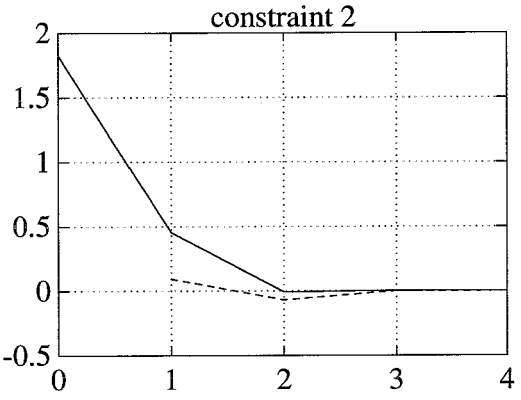

1.4
1.2
1
0.8
0.6
0.4
0.2
x2
0.2
0.4
0.6
0.8
1
1.2
1.4
x1

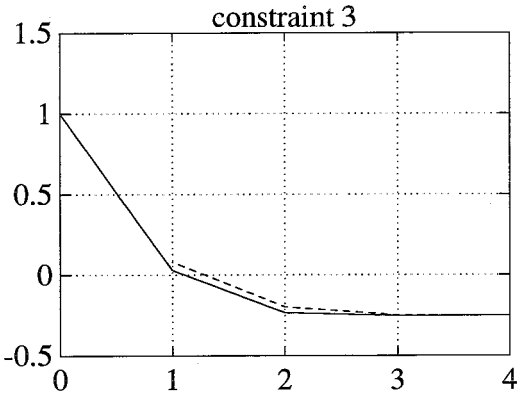
Three-bar truss, $x0$ = (0.2 , 0.2)

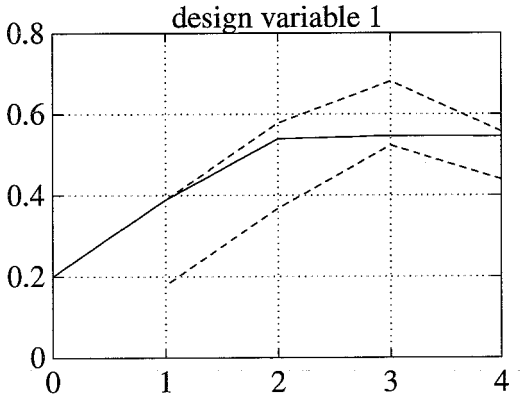
cyc	obj	viol	err	x1	x2
0	1.000	1.828e2		0.200	0.200
1		0.252e2	0.407e2		
2	1.945	0.454e2	0.249e2	0.389	0.389
3	2.733	5.746e-1	6.162	0.539	0.578
4	2.752	1.778e-1	4.277e-1	0.546	0.569
5	2.749	2.778e-2	7.178e-2	0.545	0.571

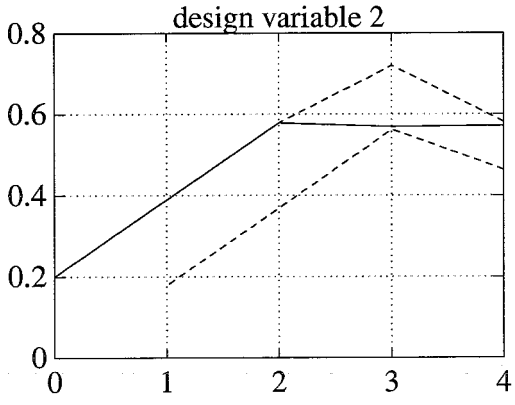


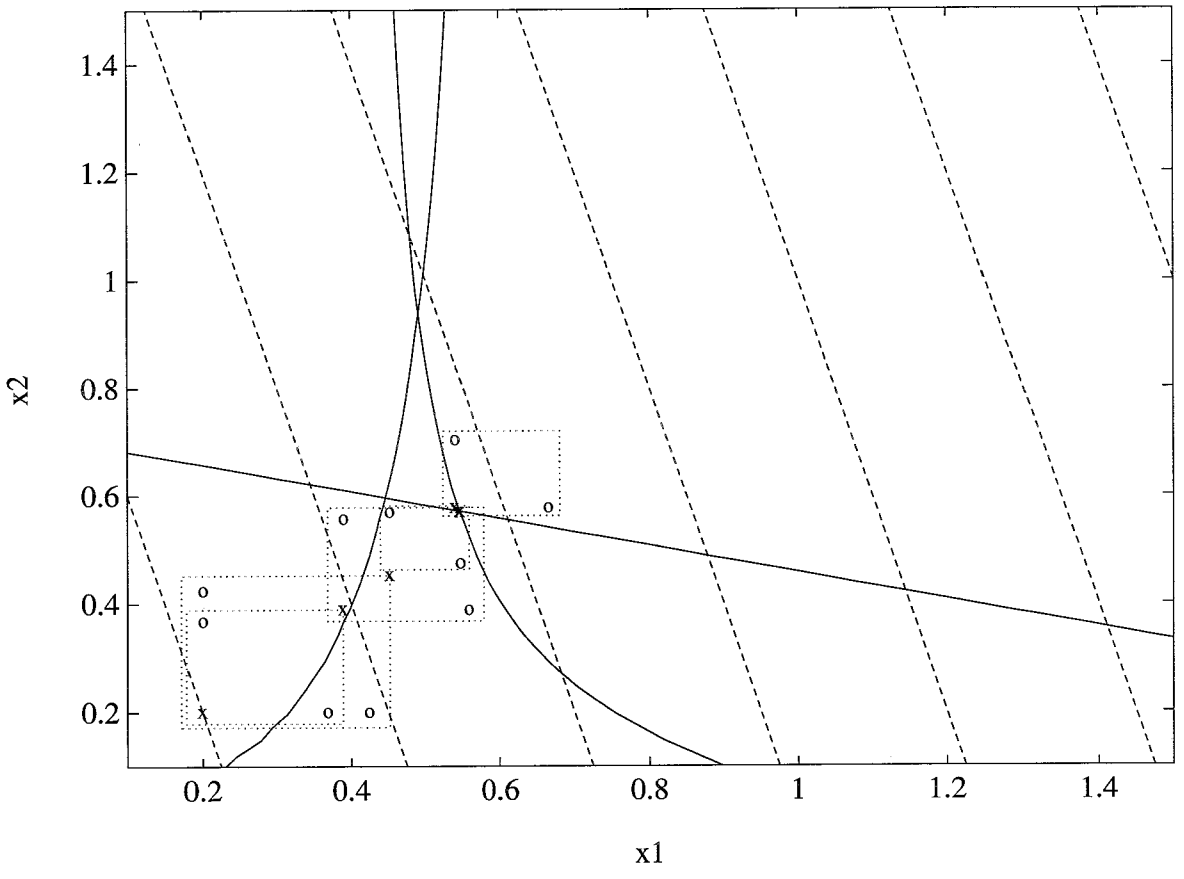






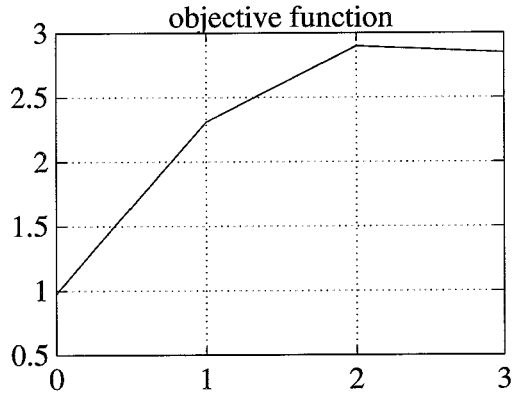


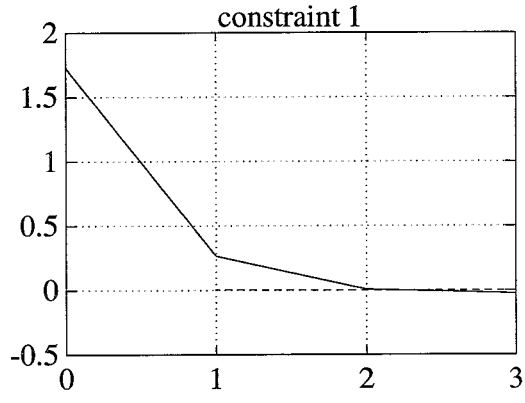


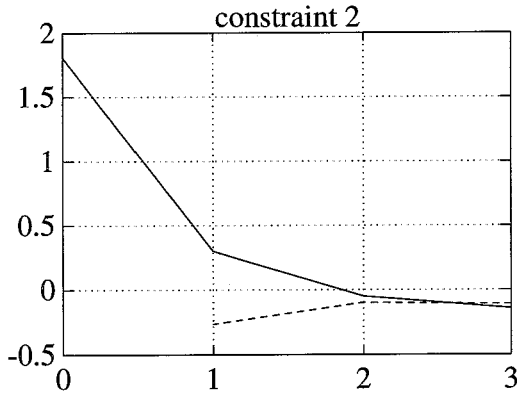

x2
x1
1.4
1.2
1
0.8
0.6
0.4
0.2
0.2
0.4
0.6
0.8
1
1.2
1.4

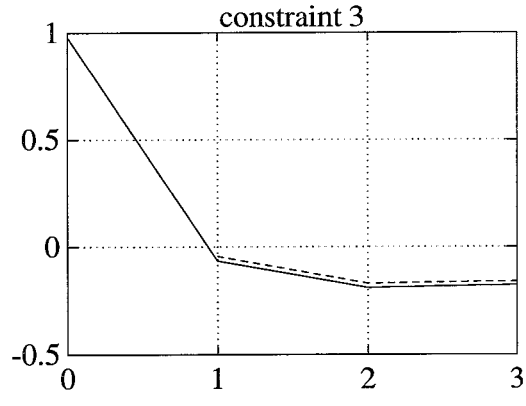
Three-bar truss with noise, x0 = (0.2 , 0.2)

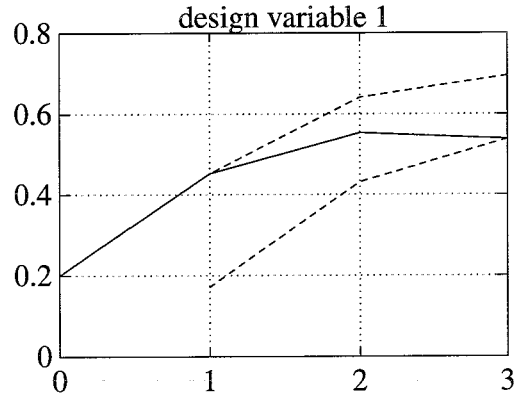
cyc	obj	viol	err	x1	x2
0	0.978	1.807e2		0.200	0.200
1	2.310	0.301e2	0.435e2	0.452	0.452
2	2.898	7.808e-1	5.118	0.553	0.641
3	2.848	-2.334	3.684	0.538	0.694

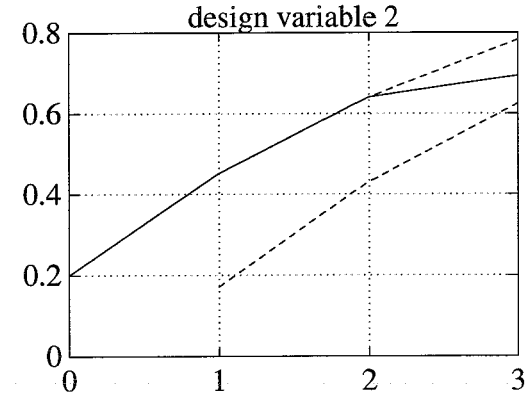


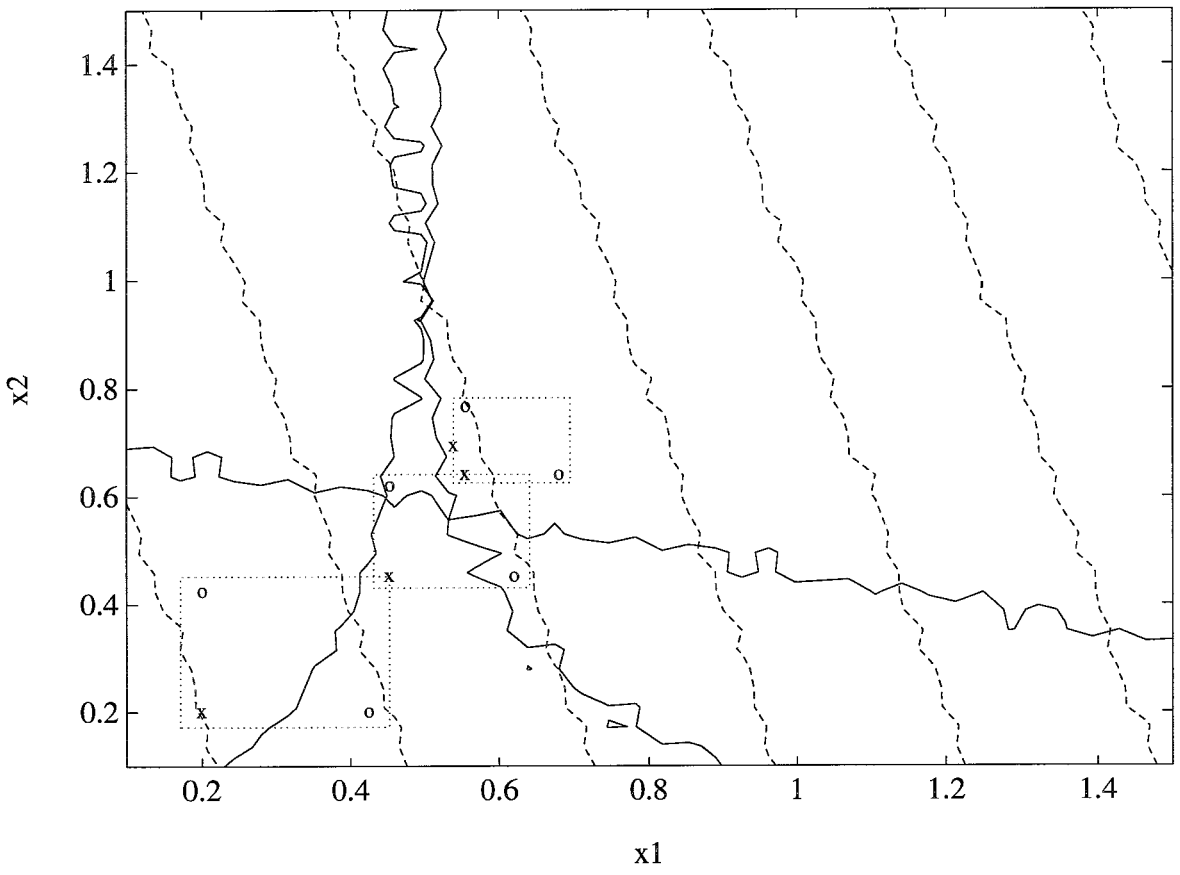











1.4
1.2
1
0.8
0.6
0.4
0.2
x2
0.2
0.4
0.6
0.8
1
1.2
1.4
x1
o
x
x
o
o
x
o
o
x
o

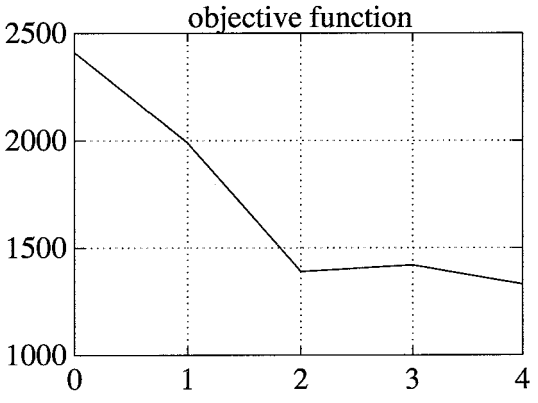
Appendix H: Full-scale frontal impact results

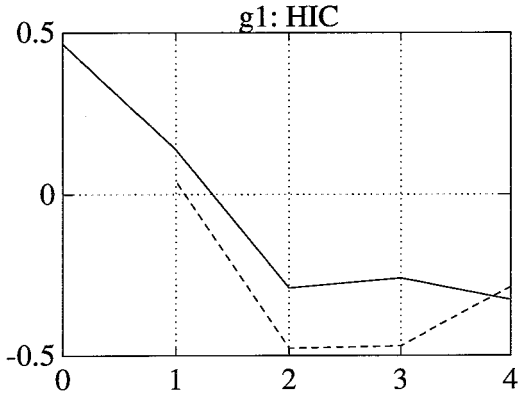
Full-scale frontal impact, $x0$ = (0.025 , 0.5)

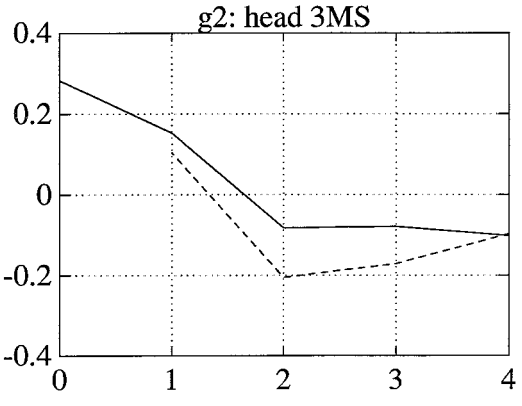
Table of objective function values, maximum constraint violations (%), maximum approximation errors (%), and design variable values for all optimisation cycles.

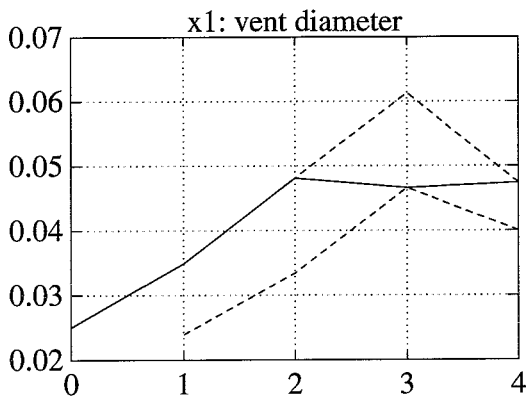
cyc	obj	viol	err	x1	x2
0	2409	0.465e2		2.500e-2	0.500
1	1988	0.153e2	8.550	3.490e-2	0.572
2	1385	-8.216	0.263e2	4.810e-2	0.668
3	1419	-7.890	0.284e2	4.663e-2	0.657
4	1334	-0.102e2	5.972	4.737e-2	0.663

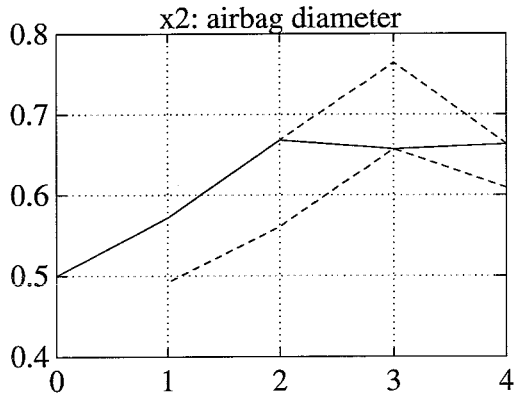
Plot of the progress of: 1) objective function, 2) exact (-) and approximate (--) constraints, and 3) optimum design variables (-) and search subregion lower and upper bounds (--). On the x-axes the optimisation cycle number.





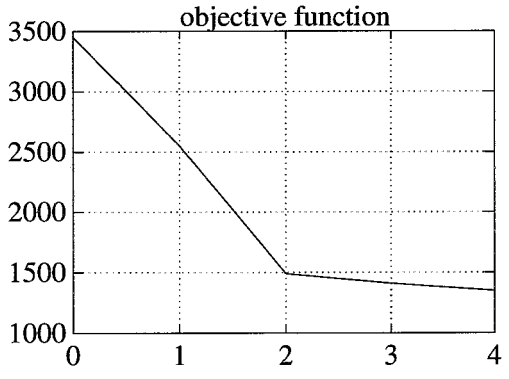


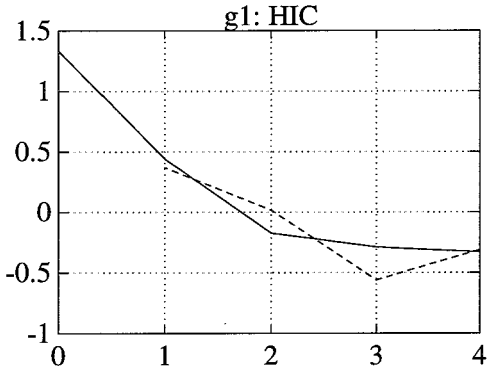

x1: vent diameter
0.07
0.06
0.05
0.04
0.03
0.02
0
1
2
3
4

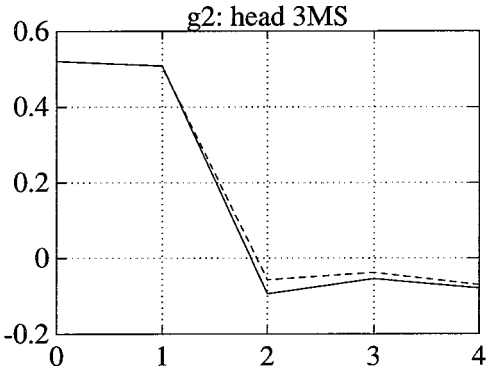

x2: airbag diameter
0.8
0.7
0.6
0.5
0.4
0
1
2
3
4

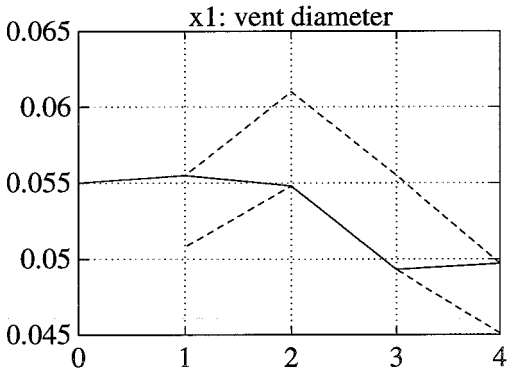
Full-scale frontal impact, x0 = (0.055 , 0.5)

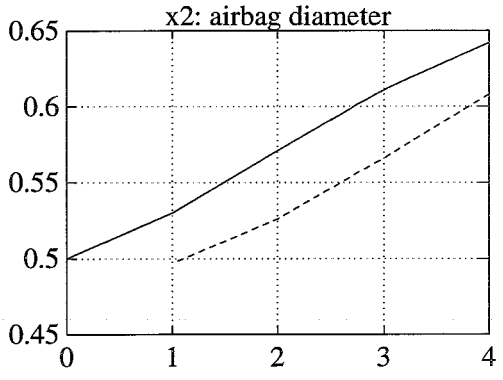
cyc	obj	viol	err	x1	x2
0	3454	1.335e2		5.500e-2	0.500
1		-1.108	2.235e2		
2		-2.807	2.991e2		
3		-4.913	2.501e2		
4	2551	0.508e2	4.991	5.546e-2	0.530
5	1493	-9.453	0.230e2	5.485e-2	0.571
6	1405	-5.430	0.382e2	4.928e-2	0.611
7	1348	-7.917	2.523	4.974e-2	0.642



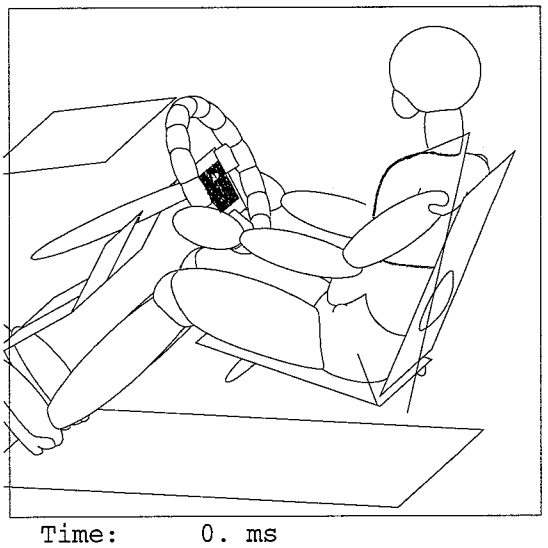




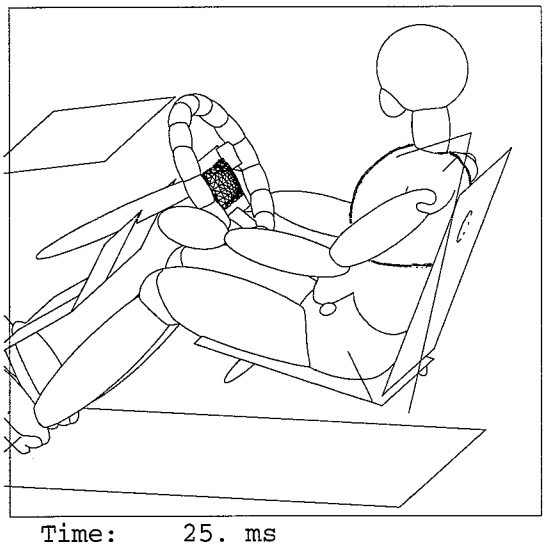




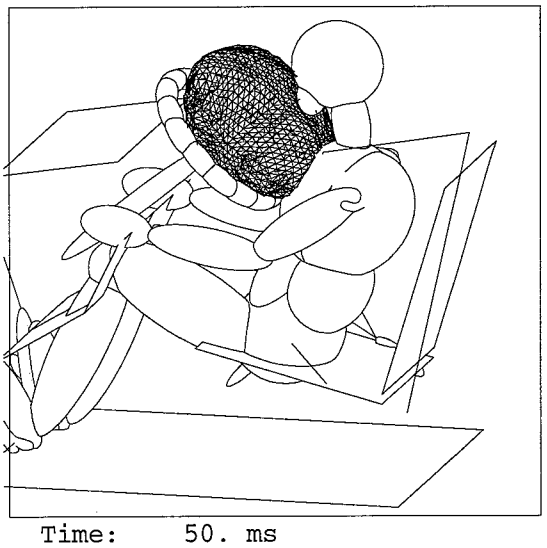
Animation with the initial safety measures design.



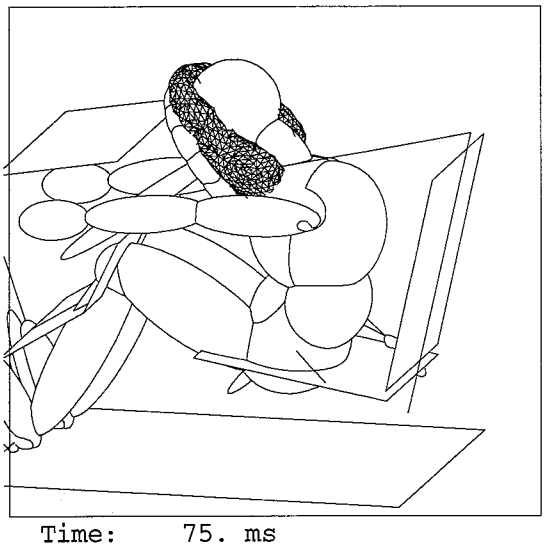
Time: 0. ms



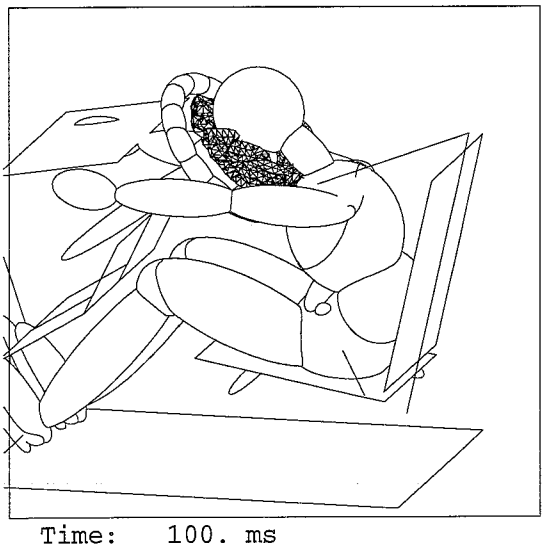
Time: 25. ms



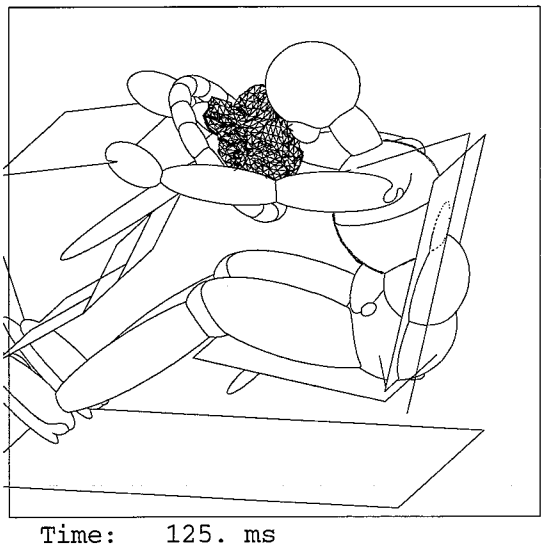
Time: 50. ms



Time: 75. ms

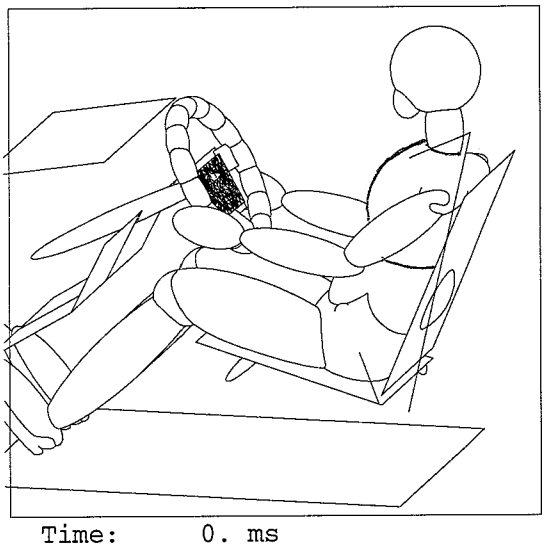


Time: 100. ms

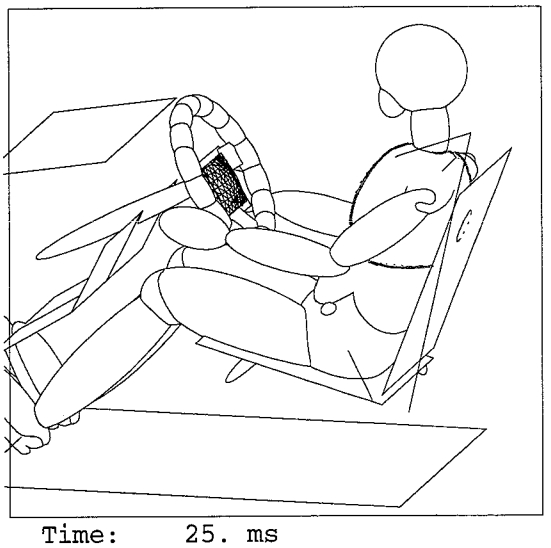


Time: 125. ms

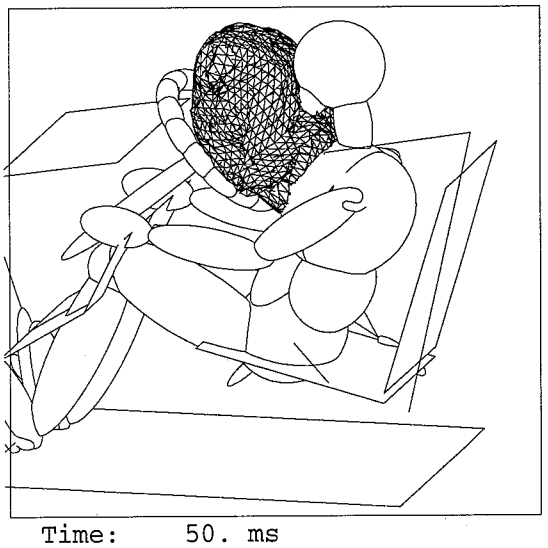
Animation with the optimum safety measures design.



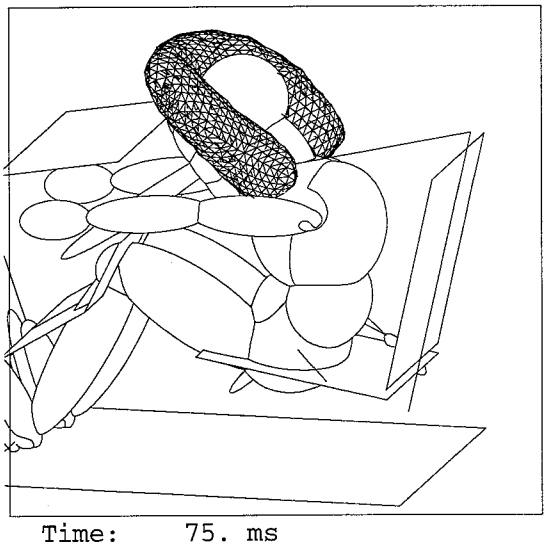
Time: 0. ms



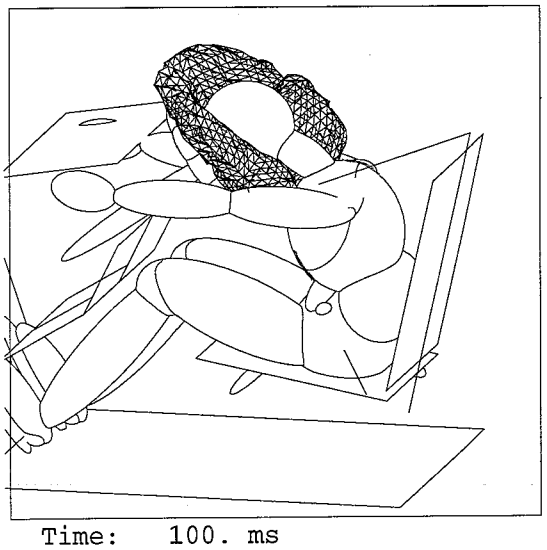
Time: 25. ms



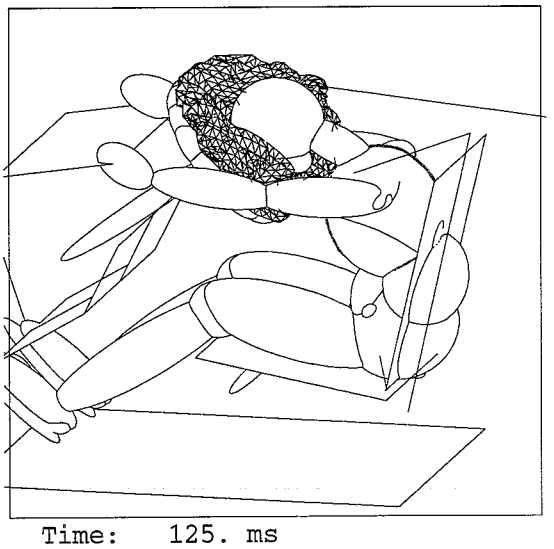
Time: 50. ms



Time: 75. ms



Time: 100. ms



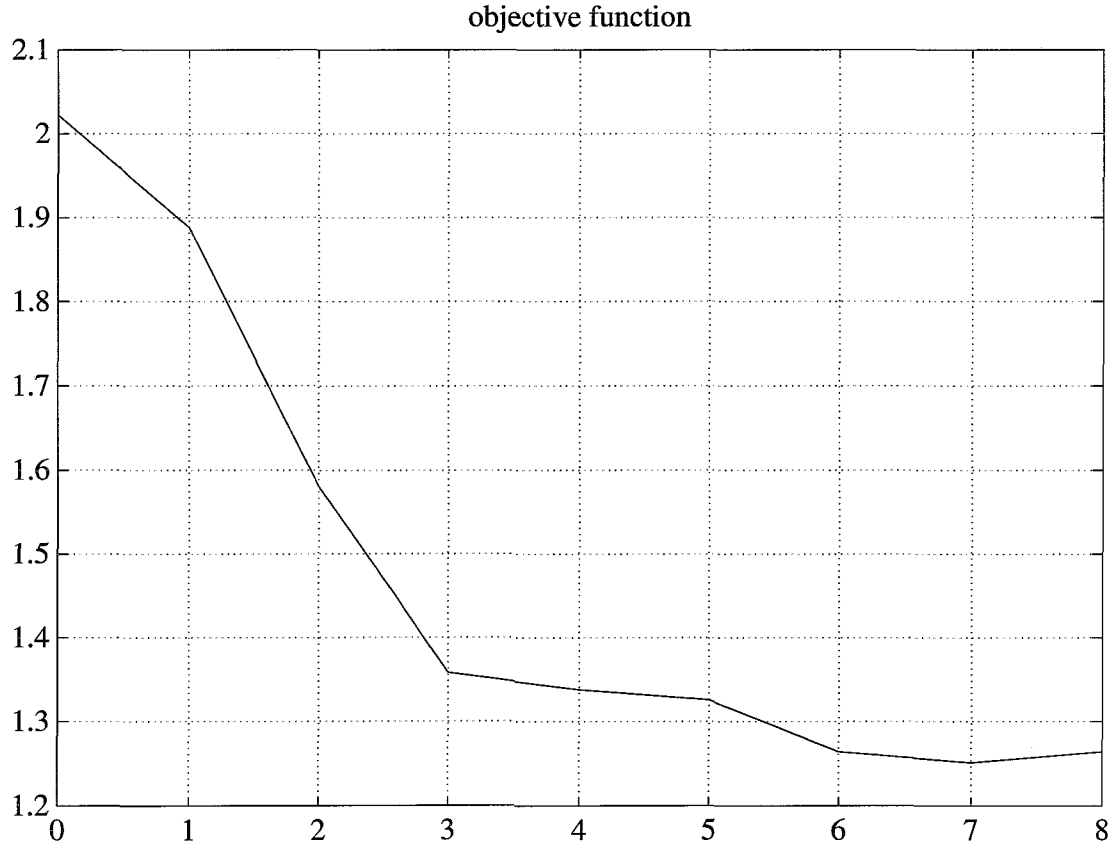
Time: 125. ms

Appendix I: Full-scale off-set impact results

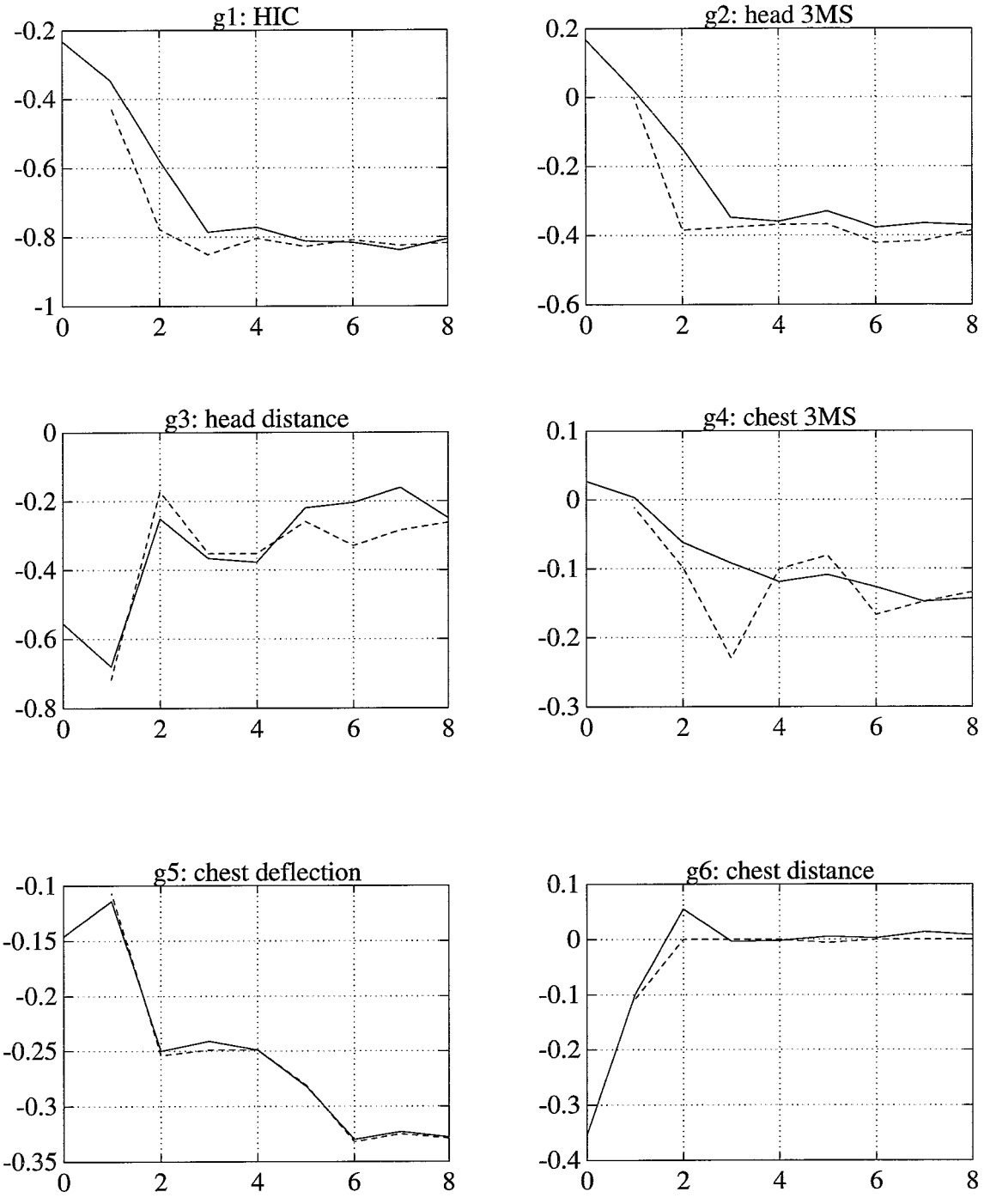
Table of objective function values, maximum constraint violations (%), maximum approximation errors (%), and design variable values for all optimisation cycles.

cyc	obj	viol	err	x1	x2	x3	x4	x5	x6
0	2.022	0.168e2		2.290e-3	0.400	0.400	4520	26800	30.00
1	1.888	2.025	0.114e2	2.993e-3	0.355	0.405	4400	26000	30.40
2		1.725	0.506e2						
3	1.581	5.478	0.474e2	3.600e-3	0.319	0.360	3320	18800	30.80
4	1.358	-3.535e-1	0.305e2	3.550e-3	0.322	0.338	3410	13400	33.50
5	1.338	-2.213e-1	0.141e2	3.575e-3	0.320	0.332	3365	12800	33.80
6	1.326	5.290e-1	8.544	3.689e-3	0.321	0.329	3163	12000	34.20
7	1.264	2.690e-1	0.104e2	3.841e-3	0.326	0.332	2893	10933	34.73
8	1.251	1.370	0.107e2	3.818e-3	0.327	0.347	2933	10000	35.00
9	1.264	8.529e-1	5.892	3.748e-3	0.328	0.345	2913	10000	35.00

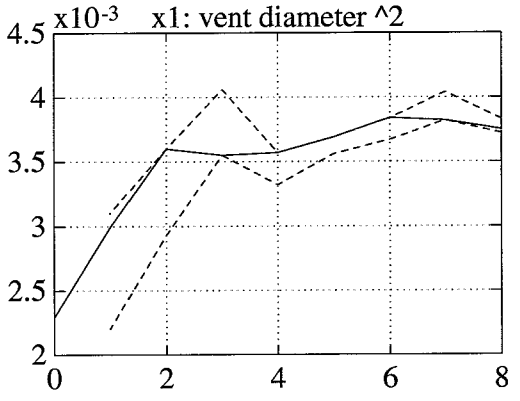
Plot of the progress of the objective function. On the x-axis the optimisation cycle number.

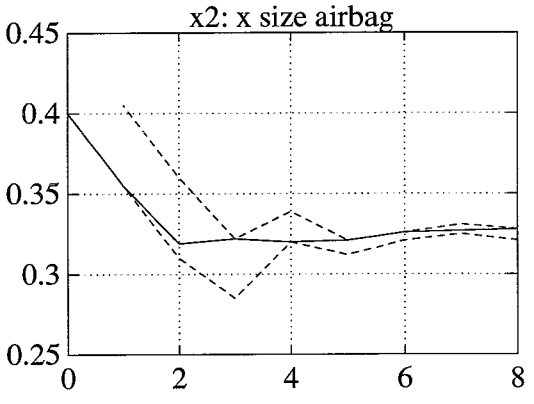


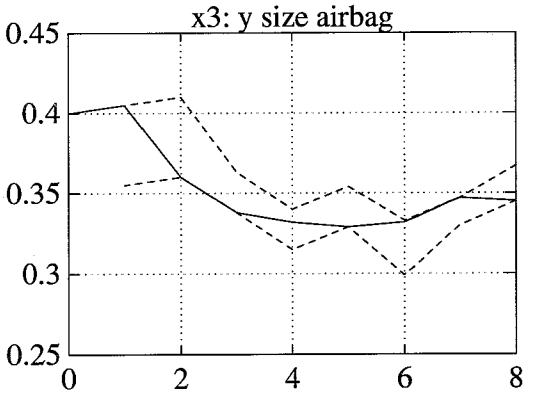
Plot of the progress of the exact (-) and approximate (--) constraints. On the x-axes the optimisation cycle number.

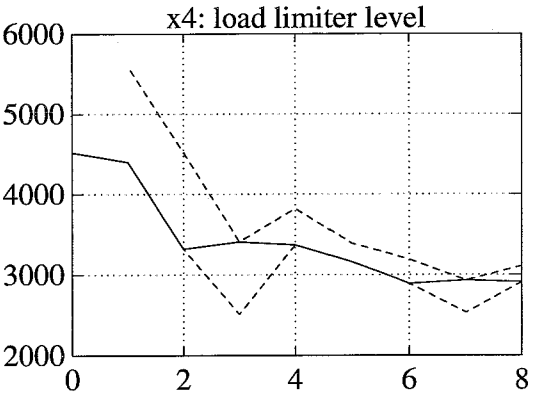


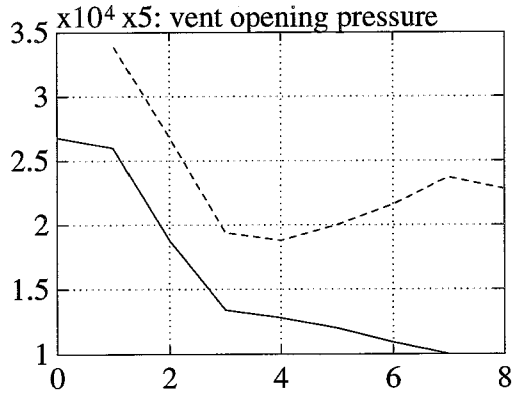
Plot of the progress of the optimum design variables (-) and subregion lower and upper bounds (--). On the x-axes the optimisation cycle number.

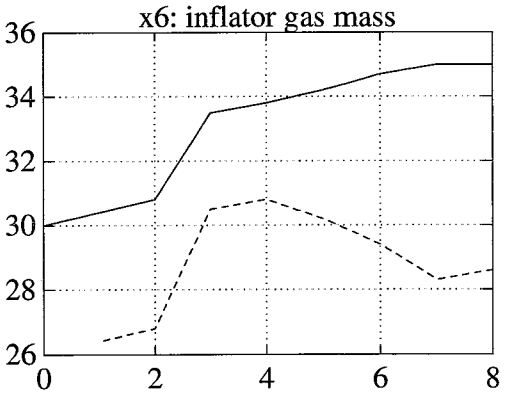




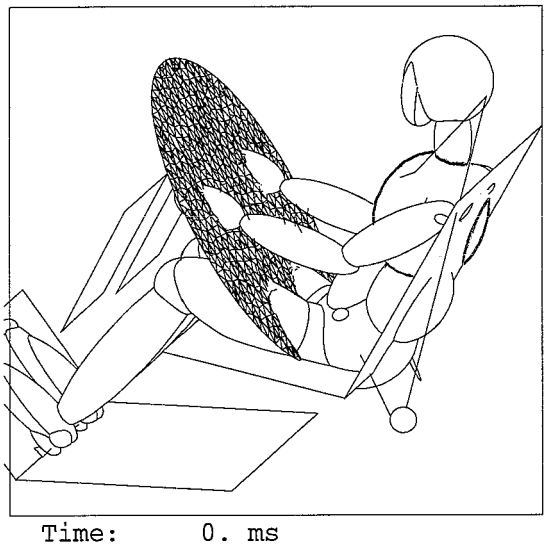




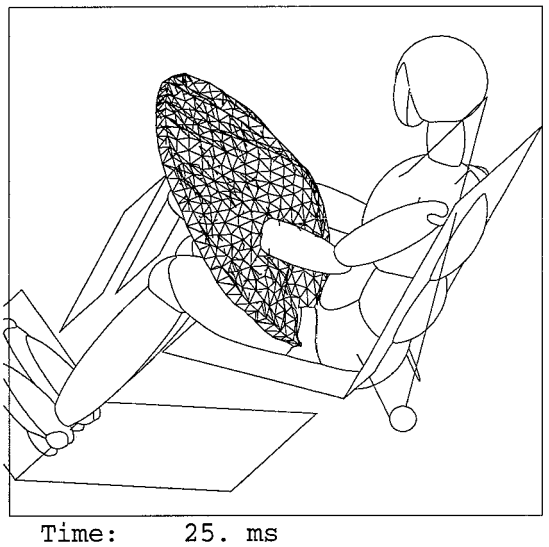




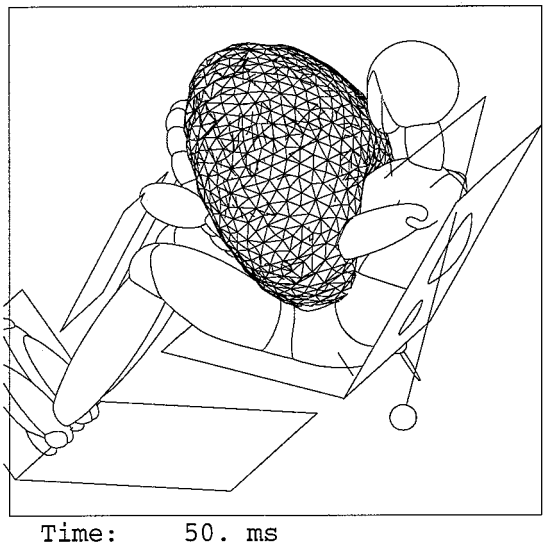
Animation with the initial safety measures design.



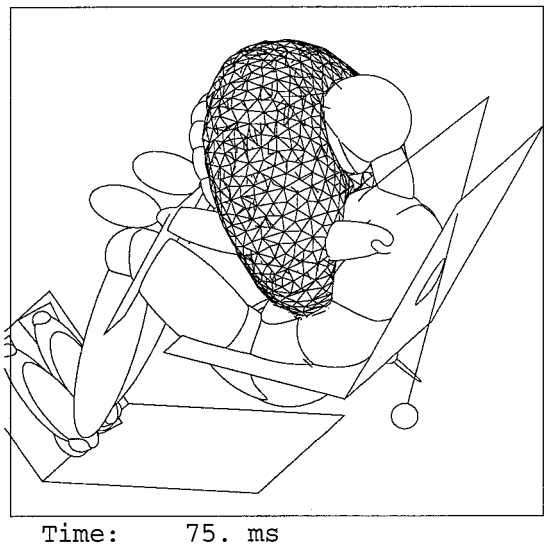
Time: 0. ms



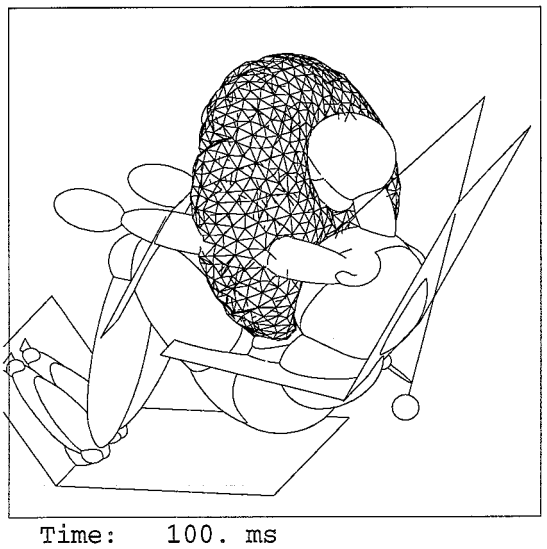
Time: 25. ms



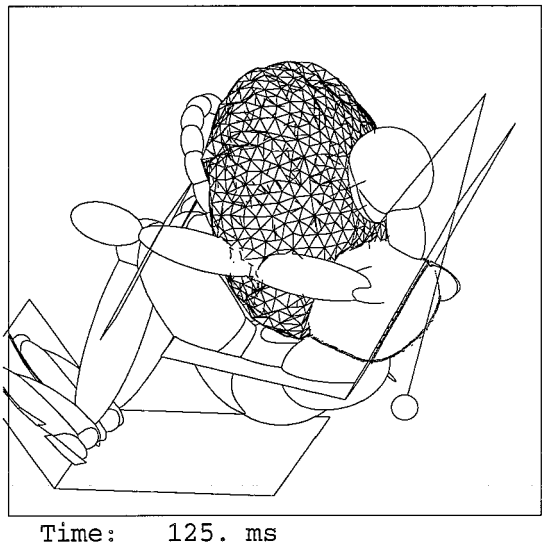
Time: 50. ms



Time: 75. ms

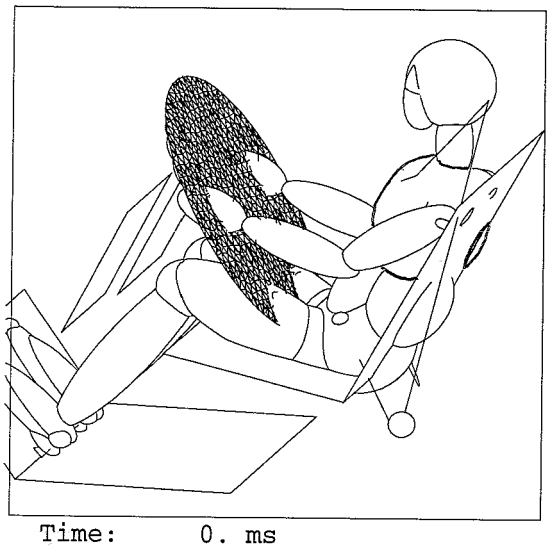


Time: 100. ms

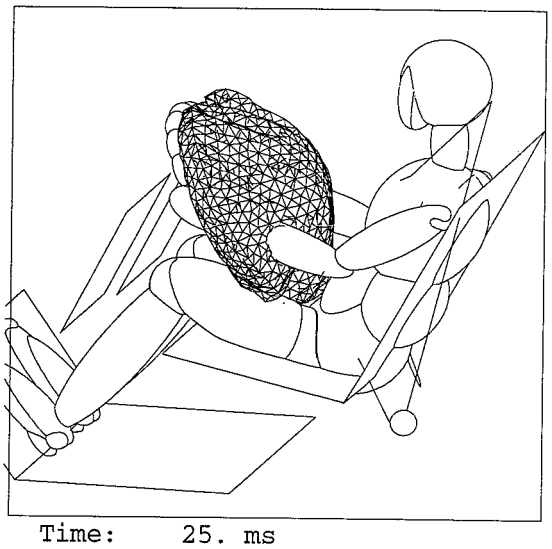


Time: 125. ms

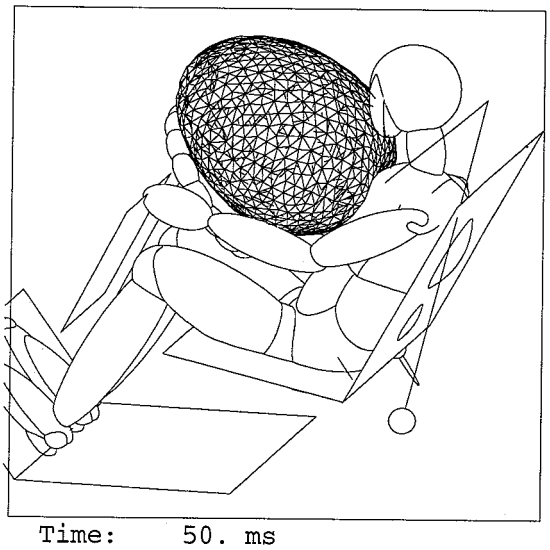
Animation with the optimum safety measures design.



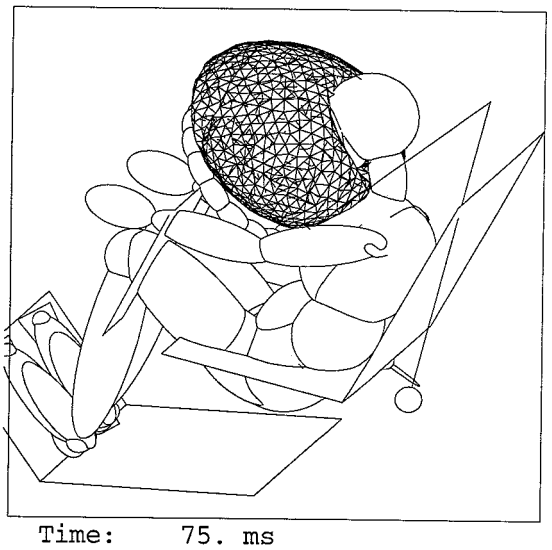
Time: 0. ms



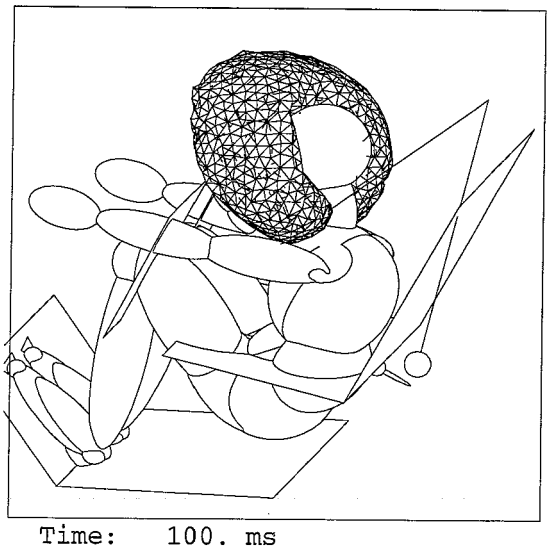
Time: 25. ms



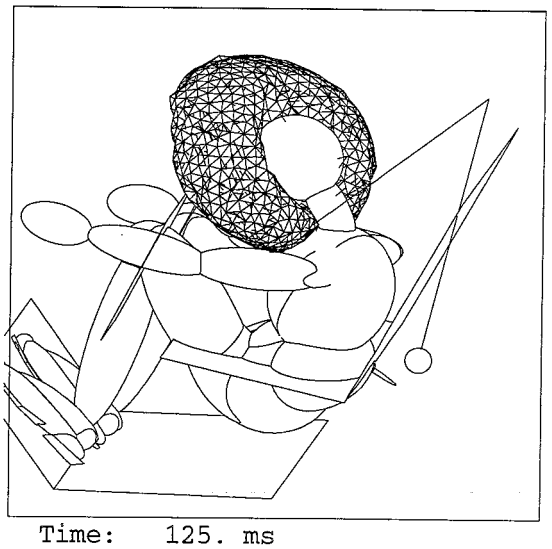
Time: 50. ms



Time: 75. ms



Time: 100. ms



Time: 125. ms