INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN
ICED 03 STOCKHOLM, AUGUST 19-21, 2003

# SIMULATION BASED OPTIMISATION FOR SYSTEM DESIGN

Petter Krus
Department of Mechanical Engineering
Linköping University, SE-58183 Linköping, Sweden
Email: petkr@ikp.liu.se

## Abstract

Modelling and simulation is of crucial importance for system design and optimisation. It has now coming to a point where whole systems can be handled. In aeronautics, simulation has been strong in the area of flight dynamics and control. Modelling and simulation of basic aircraft systems such as hydraulic systems also has a long tradition. The rapid increase in computational power has now come to a point where complete modelling and simulation of all the sub systems in an aircraft is possible.

There are several levels of design from requirement analysis and system architecture down to detail design. There is a clear danger that systems engineering activities are performed only at the top level of a design. In order to have an impact on the product development process it must permeate all levels of the design in such a way that a holistic view is maintained through all stages of the design. This can be achieved if common system model is used where the interaction with other sub-system and the whole aircraft can be studied, and where the system can be optimised from top level requirements.

In this paper it is demonstrated how the actuation system control surfaces can be simulated and optimised, using a flight dynamics model of the aircraft coupled to a model of the actuation system. In this way the system can be optimised for certain flight condition by "test flying" the system. The distributed modelling approach used, makes it possible to simulate this system much faster than real time on a 650 MHz PC. This means that even system optimisation can be performed in reasonable time.

*Keywords: Simulation, Optimisation, Constraint optimisation*

## 1 Modelling and Simulation

The use of simulation for evaluation of performance and behaviour at the design stage is growing rapidly. There also seems to be a dramatic step in model size. The reason for this can be illustrated by the graph in Figure 1.

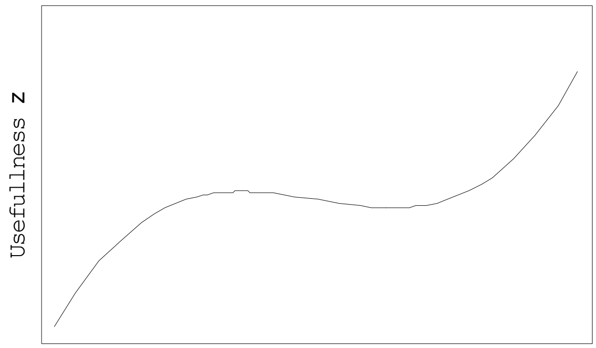


Figure 1. Model usefulness as a function of the degree of model completeness

The usefulness of simulations of a system, initially increases with the degree of completeness of the model. At a certain point, however the curve flattens off and even declines. This is because in order to understand dynamic phenomena, the simplest model that captures the phenomena is the best, since a simple model is easier to understand and to use for analytical studies. If the degree of completeness is increased even more, however, the usefulness of the model tends to increase sharply as a complete (in some sense) model is approached. In this region the model can be used for evaluation of performance and for verification of system behaviour. This requires much more complete models, and has therefore not been used in the past. This is rapidly becoming the norm in system development.

There are several levels of design from requirement analysis and system architecture down to detail design. There is a clear danger that systems engineering activities are performed only at top level of a design. In order to have an impact on the product development process it must, however, permeate all levels of the design in such a way that a holistic view is maintained through all stages of the design. This can be achieved with common model of the complete system, where the subsystem designs can be tested and optimised in an environment where the interaction with other sub-system and the whole aircraft can be studied. It does, however, impose strong requirements on robustness in the simulation models.

The rapid development in simulation methods and the general increase in hardware performance imply that design methods based on different kinds of direct-search optimisation for system design, are becoming much more important. Over the years a number of more or less advanced schemes for design optimization has evolved. There is a relative rich literature in design optimization, see for example G.V. Reklaitis, A. Ravindran, K.M. Ragsdell ,1983, Papalambros, Douglas, Wilde 1988, or C Onwubiko 2000. There are, however, situations where there is very little information regarding the nature of the object function, gradients can not be obtained explicitly, and constraints are implicit. This is true when evaluation of the object function relays on simulation of dynamic systems. In these situations direct-search methods are very attractive.

# 2 REQUIREMENTS ON SIMULATION

Optimisation based on simulation puts very high demands on the numerical efficiency and robustness of the simulation. Since a high number of simulations need to be done, typically ranging from a few hundred to tens of thousands, low simulation times are of course very important.

The following strategy that has been adopted for the development of the HOPSAN package, developed at Linköping University, for simulation based optimisation:

- Modelling on a detailed equation level using a symbolic math package to generate implementation. In order to provide highly numerically robust models
- The distributed modelling approach for partitioning of systems, based on bi-directional delay lines, see (Auslander 1968) and (Krus et al 1990). In order to be able to handle large systems efficiently.
- Different time step for different parts of the model but constant over time. In order to give short and deterministic simulation times.
- Co-simulation, in order to have an open flexible framework for connecting models from different groups using different simulation tools (Jansson 2001)

There is also ongoing development with the prime objective to introduce a model centric architecture as opposed to the traditional tool-centric architecture. The extensible markup language XML is a prime candidate for defining tools independent model-structures. The *Modelith* schema (http://hydra.ikp.liu.se/modelith/) is an effort to define a schema for simulation models. A simulation model can also be a component in a process involving several other design tools.

## 3 Optimisation

The rapid development in simulation methods and the general increase in hardware performance imply that design methods based on different kinds of numerical optimisation for system design, are becoming much more important. Numerical optimisation methods require that the object function is evaluated (using simulation) a large number of times, but they are very attractive since they can optimise complete non-linear systems and do not rely on grossly simplified models as more analytical methods do. Work in this area has shown that optimisation can be used both for parameter optimisation and for component sizing; see Krus, Jansson and Palmberg 1993. It has also been used for component selection, Andersson 2001.

If a system model in the form of a simulation model is defined, it is possible to use optimisation based on simulation. Using this method, the system is simulated using using different sets of system parameters $x_{sp}$. From each system evaluation a set of system characteristics, $y_{sc}$ are obtained and using these, an objective function $f$ is formulated.

In general the simulation is used to obtain the performance characteristics of the system.

$$Y_s = F(X_{sp}) \tag{1}$$

The object function is a function of the system characteristics.

$$f_{obj} = G(Y_{sc}) \tag{2}$$

there may also be a violation flag, that indicates if implicit constraints are violated, that also is a function of system characteristics.

$$c_{viol} = C(Y_{sc}) \tag{3}$$

Another way to deal with constraints is to use a penalty function that is included in the objective function instead.

The objective function is, as mentioned before, function of system characteristics. In many cases there are several objectives that can be more or less difficult to combine into a single objective. In these cases multi-objective optimisation can be used, see Andersson 2001. However, one way of combining several objectives is to use eq (8).

$$f_{obj,0} = \sum_{i=1}^{n} \left( \frac{f_i}{f_{i0}} \right)^{\gamma i} \qquad (4)$$

Here $f_i$ are the partial objectives. $f_{i0}$ represents nominal values used to normalise the equation. Typically, these can be established by manually tuning the system to achieve a reasonable solution. The value of these can then be adjusted until an optimal solution with a desirable combination of characteristics is found. The exponent $\gamma_i$ can be used to further control the behaviour of the solution. A high value ($\gamma$>2) will yield a more sever punishment for objectives worse then their nominal objectives.

Although direct search optimisation based on simulation may sound very time consuming it has some attractive features. Perhaps the most important is that it is possible also to include dynamic properties that can be optimised

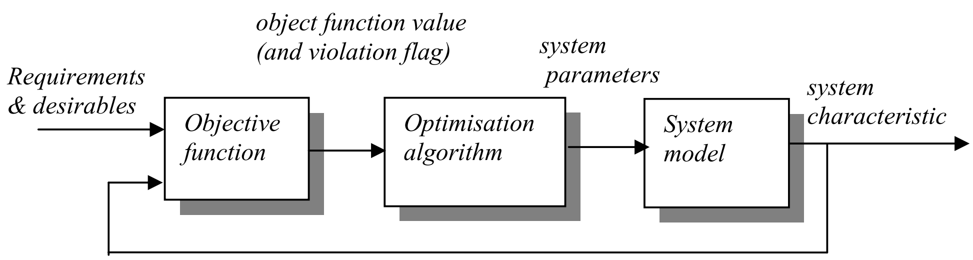


Figure 2. Optimisation based on simulation.

In the general case there exist many explicit relations between parameters in the system. In fact, in manual design, great efforts are made to obtain explicit design relations and there are many cases where system parameters are coupled and cannot be chosen independently from each other. It is therefore appropriate to define a layer of explicit design relations where relatively few independent optimisation variables (design parameters) $x_{dp}$, are expanded to the full set of system parameters, $x_{sp}$.

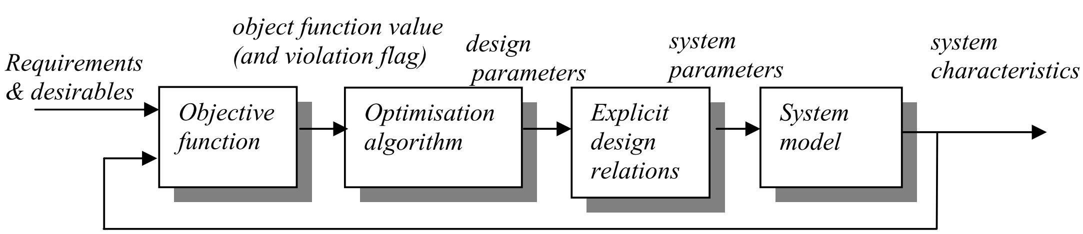


Figure 3. Optimisation based on simulation with layer of explicit design relations.

The explicit design relations can be written as :

$$X_{sp} = R(X_{dp}) \qquad (5)$$

Where $X_{dp}$ is the vector of design parameters. The whole optimisation problem can then be written as:

$$\text{Maximize } f_{obj} = G(F(R(X_{dp}))) \qquad (6)$$

# 4 The COMPLEX-RF algorithm

There are basically two families of optimization methods used in engineering. The gradient methods are widely used and are suitable for problems where the gradient of the object function can be calculated explicitly at each point. This is the case in many structure optimization applications. The other group is the non-gradient methods. These methods do not rely explicitly on gradient information in each point. These methods are therefore of more general use, since gradient information is not generally available, especially if parts of the object function are evaluated using simulation of non-linear systems. The modified version of the original COMPLEX method, by Box 1965) which based on the SIMPLEX method, has been found to be one of the simplest and most easy to use methods, and has been used for system optimization of hydraulic systems See Krus et al 1993. The implementation shown here has also been described in Krus and Gunnarsson 1993. This implementation of the COMPLEX method is also used in some large Swedish companies.

The method can be used to maximise the function.

$$F(x_1, x_2, ..., x_N) \tag{7}$$

subjected to the constraints

$$g_i < x_i < h_i \tag{8}$$

where i = 1,2,....,M. The implicit variables $x_{N+1},...,x_M$ are dependent functions of $x_1,....x_N$. For design, $x_1,...,x_N$ are the design parameters $X_{dp}$ and the dependent functions $x_{N+1},...,x_M$ are a subset of the vector of system characteristics $Y_c$. The constraints $g_i$ and $h_i$ are either constants or functions of $x_1,....x_N$. In the implementation used here, an initial COMPLEX of *m* points is generated. The variables at each point are generated using random numbers.

$$x_{ij} = g_i + r_{ij}(h_i - g_j) \tag{9}$$

Here j is an index that indicates a point in the COMPLEX and i an index that indicates a variable. $r_{ij}$ is a random number in the interval [0,1]. If the implicit constraints are not fulfilled, a new point is generated until the implicit constraints are fulfilled. The number of points *m* in the COMPLEX must be, such that $m >= N + 1$, where *N* is the number of independent variables

The object function is evaluated at each point. The point having the lowest value is replaced by a point reflected in the centroid of the remaining points by a factor α.

$$x_{ij}(new) = x_{ic} + \alpha(x_{ic} - x_{ij}(old)) \tag{10}$$

The centroid is calculated as

$$x_{ic} = \frac{1}{m-1}\left(\sum_{j=1}^{m} x_{ij} - x_{ij}(old)\right) \tag{11}$$

Box (1965) recommends α= 1.3. If a point repeats as the lowest value on consecutive trials, it is moved one half the distance towards the centroid of the remaining points. In this case:

$$x_{ij}(new') = x_{ic} + (x_{ic} - x_{ij}(new))/2 \tag{12}$$

The COMPLEX-RF optimization method used here is a modified version of the COMPLEX method by Box (1965). It is modified by introducing some randomization in the search. This avoids premature collapse of the method.

$$x_{ij}(new) = x_{ic} + \alpha(x_{ic} - x_{ij}(old)) + r \tag{13}$$

Here $r$ is random noise in an interval, which is a fraction of the mean distribution of the parameter sets in the COMPLEX. Another modification involves what happens if a point repeats as the lowest value on consecutive trials. Instead of just moved halfway towards the centroid it is also mirrored in the centroid. This handles constraints or sharp edges in the object function better since it avoids a premature collapse on the edge.

$$x_{ij}(new') = x_{ic} - (x_{ic} - x_{ij}(new))/2 + r \tag{14}$$

## 4.1 Convergence rate

The number of parameters in the COMPLEX m is function of the number of independent variables $m=\kappa n$ where typically $1.5<\kappa<2$. $\Delta x(k)$ is the spread of the COMPLEX parameter set at a particular evaluation no $k$. $\alpha$ is the reflection factor in the COMPLEX method and this has normally the value 1.3, and $n$ is the number of optimization variables (The number of parameter sets in the COMPLEX method is set to typically two times the number of optimization parameters). Expressed as a function of the original spread $\Delta x_0$ the following expression is obtained.

$$\varepsilon = \frac{\Delta x_{k+1}}{\Delta x_0} = \left(\frac{\alpha}{2}\right)^{\frac{k}{2\kappa n}} \tag{15}$$

This means that the number of calculations needed to reduce the spread down to $\varepsilon$ of the original spread can be estimated by:

$$k = -4.64 n \log \varepsilon \tag{16}$$

From this simple relationship follows that, the number of evaluations to reduce a COMPLEX a certain amount, is linearly dependent on the number of points in the COMPLEX. In reality the objective function can be much more complicated than assumed here, but this estimate gives a lower bound to the number of evaluations necessary and a fair description of the behavior near optimum.

An interesting aspect is to study the amount of information gained in each evaluation. In general the amount of information (in bits) to represent a value can be expressed as:

$$I = Log_2 \frac{x}{\Delta x} = -S \tag{17}$$

where $\Delta x$ is the uncertainty of the variable and x its nominal value. S is the entropy, and information I represents negentropy. Therefore the change in entropy, in each iteration, can be written as:

$$\Delta S = nLog_2 \left(\frac{\alpha}{2}\right)^{\frac{1}{2\kappa n}} \tag{18}$$

the multiplication with n comes from the fact that all n variables gain information. This can be simplified to:

$$\Delta S = Log_2\left(\frac{\alpha}{2}\right)^{\frac{1}{2\kappa}} \tag{19}$$

with α=1.3 and k =2 yields

$$\Delta S = Log_2\left(\frac{1.3}{2}\right)^{\frac{1}{4}} = -0.155 \tag{20}$$

This means that the system is gaining 0.155 bits of information at each evaluation (which may seem like a very small value). Note that this is independent of the number of optimization variables. However, for more optimization variables it takes longer time to converge since more information is needed. This represents an upper theoretical limit for the amount of information gained in each function evaluation. In reality even a benign object function gives a convergence rate several times lower than this.

# 5 Example: Simultaneous optimisation of control system and actuation system.

The system model is consisting of a six degree of freedom model of a fighter aircraft and a model of its hydraulic actuation system. See Figure 4. There is also a flight control unit. This could either represent an actual flight control unit or just a system needed to represent a pilot to fly the aircraft through the simulation. Even if the aim of this optimisation is not the design of the flight control system, it needs to be included in the optimisation since different controllers may be needed for different actuation system parameters. There is also a simple engine model to represent the two engines in the aircraft.

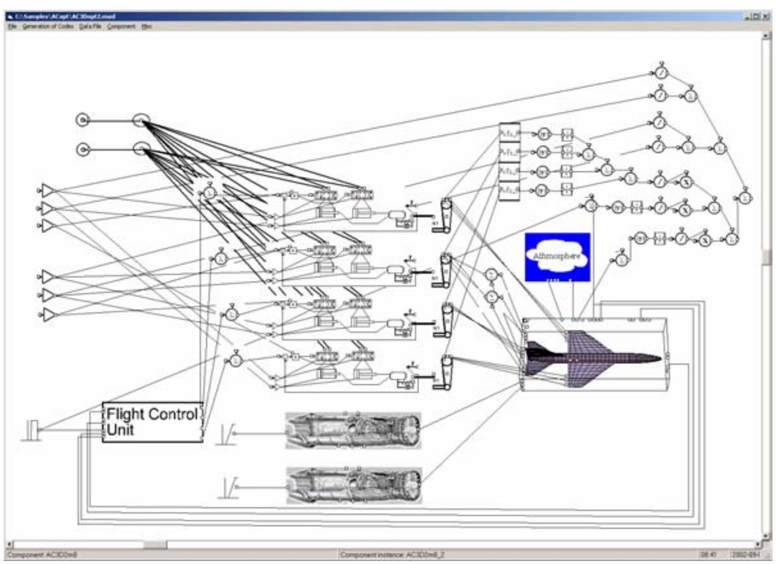


Figure 4. The HOPSAN simulation model

## 5.1 The explicit design relations

In this example there are many explicit design relations that can be used to reduce the number of optimisation variables x. The most obvious one is the symmetry relations. The symmetry requirement is imposed means that there is a left-right symmetry in the control system which means many of the design variables are transformed into two system variables. Another useful mechanism for parameter reduction that also falls into this category is the use scaling. A component such as a servo valve has many design parameters but the driving requirements for a servo valve are usually only flow capacity and bandwidth (speed). This means that is can be assumed that most real valves can be described by only two performance parameters and in this case only one is used which is size. The pistons are also only described by one parameter, which is the piston area.

## 5.2 The objective function

The main objective is to produce an actuation system that can turn the aircraft as fast as it is possible while having as low weight as possible. That means that the components should have as small size as possible. In addition the pressure variation in the actuators is something that should be kept down in order to promote stable systems. In this example there are no constraints except in the explicit design relations. The objective function can be written as

$$f_{obj} = -\left( \frac{Ie_{\varphi}}{Ie_{\varphi 0}} + \frac{Ie_{\theta}}{Ie_{\theta 0}} + \frac{Ip}{Ip_0} + \frac{m_s}{m_{0s}} \right) \tag{21}$$

Here $Ie_{\varphi}$ is the integrated error in yaw angle, $Ie_{\theta}$ is the integrated error in tip. *Ip* is the sum of integrated pressure variations in all the actuators (high pass filtered to remove the DC component). Finally, $m_s$ is the total weight of the actuator system. The optimisation algorithm is set up for finding maximum, hence the negative sign in front of the expression.

There are ten design parameters used for the optimisation in this example. They are

- Size of the aileron pistons
- Size of the elevator and rudder pistons
- The size of the aileron valves
- The size of the elevator and rudder valves
- Gain of the aileron servos
- Gain of the elevator and rudder servos
- FCU gain in pitch
- FCU gain in roll
- FCU gain in yaw
- FCU coupling gain between yaw and roll

In order to be more efficient it is often useful to let the optimisation algorithm operate on the logarithm of the design parameters. This is especially useful when the design parameter space spans several orders of magnitude. The design space for all these parameters was at least one order of magnitude.

## 5.3 Results

The aircraft actuation system was optimised for the case of a sharp 90 deg turn. The simulation started at the time –30 second and the turn was to be initiated at the time 0. The system was then simulated for an additional 40 seconds bringing the total up to 70 seconds. The optimium solution was found after 500 simulation runs.

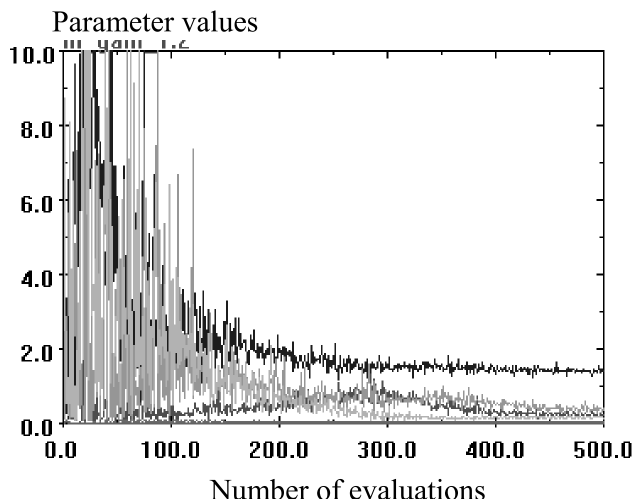


Figure 5. The convergence of the parameter values as a function of number of calculations.

The optimised flight path is shown in the figure below. From this plots it can be seen that the system behaviour is satisfactory. The original parameter design space was very large and most of the initial parameter sets was unable to fly stable at all.

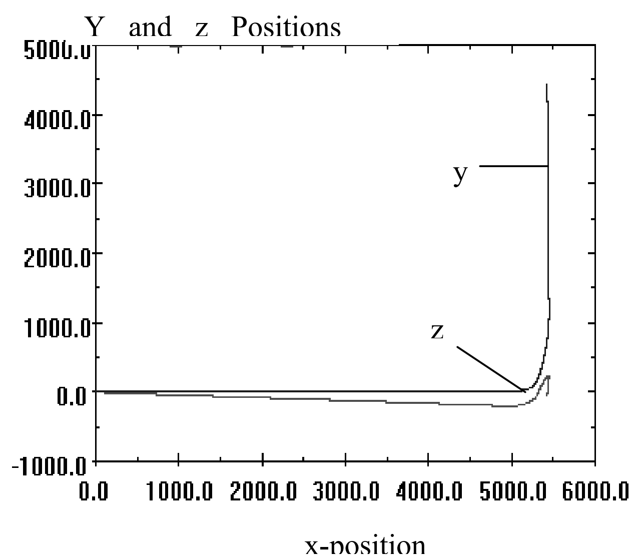


Figure 6. The flight-path of the optimised aircraft.

Looking into the subsystems it is also possible to study the detailed behaviour of components and subsystems

# 6 Conclusions

Optimisation techniques are at the core of computational engineering design and in this paper it has been demonstrated that direct-search optimisation can be used on full-scale simulation models for system optimisation.

Simulation based optimisation does, however, puts special requirements on the models and methods used for simulation. Several stages in the design process can be identified, such as object function formulation and parameterisation of the model. The direct-search optimisation method found most suited for these kinds of problems is the COMPLEX-RF method which also has been analysed in this paper.

**References**


[1.] G.V. Reklaitis, A. Ravindran, K.M. Ragsdell. 'Engineering Optimization', John Wiley & Sons Inc. ISBN 0-471-05579-4. 1983.

[2.] P.Y. Papalambros, D.J. Wilde, 'Principles of Optimal Design', Cambridge University Press, ISBN 0-521-42362-7, 1988

[3.] C. Onwubiko, 'Introduction to Engineering Design Optimization', Prentice Hall, ISBN 0-201-47673-8, 2000.

[4.] D M Auslander, "Distributed System Simulation with Bilateral Delay-Line Models" Journal of Basic Engineering, Trans. ASME p195-p200, June 1968.

[5.] M. J. Box. "A new method of constrained optimisation and a comparison with other methods". Computer Journal*, 8:42--52,* 1965.

[6.] P Krus, A Jansson, J-O Palmberg, "Optimization Using Simulation for Aircraft Hydraulic System Design", Proceedings of IMECH International Conference on Aircraft Hydraulics and Systems, London, UK, 1993

[7.] P Krus, A Jansson, J-O Palmberg, K Weddfelt. 'Distributed Simulation of Hydromechanical Systems'. Presented at "Third Bath International Fluid Power Workshop", Bath, UK 1990.

[8.] J Larsson, "Concept for Multi-domain simulation with application to construction machinery", Thesis No 870, ISBN 91-7219-955-5, Linköping 2001.

[9.] J Andersson, "Multiobjective Optimization in Engineering Design – Applications to Fluid Power Systems", Dissertations No 675, ISBN 91-7219-943-1, Linköping 2001



Petter Krus
Department of Mechanical Engineering
Linköpings Universitet
SE-58183, Linköping, Sweden
Phone: +46(013)281792, Fax: +46(013)285670
petkr@ikp.liu.se