Lappeenrannan teknillinen korkeakoulu
Lappeenranta University of Technology

Juha Kilkki

AUTOMATED FORMULATION OF OPTIMISATION MODELS FOR STEEL BEAM STRUCTURES

Thesis for the degree of Doctor of Science (Technology) to be presented with due permission for public examination and criticism in the Auditorium 1381 at Lappeenranta University of Technology, Lappeenranta, Finland on the 13th of December, 2002, at noon.

Acta Universitatis
Lappeenrantaensis
140

Supervisor	Professor Dr. Gary Marquis Department of Mechanical Engineering Lappeenranta University of Technology Finland
Reviewers	Professor Dr. Károly Jármai Department of Materials Handling and Logistics University of Miskolc Hungary
	Professor Dr. Jack Samuelsson Department of Lightweight Structures Royal Institute of Technology Sweden
Opponent	Professor Dr. Károly Jármai Department of Materials Handling and Logistics University of Miskolc Hungary

ISBN 951-764-713-1
ISSN 1456-4491

Lappeenrannan teknillinen korkeakoulu
Digipaino 2002

ABSTRACT

Juha Kilkki

AUTOMATED FORMULATION OF OPTIMISATION MODELS FOR STEEL BEAM STRUCTURES

Lappeenranta 2002
85 pages and 14 appendices at 30 pages
Acta Universitatis Lappeenrantaensis 140
Diss. Lappeenranta University of Technology
ISBN 951-764-713-1, ISSN 1456-4491

Over 70% of the total costs of an end product are consequences of decisions that are made during the design process. A search for optimal cross-sections will often have only a marginal effect on the amount of material used if the geometry of a structure is fixed and if the cross-sectional characteristics of its elements are property designed by conventional methods. In recent years, optimal geometry has become a central area of research in the automated design of structures. It is generally accepted that no single optimisation algorithm is suitable for all engineering design problems. An appropriate algorithm, therefore, must be selected individually for each optimisation situation.

Modelling is the most time consuming phase in the optimisation of steel and metal structures. In this research, the goal was to develop a method and computer program, which reduces the modelling and optimisation time for structural design. The program needed an optimisation algorithm that is suitable for various engineering design problems. Because Finite Element modelling is commonly used in the design of steel and metal structures, the interaction between a finite element tool and optimisation tool needed a practical solution. The developed method and computer programs were tested with standard optimisation tests and practical design optimisation cases.

Three generations of computer programs are developed. The programs combine an optimisation problem modelling tool and FE-modelling program using three alternate methdos. The modelling and optimisation was demonstrated in the design of a new boom construction and steel structures of flat and ridge roofs.

This thesis demonstrates that the most time consuming modelling time is significantly reduced. Modelling errors are reduced and the results are more reliable. A new selection rule for the evolution algorithm, which eliminates the need for constraint weight factors is tested with optimisation cases of the steel structures that include hundreds of constraints. It is seen that the tested algorithm can be used nearly as a black box without parameter settings and penalty factors of the constraints.

Keywords: optimisation model, steel structure, differential evolution, evolution algorithm, optimisation

UDC 519.85 : 624.014.2

ACKNOWLEDGEMENTS

This study was carried out at the Laboratory of Steel Structures at Lappeenranta University of Technology.

I wish to express my gratitude to Professor Dr. Heikki Handroos, Professor Dr. Aki Mikkola and Professor Dr. Heikki Martikka who have helped me in my research work. Especially, I want to thank Professor Dr. Gary Marquis who supervised the final stages of my research work, proof-read the manuscript and whose guidance made this study possible. I thank Dr. Teuvo Partanen and Emeritus Professor Erkki Niemi for numerous discussions and encouragement.

I thank Mr. Jani Tietäväinen for his programming work. I also wish to thank Dr. Huapeng Wu and Mr. Altti Lagstedt for good co-operation.

The Ministry of Education, the Graduate School of Concurrent Mechanical Engineering and Lappeenranta University of Technology are acknowledged for the financial support and opportunity to finish this thesis. Software development was partially financed by the Finnish Technology Development Centre (TEKES). I am grateful for the financial support provided by the Foundation of Technology in Finland (TES).

Particular thanks go to my mother, Tyyne Kilkki.

Finally, this thesis is dedicated to the memory of my father, Eino Kilkki.

Lappeenranta, November 2002

Juha Kilkki

CONTENTS

NOMENCLATURE

a	constant, exponent
b	width
$c(\mathbf{x})$	cost function or fitness
d	diameter of the piston rod
d_i	discrete value of the i-th discrete variable
d_p	distance between satisfying solution and ideal solution
e	eccentricity
$f(\mathbf{x})$	objective function
f_d	design stress
f_ck	critical stress of compression
f_max	function value of the worst feasible solution in a population
f_j	violation of the j-th constraint
f^0	ideal solution
$\mathbf{f}$	objective function vector
$\mathbf{f}'$	part of objective function vector
f_i	i-th objective function ($i = 1,...,n_\mathrm{f}$)
f_y	yield stress
g	gap
g_i	i-th inequality constraint ($i = 1,...,m$)
$\underline{g}_i$	i-th inequality geometric constraint ($i = 1,...,m$)
g_min	minimum gap
$\mathbf{g}$	inequality constraint vector
$\mathbf{g}'$	part of inequality constraint vector
$g(\mathbf{x})$	inequality constraint function
h	height
h_i	i-th equality constraint ($i = 1,...,q$)
$\mathbf{h}$	equality constraint vector
$h(\mathbf{x})$	equality constraint function
i	index, radius of gyration
j	index
j_rand	randomly generated index
k	index
l	length
m	mass, number of inequality constraints
m_elem	element mass
m_tot	total mass
n	number of design variables, length of the code vector, number of parameters
n_h	safety factor against buckling of the hydraulic cylinder
n_D	number of discrete variables
n_C	number of continuous variables
n_f	number of objective function in multi-criteria optimisation
$n_{\mathrm{p}k}$	number of discrete parameter in set D_k
p	factor on global criterion, hydraulic pressure, weighting exponent
p_max	maximum pressure
p_min	minimum pressure
q	index, number of constraints
r	penalty coefficient

r_1, r_2, r_3	randomly chosen population indices
t	thickness
$\mathbf{u}$	trial vector
u_i	*i*-th design variable of the trial vector
w_i	*i*-th weighting coefficient
x^*	point corresponding to the maximum or minimum value
x	scalar variable
$\mathbf{x}_\mathrm{C}$	vector of continuous variables
$\mathbf{x}_\mathrm{D}$	vector of discrete variables
$\mathbf{x}$	design variable vector
$\mathbf{x}'$	part of design variable vector
x_i	*i*-th design variable
x	coordinate in global coordinate system
x'	coordinate in element coordinate system
x_i^l	lower limit of *i*-th design variable vector
x_i^u	upper limit of *i*-th design variable vector
$\mathbf{x}_\mathrm{C}$	vector of continuous variables
$\mathbf{x}_\mathrm{D}$	vector of discrete variables
y	coordinate in global coordinate system
y'	coordinate in element coordinate system
$\mathbf{y}$	input vector, vector
$\mathbf{y}'$	part of input vector
z	coordinate in global coordinate system
z'	coordinate in element coordinate system
A	cross-sectional area
C	constant
CR	mutation probability
D	domain, diameter of the piston, set of discrete variables
D_i	set of discrete values for the *i*-th variable
E	modulus of elasticity
F	force, differential factor, working or nominal load
F_cr	buckling load
F_d	design load
$F_\mathrm{y}, F_\mathrm{z}$	shear forces
G	shear modulus, generation
G_max	last generation
h	scheme, height
H	scheme string
I	moment of inertia
L	span length
L_c	buckling length
M	bending moment
M_R	bending resistance
M_x	torsional force
$M_\mathrm{y}, M_\mathrm{z}$	bending moments
N	normal force, fatigue life
N_R	normal force resistance
N_Rt	tension resistance
NF	number of function evaluations
NP	population size

PL	cross section class
Q_i	penalty of the *i*-th constraint
P	population
R	strength resistance
$\mathbb{R}$	set of all real numbers
$\mathbb{R}^n$	n-dimensional Euclidean vector space
V	shear force
W	Elastic section module
W_p	Plastic section module
α	factor (buckling), penalty exponent, rotation angle of a beam, coefficient
β	penalty exponent, angle, coefficient
β_1, β_2	constants in a feedback function
γ	buckling factor
γ_F	partial safety factor of the load
γ_J	partial safety factor of the joint
γ_m	partial safety factor of the material
δ	deflection
δ_max	maximum deflection
ε	error
η	efficiency factor
θ	angle of rotation, angle between beams
$\theta(G, x)$	iteration dependent function
λ	slenderness ratio
$\bar{\lambda}$	reduced slenderness ratio
$\lambda(G)$	feed back function
ν	Poisson's ration, number of violated constraints
σ	stress
ψ	factor
$\mathscr{F}$	feasible area in optimisation problem
$\Delta\sigma_\mathrm{eq}$	equivalent stress range
$\Delta\sigma_\mathrm{R}$	fatigue resistance
$\Delta\sigma_\mathrm{R,k}$	the characteristic stress range at the required number of stress cycles
$\Delta\sigma_\mathrm{S,d}$	design value of stress range caused by actions
∅	diameter
*	symbol corresponding zero or one

MATHEMATICS SYMBOLS

$\forall$	universal quantifier	($\forall x$, for all x,...)
$\exists$	existential quantifier	($\exists x$, there exists an x such that)
$\wedge$	conjunction	($P \wedge Q$, P and Q)
$\vee$	disjunction	($P \vee Q$, P or Q)
∞	infinity	
$\subset$	subset	(A is a subset of B)
$\in$	element of region	($x \in A$, element x belongs to the set A)
$\rightarrow$	mapping	
$\|x\|$	absolute value	
$\|\|\mathbf{x}\|\|$	vector norm	
Σ	sum	
Π	product	
rand(x)	random number of $[0, x)$	

SUBSCRIPTS

ap	attachment plate
c	compression
d	design value
e	effective
eq	equivalent
el	elastic
elem	element
f	flange
hc	hydraulic cylinder
j	joint
k	characteristic
min	minimum value
max	maximum value
node	node
opt	optimal
p	plastic
rand	randomly selected
t	tension
tel	telescope
v	shear
w	web
F	load
C	continuous variable
D	discrete variable
R	resistance

SUPERSCRIPTS

l	lower bound
u	upper bound
*	optimal value

ABBREVIATIONS

3D	Three dimensional
AGIFAP	Advanced graphical interactive frame analysis package
CAD	Computer-aided design
CHS	Circular hollow section
CPU	Central processing unit
CL	Symmetry line
DC	Direct current
DE	Differential evolution
DEC	Differential evolution component
DLL	Dynamic link library
DOF	Degree of freedom
EA	Evolution algorithm
FAT	Fatigue class
FE	Finite element
FEA	Finite element analysis
FEC	Finite element component
FEM	Finite element method
GA	Genetic algorithm
ID	Identifier
LUT	Lappeenranta University of Technology
MESO	Modified evolutionary structural optimisation method
MC	Model component
OGL	Open graphic library
OOP	Object oriented programming
OM	Optimisation model
PC	Personal computer
RHS	Rectangular hollow section
TC	Table component
UNIX	multi-user operating system

1 INTRODUCTION

1.1 Background

A customer sets requirements and wishes for capacity, dimension and mass. The laws of physics set strict constraints. A new construction must be designed with adequate strength and, if public or environmental safety is a concern, standards and sometimes legislation set strict requirements for reliability and safety. Manufacturers and designers pursue the lowest possible fabrication costs and good profit while end users are concerned with total life-cycle cost. There are always numerous design alternatives that fulfil these requirements without exceeding constraints and the process of selection constitutes a problem of *optimisation*.

It is a generally accepted truth that no single optimisation algorithm is suitable for all engineering design problems. An appropriate algorithm, therefore, must be selected individually for each optimisation situation. Choosing the optimisation algorithm and formulating the problem requires, at least, some basic knowledge about optimisation theory and a certain degree expertise about the structure itself. For this reason there is normally a high threshold for using optimisation algorithms in engineering work.

The total cost of a steel structure includes those for material, fabrication, transportation and erection. Material cost includes both unfinished materials and semi-finished members such as beams, columns and bracings. Fabrication costs include in-shop processes like cutting, welding and painting as well as transporting the fabricated sections to the construction site. Erection costs include the costs of the connection elements like bolts and electrodes and the labour cost. Sarma and Adeli have prepared a recent review of cost optimisation of for steel structures (Sarma, K. 2000).

It has been argued that over 70% of the total costs of an end product are consequences of decisions that are made during the design process. For this reason significant investments are usually made to ensure the effectiveness of this process and to train designers. Researchers in this field have proposed different flow charts both to describe and assist in the design process. These normally consist of sequential steps and feedback loops that should be followed to reach the design goal. Also, different kinds of question lists, tables or image maps are employed to stir up a designer's imagination and creative skills to help him find some real new alternatives (Eskelinen, H. 1999). Taipale has emphasized that the optimisation of a structure must also always include an economic study (Taipale, J. 1999).

For fabricated metal structures, a search for optimal cross-sections will often have only a marginal effect on the amount of material used if the geometry of a structure is fixed and if the cross-sectional characteristics of its elements are properly designed by conventional methods. The geometry of the structure determines, to a large extent, its structural efficiency. In recent years optimal geometry has become a central area of research in the automated design of structures. (Fuchs, M. B. 2001).

1.2 Relevant published works

Computer aided design and analysis programs are feature common in most types of design work. This may include functional simulation by means of highly accurate virtual prototypes (Mikkola, A. 1997). Linear or non-linear finite element analysis (FEA) is used to determine both stresses and deflections. For fatigue-loaded structures, crack growth can be simulated with finite element (FE) based fracture mechanics.

A large number of potential optimisation algorithms exist. For more complicated systems, it is possible to link programs and algorithms together (Dulikravich, G. S. et al. 1999). It has been shown that evolution based optimisation algorithms can be used to achieve good solutions, but there is usually a serious problem with analysis time. When the time for a single analysis is short, i.e., a few milliseconds, there is usually no problem with evolution-based algorithms. However, virtual prototypes are non-linear systems consisting of complex interaction between flexible element, rigid motion, hydraulic circuits and control systems. Simulating one second of prototype working time usually requires 5 to 15 minutes of central processing unit (CPU) time. In some cases the computation time can be reduced by distributed cost function evaluation.

In order for an optimisation algorithm to be used in the process of design, the engineer must create a general configuration in which the numerical values of the independent design variables have not been fixed. Steps for formulating the optimisation problems are (Siddal, J. N. 1982): 1) A configuration of general form is selected. 2) The design variables must then be explicitly defined. These are the quantities x_1, x_2, ..., x_m that the designer knows can be adjusted during the design process. 3) The input specifications must also be carefully defined. Many quantities, like loads on the structure, are fixed inputs. 4) Next, design characteristics considered important as optimisation criteria are defined and written with a functional expression. 5) Potential failure modes are identified and formulated as constraints. 6) Additional constraints are formulated as required to ensure that the configuration is not violated and the design does not lie outside the region of validity of the mathematical model. Other innate requirements of the design, e.g., geometric constraints, are also formulated. The easiest type is a simple size limitation. For example, a configuration constraint for a tube is that the outside diameter must be greater than the inside diameter. The design has to stay in the region where the mathematical model is known to be valid. 7) Equality constraints and state variables are eliminated by substitution where possible. Any state variables that cannot be eliminated must be added to the list of design variables. The expressions are converted to any standard form required by the computer program being used.

Figure 1.1 provides one method of visualising the planning and design process developed by Pahl et al. (Pahl, G. et al. 1996). The planning and design process can be divided into the phases: planning and clarifying the task, conceptual design, embodiment design and detail design. At the initial phases of product development, a product idea is needed that looks promising given the market situation, company needs and economic outlook. The purpose of clarification of the task is to collect information about the requirements that have to be fulfilled by the product, and also about the existing constraints and their importance.

The conceptual design phase determines the principle solution. If required by the optimisation technique, a set of starting values for the independent variables is selected. This concretisation involves selecting preliminary materials, producing a rough dimensional layout, and considering technological possibilities. It is possible to assess the essential aspects of a solution principle and review the objectives and constraints. The construction structure or overall layout of a technical system is determined during the embodiment design. Detail design is the phase of the process in which the arrangement, forms, dimensions and surface properties of all individual parts are finally decided, the materials specified, production possibilities assessed, and costs are estimated. Quite often correction must be made and the preceding steps repeated. The crucial activities are optimisation of the principle, layout, forms, materials and production. They influence each other and overlap to a considerable degree. An example of this type of systematic approach toward design is presented in Figure 1.1 and is general enough to be utilised within most technical research areas.

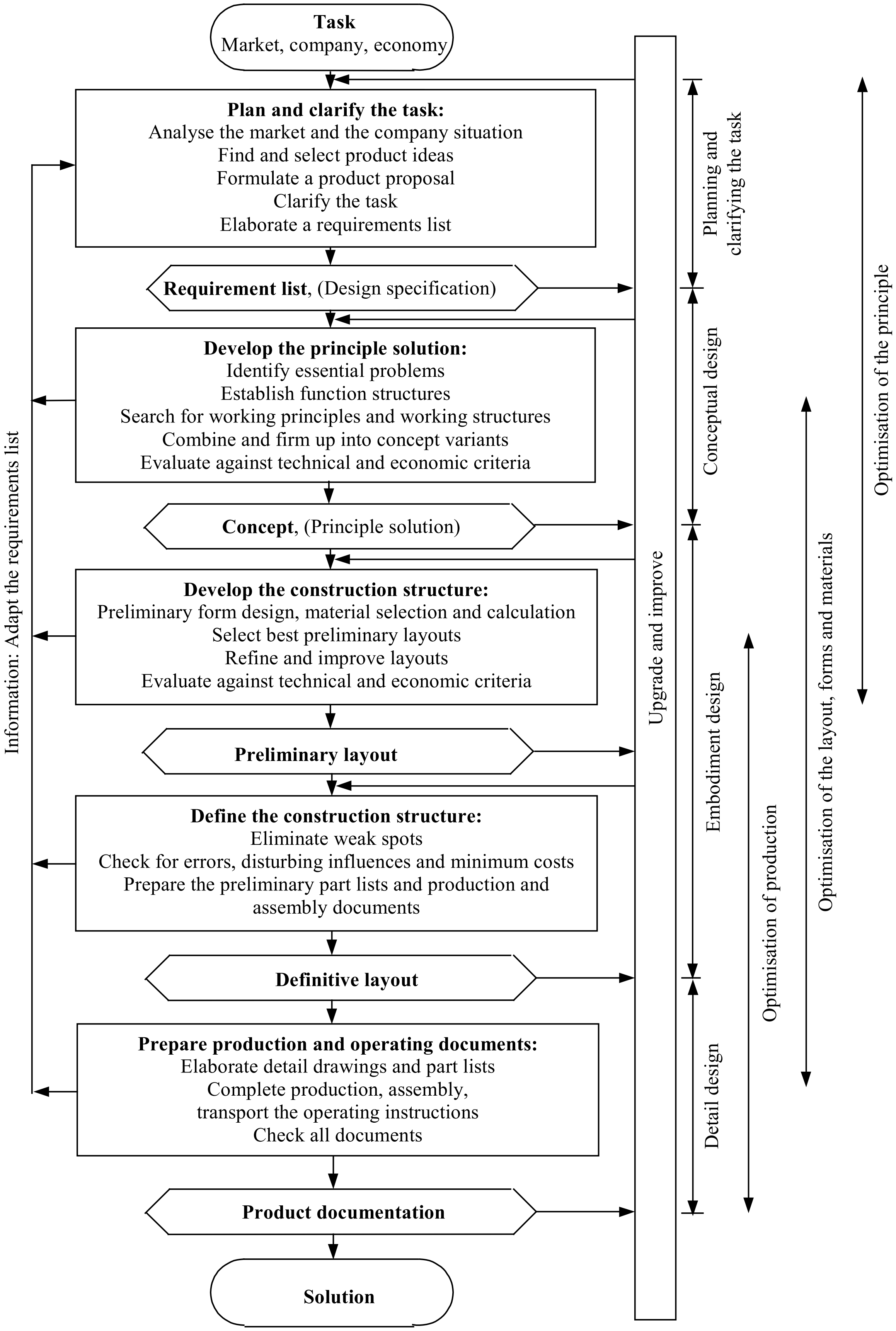

Task
Market, company, economy
Plan and clarify the task:
Analyse the market and the company situation
Find and select product ideas
Formulate a product proposal
Clarify the task
Elaborate a requirements list
Requirement list, (Design specification)
Develop the principle solution:
Identify essential problems
Establish function structures
Search for working principles and working structures
Combine and firm up into concept variants
Evaluate against technical and economic criteria
Concept, (Principle solution)
Develop the construction structure:
Preliminary form design, material selection and calculation
Select best preliminary layouts
Refine and improve layouts
Evaluate against technical and economic criteria
Preliminary layout
Define the construction structure:
Eliminate weak spots
Check for errors, disturbing influences and minimum costs
Prepare the preliminary part lists and production and assembly documents
Definitive layout
Prepare production and operating documents:
Elaborate detail drawings and part lists
Complete production, assembly,
transport the operating instructions
Check all documents
Product documentation
Solution
Information: Adapt the requirements list
Upgrade and improve
Planning and clarifying the task
Conceptual design
Embodiment design
Detail design
Optimisation of production
Optimisation of the layout, forms and materials
Optimisation of the principle

Figure 1.1 Steps of the planning and design process according to Pahl et al. (Pahl, G. et al. 1996).

In discussing constraints, it should be pointed out that the definition requires an understanding of the technology. In this way structural design is much more demanding than conventional mechanical engineering design. Every possible mode of failure must be included since the computer will go on blindly searching for optimum, unmonitored by the designer's judgement to stop it from entering dangerous uncharted regions. However, if the designer is faced with a failure mode that he or she cannot formulate mathematically, it still may be possible to preclude failure by a very approximate conservative constraint, although at the risk of some loss of optimality (Siddal, J. N. 1982). That leads to the situation where the optimisation model consists on hundreds of constraints. High number of constraints is not a problem when using evolutionary algorithms but the modelling is time consuming.

There are numerous potential difficulties in formulating the optimisation model. Physical or engineering expressions often include dangerous mathematical formulas that cause problems if a variable is allowed to become negative:

$$\sqrt{x}, \quad \log(x) \quad \text{and} \quad x^a$$

and when x is zero in cases

$$\log(x) \quad \text{and} \quad \frac{a}{x}$$

The qualification area of the formulas has to be taken into account.

Integer and discrete variables are very common in engineering design. For example, an integer variable occurs when the *quantity* of some identical components is a variable. Discrete variables usually arise from discrete standard sizes of readily available materials. Integer programming methods are available, but these are slow and unreliable. There is some risk that the rounding will move the design away from its optimum value or into infeasibility if integer and discrete variables are treated as continuous during optimisation and rounded to the nearest integer or discrete value. (Siddal, J. N. 1982). Some discrete values are also dependent on other previously selected values, e.g., the rod and the piston rod diameter of a hydraulic cylinder are directly dependent on the chosen cylinder diameter.

One of the first problems in defining design variables is to decide which quantities should be given initially specified values and which should be considered variables. Material properties can be varied, but in most cases it is more practical to pre-select materials and their properties based on experience. Variables may also be limited arbitrarily to reduce the complexity of the problem. On the other hand, there is the danger that some variables will be overlooked. (Siddal J. N. 1982).

Many elegant solutions and methods for optimisation of steel structures are presented in optimisation textbooks, conference proceedings and scientific papers. The problem is that these methods are quite often focused on one specific optimisation case and are based on some specific codified design norm. Farkas and Jármai have presented a large collection of solutions for optimum steel structures and cost calculation and optimisation of welded steel structures (Farkas, J. et al. 1997, Jármai, K. et al. 1999). Shrestha and Ghaboussi (Sherestha, S. M. et al. 1998) have proposed a methodology, which uses a genetic algorithm (GA) to evolve optimum shape designs for skeletal structures. In this method, all three shape aspects of skeletal structures, sizing, geometry and topology, are simultaneously considered. The members are chosen from a set of discrete member sizes. The local strengths of the joints are not considered. Tanskanen (Tanskanen, P. 2000) has proposed a modified evolutionary structural optimisation method (MESO). The method is implemented in problems involving linearly elastic planar structures under static single loading conditions. Ohsaki et al. (Ohsaki, M. et al. 1998) have formulated the topology optimisation problem of trusses for specified eigenvalue of vibration by means of semi-definite programming.

Takada et al. have presented an optimisation of shear wall allocation in three dimensional (3D) frames by the branch-and-bound method. The allocation design of shear walls in a multi-storied 3D building system has been reduced to a design problem of appropriate selections of wall sections from a large number of discrete candidates. The problem is one of combinatorial optimality. (Takada, T. et al. 2001).

Optimisation of both geometry and cross-section of a truss structure has made by Gil. The geometric design problem is defined by unknown nodal coordinates and is combined with a parametric design problem defined by the cross-sections. The methodology combines a full stress design optimisation with a conjugate gradient optimisation. (Gil, L. et al. 2001).

Several commercial design and optimisation software packages provide tools for optimising specific complex engineering systems. Software *Engineus*, for example, combines genetic algorithms, expert systems, and object-oriented programming with numerical optimisation and annealing simulation. It has been applied in the design of an aircraft engine turbine, a molecular electronic structure, a cooling fan, a direct current (DC) motor, an electrical power supply, nuclear fuel rods, and the aerodynamic and mechanical 3D design of turbine blades.

Tong has presented an optimisation procedure for the minimum weight optimisation with discrete variables for truss structures subjected to constraints with respect to stresses, natural frequencies and frequency response (Tong, W.H. 2000). The first step in this method is to find a feasible basic point by defining a global normalised constraint function and using a difference quotient method. The second step is to determine the discrete values of the design variables by analysing the difference quotient at the feasible basic point and by converting the structural dynamic optimisation process into a linear zero-one programming. A binary number combinatorial algorithm is employed to perform the zero-one programming.

Fuchs deals with optimisation for maximum stiffness of controlled truss-type structures subjected to a class of unknown disturbances. A constant volume constraint was imposed on the truss. Because the selection of a single "optimal" structure is very sensitive, he presents a methodology to design numerous sub-optimal, or near-optimal, structures (Fuchs, M. B. 2001).

Erbatur et al. report the development of a computer-based systematic approach for discrete optimal design of planar and space structures composed of one-dimensional elements. A genetic algorithm is used as the optimiser. An approach based on a proposed multilevel optimisation is tested (Erbatur, F. et al. 2000).

Hayalioglu has presented the optimum design of geometrically non-linear elastic-plastic steel frames with discrete design variables. Large displacement restrictions are considered in the optimum designs. However, the algorithm is time consuming and requires non-linear analyses of a large number of frames. (Hayalioglu, M. S. 2000).

Manickarajah has used an evolutionary method in the optimum design of frames with multiple constraints. The optimisation proceeds by slowly removing inefficient or low stressed material and/or gradually shifting material from the strongest part of the structure to the weakest part. The method involves two steps. First, design variables are scaled uniformly to satisfy the most critical constraint. In the second step, a sensitivity number is computed for each element depending on its influence on the strength, stiffness and buckling load of the structure (Manickarajah, D. et al. 2000). A related method for evolutionary structural optimisation to resist buckling has been proposed by Rong J. H. et al for maximising the critical buckling load of a structure of constant weight (Rong, J. H. et al. 2000).

A method for optimum design of steel frames with frequency constraints has been proposed by Salajegheh (Salajegheh, E. 2000). In order to reduce the number of frequency analyses that are required in the optimisation process, the frequencies are approximated during each design cycle (Salajegheh, E. 2000). Kameshki has used a genetic algorithm in the design of non-linear steel frames with semi-rigid connections. The design algorithm eventually achieves a frame of minimum weight by selecting appropriate sections from a catalogue of standard steel sections. He has used a non-linear empirical model to include the moment-rotation relation of beam-to-column connections. (Kameshki, E. S. et al. 2001). The conceptual design of buildings based on genetic algorithms has been presented by Miles et al. (Miles, J. C. et al. 1999). The system, called BGRID, employs a genetic algorithm to search for viable design options. The design process is focussed on determining the layout of columns based on a large number of criteria. These include lighting requirements, ventilation strategies, limitations introduced by the available sizes of typical building materials and the available structural system. The genetic algorithm is used more as a search engine as opposed to an optimisation tool.

The use of discrete optimisation techniques in reliability-based design of truss structures has been studied by Strocki et al. The problem is formulated as the minimisation of structural volume subjected to constraints on the computed reliability of individual components. Cross-sectional areas of truss bars and coordinates of the specified truss nodes are considered as discrete and continuous design variables. The specified allowable reliability indices are associated with specific limit states. These limit states are 1) admissible displacements of the chosen truss nodes, 2) admissible stress or local buckling of the elements, or 3) global loss of the stability. Transformation and controlled enumeration methods are employed to solve the optimisation problem (Stocki, R. et al. 2001).

The finite element method (FEM) is a widely used engineering tool. However, finite element analysis (FEA) does not provide direct and clear conclusions about the strength of a steel structure. The FE-results have to be processed based on strength analysis principles. Mesh sizes and element types have to be chosen based on experience at the start of the modelling process. Beam elements are commonly used in the analysis of the steel structures. Solid or thin shell elements are selected commonly in the fatigue analysis when the area subject to damage is highly local. Typically, the size of this local area is equal to the plate thickness. However, most design code based strength computations are based on nominal forces or stresses in a structure. In the case of non-redundant structures, the nominal forces can usually be solved by hand calculation while computerized FE analysis is used for statically indeterminate structures. In most cases the designer must first construct a geometric model, which is then solved for the various load cases to determine the limiting case for the structure. Strength of the structure is often calculated by comparing the calculated resistances of the details of the structure to the forces calculated by the FE analysis. This procedure is time consuming and does not assist in the formulation of an optimisation model.

Several commercial FE-programs contain optimisation packages, but these packages are difficult to link to common design code based calculations. Another disadvantage is that the optimisation models are still complex and time consuming to build. The common trend in FE-program development is toward better efficiency. FE-models can contain more degrees of freedom and the analysis can be non-linear. These kinds of programs are very useful in the aviation and car industries where a single FE-model may contain over 100 million elements. These FE-models usually require parallel computing and expensive hardware. However, for the vast majority of engineering applications, more economical programs and methods for the design and optimisation of the steel structures is required.

Object oriented programming can greatly improve the implementation efficiency, the ability to extend and ease of maintenance of large software systems (Lichao, Y. 2001). In this research project, the programs have been developed using object-oriented programming. This greatly simplifies the task of utilising already developed classes in the future versions of the work. Lämmer et al. have presented a means for integrating object-oriented construction and simulation models. In their study, it was noticed that a common product model that can reflect all the necessary facets and stages of the process does still not yet exist. The integration of design and simulation models in structural engineering, primarily based on computer-aided design (CAD) and finite element modelling, is an essential requirement for efficient data flow between existing program solutions (Lämmer, L. et al. 2001).

1.3 Scope of the thesis

This thesis focuses on the technical fields of mechanical engineering design and structural engineering design. Subjects addressed in this work are common for machines fabricated from steel components and actuators such as hydraulic cylinders, pneumatic cylinders and electric motors, gears and bearings. The design problems treated are the dimensioning and selection of components and materials.

The early version of the optimisation program, OPTIMAZE, was developed in the research and development project, "On-line optimisation of metal structures" (Kilkki, J. 2000). The FE-program Advanced Graphical Interactive Frame Analysis Package (AGIFAP) was first developed in the Laboratory of Steel Structures at Lappeenranta University of Technology (LUT) but was later further developed by the author (Kilkki, J. 2001).

In frame structures, elements carry bending loads while in truss structures elements carry axial loading. A typical steel structure and an idealised truss are shown in Figure 1.2. In most practical design problems, the design variables are discrete due to the availability of standard sizes for the steel members and practical limitations related to both construction and manufacturing. The phases of the design of the truss are presented in Figure 1.3. These are:

1) Loads on the structure are clarified and the most dangerous load combinations are determined.
2) Height of the structure is decided. The type of the truss and cross member division is selected.
3) Preliminary selection of the beams. Resistance of the most heavily loaded joint is calculated
4) Real forces of the beams are calculated
5) Strengths of the joints are calculated
6) Deflection of the truss is calculated
7) Cross members and the joints of the cross members are designed

In this thesis, phases 3 to 6 are automated by incorporating a suitable automation algorithm according to Figure 1.4.

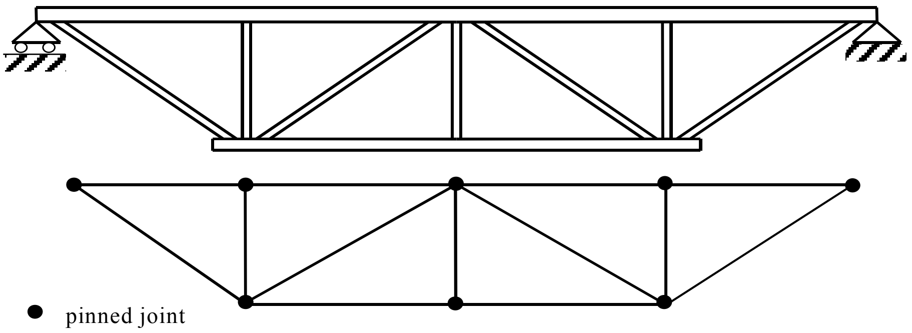


Figure 1.2 A typical steel structure and the idealised truss structure. Pinned joints do not carry bending moments.

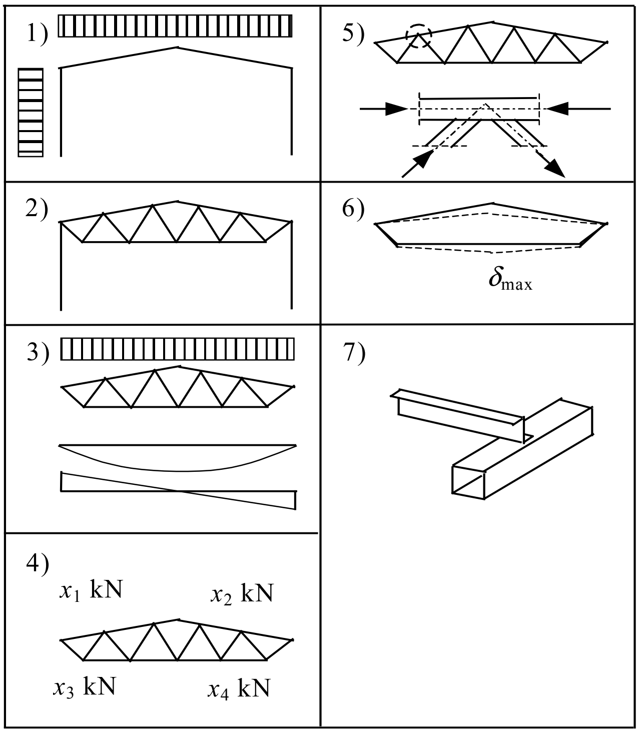


Figure 1.3 Phases of the design of the truss structure in common design method.

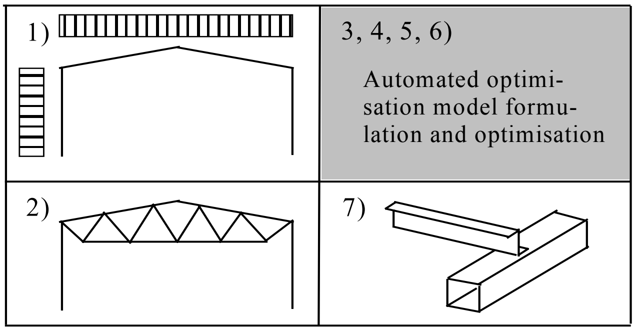


Figure 1.4 The new automated optimisation model formulation of the truss structure in the new design method.

1.4 The research methods

The research problem has been to formulate and solve a model for steel truss structures using automated optimisation. Efficiency of the optimisation algorithm itself has been checked using common test functions for evolution algorithms. The main goal of the research project was to investigate the possibilities for automating the modelling of the optimisation models. Modelling tools were tested with real-life steel structure design problems and defects and inadequacies of the modelling tool were examined and used as inputs for further development of the later modelling tools. Quantitative assessment of the modelling tools is a difficult task and the suitability of the developed modelling tools has been reported by presenting the good and bad features in this thesis.

The effect of the modelling tools enhancement was tested and is presented in this thesis through the optimisation problems of one boom structure and three truss structures. Previously reported applications for the modelling tools include, e.g., the optimisation of an I-beam cross section (Kilkki, J. et al. 2001). Unpublished but interesting cases also include the optimisation of a harbour crane and the shape optimisation of a back box for a paper machine. In the paper machine case, the developed modelling tool has been linked to the commercially available Fluent 5.0 and Gambit programs. Analysis time of the objective function was about 20 minutes. Objective function evaluations were distributed in parallel to three multi-user operating system (UNIX) workstations. The modelling software and optimisation program have also been used in a virtual prototype study of damping parameters for a paper machine roll (Sopanen, J. et. al. 2000).

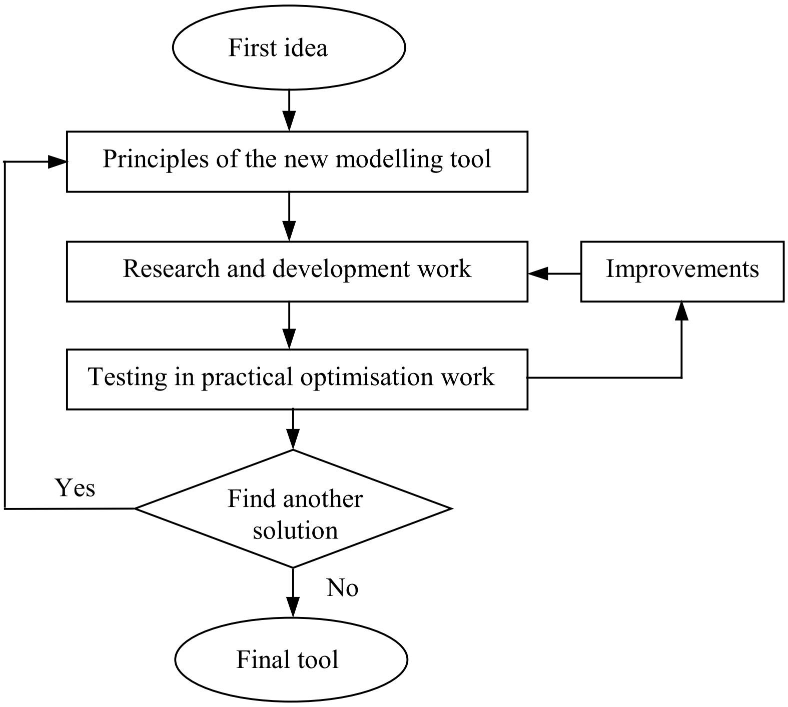

First idea
Principles of the new modelling tool
Research and development work
Improvements
Testing in practical optimisation work
Yes
Find another solution
No
Final tool

Figure 1.5 Process of the research and development work.

Figure 1.5 presents a flow chart of how the research has proceeded. After the initial idea to develop the common optimisation program was conceived, a first version of the modelling and optimisation program was developed. This program was tested on several practical optimisation cases. Experience with the program was used as feedback to define needed improvements for the program. Eventually, further improvements to the modelling tool proved futile so that a new type of solution was required. Advantages and disadvantages observed in one modelling tool provided important input for the development of newer tools. The eventual modelling tool required three of the large iteration loops shown in Figure 1.5 and countless smaller improvements.

1.5 Overview of the dissertation

This thesis focuses on the subject of the integration of optimisation in design engineering work. A new method for integrating the global optimisation algorithm in the design of steel structures is presented. The optimisation algorithm itself is a version of the Differential Evolution (DE) algorithm.

Chapter 1 in this thesis presents the background, motivation of the writer to the presented study, an introduction to the research method used, and a summary of the features of the work though to be original.

In Chapter 2, those methods and tools that are fundamental to the thesis research are presented. The optimisation methods are discussed briefly together with the global optimisation methods. Specific attention is given to the theoretical background and limitation of the evolution algorithms. Characteristics of the optimisation algorithm form one set of constraints for the modelling tools and these are presented in some detail.

The optimisation model definition method developed during this research project is presented in Chapter 3. This chapter presents two computer programs. The first is an optimisation modelling and solving program that automatically formulates the design while the second is a modified finite element FE analysis program that is used to evaluate the constraints and objective functions.

Chapter 4 presents the numerical output produced by the developed and modified programs. The optimisation portion of the program is first tested using numerous standard test functions and optimisation problems. The optimisation system, consisting of both the modelling and FE analysis programs, is then tested on several large steel structures optimisation problems. These structures are a hydraulically driven multi-redundant boom and several truss structures.

Results of the new modelling and optimisation tool are discussed in Chapter 5. Advantages and disadvantages of the method are presented and evaluated. This chapter presents important information for future development of design optimisation tools for steel structures.

Chapter 6 presents a summary and important conclusions of this dissertation.

1.6 Contribution of the dissertation

In this thesis, a method is presented to utilise the differential evolution optimisation algorithm in the design of mechanical engineering steel structures. The main problems associated with optimisation in mechanical engineering are presented. Two significant problems are 1) the interaction between designer, optimisation model and optimisation algorithm, and 2) the definition and formulation of the optimisation model. A solution of these problems is presented. Three modelling programs have been developed that assist in the interaction between an optimisation model and the designer. These programs help the designer to formulate and solve an optimisation model. The first program uses graphical components, which consists of mathematical formulas, - tables of discrete components and finite element solvers. The finite element method is commonly used in the design of the steel structures. This thesis presents an automated optimisation model formulation of the FE-model. Modelling tools have been developed taking advantage of object oriented programming techniques. The optimisation modules of these programs are tested with evolution algorithm test problems.

The first assumption is that design costs are lower when the strength of the design can be checked immediately after the finite element model is completed. The second assumption is that design costs decrease further if the finite element model can be optimised right after modelling.

The common aim of this thesis is to combine a modern evolution based optimisation algorithm, engineering design and the finite element analysis. This thesis focuses on the technical area represented by the intersection of the three ellipses in Figure 1.6.

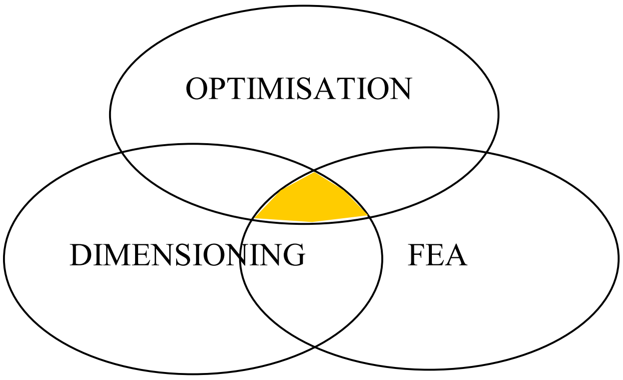


Figure 1.6 The intersection of the ellipses is the focus of this thesis.

The following claims in this thesis are considered to be original:

1. Three unique combinations of optimisation problem modelling tools and FE-modelling program have been created.

2. An optimisation tool consisting of editable components is developed and demonstrated in engineering design applications.

3. A compiled optimisation tool for FE-modelling and optimisation is created and demonstrated in mechanical equipment.

4. An automated formulation of the optimisation model of the steel beam structures is developed and tested in real optimisation problems of steel structures. The most time consuming modelling time is reduced.

5. A new selection rule for the evolution algorithm that eliminates the need for constraint weight factors is tested with optimisation cases of the steel structures. That is a remarkable advantage for optimisation of steel structures which contain hundreds of constraints. Result is that the algorithm can be used nearly as a "black box" and reduces the optimisation and modelling time.

2 SURVEY OF EXISTING METHODS

2.1 Finite element method

The finite element method is a numerical procedure for solving continuum mechanics problems with an accuracy acceptable to engineers. In structural analysis the displacement method is normally used. This means that displacements of discrete locations within the structure are the primary unknowns to be computed. Stress is a secondary variable and is computed from displacements based on a suitable constitutive relationship (Cook, R. D. 1981).

Real-life structures can normally be modelled as the sum of individual parts. Figure 2.1 shows an example of a truss structure, which is labelled using the symbols used throughout this manuscript. Truss distortion can be defined completely based on displacements at the nodes of the model. Forces and moments can be defined for both the i- and j-ends of the beams. These forces and moments can be utilised directly to evaluate the failure resistance of the structure, e.g., according a desired design code or norm.

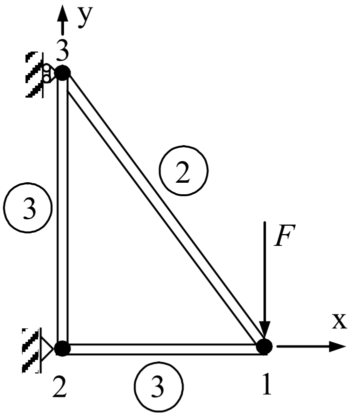


Figure 2.1 Nodes and elements of the finite element model. Element identifiers (IDs) are circled.

2.2 Limit state design

The central concepts of the limit state design are explicit reference to 'limit states', the definition of the nominal loads and stresses used in calculations in terms of statistical concepts and the use of the partial safety factor format.

'Limit sates' are the various conditions in which a structure would be considered to have failed to fulfil the purposes for which it was build. There is a general division into ultimate and serviceability limit states. The former are those catastrophic states which require a large safety factor in order to reduce their risk of occurrence to a very low level and the latter are the limits on acceptable behaviour in normal service. All these limit states require structural calculations (Dowling, P. J. 1988).

The limit states for which steelwork is to be designed are ultimate limit states and serviceability limit states. Ultimate limit states are: strength (included general yielding, rupture, buckling and transformation into a mechanism, stability against overturning and sway, fracture due to fatigue, excessive deflections and brittle fracture. When the ultimate limit states are reached, the whole structure or part of it collapses. Serviceability limit states are deflection, vibration (for example, wind-included oscillation), reparable damage due to fatigue, corrosion and durability, plastic deformations and the slip of the friction joints. The serviceability limit states, when reached, make the structure or part of it unfit for normal use but do not indicate that collapse has occurred (Mac Kinley, T. 1987, B7 1996).

Factored loads are used in design calculations for strength and stability. Factored load F is a working or nominal load multiplied by relevant overall load factor γ_F. The overall load factor takes account of the unfavourable deviation of loads from their nominal values and the reduced probability, that various loads will all be at their nominal value simultaneously.

The uncertainty of the material is taken account by the partial safety factor γ_m. The design strength of the material is taken account by

$$f_d = f_y / \gamma_m \tag{2.1}$$

The strength resistance R of the detail is calculated using the design strength f_d. The structure is supportable if the strength resistance is greater than the design load F (factored load).

$$R(\gamma_m, x) > F(\gamma_F, x) \tag{2.2}$$

2.3 Optimisation

The aim of structural optimisation is always the minimisation or maximisation of a defined objective function, e.g., cost of materials and labour, structural weight, or storage capacity. Problems of structural optimisation may be generally classified as sizing, shape or layout optimisation. Sizing optimisation relates to the cross-sectional dimensions of one- or two-dimensional structures. The cross-sectional geometry is partially prescribed so that the cross-section can be fully described by a finite number of variables. Geometric shape optimisation refers to the shape of the centroidal axis of bars and the middle surface of shells. It also includes boundaries of continua or interfaces between different materials in composites. Layout optimisation consists of three simultaneous operations: 1) topological optimisation, i.e., the spatial sequence or configuration of members and joints, 2) previously mentioned geometrical shape optimisation, and 3) optimisation of the cross-sections. (Rozvany, G. I. N. 1992).

Optimisation is the act of obtaining the best result under a given set of circumstances or restraints. The ultimate goal is to minimise the effort required or maximise the desired benefit. Optimisation can be defined as the process of finding the conditions that give the maximum or minimum value of a function. The minimised or maximised function is termed an objective function. Point x^* corresponds to the minimum value of function $f(x)$ in Figure 2.2. The identical point x^* corresponds to the maximum value of the function $-f(x)$ (Rao, S. S. 1978).

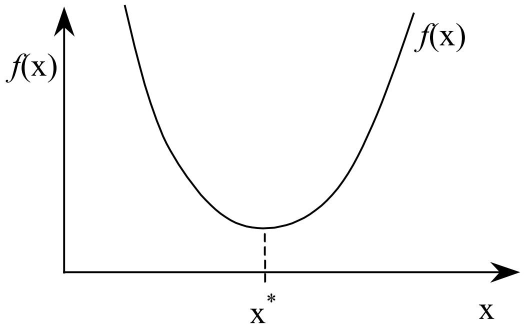


Figure 2.2 The function $f(x)$ and the optimum point x*.

During an optimisation procedure, a search is made for the objective function that satisfies the inequality and equality constraints as follows:

$$\begin{array}{ll} \min f(\mathbf{x}) & \mathbf{x} = (x_1,...,x_i,...,x_n) \\ g_j(\mathbf{x}) \geq 0 & j = 1,2,...,m \\ h_j(\mathbf{x}) = 0 & j = m+1,...,q \end{array} \tag{2.3}$$

where n is number of unknowns and q is the number of constraints. The functions may be continuous or the unknowns may be defined by series of discrete values. Typically it is requited that the variables are positive ($x_\mathrm{i} \geq 0$), or that their upper and lower limit may be prescribed with box limits

$$x_i^\mathrm{l} \leq x_i \leq x_i^\mathrm{u} \tag{2.4}$$

The functions f, g, h may be linear or non-linear. In structural synthesis problems the number of constraints is characteristically larger than that of variables ($q \geq n$).

2.3.1 Discrete and continuous variables

In optimisation problems, functions can be continuous or discrete. Discrete variables are, e.g., the thickness, width and height of a fabricated hollow section while the cut length is usually a continuous variable. Discrete variables may also be connected to other variable or variables. For example, material cost is usually dependent on material strength and quality.

Integer programming methods are usually slow and unreliable. One practical approach is to initially treat the discrete variables as continuous. After the optimum solutions are found, the continuous values can be rounded to nearest acceptable discrete value. There is a risk, however, that the rounding procedure moves the solution away from optimum or moves to an infeasible solution. It is, therefore, necessary to check values against to the constraints.

Each design variable may be regarded as one dimension in a design space. In cases with two variables, the design space reduces to a planar problem. In the general case of n variables, an n-dimensional hyperspace is required.

The optimal design problem can be expressed in the following form:

$$\begin{array}{lll} \text{Minimize} & f(\mathbf{x}_\mathrm{C},\mathbf{x}_\mathrm{D}) & \\ \text{Subject to} & g_i(\mathbf{x}_\mathrm{C},\mathbf{x}_\mathrm{D}) \leq 0 & i = 1,...,m \\ & x_j^\mathrm{l} \leq x_{\mathrm{C}j} \leq x_j^\mathrm{u} & j = 1,...,n_\mathrm{C} \\ & x_{\mathrm{D}k} \in D_k, \quad D_k = \left(d_{k,1},d_{k,2},...,d_{k,n_{\mathrm{p}k}}\right) & k = 1,...,n_\mathrm{D} \end{array} \tag{2.5}$$

where f and g_i are objective and constraint functions, respectively. Components of the mixed variable vector $\mathbf{x}$ are divided into n_C continuous variables expressed as $\mathbf{x}_\mathrm{C} \in \mathbb{R}$, where $\mathbf{x}^\mathrm{l}$ and $\mathbf{x}^\mathrm{u}$ are lower and upper limits, and n_D discrete variables, expressed as $\mathbf{x}_\mathrm{D}$. D_k is the set of discrete values for the k-th discrete variable. The set D_k consists of $n_{\mathrm{p}k}$ discrete parameters. Values for these parameters are, for example, selected from a table of standard sizes. Values corresponding to these parameters depend directly on the choice of one of the discrete variables $x_{\mathrm{D}k}$ ($k \in [1, n_\mathrm{D}]$). For example if a certain beam cross-section is chosen as one of the discrete n_D parameters, beam values like section modulus and area are fixed. The derivatives $\partial f/\partial x_{\mathrm{D}k}$ and $\partial g_i/\partial x_{\mathrm{D}k}$ ($i = 1,...,m$) cannot be computed (Giraud-Moreau, L. et al. 2002).

2.3.2 Objective Functions

Optimisation means minimisation or maximisation of the real value objective function f:$\mathbb{R}^n \rightarrow \mathbb{R}$ over the vector space $\mathbb{R}^n$. The goal is to obtain a minimum or maximum value for $f(\mathbf{x})$, when $\mathbf{x} \in \mathcal{F} \subset \mathbb{R}^n$. The objective function should usually be formulated in such a way that it closely describes the optimisation goal. In structural engineering problems weight or total cost are usually chosen. In practical applications, one objective function rarely represents the only measure of the performance of a structure. Objective function of the problem $f(\mathbf{x})$ or constraint functions $\mathbf{g}(\mathbf{x})$: $\mathbb{R}^n \rightarrow \mathbb{R}^p$ and $\mathbf{h}(\mathbf{x})$: $\mathbb{R}^n \rightarrow \mathbb{R}^q$ can be non-linear. Non-linear in this sense means that a valid function does not exist such that $f(\mathbf{x} + \mathbf{y}) = f(\mathbf{x}) + f(\mathbf{y})$ for all $\mathbf{x}$, $\mathbf{y}$ or such that $f(\alpha\mathbf{x}) = \alpha f(\mathbf{x})$ for all $\mathbf{x}$. For non-linear optimisation problems, the logarithm and exponent functions often cause severe scaling problems because small differences in values for some variable can cause large changes in objective functions (Haataja, J. 1995).

2.3.3 Constraints

The objective function can be computed over an entire vector space; however, some solutions to the function are not feasible for technical reasons. Constraints are often associated with the violation of some physical law. The set of all feasible designs forms the feasible region $\mathcal{F}$ or the set of all points which satisfy the constraints constitutes the feasible domain of $f(x)$. Boundary points satisfy the equation $g_j(x) = 0$. Interior points satisfy the equation $g_j(x) < 0$. Exterior points satisfy the equation $g_j(x) > 0$. An example of an inequality constraint is presented in the Figure 2.3.

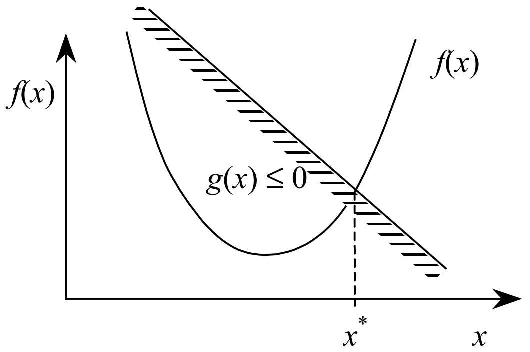


Figure 2.3 Function $f(x)$ with inequality constraint $g(x)$.

Several methods have been proposed for handling constraints. These methods can be grouped into five categories: 1) methods preserving the feasibility of solutions, 2) methods based on penalty functions, 3) methods that make a clear distinction between feasible and infeasible solutions, 4) methods based on decoders and 5) hybrid methods. Three methods for handling constraints are presented in this thesis, one based on penalty functions, one method on search for a feasible solutions and a new hybrid method presented by Lampinen (Lampinen, J. 2002).

2.3.3.1 Methods based on penalty functions

The most common approach for handling constraints, especially inequality constraints, is to use penalties. The basic approach is to define the fitness value of an individual *i* by extending the domain of the objective function *f* using

$$f_i(\mathbf{x}) = f(\mathbf{x})_i \pm Q_i \tag{2.6}$$

where Q_i represents either a penalty for an infeasible corresponding variable *i*, or the cost for repairing such a variable, i.e., the cost of making it feasible. It is assumed that if variable *i* is feasible, then Q_i

= 0. There are at least three main choices to define a relationship between an infeasible individual and the feasible region of the search space (Coello, C. A. 1998, Michalewicz, Z et al. 1999):

1) an individual might be penalised just for being infeasible, i.e., no information is used about how close it is to the feasible region;

2) the degree of infeasibility can be measured and used to determine its corresponding penalty; or

3) the effort of repairing the individual, i.e., the cost of making it feasible, might be taken into account.

The following guidelines related to the design of the penalty functions have been derived (Coello, C. A. 1998, Michalewicz et al. 1999):

1) Penalties that are functions of the distance from feasibility perform better than those, which are merely functions of the number of violated constraints.

2) For a problem having few constraints and few full solutions, penalties that are solely functions of the number of violated constraints are not likely to find solutions.

3) Successful penalty functions are constructed from two quantities: the maximum completion cost and the expected completion cost. The completion cost is defined as the cost of making an infeasible solution feasible.

4) Penalties should be close to not less than the expected completion cost. The more accurate the penalty, the better will be the final solution. When a penalty often underestimates the completion cost, a search may not yield a solution.

Usually, the penalty function is based on the distance of a solution from the feasible region, $\mathscr{F}$. A set of functions f_j ($1 \leq j \leq m$) is used to construct the penalty, where the function f_j measures the violation of the j-th constraint as follows:

$$f_j(x) = \begin{cases} \max\left(0, g_j(\mathbf{x})\right) & \text{if} \quad 1 \leq j \leq m \\ \left|h_j(\mathbf{x})\right| & \text{if} \quad m+1 \leq j \leq q \end{cases} \tag{2.7}$$

Dynamic penalty techniques also exist in which penalties change over time. Individuals are evaluated at generation using:

$$c(\mathbf{x}) = f(\mathbf{x}) + (C \times G)^{\alpha} \sum_{j=1}^{m} f_j^{\beta}(\mathbf{x}) \tag{2.8}$$

where C, α and β are constants defined by the user and m is the number of inequality constraints. This dynamic function progressively increases the penalty from one search generation to the next. In this case, the quality of the discovered solution is very sensitive to changes in the values of the parameters.

Adaptive penalty functions are constructed so that one component receives a feedback from the search process. Feedback for the penalty function is constructed as:

$$c(\mathbf{x}) = f(\mathbf{x}) + \lambda(G) \sum_{j=1}^{m} f_j^2(\mathbf{x}) \tag{2.9}$$

The function $\lambda(G)$ in the above expression is updated every search generation G as:

$$\lambda(G+1)=\begin{cases}(1/\beta_1)\cdot\lambda(G) & \text{if case 1}\\ \beta_2\cdot\lambda(G) & \text{if case 2}\\ \lambda(G) & \text{otherwise}\end{cases} \tag{2.10}$$

where cases 1 and 2 denote situations for which the best individual in the last generation was always feasible (case 1) or was never feasible (case 2). Parameters β_1 , $\beta_2 \geq 1$, and $\beta_1 \neq \beta_2$ to avoid cycling. The penalty component $\lambda(G+1)$ for the generation $G+1$ is decreased if all best individuals in the last generation were feasible or is increased if they were all infeasible. The drawback of this dynamic penalty approach is how to choose the generational gap and how to define the values of β_1 and β_2. (Coello, C. A. 1998).

2.3.3.2 Methods based on a search for feasible solutions

There are few methods that emphasise the distinction between feasible and infeasible solutions in the search space. In one method, each individual is evaluated by the formula:

$$c(\mathbf{x})=f(\mathbf{x})+r\sum_{j=1}^{m}f_j(\mathbf{x})+\theta(G,\mathbf{x}) \tag{2.11}$$

where r is a constant. The original component $\theta(G, \mathbf{x})$ is an additional iteration dependent function, which influences the evaluations of infeasible solutions. A modification to this approach is implemented with the tournament selection operator and with the following evaluation function:

$$c(\mathbf{x})=\begin{cases}f(\mathbf{x}), & \text{if } \mathbf{x} \text{ is feasible}\\ f_{\max}+\sum_{j=1}^{m}f_j(\mathbf{x}), & \text{otherwise}\end{cases} \tag{2.12}$$

where $f_{\max}$ is the function value of the worst feasible solution in the population. An objective function is not considered in the evaluation an infeasible solution. There is no need for the penalty coefficient r here, because the feasible solutions are always evaluated to be better than infeasible solutions and infeasible solutions are compared purely based on their constraint violations (Michalewicz, Z. et al. 1999).

The technique is expected to behave well if the assumption of the superiority of feasible solutions over infeasible ones holds. The technique will fail in cases where the ratio between the feasible region and the whole search space is too small, unless a feasible point is introduced in the initial population.

2.3.3.3 Methods without penalties

It is also possible to work with constraints without the aid of penalty functions. For the sub-population guided by the objective function, the evaluation of such a function for a given vector **x** is used directly as the fitness function, with no penalties of any sort. For all the other sub-populations, the algorithm used was the following:

$$\begin{aligned}&\text{if } g_j(\mathbf{x})<0.0 \text{ then } c(\mathbf{x})=g_j(\mathbf{x})\\ &\text{else if } v\neq 0 \text{ then } c(\mathbf{x})=-v\\ &\qquad\qquad\qquad\quad c(\mathbf{x})=f(\mathbf{x})\end{aligned} \tag{2.13}$$

where $g_j(\mathbf{x})$ refers to the constraint corresponding to sub-population $j+1$, and v refers to the number of constraints that are violated. Each sub-population associated with a constraint will try to reduce the amount by which that constraint is violated. If the evaluated solution is infeasible but does not violate the constraint corresponding to that sub-population, then the sub-population will try to minimise the total number of violations. This in turn influences other sub-populations in the effort of driving the genetic algorithms to the feasible region. This approach aims at combining the distance from feasibility with information about the number of violated constraints, which is the same heuristic normally used with penalty functions.

Because the number of constraints will normally be greater than one, the other sub-populations will drive the GA toward the feasible region. In fact, the sub-population evaluated with the objective function will be useful to keep diversity in the population and will render the use of sharing techniques unnecessary. The behaviour expected under this scheme is that, even in the event that there are only few feasible individuals at the beginning, gradually more solutions will be produced that are feasible with respect to some constraints. (Coello, C. A. 2000)

2.3.3.4 Hybrid method

Lampinen has introduced a new method for constraint handling (Lampinen, J. 2002). The proposed modification for the differential evolution DE algorithm's selection rule is mathematically expressed as follows:

$$x_{i,G+1} = \begin{cases} u_{i,G+1} & \text{if} \quad \begin{cases} \begin{cases} \forall j \in \{1,\ldots,m\} : g_j(u_{i,G+1}) \le 0 \wedge g_j(x_{i,G}) \le 0 \\ \wedge \\ f(u_{i,G+1}) \le f(x_{i,G}) \end{cases} \\ \vee \\ \begin{cases} \forall j \in \{1,\ldots,m\} : g_j(u_{i,G+1}) \le 0 \\ \wedge \\ \exists j \in \{1,\ldots,m\} : g_j(x_{i,G}) > 0 \end{cases} \\ \vee \\ \begin{cases} \exists j \in \{1,\ldots,m\} : g_j(u_{i,G+1}) > 0 \\ \wedge \\ \forall j \in \{1,\ldots,m\} : \max\left(g_j\left(u_{i,G+1}\right),0\right) \le \max\left(g_j\left(u_{i,G}\right),0\right) \end{cases} \end{cases} \\ x_{i,G} & \text{otherwise} \end{cases} \tag{2.14}$$

Thus, when compared with the corresponding member, $x_{i,G}$, of the current population, the trial vector, $u_{i,G+1}$, will be selected if any one of the following three conditions are satisfied:

1. It satisfies all constraints and provides a lower or equal objective function value. In this case both of the compared solutions are feasible, or
2. It is feasible while $x_{i,G}$ is infeasible, or
3. It is infeasible, but provides a lower or equal value for all constraint functions

In the case of an infeasible solution, the selection rule does not compare the objective function values. No selective pressure exists towards the search space regions providing low objective values combined with infeasible solutions. However, a selective pressure towards regions where constraint violation decreases does generally exist. For this reason an effective selection pressure will be applied for finding the first feasible solution. The result is fast convergence toward the feasible regions of the search space.

In this algorithm, if both the compared solutions are feasible, the one with lower objective function value is selected as being better. A feasible solution is considered better than infeasible one. If both the compared solutions are infeasible, the situation is less obvious. The candidate vector can be considered less infeasible, and thus better than the current vector, if it does not violate any of the constraints to a degree greater than the current vector or if it violates at least one fewer of the constraints.

When the candidate vector can be considered equally as good as the compared current population member, it is allowed to continue into the new population. This rule helps avoid the stagnation phenomena (Lampinen, J. 2000).

2.3.4 Local and global optimisation

An optimisation problem involves searching for a local minimum point for the problem given by, $\mathbf{x}^* \in \mathscr{F} \subset \mathbb{R}^n$, where

$$f(\mathbf{x}^*) \leq f(\mathbf{x}) \quad \text{with all} \quad \mathbf{x} \in \mathscr{F} \subset \mathbb{R}^n \text{ so that } \left\|\mathbf{x}^* - \mathbf{x}\right\| < \varepsilon, \quad \varepsilon > 0. \tag{2.15}$$

is valid. However, if the objective function is non-convex there may be numerous local minima and global optimisation methods will need to be used to ensure that the solution represents the absolute minimum of the objective function that is feasible for the given set of constraints.

Figure 2.4 shows a function $f(x)$ that has various extreme values. Local minima of the function exist at points x_1 and x_4 while point x_2 represents a local maximum. At point x_3 the first derivative is zero but a sign change does not occur so no local maximum or minimum exists. This point is an inflection point because the second derivative changes sign. Point x_6 represents the global minimum while point x_7 represents the global maximum.

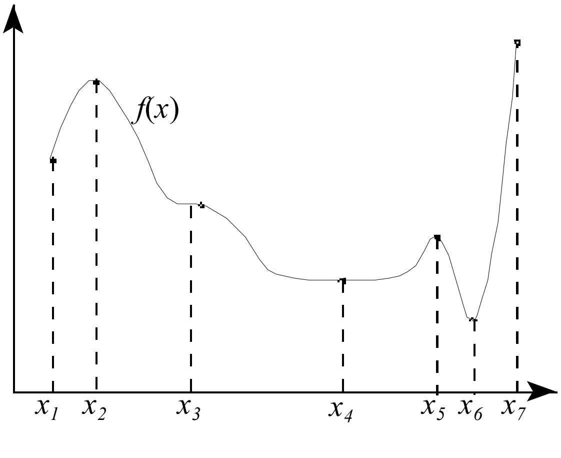


Figure 2.4 Function $f(x)$.

Global optimisation methods may deterministic or heuristic. Some methods use deterministic methods in a part of the heuristic method. An optimisation problem is deterministic if values of the objective functions are known exactly and they can be summarised.

2.3.5 Multi-criteria optimisation

Multi-criteria optimisation goes back as far as the work of Parades in 1898. Interest in the field of optimisation theory increased dramatically in the late 1960s. Since then, many studies have been published on multi-criteria optimisation. Most of these deal with the theory of decision making from a general point of view. A few publications can be found in the field of optimum engineering design (Eschenauer, H. et al. 1991, Farkas, J. et al. 1997).

A multi-criteria optimisation problem can be formulated as follows: Find **x** such that

$$f(\mathbf{x}^*) = \text{opt}\, f(\mathbf{x}) \tag{2.16}$$

such that

$$\begin{aligned} g_j(\mathbf{x}) &\geq 0 \quad j = 1,\dots,m \\ h_i(\mathbf{x}) &= 0 \quad i = m+1,\dots,q \end{aligned} \tag{2.17}$$

solutions of this problem are termed Pareto optima. Different methods for generating Pareto optimal solutions to some multi-criterion optimisation problems have been developed, e.g., linear weighting methods, distance methods, constrained methods and hybrid methods.

Linear weighting methods define a weighted objective function which is the sum formed by linearly scaling all the objective criteria. If the weighting coefficients are denoted by w_i, $i = 1,2,\dots,n_\mathrm{f}$, this scalar optimisation problem takes the form (Farkas, J. 1997, Haataja, J. 1995)

$$\min \sum_{i=I}^{n_\mathrm{f}} f(\mathbf{x}) = \min \sum_{i=1}^{n_\mathrm{f}} w_i f_\mathrm{i}(\mathbf{x}) \quad \text{where} \quad w_i \geq 0 \;\; \text{and} \;\; \sum_{i=1}^{n_\mathrm{f}} w_i = 1 \tag{2.18}$$

It is possible to determine the Pareto optimum for a problem by varying these weight coefficients. The effect of weighting coefficients on the weighted objective is slight if the values of the individual component objective functions differ greatly. Normalising Eq. 2.19 with virtual ideal-solution f_i^0 can avoid this problem. Normalised weighting better reflects the importance of the weighting coefficient w_i than does non-normalised weighting

$$\min \sum_{i=I}^{n_\mathrm{f}} f(\mathbf{x}) = \min \sum_{i=1}^{n_\mathrm{f}} \frac{w_i f_i(\mathbf{x})}{f_i^0} \quad \text{where} \quad w_i \geq 0 \;\; \text{and} \;\; \sum_{i=1}^{n_\mathrm{f}} w_i = 0 \quad \text{and} \quad f_i^0 \neq 0 \tag{2.19}$$

Distance methods can also be used to generate the Pareto optimal solution. These methods are based on the minimisation of the distance d_p between the attainable set and some chosen reference point f_i^0 in the criterion space. The method is expressed as

$$\min \; d_\mathrm{p}(\mathbf{x}) \tag{2.20}$$

with the common form of distance function written as

$$d_\mathrm{p}(\mathbf{x}) = \sum_{i=1}^{n_\mathrm{f}} \left[\frac{f_i^0 - f(\mathbf{x})}{f_i^0} \right]^p \quad when \;\; p = 1,2,3,\dots \tag{2.21}$$

The solution differs greatly depending to the chosen value of p. Deviations in the absolute sense are as follows:

$$d_\mathrm{p}(\mathbf{x}) = \left[\sum_{i=1}^{n_\mathrm{f}} |f_i^0 - f_i(\mathbf{x})|^p \right]^{1/p} \quad \text{when} \;\; 1 \leq p \leq \infty \tag{2.22}$$

$$d_{\mathrm{p}}(\mathbf{x}) = \left[\sum_{i=1}^{n_{\mathrm{f}}} \left| \frac{f_i^0 - f_i(\mathbf{x})}{f_i^0} \right|^p \right]^{1/p} \quad \text{when } 1 \le p \le \infty \tag{2.23}$$

When $p = 1$ the equation is Euclidean metric and with $p = \infty$, Chebysev metric. Jármai has presented the use of relative deviation (Jármai, K. 1989):

$$d_{\mathrm{p}}(\mathbf{x}) = \left[\sum_{i=1}^{n_{\mathrm{f}}} w_i \left| \frac{f_i^0 - f_i(\mathbf{x})}{f_i^0} \right|^p \right]^{1/p} \quad \text{when } 1 \le p \le \infty \tag{2.24}$$

One possibility is to replace the original multi-criterion problem with a scalar problem where one criterion is chosen as the objective function and all the other criteria are moved into the constraints.

2.3.6 Stopping conditions

In order to determine when an optimal or at least a sufficiently good solution has been achieved, a stopping condition must be introduced. A stopping condition is based on the assumption that the probability of improving optimality can not be increased without an undue amount of additional effort. The possibility to raise optimality is low in the case of evolution algorithms if individuals of the population are almost identical. An optimisation problem should be solved several times to ensure the optimality.

2.3.7 Optimality

The design of complex, real world systems involves a search for “just the right” combination of components that achieves the property known as “optimality”. Yet, optimality in the conventional sense usually ignores a vital measure of system performance: stability. The designs that are optimal in the conventional sense are often particularly vulnerable to the phenomenon of instability. Figure 2.5 presents a non-linear function of a single variable. The possible values of x range from 0.0 to 1.0. The function has two main peaks. The highest is centred about 0.2 while another, somewhat lower but much broader peak is centred at about 0.7. A designer must perform a sensitivity analysis to determine the suitability of the optimal area. In most cases, a slightly lower but broader peak is more optimal than a mathematically optimal peak which is high but narrow (Ignizio, J. 1999). Production tolerances or restrictions may result that a mathematically optimum solution is useless from an engineering point of view.

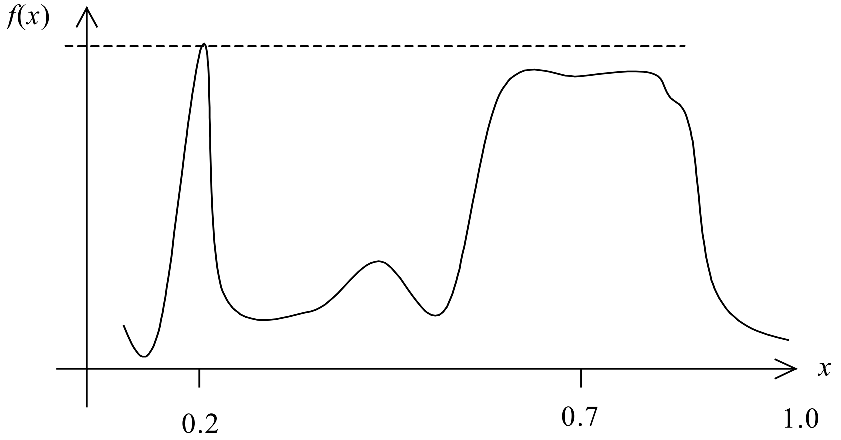


Figure 2.5 The illusion of the optimality. The lower and broader peak is more optimal than the narrower but higher peak in the practical matter.

2.3.8 Optimisation

In mechanical engineering work, many variables are discrete and functions can be discontinuous. These features of a mechanical design problem limit the choice of the optimisation method. Random search algorithms are usually suitable when variables are discrete and functions are discontinuous. Random search algorithms do not ensure that the optimal point found is the exact solution, but many authors have demonstrated that the convergence probability approaches unity as the number of iterations becomes large, $G \rightarrow \infty$ (Viitanen, S. 1997). Although these algorithms need numerous iterations to converge, they are suitable for many kinds of optimisation problems.

Efficiency of an algorithm depends heavily on the optimisation problem. Random search methods are less sensitive to the characteristics of the problem than are other optimisation algorithms. However, the use of random search algorithms in mechanical engineering work is very limited. This is largely the result that these algorithms are unknown and their benefit has been demonstrated in only a few special cases. In these cases the problem and solving algorithm are usually modified for the case. This arises the potential that random search algorithms could be much more widely employed in day-to-day engineering work.

Optimisation methods in engineering applications have the potential to increase efficiency, economy and reliability and thus the overall competitiveness of an engineering construction. Correct material and component selection result in lower product costs. Simultaneously the resistance of a structure to higher loads is possible.

2.3.9 Evolutionary algorithms

Evolutionary computation techniques are stochastic optimisation methods, which are conveniently presented using the metaphor of natural evolution: a randomly initialised population of individuals evolves following the Darwinian principle of the survival of the fittest. The probability of survival of the newly generated solutions depends on their fitness. Fitness or cost describes how well a solution performs with respect to the optimisation problem at hand. Better solutions are kept with a high degree of probability while the worst are rapidly discarded. One of the main advantages of evolutionary computation techniques is that they do not impose severe mathematical requirements for the optimisation problem. They require only an evaluation of the objective function. They can handle non-linear problems defined on discrete, continuous or mixed search spaces. These may be unconstrained or constrained. Evolutionary algorithms are naturally global optimisation algorithms (Michalewicz, Z. et al. 1999).

Genetic algorithms are useful when the variable space is large. Optimised functions can be discontinuous, or variables can be discrete.

Differential Evolution DE is a simple yet powerful population-based, direct-search algorithm for globally optimising functions defined on totally ordered spaces, including especially functions with real-valued parameters (Price, K. V. 1999).

2.3.9.1 Genetic algorithms

Genetic algorithms have been used to solve structural optimisation problems. They apply the principle of survival of the fittest into the design of structures. They also have the ability to deal with discrete optimum design problems and do not require functions to be derivated as in classical optimisation. The genetic algorithm was originally proposed by Holland in 1975.

Genetic algorithms are based on an artificial model of the evolution process in nature. A genetic algorithm maintains a population of alternative solutions for the optimisation problem to be solved. Alternative solutions are individuals of the population. News generations of solutions will be created about individuals using a specific reproduction scheme. The dominating principle is that the best individual solution of the population has the best chances to survive to the next generations of populations. The better or "fittest" is considered to be that with the lowest cost-function. The cost-function is assigned to each individual with respect to the specified design targets and design constraints for the optimisation problem. An individual with a low cost-function value has a higher probability of surviving to the next generation than does an individual with a high value. It is termed elitism if some of individuals are transferred directly to the next generation.

Less "fit" individuals do not survive to the next generation. They will be replaced by the re-combinations (or crossovers) of better individuals. Combining randomly chosen parts of the better individuals creates a new individual. Two good solutions have a high probability of combining the best properties of each. The child is better than its parents. Only the solutions that are possible to recombine with existing individuals are permitted. Mutations are created in the population at random intervals. A better individual will replace a weak mutation but, at the same time, a new solution has been created that was not possible based on the recombination of existing individuals.

One of the characteristics of genetic algorithms is that the coding must be done based on the parameter *set* rather than on the parameters themselves. A second characteristic is that a search is made from a population of points rather than a single point. A third characteristic is that GA uses objective function information, not derivatives or other auxiliary knowledge. The final characteristic is that it uses probabilistic transition rules, not deterministic rules.

Genetic algorithms require the natural parameter set of the optimisation problem to be coded as a finite-length string over some finite alphabet. Solution space vectors can be coded to a genetic code using values zero and one, $\mathbf{x} \in \{0,1\}^n$ or $\mathbf{x} \in \mathbb{R}^n$. The similarity of the code vectors can be illustrated with schemes where the scheme is a string

$$H = (h_1, h_2, ..., h_n), \; h_i \in \{0,1,*\}, \; i = 1,...,n, \tag{2.25}$$

In this equation the symbol * correspond to zero or one. The schema (0,*,1,1,*) correspond to the set of code vectors {(0,0,1,1,0), (0,0,1,1,1), (0,1,1,1,0), (0,1,1,1,1)}. The set of the binary code vectors corresponds to 3^n different schema, where n is a length of the code vector.

GA uses probabilistic transition rules to guide their search. A random choice is used as a tool to guide the search space with likely improvement.

A simple genetic algorithm is composed of three operators: reproduction, crossover and mutation. Reproduction is a process in which individual strings are copied according to their objective function values or fitness values. Copying strings according to their fitness values means that strings with a higher value have a higher probability of contributing one of more offspring in the next generation. The reproduction operator can be implemented in algorithmic form by creating a biased roulette wheel where each current string in the population has a roulette wheel slot sized in proportion to its fitness. Each time a new offspring is required, a "spin of the roulette wheel" yields the reproduction candidate.

After reproduction, simple crossover proceeds in two steps. First, members of the new reproduced strings in the mating pool are mated at random. Second, each pair of strings undergoes crossing over as follows: an integer position k along the string is selected uniformly at random between 1 and the string length less one $[1, n - 1]$. Two new strings are created by swapping all characters between positions $k + 1$ and l inclusively.

Mutation is the occasional random alteration of the value of a string position with small probability. This means changing a 1 to a 0 and vice versa. Mutation is needed because some potentially useful genetic material is lost in spite of the fact that mating only occurs between better solutions.

A characteristic set of individuals or parameter vectors $\{\mathbf{x}^i\}$, $i = 1,..., NP$ are selected for an initial population P_1. After that, two individuals are crossbred. Individuals are selected with a suitable strategy. The descendant is transformed with an occasional mutation. Less successful individuals can be removed from the population. The candidates of the solution are compared with the objective function $f(\boldsymbol{x})$. (Goldberg, D. E. 1989, Haataja, J. 1995)

```
Genetic algorithm

1)  Generate start population $P_1 = \{\mathbf{x}^i\}, i = 1,..., NP$
2)  Set generation $G = 1$
3)  Evaluate $P_G$
4)  If solution satisfies termination condition set $\mathbf{x}^{*i} = \mathbf{x}^i$ and terminate iteration else select best individuals from population $P_G$ to population $P_{G+1}$
5)  Crossover individuals of the population $P_{G+1}$
6)  Generate mutations in population $P_{G+1}$
7)  $G = G + 1$ and go to step 3.
```

2.3.9.2 Differential evolution algorithm

The differential evolution algorithm DE was first introduced by Storn and Price (Storn, R. 1995). It can be categorised into the class of evolutionary optimisation algorithms (EA). It is a simple yet powerful population based, direct-search algorithm for globally optimising functions defined on totally ordered spaces, including, especially, functions with real-valued parameters.

DE generates a randomly distributed initial population $P_{G=0}$ of NP n-dimensional object variable vectors $x_{j,i,G}$. The term $\text{rand}_j[0,1]$ represents a uniformly distributed random variable that ranges from zero to one. A new random variable is generated for each value of j. After initialisation, the population is subjected to repeated generations, $G = 1,2,...,G_{\max}$ of mutation, recombination and selection. DE employs both mutation and recombination to create one trial vector $u_{j,i,G+1}$, for each vector $x_{j,i,G}$. The indices r_1, r_2 and r_3 are randomly chosen population indices that are mutually different and also different from i, which indexes the current object vector. Both CR and F are user-specified control variables. CR represents a probability and ranges from 0 to 1. F is a scaling factor that belongs to the interval (0, 2). An individual of the next generation is created by adding a weighted difference of the corresponding components of two other individuals or parameter vectors according equation

$$x_{i,j,G+1} = \begin{cases} x_{j,r3,G} \ + \ F\ \left(x_{j,r1,G} \text{ - } x_{j,r2,G}\right) & \text{if } \text{rand}_j[1,0] \ \le \ CR \vee j = j_{\text{rand}} \\ x_{j,i,G} & \text{otherwise} \end{cases} \tag{2.26}$$

After each child vector is evaluated according to the objective function, its cost is compared to the cost of its parent. If the child vector has an equal or lower cost to the parent vector, it replaces its parent vector in the population. If the cost of child vector is greater than the cost of its parent, the parent vector is retained.

The selection scheme is elitist because all individuals in next generation $G + 1$ are equal or better than counterparts in the current G population according equation

$$\mathbf{x}_{i,G+1} = \begin{cases} \mathbf{u}_{i,G+1} & \text{if } f\left(\mathbf{u}_{i,G+1}\right) \ \le f\left(\mathbf{x}_{i,G}\right) \\ \mathbf{x}_{i,G} & \text{otherwise} \end{cases} \tag{2.27}$$

The evolutionary cycle in DE repeats until the task is solved, all vectors converge to a point, or no improvement is seen after many generations.

A reason for stagnation of the DE algorithm is that the reproduction operation is capable of providing only a finite number of potential trial solutions. If none of the new solutions is able to replace a member of the current population during the selection operation, the algorithm will stagnate. The probability of stagnation depends on how many different potential trial solutions are available and their capacity for entering the population of the following generation. Thus, it depends on population size *NP*, differential factor *F*, mutation probability *CR,* current population and objective function (Lampinen, J. 2000).

```
Algorithm DE/rand/1/bin

Input:       $D,\ G_{\max}, NP > 4, F \in (0,1+), CR \in [0,1]$,  and initial bounds: $\mathbf{x}^{\mathrm{l}}, \mathbf{x}^{\mathrm{h}}$

Initialize:  $\forall i \le NP \wedge \forall j \le D : x_{j,i,G=0} = x_j^l + rand_j[0,1] \cdot (x_j^{\mathrm{h}} - x_j^{\mathrm{l}})$
             $i = \{1,2,\ldots,NP\}, j = \{1,2,\ldots,D\}, G = 0, \quad rand_j[0,1] \in [0,1]$

While: $G < G_{\max}$
    $\forall\ i \le NP$
        Mutate and recombine:
        $r_1, r_2\ r_3 \ \in \ \{1,2,...,D\}$, randomly selected, exept: $r_1 \ne r_2 \ne r_3 \ne i$
        $j_{rand} \in \{1,2,...,NP\}$,   randomly selected once each $i$
        $\forall j \le D, u_{j,i,G+1} = \begin{cases} x_{j,r_3,G} + F \cdot (x_{j,r_1,G} - x_{j,r_2,G}) & \text{if } (rand_j[0,1) < CR \vee j = j_{rand}) \\ x_{j,i,G} & \text{otherwise} \end{cases}$
        Select:  $\mathbf{x}_{i,G+1} = \begin{cases} \mathbf{u}_{i,G+1} & \textit{if } \ f(\mathbf{u}_{i,G+1}) \le f(\mathbf{x}_{i,G}) \\ \mathbf{x}_{i,G} & \text{otherwice} \end{cases}$   can be replaced eqn. (2.14)
    $G = G + 1$   new generation
```

2.4 Parallel processing

Computational time and cost is one of the major limitations when applying evolutionary optimisation algorithms with computationally costly objective functions. Evolutionary algorithms require a relatively high number of objective function evaluations during the evolutionary process and parallel computation is often essential (Lampinen, J. 1999b).

The parallel computing with optimisation with evolution algorithm is researched widely. In this thesis, the possibilities of the parallel computing are presented briefly to demonstrate how the higher efficiency of the optimisation can be achieved when the capacity of one personal computer (PC) is not adequate.

Evolutionary algorithms are well suited for parallel computational problem solving techniques. Their parallel implementation is generally considered to be straightforward. It has been shown, that the parallel computation of DE-algorithm will be both easy to implement and the efficiency of the distributed computations will be near 100 percent (Lampinen, J. 1999b).

The advantages of the distributed DE algorithm are effectiveness, efficiency, robustness and flexibility. The reduction in computation time is approximately linear with respect to added processor capacity. A distributed DE algorithm does not require any disadvantageous modifications to the differential evolution algorithm. Both steady-state and generational reproduction of individuals can be used

In terms of efficiency, the computational overhead due to distribution is low and load balancing is good. Usage of a heterogeneous network of computers does not degrade the performance. High variance of the evaluation times of the objective function does not degrade the performance. Waiting time due to file access collisions, with a high number of slave processes is a trivial problem that can be solved by using two or more pairs of shared communication interface files.

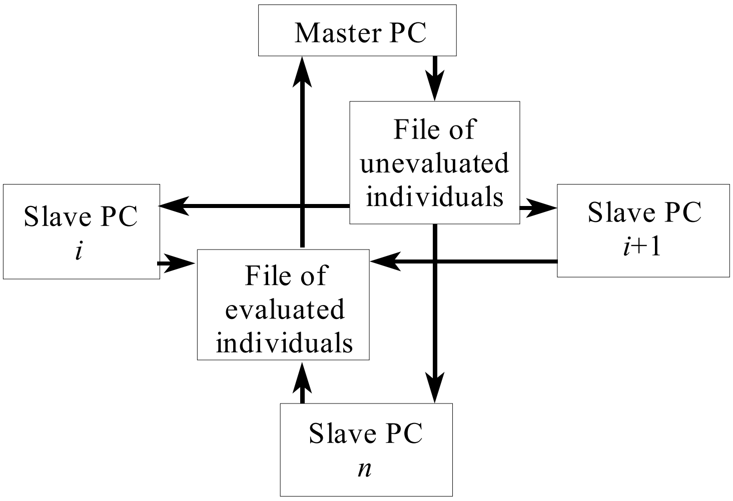

Master PC
File of unevaluated individuals
Slave PC i
Slave PC i+1
File of evaluated individuals
Slave PC n

Figure 2.6 Example about distributed optimisation (Lampinen, J. 1999b).

The computations can be considered robust in that a software or hardware failure of a slave process does not crash the whole system. Two or more master processes can be used to make the system resistant against master process failures. Slave processes can be added or removed during the optimisation process, thus improving flexibility. The number of slave processes is not limited by the population size and more than one master process can be used. DE may run in background or with a low priority in multitasking environments. Heterogeneous cluster of PCs in Ethernet can be used.

There are a few problems associated with the distributed DE algorithm. The distributed DE algorithm is neither yet well known nor thoroughly proven. It can results in some more traffic on a network. Updating slave processes is laborious if a high number of workstations are used for computation. Distributed DE algorithm is suitable only in case of a computationally expensive objective function.

2.5 Summary

This chapter has provided a brief technical background for finite element, limit state design, optimisation algorithms and computational techniques using in the current research project.

The Differential Evolution optimisation algorithm has been selected as the optimisation algorithm because it is a powerful direct-search algorithm. A Genetic Algorithm is a binary coded while DE is a floating point coded algorithm. The optimisation algorithm used in this thesis, can handle both discrete and continuous variables.

A penalty based constraint-handling method was tested in the first modelling and optimisation program generation. The final program version included only the new constraint handling method without penalties. This has reduced the modelling time.

Six multi-criterion optimisation methods were available and tested in the first modelling and optimisation program generation. The user has to select the criterion and corresponding weighting coefficients and ideal values.

The DE is a probabilistic optimisation algorithm. The property values of the individuals are monitored. The user makes the decision if the optimisation is stopped after a specified number of generations or if the deviation of the properties of the individuals gets lower than a pre-set deviation value.

The design of complex, real world systems involves a search for “just the right” combination of components that achieves the property known as “optimality”. In most cases, a slightly lower but broader peak is more optimal than a mathematically optimal peak which is high but narrow. In this thesis, the stability of the optimal solution with real-life engineering cases has not considered.

Evolution algorithms are naturally parallel optimisation algorithms. The distributed optimisation has been demonstrated with time consuming objective function, which included heavy virtual prototype simulation. An automated building of the optimisation model was included into the FE-modelling program.

3 DEVELOPMENT OF A MODELLING AND OPTIMISATION TOOL

3.1 General

As discussed in Chapter 1 of this thesis, the main goal of the research project has been to investigate the possibilities for integrating automated modelling techniques, FE analysis, and an optimisation tool. The research problem has been the formulation and solving of a model for steel truss structures using automated optimisation. In order to accomplish this goal, it was necessary to develop three different versions of the optimisation program. This chapter describes some important innovations made from one program generation to the next.

The object-oriented programming (OOP) technique has been shown to significantly improve the extendibility and reusability of software. It also enables and encourages modular design of software so that the modules or components can be reused for multiple purposes. A carefully designed framework or architecture can significantly reduce the effort required for maintaining and extending the software (Gorlen, K. E. et al. 1990). The FE-program is inherently modular and is particularly amenable to OOP (Yu, L. et al. 2001). For these reasons OOP techniques were employed during program development.

The modelling tool developed in this research study was linked to an easy-to-use FEA-program developed at Lappeenranta University of Technology. The FE-program, called AGIFAP, is a finite element analysis package for three dimensional frame type structures. Two-node beam type elements are used in the AGIFAP program. Each element has 14 degrees of freedom: 6 translation, 6 rotation and 2 warping degree of freedom (AGIFAP Version 5.51, User guide 1995). The coordinate system and positive forces and moments of the element are presented in Figure 3.1. The interface and the data handling of the AGIFAP program have been developed during this project so as to reflect object oriented programming. Only linear finite element modelling and solving has been used in this study.

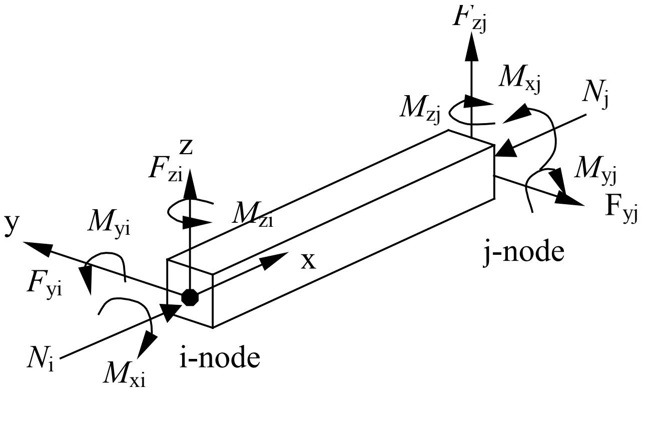


Figure 3.1 Forces of the beam element of the AGIFAP finite element program.

As mentioned in Chapter 2, most current structural design codes used in Europe are based on limit state design concepts. For this reason, design computations are based primarily on the limit state design method. Some exceptions to this rule have been made because machines components, like hydraulic cylinders, are generally designed based on allowable load rather than limit state design concepts. The design program transcends both the structural design and machine design worlds and, therefore, includes the different design philosophies used by these different design spheres. For this reason geometric constraints have not been processed with safety factors.

3.2 First generation program

The first phase, in what turned out to be a three-stage optimisation program development process, was to construct a modelling tool that uses graphical components for the model formulation. This program version was called “Optimaze”. The graphical components were constructed so that the design could graphically select first the type of component desired, and then easily define the parameters of that component. Three different types of components were available to describe a structural model. These included math components, which represent mathematical objective functions or restraints, e.g. buckling resistance of a beam is given as a mathematical relationship between applied loads and beam cross-section properties. Math component formulas can be edited as needed by the user in this program. Table components are also used. This means that properties of the component are determined based on numerical information in table form. A typical table component would be a rectangular hollow section (RHS) beam for which the manufacturer defines discrete beam sizes, thickness, yield strength, etc. A component is selected from the table component during the optimisation.

The final component type is a finite element component. The optimisation model was build by linking the input and output variables of the components. Tables of the components can include both geometric and performance properties of the components. For example, the hydraulic cylinder catalogue includes this type of information. This feature allows the program to automatically select a cylinder from the catalogue during the optimisation.

The FE-components served as interface components between the optimisation model and the FE analysis program. This is illustrated in Figure 3.2. The input and output variables of the FE-model were linked to the optimisation model using these components. For example, in the case of the geometric design of a truss structure, the optimisation algorithm provided the set of instructions to the FE-component, which then formulated a FE-model that could be processed by the FE-analysis software. As the FE analysis program was evaluating the designed structure, the FE component produced a data file for the optimisation program describing the developed model. Results from the FE analysed were then processed by the FE-component before being communicated to the optimisation algorithm.

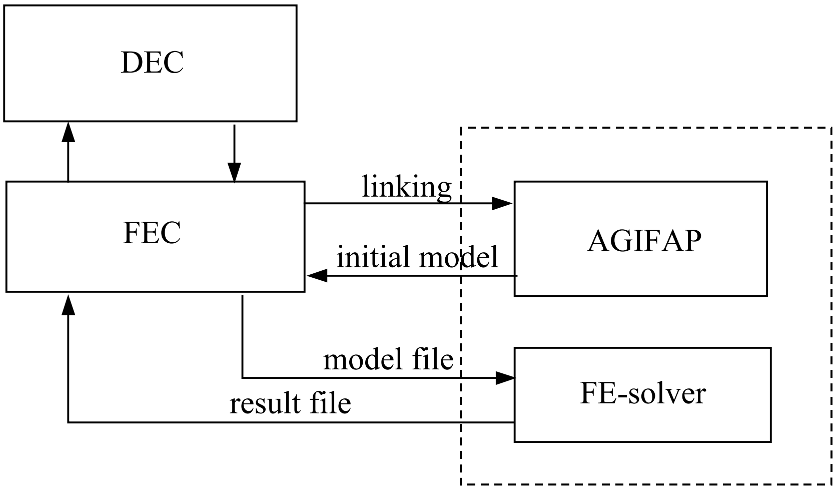


Figure 3.2 Data transfer between DE-component (DEC), FE-component (FEC), optimisation model, AGIFAP and FE-solver.

One special math component in the model is the differential evolution component (DEC). This component contains the optimisation algorithm. The DEC component consists of three parts: 1) the definition of the design variables, 2) the objective functions, and 3) the constraints. Other components are linked to the DEC and other components with special relations. For example, the outputs of certain components will naturally form the inputs of other components. Output from a table component that searches a catalogue for some suitable design element will be linked to the input of a subsequent component. This link transfers the parameters that describe the selected element. This linking between elements is illustrated in Figure 3.3. In this Figure (MC) represents math components, (TC) represents table components and (FEC) is a FE component. The optimisation model, or the group of objectives and constraints, is constructed using components: MC, TC and FEC. Values of the objectives and constraints are transmitted to the corresponding components of the differential evolution algorithm.

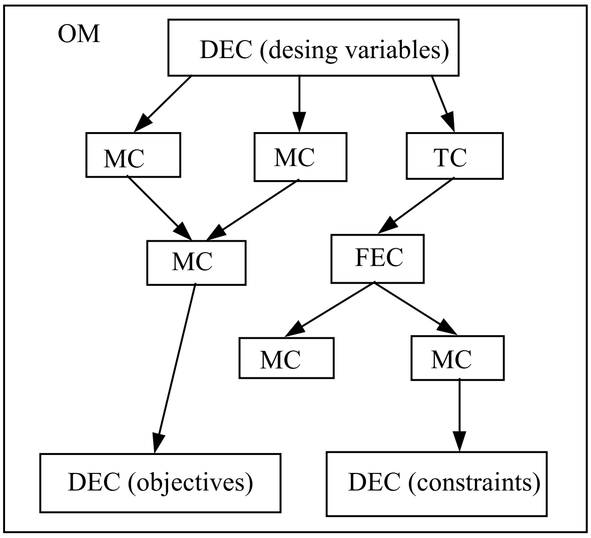


Figure 3.3 An example about the optimisation model (OM).

The input vector of a component may contain part of the design variable vector or the entire vector. Similarly the input may contain all or parts of the constraint vector or other intermediate variable vectors created by other components and the evolution algorithm. The components of the optimisation model and the input and output vectors are illustrated in the Figure 3.4. Inputs for the DEC component contain all constraint, **g**, and objective function, **f**, information. The output is a population of design vectors, **x**. Other components, like math components MC_k do not necessarily have the entire design vector as an input, but instead may require only a portion of that vector, **x'**.

The vectors, **y'**, in this figure represent intermediate pieces of information that are created by some components and utilised by other components. Similarly, a single math components may produce as an output a subset of design constraints, **g'**.

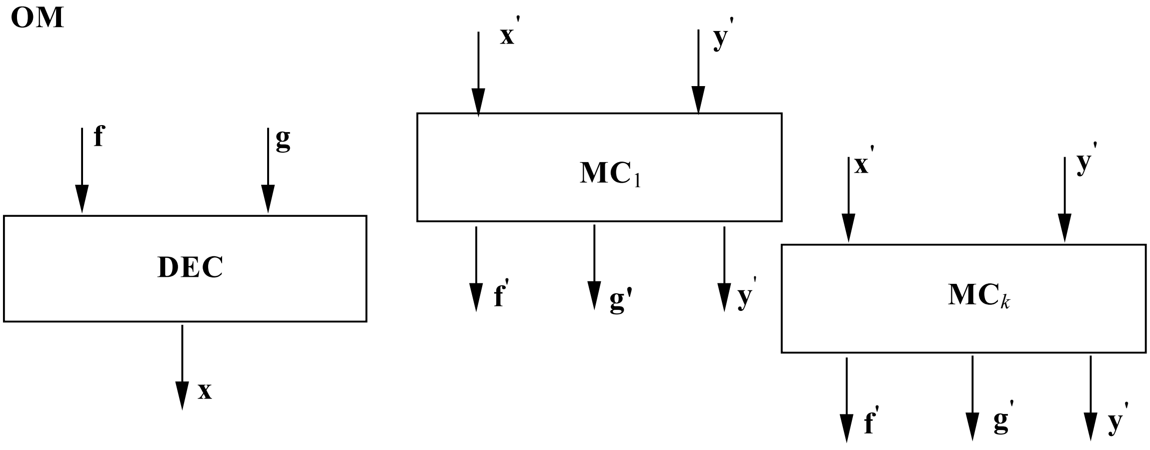


Figure 3.4 The components of the Optimaze-program. The differential evolution component (DEC) and the model components (MC_i) lie on the optimisation model (OM) form.

Meriäinen has used the Optimaze program in the virtual design of the mechatronic machine. The modelling tool includes a special component (ADAMS-component), which makes use of the ADAMS dynamic simulation software. The ADAMS component in this case operated very much like a FE-component. The component first constructed an input-file for the virtual prototype solver. A virtual prototype using ADAMS was made and evaluated for every objective function evaluation. The ADAMS-component was then used to bring the simulation results back into the optimisation program. Combining optimisation and simulation software using internal macros and text files created the software environment.

In this case the optimisation problem was the design of a hydro-mechanical crane. Results of the optimisation process for this example problem were especially promising and indicated that the virtual prototypes can be used in the optimisation of machines with the used software environment (Meriläinen, V. 2000). A sketch of the crane is presented in Figure 3.5. Virtual prototyping can be defined as a software-based engineering discipline that includes modelling a system as well as simulating and post-processing the results. The modelling of a system involves creating a set of equations that define the physics of the system being studied. The simulation consists of the numerical solution of these equations as a function of time. The post-processing of the results refers to the visualisation of three-dimensional behaviour by traditional diagrams or more illustrative animation (Mikkola, A. 1997).

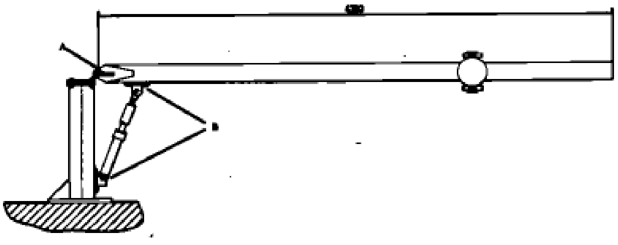


Figure 3.5 A hydromechanical crane (Meriläinen, V. 2000).

3.3 Second generation program

The main innovation between first and second-generation programs was the inclusion of a new open graphic library (OGL) interface for the FE analysis program. In the first version, the graphical user interface was a separate program, but in the second generation the user interface was integrated. The input and output vectors are immediately of use in the optimisation program. This step eliminated some of the relational definitions between variables that were previously required. The main disadvantage in this approach, however, was that the math and other components needed to be pre-compiled.

The structure of the second-generation program is shown in Figure 3.6. Use of OGL required that the FE solver, AGIFAP, needed to be modernised. The previously used FE components were no longer used and these functions were now built directly into the AGIFAP-program. The objective function and constraint functions vectors are part of AGIFAP in this generation.

The new modelling tool was developed using pre-compiled math components and the finite element solver was linked to the graphics of the new AGIFAP version. The FE-model was built in a pre-compiled math component and the FEA-results were read to the optimisation program through this component.

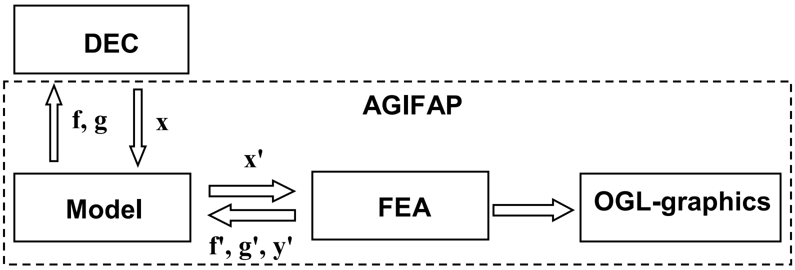


Figure 3.6 The communication between the optimisation model and FEA in the phase two optimisation program.

3.4 Third generation program

As work on the various optimisation programs progressed, the advantages that could potentially be provided by automatic FE-optimisation modelling became obvious. Manually formulating the optimisation model was a particularly laborious task and an automatic tool would be a great time saving tool. The developed FE-optimisation modelling tool is able to generate the objective functions, constraints and the design variables automatically as part of the conventional FE-modelling operation. The user activates the objectives, constraints and design variables during the FE modelling process. As the structural optimisation proceeds, the FE-model monitors the changes of the structure during the optimisation.

In the third-generation program, the optimisation modelling tool is integrated with the FE-modelling program. An FE-model is constructed in a conventional manner, but the program is able to generate simultaneously and automatically also an optimisation model based on the user-built FE-model. At the start of a design task, the designer constructs a preliminary FE model in much the same way as he would using some commercially available pre-processing software. After the FE-model is ready, the user can select and activate the objective functions, constraints and design variables of the FE-model. The FE-model automatically generates the input and output vectors for the optimisation function.

Because the optimisation algorithm was implemented in the FE-optimisation modelling tool, there was some loss of flexibility from the math components. However the optimisation-modelling tool can be implemented to the original optimisation-model-modelling tool as new FE-component. The forces and deflections of the structure can be examined by the user directly with the optimised FE-model. The constraints of the FE-model are programmed according the Finnish design code B7 (B7 1996).

The second major innovation in moving from the second to the third generation of the program was the inclusion of constraint classes. Previous versions defined constraints in terms of functions. With automated modelling techniques, however, the inclusion of constraint classes is much less time consuming for the designer and adds flexibility.

3.4.1 Optimisation algorithm

For all three program versions, the selected optimisation algorithm is a floating-point encoded evolutionary optimisation algorithm. There are several variants of DE and the particular version used was *DE/rand/1/bin scheme* as defined by Price (Price, K. V.1999).

In the first two generations of the program, penalty function based constraint handling was used. In the final version, constraints were handled without penalty factors by a hybrid constrain handling method developed by Lampinen (Lampinen, J. 2002). Details of this constraint handling method have been previously presented in section 2.3.3.4 of this thesis. The new constraint handling method made it possible to reduce the optimisation time, because useless FEA calculations can be recognised in advance thus bypassing the time-consuming calculation process. (Lampinen, J. 2002).

In practical applications, the objective and constraint functions are often computationally expensive. It is unnecessary to evaluate the objective function for candidate individuals already determined to be infeasible. The strategy outlined in Figure 3.7 was, therefore, implemented in the third program version as a means of avoiding unnecessary and time-consuming evaluations of the objective and constraint functions.

Numerous pieces of information were monitored during the optimisation process: population generation, fitness, best population, performance of the FE-model, selection criteria data, and statistics about design variables in the population. Statistical information included minimum and maximum values together with mean value and the standard deviation. The fitness function is a measure of how well the new populations perform with respect to the objective functions and is the sum of the deviations of the design variables of the population. If the value of the fitness function is not decreasing, it is an indication that some numerical problem exists or that an optimum solution has been reached. In either case, continued optimisation is useless if the value of the fitness function is stagnate or increasing.

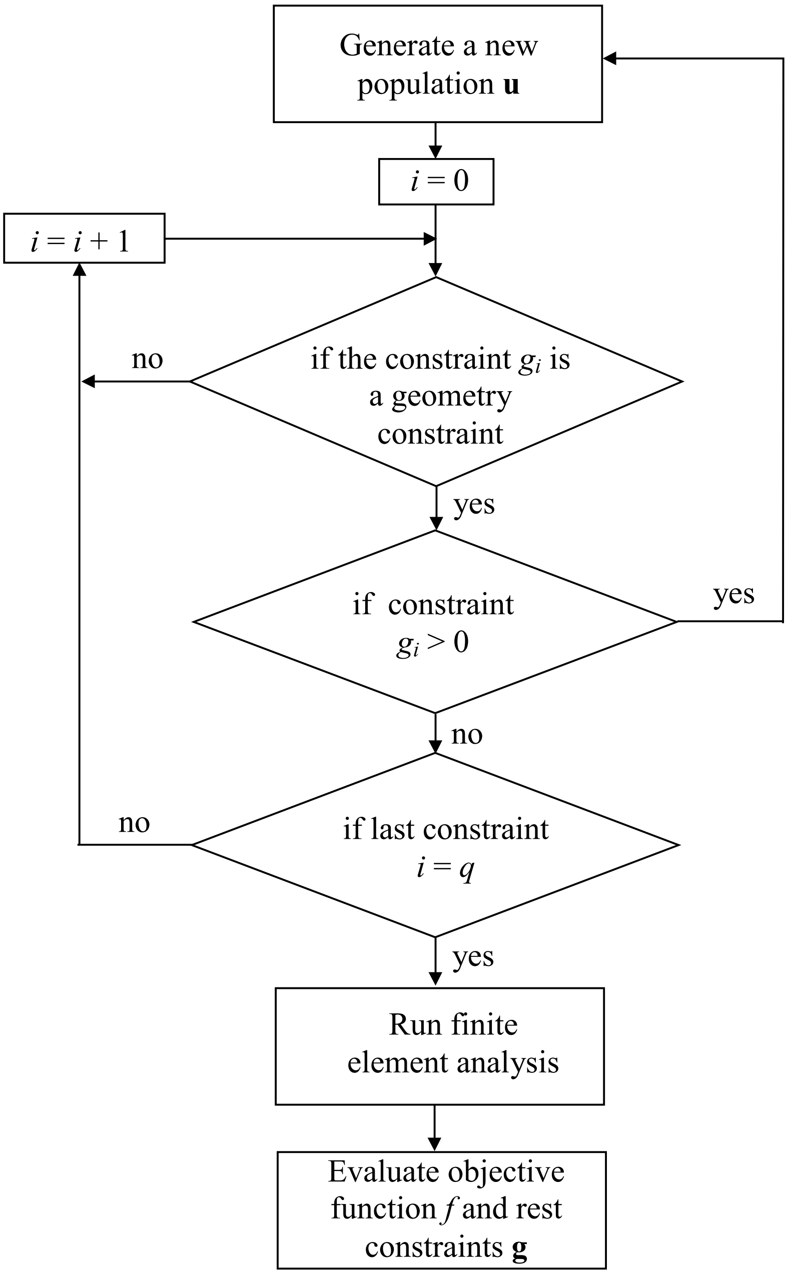

Generate a new population **u**
$i = 0$
$i = i + 1$
if the constraint g_i is a geometry constraint
no
yes
if constraint $g_i > 0$
yes
no
if last constraint $i = q$
no
yes
Run finite element analysis
Evaluate objective function f and rest constraints **g**

Figure 3.7 The optimised implementation of the method was used for avoiding unnecessary evaluations of the objective/constraint functions involved.

3.4.2 The structure of the developed FE-program

In the following sections, various aspects of the AGIFAP finite element analysis program are presented. These are organised in a hierarchical structure consisting of classes. In the following sections, classes are denoted by bold text. The main class hierarchy of the FE-program is:

```
TMain
   TOGLwindow
   TModel
    TElements
      TNode
      TCSection
      TMaterialProperty
      TNodalForceOfElement
    TNodes
      TRestraint
      TNodalLoad
      TNodalPointCoordinate
      TDisplacement
    TMass
    TArea
    TNodalLoads
    TCSections
    TMaterial
    TInputVector
    TOutputVector
```

The **OGLwindow** and **TModel** belong to the primary or first class of the program. The **TModel** class contains model data: **TElements**, **TNodes**, **TNodalLoads**, **TCSections**, **TMaterials**, **TInputVector** and **TOutputVector**. These secondary classes are visible and the member functions of these classes are available throughout the model. **TInputVector** class and contains data of the design variables of the optimisation model while **TOutputVector** contains information about restraints, design variables, and upper and lower limits of the design variables and the objective functions. The optimisation program is linked to the FE-model through **TInputVector** and **TOutputVector**.

The **TMass** class calculates the sum of the mass of the elements in the model. The **TArea** class is the sum of the area of the elements that would require painting. **TCSections** class includes calculation routines of cross section properties and the geometry of the available cross sections in the FE-model. **TCSection** class of the TElements class contains the cross section properties and geometry of the element.

The **OGLwindow** class is the 3D-graphical interface to the structure of the finite element model. The graphical interface uses the open graphic library functions (OGL-functions). The graphical interface can be used during the optimisation as a means of monitoring the finite element model.

The AGIFAPwin is a finite element program which consists of the pre- and post-processors and the analysis program. The AGIFAPwin can be used on ordinary finite element modelling and analysis. The program is compatible with classes of the Opitmaze-modelling and optimisation program. The optimisation model of the steel structure is easy to build. First, the model of the finite element model is build by using AGIFAPwin program. The finite element model is saved to a text file. The content of the text file is same as the model file of the AGIFAPwin-program. The design variables and the properties of the design variables can be selected. Now the optimisation program is opened and a new FE-component is created on the canvas of the optimisation model. The AGIFAPwin model is loaded to the FE-component. The component is now ready to link to the optimisation model if the design variables, constraints and the objective functions were defined in the modelling phase of the AGIFAPwin-model. The constraints of the FE-model are pre-programmed and in the class of the TElements and the **TNodes**. The graphical view of the FE-model can be monitored during the optimisation.

After the FE-model is generated, the restraints and the objective functions can be calculated with various values of the design variables.

3.4.3 Structure of the program

Figure 3.8 illustrates the progression of the optimisation program. The design variable vector **x** is transferred from the output vector of the differential evolution algorithm to the FE-modelling tool. The FE-model is then automatically updated first by adjusting the x, y, and z nodal coordinates. The nodal position is updated only if it is a design variable and the node is not associated with geometric constraint in the FE-model. Geometrically constrained node coordinates are associated with the master nodes. Based on the output vector of the differential evolution algorithm, the dimensions of the profiles are updated and the cross-section properties and the geometric constraints are solved. The finite element model is produced only if none of the geometric constraints is violated. Otherwise the FE-model is unnecessary because the result is useless. The FE-model is solved and result is read back to the FE-model.

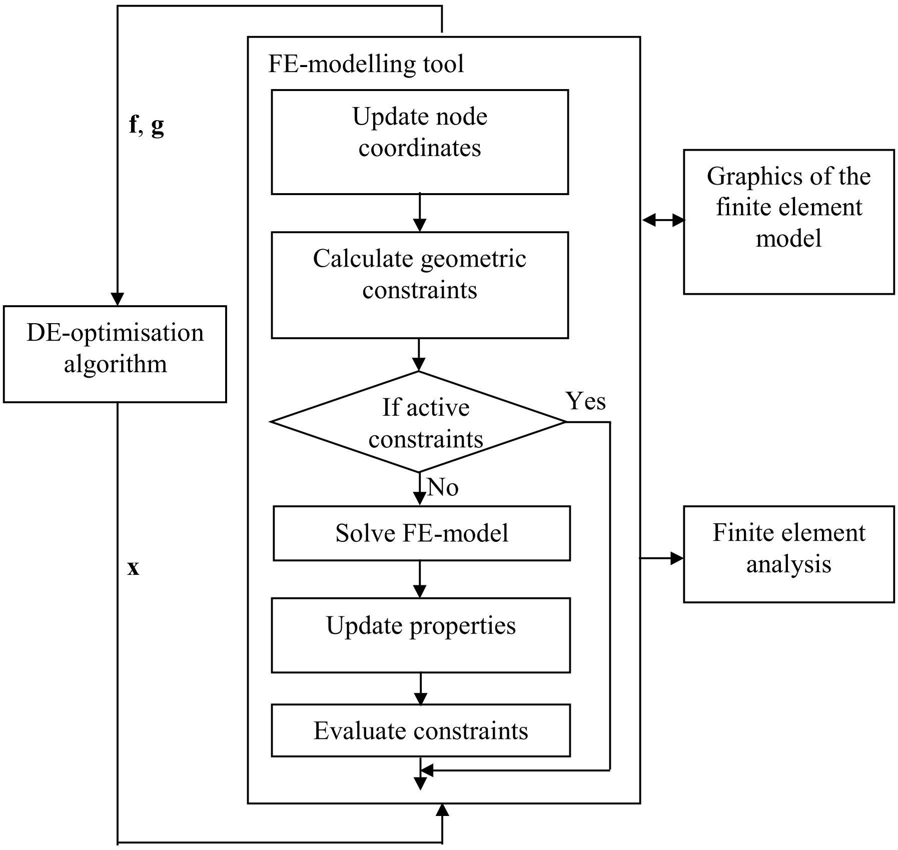


Figure 3.8 DE-optimisation algorithm uses finite element modelling tool and finite element analysis through design vector **x**, objective function vector **f** and constraint vector **g**.

Yield stresses of beams are selected from a table (Appendix 1) depending on the material and the thickness of material. All remaining constraints are solved according the limit state dimensioning.

The current version of the programs evaluates numerous constraint conditions for the beams: eccentric buckling, tension-, compression-, and bending resistances. Local strength of the T-, K- and KT-joints are also evaluated using formulas of the appendices (7, 8 and 9).

Lengths and masses of the beams are calculated for later use. The total mass of the structures is evaluated as an objective.

Following this extensive set of analysis, the values of the objective function ***f*** and constraint functions **g,** are moved to the input vector of the differential evolution algorithm. The optimisation algorithm controls the finite element model graphics updating. Usually the graphics is updated according the best-known solution.

3.4.3.1 Commands and classes

Essential classes and commands are presented in the sections 3.4.3.2 - 3.4.3.11 of this thesis. Constraint classes are presented by command tables as shown in Table 3.1. The class where the command belongs is presented on the first line of the table. The needed parameters are listed after a command presented on the second line. Extra data is presented on the last line if necessary.

Table 3.1 Structure of the command table.

TclassName				
COMMAND	Parameter 1	Parameter 2	....	Parameter *n*
Extra data				

3.4.3.2 Class TElement

An element is presented in Figure 3.9. The element has an element ID-number and ID-number for both i-node and j-node.

i-Node ID
Element ID
j-Node ID

Figure 3.9 Definition of the i- and j-nodes of the element.

A **TElement** class contains several classes, which contains the constraint handling of the elements. There are also classes for parameters, objectives and databases. **TGeometry** class defines the cross section of the element and **TProperty** contains physical properties of the cross section. **TMaterial** class includes the material properties. Nodes of the element are defined by **TNode** class. Constraint classes of the element are **TBendignResistance**, **TCentricBuckling**, **TTensileResistance**, **TCompressionResistance**, **THydraulics**, **TObjectives** class contains the mass calculation of the element. **TDatabase** class contains the cross section dimensions of the selectable rectangular hollow section profiles. The **TElement** class is presented in Table 3.1.

The element definition and classes are presented in Table 3.2. A command ELEMENT creates an element, which has two node IDs, material ID, profile ID and rotation angle.

Table 3.2 Definition of the element.

TElement						
ELEMENT	Element ID	i-Node ID	j-Node ID	Material ID	Profile ID	Rotation

```
class TElement
{
    public:
        TGeometry *Geometry;
        TProperty *Property;
        TMaterial *Material
        TNode  *iNode;
        TNode  *jNode;
        TBendingResistance *BendingResistance;
        TCentricBuckling *CentricBuckling;
        TTensileResistance *TensileResistance;
        TCompressionResisance *CompressionResistance;
        THydraulics *Hydraulics;
        TObjectives *Objectives;
        TDatabase *Database;
}
```

3.4.3.3 Class TNode

A node in Figure 3.10 is defined by coordinates in a cartesian global coordinate system.

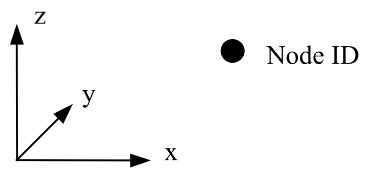


Figure 3.10 Definition of the node in the cartesian global coordinate system.

A class **TNode** contains the coordinates of the node and the joint constraints **TKjoint**, **TTjoint** and **TKTjoint**. A command NODE creates a node, which is defined by x, y and z coordinates. The **TNode** class and NODE-command are presented in Table 3.3.

Table 3.3 Definition of the node.

Tnode				
NODE	Node ID	x-coordinate	y-coordinate	z-coordinate

```
class TNode
{
    public:
        TNodalPointCoordinate *NodalPointCoordinate;
        TKjoint *Kjoint;
        TTjoint *Tjoint;
        TKTjoint *KTjoint;
}
```

The T-, K- and KT-joints can also be a part of a 3D-structure. The current program is limited in that joints must be planar. 3D-joints, like double K-joints, for example would need a new joint class but are not considered on this thesis.

3.4.3.4 Class TCentricBuckling

The parameters of the buckling constraints are presented in Table 3.4. The constraint is activated for element ID. A command CENTRICBUCKLINGY belongs to a class **TCentricBuckling**. The constrained buckling direction is to direction of the element y-axis. Parameters γ and α are buckling factors, γ_m is a safety factor. The factor α represents the shape of the beam cross section and factor γ depends on the fixing of the beam ends according the Figure 3.11. The constraint formulas are presented in Appendix 2. Forces and an element are presented in Figure 3.12.

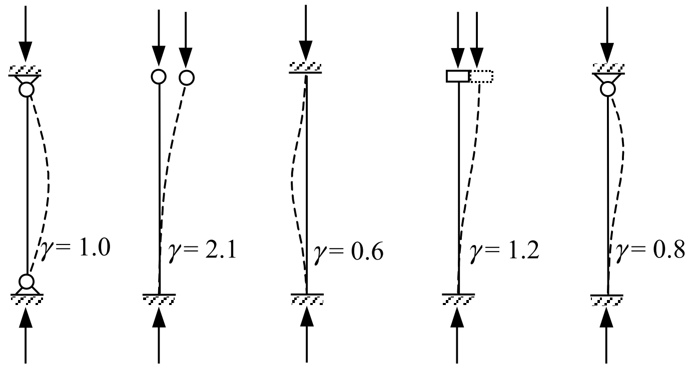


Figure 3.11 Buckling length $L_c = \gamma \cdot L$ factors.

F
Element id
F

Figure 3.12 Forces and element ID of the buckling element.

Table 3.4 Definition of the centric buckling constraint.

TcentricBuckling					
CENTRICBUCKLINGY	Element ID	γ	α	γ_m	
Default value of the safety factor γ_m, is 1.0.					

3.4.3.5 Class TBendingResistance

A bending moment of the beam element is presented in Figure 3.13. The bending moment resistance is calculated individually for both element ends because the ratio of normal force and moment can be different on i-end and j-end.

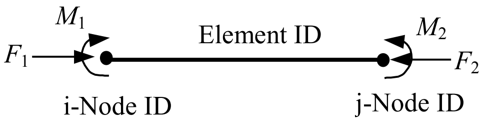


Figure 3.13 Bending moment of the beam element.

A command BENDINCONST and class **TBendingResistance** are presented in Table 3.5. The formulas of the bending resistance M_R are presented in Appendix 3. The equations do not handle the combination of the bending and shear forces. The bending resistance formulas are for cross section classes 1, 2 and 3. Parameters of the command are node ID and material safety factor γ_m.

Table 3.5 Definition of the bending constraint.

TbendingResistance					
BENDINGCONST	Element ID	Node ID	γ_m		
Default value of the safety factor γ_m, is 1.0.					

3.4.3.6 Class TTensileResistance

An element under tension is presented in Figure 3.14. The only parameter of the command TENSIONCONSTRAINT is the partial safety factor of the material. Constraint is presented in Table 3.6 and formulas of the tension resistance N_{Rt} is presented in Appendix 4.

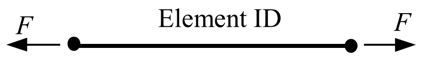


Figure 3.14 Definition of the tensile constraint in the optimisation model.

Table 3.6 Definition of the tensile resistance constraint.

TtensileResistance					
TENSIONCONSTRAINT	Element ID	γ_m			
Default value of the safety factor γ_m, is 1,0.					

3.4.3.7 Class TCompressionResistance

A constraint of the compressed element is quite similar as element under tension. A compressed element is presented in Figure 3.15 and the formulas of the tension resistance N_{Rt} is presented in Appendix 5.

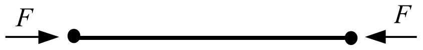


Figure 3.15 Definition of the compression constraint in the optimisation model.

The only parameter of the command TENSIONCONSTRAINT is a material safety factor γ_m presented in Table 3.7.

Table 3.7 Definition of the compression resistance constraint.

TTensileResistance					
TENSIONCONSTRAINT	Element ID	γ_m			
Default value of the safety factor γ_m = 1,0					

3.4.3.8 Class THydraulics

A hydraulic cylinder and corresponding element are presented in Figure 3.16. The command CYLINDERCONSTRAINT creates buckling constraint of the hydraulic cylinder. The parameters of the command are presented in Table 3.8. In the FE-model the hydraulic cylinder is a circular hollow section beam. The beam is selected using the element ID number. The piston node ID is needed to set the direction of the orientation of the hydraulic cylinder. The diameter of the circular beam in the FE-model is equal to the diameter of the piston rod. The piston diameter and the piston rod diameter are selected from the table of the hydraulic cylinders using table ID. The table of the hydraulic cylinders is presented in the Appendix 6. Forces of the hydraulic cylinder are calculated using the maximum pressure p_{max} and minimum pressure p_{min} of the hydraulic system and the diameter D of the piston and the diameter d of the piston rod. The friction and the other losses of the force are calculated using the efficiency factor η. The material of the hydraulic cylinder is defined by the factors E and f_y. The safety factor n_h is applied. Constraint formulas of the hydraulic cylinder is presented in Appendix 7.

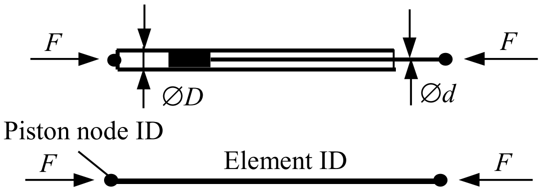


Figure 3.16 Definition of the hydraulic cylinder in the optimisation model.

Table 3.8 Definition of the hydraulic cylinder.

THydraulicCylinder					
CYLINDERCONSTRAINT	Elem ID	Piston node ID	Table ID	η	p_{min}
	p_{max}	E	f_y	n_h	d
	D				
Default value of the safety factor n_h, is 3.5.					

3.4.3.9 Class TTjoint

A T-joint can fail in several ways. The failure modes are listed in Figure 3.17. An upper flange of the chord may break by yielding if the ratio of diagonal width and a chord width is small (Figure 3.17a). In Figure 3.17b is presented a wide chord with thin web. The diagonal is smaller than chord. The diagonal may cut through the upper flange of the chord (Figure 3.17c). The mode presents a joint with strong chord with thin diagonal when the diagonal under tension can break. The wide and thin walled diagonal may buckle locally like in Figure 3.17d. The cross section of the low and thin walled chord may ultimately yield as in Figure 3.17e. In Figure 3.17f, the widths of the chord and diagonals are equal. The high and thin walled chord can buckle locally. The compressed flange of the thin walled and wide chord may buckle locally (Figure 3.17f). These failure modes are used in the classes **TTjoint, TKjoint and TKTjoint** sections. (Rautaruukki steel products designers guide 1998)

In this thesis and in the computer program, the cross-sections are limited to square and rectangular hollow sections. The program could be extended to also handle circular hollow sections (CHSs) with only a moderate amount of work.

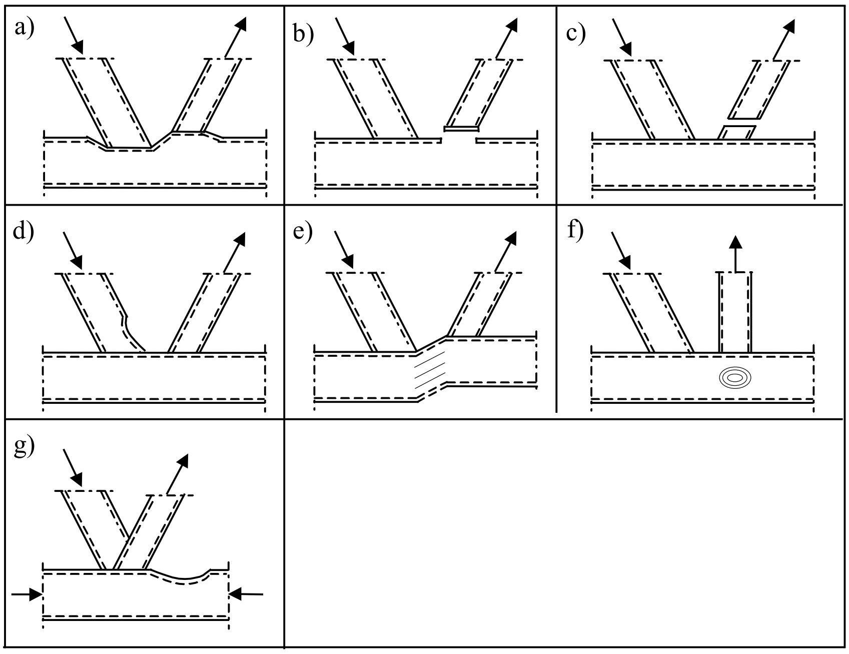


Figure 3.17 Local braking modes of the joints of the chord and diagonals (Rautaruukki steel products designers guide 1998).

Dimensions of the T- and Y-joint are presented in Figure 3.18. The corresponding element model and element numbering are presented in Figure 3.19. A user has to select one node and three connecting elements from the FE-model.

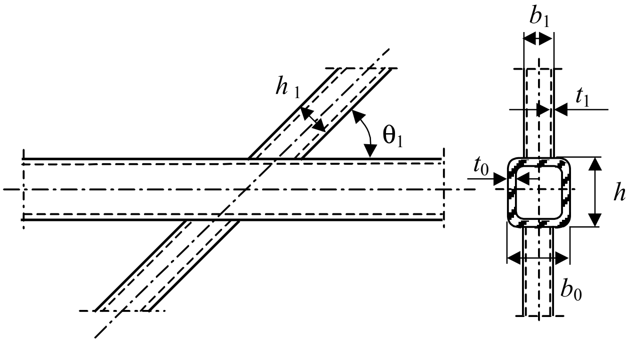


Figure 3.18 Dimensions of the T- or Y- joint.

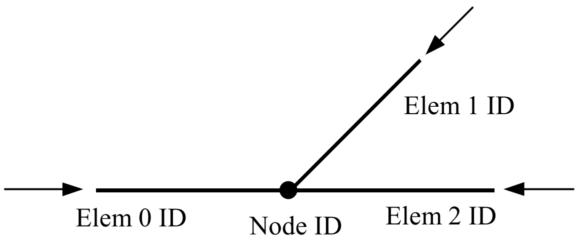


Figure 3.19 Definition of the T-or Y-joint in the FE-model.

Parameters of the command TJOINT are presented in Table 3.9. Element IDs 0, 1 and 2 have to be selected in a clockwise orientation starting from the chord beam. Selected elements must be connected to the same node. A joint has its own safety factor, γ_j, while the material safety factor is $\gamma_{m0.}$ The constraint formulation is presented in Appendix 8.

Table 3.9 Definition of the T-joint.

TTjoint						
TJOINT	Node ID	Elem ID$_0$	Elem ID$_1$	Elem ID$_2$	γ_{m0}	γ_j
Default values of the material safety factor γ_{m0} and the joint safety factor γ_j, are 1,0.						

3.4.3.10 Class TKjoint

Dimensions of the K-joint are presented in Figure 3.20. The joint consists of the upper or lower chord and two bracing. Manufacturability sets minimum gap *g* between braces. The gap causes the eccentricity *e*, which causes additional moments in the joint. Corresponding FE-model and element numbering are presented in Figure 3.22. Accurate joint model should be constructed with rigid element as shown in Figure 3.21. In this thesis, the affect of the eccentricity is ignored.

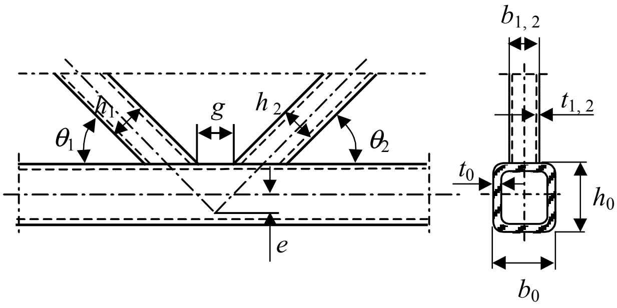


Figure 3.20 The geometry and the dimensions of the K-joint.

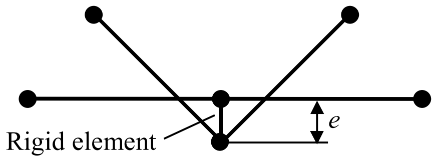


Figure 3.21 Finite element model of the K-joint with eccentricity *e*.

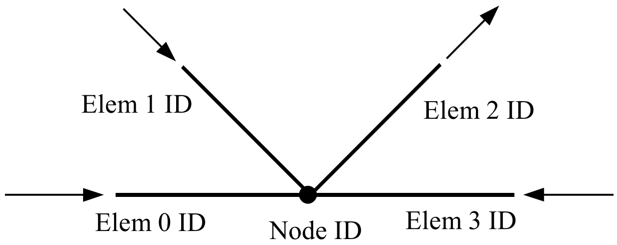


Figure 3.22 Definition of the K-joint in the optimisation program.

The definition of the command KJOINT is presented in Table 3.10. The node ID is the intersection node of the chord and diagonal elements. The gap is not modelled but it is calculated in the constraint equations. The eccentricity is not modelled in the finite element model and the additional bending moment is not considered. The joint has a safety factor γ_j, which is automatically set to 1,0. The material safety factor γ_{m0} is usually set to 1,0. The constraint formulas are presented in Appendix 9.

Table 3.10 Definition of the K-joint.

TKjoint						
KJOINT	Node ID	Elem 0 ID	Elem 1 ID	Elem 2 ID	Elem 3 ID	Gap g
	γ_{m0}	γ_j				
Gap g is the measure in Figure 3.20. Default values of the material safety factor γ_{m0} and the joint safety factor γ_j, are 1,0.						

3.4.3.11 Class TKTjoint

Dimensions of the KT-joint are presented in Figure 3.23. The strength calculation of the KT-joint is quite similar than K-joint. KT-joint has two gaps g_1 and g_2. The gap depends on dimensions of the intersecting beams and the angles θ_1 and θ_2 and selected eccentricity e. The user defines a minimum gap, $g_{\min}$, in this application. The formulation of the gap and constraints are presented in Appendix 8. The element numbering of the elements 0 - 4 is presented in Figure 3.24.

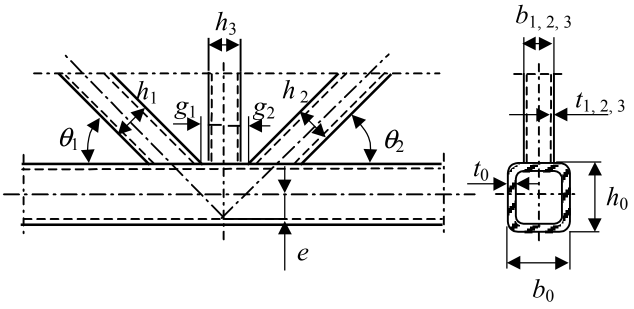


Figure 3.23 The geometry and the dimensions of the KT-joint.

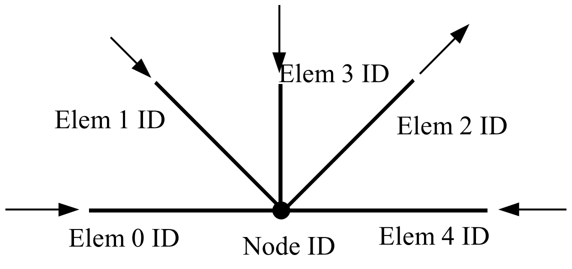


Figure 3.24 Definition of the KT-joint in the FE-model.

Parameters of the command KTJOINT are presented in Table 3.11. The elements 0 - 4 must joint at the same node. The user has to define only the minimum gap $g_{\min}$. Material safety factor is γ_{m0} and joint safety factor γ_j. The constraint formulas are presented in Appendix 10.

Table 3.11 Definition of the KT-joint.

KT-joint						
KTJOINT	Node ID	Elem 0 ID	Elem 1 ID	Elem 2 ID	Elem 3 ID	Elem 4 ID
	g_{min}	γ_{m0}	γ_{j}			
Default values of the material safety factor γ_{m0} and the joint safety factor γ_{j}, are 1,0.						

3.4.3.12 Geometry constraints of the FE-model

Geometry constraints of the FE-model are needed to keep nodes on a particular line or plane in the FE-model space during the optimisation. A situation is illustrated in the Figure 3.25. The slave nodes from 2 to 7 are kept on the line between master nodes 1 and 8. The nodes of the lower chord are kept on same xy-plane with node 9. The coordinate of the master node is commonly the design variable in the optimisation model. The master node is first moved to the new position during the optimisation and slave nodes move to new positions according to this.

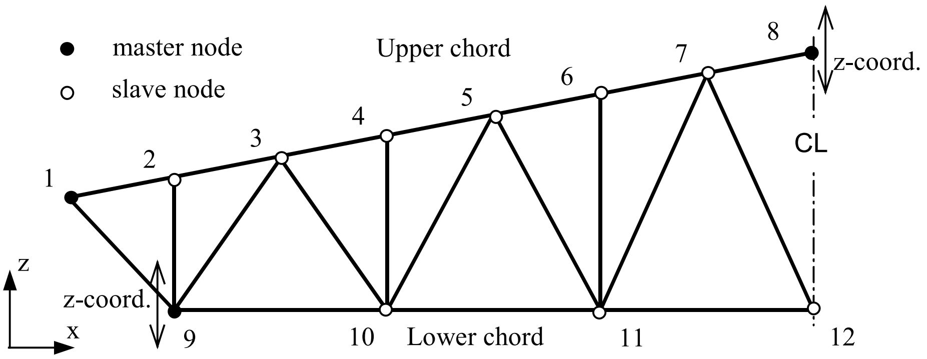


Figure 3.25 The definition of the geometric constraints in the FE-model. The master nodes correspond to design variables.

Parameters of the command GEOMCONST are presented in Table 3.12. Constrained node is interpolated between x, y and z-positions of the nodes A and B.

For example, the geometry constraint of the node 2 of the upper chord is defined by command GEOMCONST,2,0,0,0,0,1,8 and the geometry constraint of the node 10 is defined by command GEOMCONST,10,0,0,0,0,9,9. The z-position of the node 2 is interpolated between node 1 and 8. The z-position of the node 10 is same as the z-position of the node 9.

Table 3.12 Definition of the FE-model geometry constraint.

TGeomConstraint					
GEOMCONST	Node ID	Node Ax ID	Node Bx ID	Node Ay ID	Node By ID
	Node Az ID	Node Bz ID			
The Node ID coordinate is same than Node A ID if Node A ID = Node B ID					

3.5 Summary

Finite element analysis plays an important part in the design of steel structures as a means of calculating deflections and forces. The FE-analysis program called AGIFAP has been selected to form one basic element of the FE-analysis tool. An object-oriented framework for AGIFAP has been applied to ensure that routines in the program are re-useable and extendable.

Early versions of the AGIFAP program were developed by Steel Structures Laboratory at Lappeenranta University of Technology. The Windows version with of the program with OpenGL-graphics was developed in co-ordination between OME-Software Company and the author of this thesis. The automated formulation program for the optimisation models for steel beam structures (third generation program) was developed by the author.

The optimisation and modelling program has evolved through three program versions during this study. The first generation modelling tool uses graphically editable components. Components can include mathematical formulas, tables and links to external programs like FE- and simulation software. It is a time consuming process to link the optimisation model to the external FE-model, if many nodal coordinates, element properties and material properties are defined as variables. The program is without the FE-model monitoring during the optimisation. The differential evolution algorithm with penalty factor based constraint handling was selected as the optimisation algorithm.

A FE-model was monitored with 3D-graphics during the optimisation in the second-generation optimisation program. The FE-model and mathematics were ready-compiled and non-editable. The optimisation algorithm was The Differential Evolution algorithm with penalty factor based constraint handling.

In the third program generation, the time consuming process of linking the optimisation model to the FE model was automated. Elements and nodes of the FE-modelling program include the classes of the constraints and objectives, for example, **TTjoint**, **TCentricBuckling** and **TObjectives**. In this way, the optimisation model is built up simultaneously with the FE-modelling. A FE-model was monitored with 3D-graphics during the optimisation. Constraints were handled without penalty factors by a new constraint handling method. The new constraint handling method made it possible to reduce the optimisation time, because useless FEA calculations can be recognised in advance thus bypassing the time-consuming calculation process. Limit state design method, including ultimate limit states and serviceability limit states, is the most common design method for steel structures and is therefore used in this program.

The automated FE-modelling and optimisation tool includes several constraint classes, which are included into classes of the FE-program. Each node and element in the FE-program and FE-model have their own constraints. The formulas of the constraints are presented in appendices. The development of these classes has eliminated the time consuming work of trying to define all these constraints for an optimisation FE-model. The FE-modelling tool includes additional functions for defining master and slave nodes for truss structures. Slave node or nodes follow the geometry defined by master node or nodes, during the optimisation when master nodes moves in a design space.

4 VERIFICATION AND APPLICATIONS

This chapter demonstrates the numerical capabilities of the automated optimisation program. The optimisation portion of the program is first tested using eight standard test functions and optimisation problems. The optimisation system, including both the modelling and FE analysis programs, is then tested on several large steel structures optimisation problems. These structures are a hydraulically driven multi-redundant boom and two truss structures.

4.1 Test functions

The optimisation program and the computer code for the differential evolution algorithm is tested using eight common test functions for evolutionary algorithms. These test functions are presented in Table 4.1.

Table 4.1 Test functions.

Function no.	Reference	Additional data
1	-	Rastrigin 10 D
2	Floudas C.A. and Pardalos P.M. 1987	-
3	HockW. and Schittkowski K. 1981	problem no. 113
4	Hock W. and Schittkowski K. 1981,	problem no. 116
5	Koziel S. and Michaelewics Z. 1999	-
6	Hock, W. and Schittkowski, K. 1981	problem no. 100
7	Himmelblau, D. 1972	problem no. 16
Functions 2-7 are referenced and evaluated by Lampinen. (Lampinen, J. 2002)		

Table 4.2 Values of eight test functions.

f	*CR/F/NP*	*NF*	***f*(x*) test values**	***f*(x*) best know values**
1	0.05/0.8/20	450	*f*(0,0,0,0,0,0,0,0,0,0) = 0	*f*(0,0,0,0,0,0,0,0,0,0) = 0
2	0.9/0.9/20	80 000	*f*(1,1,1,1,1,1,1,1,1,3,3,3,1) = -15	*f*(1,1,1,1,1,1,1,1,1,3,3,3,1) = -15
3	0.95/0.9/25	10 000	*f*(2.33049, 1.95136, -0.47740, 4.36576, -0.62455, 1.03814, 1.59420) = 680.63006	*f*(2.330499, 1.951372, 0.4775414, 4.365726, -0.6244870, 1.038131, 1.594227) = 680.6300573
4	0.95/0.9/35	80 500	*f*(579.3067, 1359.9706, 5109.9708, 182.0177, 295.6012, 217.9823, 286.4165, 395.6012) = 7049.2481	*f*(579.3040, 1359.975, 5109.970, 182.0175, 295.6012, 217.9825, 286.4163, 395.6012) = 7049.248021
5	0.95/0.9/15	4 455	*f*(14.0950, 0.84296) = -6961.814	*f*(14.095, 0.84296) = -6961.81381
6	0.95/0.9/25	175 000	*f*(1.95068, 2.78901, 8.80989, 5.05910, 1.02631, 1.42293, 0.96093, 9.57055, 9.57323, 7.34134) = 15.17491	*f*(2.171996, 2.363683, 8.773926, 5.095984, 0.9906548, 1.430574, 1.321644, 9.828726, 8.280092, 8.375927) = 24.3062091
7	0.95/0.9/15	18 825	*f*(78.000, 33.000, 27.071, 45.000, 44.969) = -31025.5602	*f*(78.000, 33.000, 27.071, 45.000, 44.969) = -31025.5602
CR mutation probability, *F* differential factor, *NP* population size, *NF* function evaluations				

Results from the numerical experiments are given in Table 4.2. Details of the different test functions are presented in Appendix 11. These tests are intended to confirm that the optimisation algorithm is utilised properly. In this table, *CR* is the mutation probability, *F* is the differential factor, *NP* is the population size, and *NF* is the number of function evaluations. In all cases the test values of the optimisation function and the design variables are identical or nearly identical to the best known published values.

4.2 Design of new industrial product – the multi-redundant boom

4.2.1 The structure

The first industrial case to be presented is that of a multi-redundant tripod boom. This type of boom could, for example, be employed in a drilling tractor used in mining and quarrying. In principle, the tripod boom is an open loop spatial manipulator, which includes two three-degree of freedom (DOF) parallel modules connected in series. This type of mechanism can cover a large workspace and act redundantly. In practice, the tripod boom consists of two modules. Each module consists of three hydraulic cylinders, one telescope, and triangular connection plates. A schematic diagram of the structure and these different components is presented in Figure 4.1.

The modular concept is similar to that presented by Innocenti (Innocenti, et al. 1993). The three cylinders are connecting to the vertices of the connection plates while the telescope is connected to the middle of the plate. Different types of joints would be needed in order for the boom to reach different working areas and to achieve different degrees of freedom. The tripod boom - structure presented here differs significantly from conventional parallel and special structures.

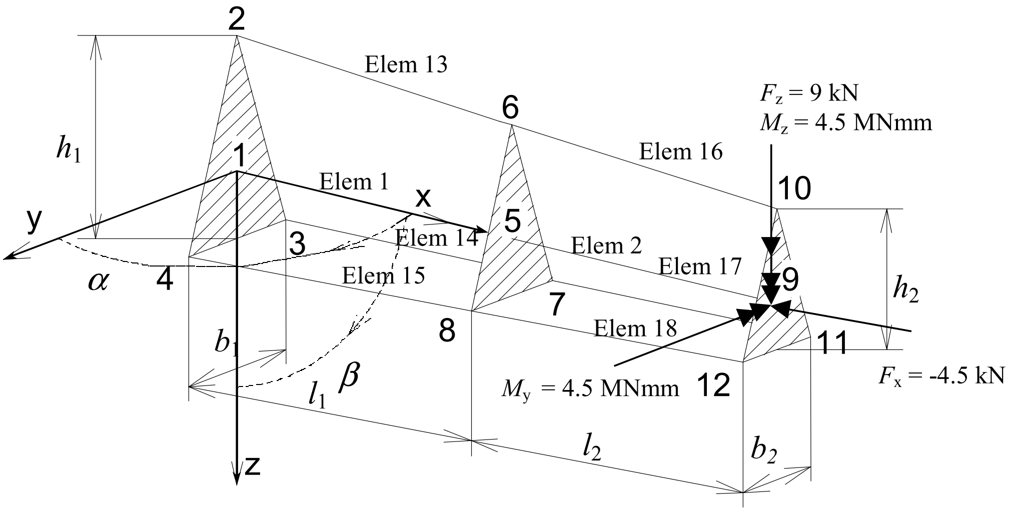


Figure 4.1 The geometry, dimensions and the loads of the multi redundant boom.

The principle difference between a tripod boom and a normal parallel structure is with regard to the achievable working area. Depending on the joint selections, the tripod boom can reach a very versatile working area. It can even reach behind corners; an operation impossible with parallel structures. Several of the numerous joint combination alternatives for the telescopes are presented in Figure 4.2. Ultimately, the selected joint was the joint combination I. It was determined that this would produce a boom more economical to fabricate. The working area for this type of boom structure is illustrated in Figure 4.3.

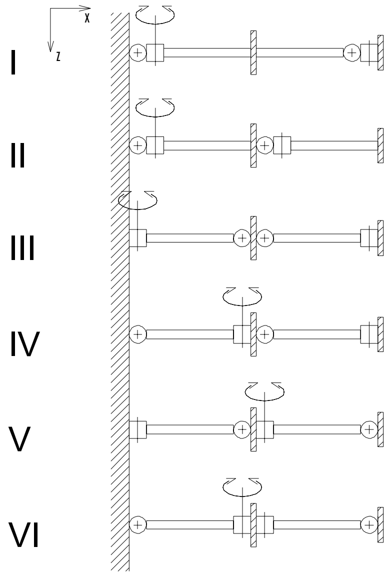


Figure 4.2 Six joint combination alternatives of the telescopes (Lagsted, A. et al. 1999b).

One significant advantage of a tripod boom over a normal spatial structure is the increased flexibility. With properly selected joints a tripod boom structure can compensate the deformation of the end of the boom. The cylinders cause mechanical synchronisation, which rotates the connection plates so that undesirable rotation of the boom tip is eliminated. It is also possible to vary the angle of the cylinders so as to increase or decrease the rigidity of the structure. Structural stiffness is converted to hydraulic stiffness.

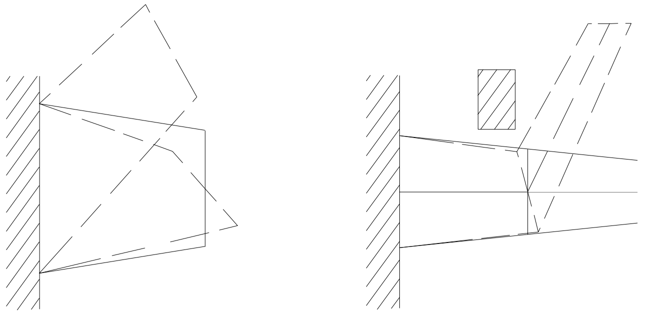


Figure 4.3 The working area of the multi-redundant boom (Lagsted, A. et al. 1999b).

4.2.2 Parameters

To describe the boom structure, a total of 9 design parameters were required. Bounds had to be set for discrete variables x_i. Lower and upper bounds have to set for discrete variables and for continuous variables. For components selected from a table of available sizes, the lower bound is usually $x_i^1 = 1$ and the upper bound, x_i^u, is the number of possible discrete values. Limits are set in practice automatically when the user is selecting allowed components from a database.

Design variables of the boom are presented in Table 4.3. Height of the mounting and end plates is between 300 and 2 000 mm. The end plate is expected to be smaller than the mounting plate. The length of the telescopes is allowed to vary between 1 000 mm to 3 000 mm and the total length of the boom is expected to be from 2 000 mm to 6 000 mm. Lower hydraulic cylinders in both modules are identical.

Table 4.3 Design variables of the boom. Element numbers corresponds to numbering in Figure 4.1.

Design variable	**Variable name**	**Element no.**	$\mathbf{x^l \leq x_i \leq x^u}$
Height of the mounting plate	h_1	-	$300 \leq x_1 \leq 2000$ mm
Height of the end plate	h_2	-	$300 \leq x_2 \leq 2000$ mm
Length of the telescopes	$l_1 = l_2$	-	$1000 \leq x_3 \leq 3000$ mm
Hydraulic cylinder	Cyl_{12}	14	$1 \leq x_4 \leq 20$
Hydraulic cylinder	Cyl_{13}	13	$1 \leq x_5 \leq 20$
Hydraulic cylinder	Cyl_{14}	17	$1 \leq x_6 \leq 30$
Hydraulic cylinder	Cyl_{15}	16	$1 \leq x_7 \leq 20$
Telescope profile	Tel_1	1	$1 \leq x_8 \leq 125$
Telescope profile	Tel_2	2	$1 \leq x_9 \leq 125$

4.2.3 Constraints

Constraint exists between several sets of variables, for example the yielding strength of a steel plate is dependent on the plate thickness. When components are selected from a component database, most of the dimensions for the component depend on the selected component. For example, when using a hydraulic cylinder database, the diameter of the cylinders depends on the selected cylinder and cannot be freely selected by the designer. In some cases the length of a cylinder may be continuous variable within some allowable region. Table 4.4 summarises the constraints used for the boom structure.

For the boom structure, the constraints relate primarily to the performance of the six individual cylinders in terms of adequate buckling strength and force. Fatigue capacity of the two telescoping beams and maximum deflection of the boom end under full load are also important constraints.

Fatigue strength of the telescopes is evaluated by formulas presented in Appendix 12. Material partial safety factor $\gamma_m = 1.3$ and $\gamma_F = 1.0$. The fatigue class (FAT) for the welded joint of the rectangular section and plate is 45. In order to achieve an infinite fatigue life, $N > 5 \cdot 10^6$, the fatigue resistance, $\Delta\sigma_R$, is computed as 33.2 MPa. (Hobbacher, A. 1996)

Table 4.4 Constraints of the boom structure.

Constraint	Constraint name	Constraint
buckling of hydraulic cylinder	g_1	$N_{12} \leq F_{cr12}$
buckling of hydraulic cylinder	g_2	$N_{13} \leq F_{cr13}$
buckling of hydraulic cylinder	g_3	$N_{14} \leq F_{cr14}$
buckling of hydraulic cylinder	g_4	$N_{15} \leq F_{cr15}$
buckling of hydraulic cylinder	g_5	$N_{16} \leq F_{cr16}$
buckling of hydraulic cylinder	g_6	$N_{17} \leq F_{cr17}$
adequate of hydraulic force	g_7	$F_{12} \leq F_{min12}$
adequate of hydraulic force	g_8	$F_{13} \leq F_{min13}$
adequate of hydraulic force	g_9	$F_{14} \leq F_{min14}$
adequate of hydraulic force	g_{10}	$F_{16} \leq F_{min16}$
adequate of hydraulic force	g_{11}	$F_{17} \leq F_{min17}$
adequate of hydraulic force	g_{12}	$F_{15} \leq F_{min15}$
fatigue life for telescope 1	g_{13}	$\Delta\sigma_{S,d1} \leq \Delta\sigma_{R,d1}$
fatigue life for telescope 2	g_{14}	$\Delta\sigma_{S,d2} \leq \Delta\sigma_{R,d2}$
maximum deflection of boom end and full force 50 mm	g_{15}	$\delta \leq \delta_{max}$

4.2.4 Objectives

Table 4.5 presents the objectives of the optimisation problem. The weight objective includes the weight of the steel structure and the total length of the boom. The weight of various working attachments for the boom is difficult to determine and uncertain. In this optimisation exercise, attachment weigh is defined as 10% of the total weight of the steel structure.

Working area for the boom is related to the length and is set to be as large as possible. In a tunnel mining operation, for example, the working will not vary significantly. However, when a hall is quarried the working area is more significant. Minimum adjusting time, as the boom moves from one position to the next, is an important consideration because a vehicle will typically drill dozens of charge hole from a single location. Drilling accuracy may be unsatisfactory if deflection is too large. This requirement is not part of the objective function to be minimised but was previously set as a constraint.

Table 4.5 Objectives of the optimisation model.

Objectives		Weight w_i	Ideal value f_i^0
Total weight of the steel structure $i = 1$	minimisation	0.05	400 kg
Total length of the boom $i = 2$	maximisation	0.95	4 000 mm

The total mass of the boom is calculated by the equation:

$$f_1 = \sum_{i=1}^{6} m_{\mathrm{hc},i} + \sum_{i=1}^{2} m_{\mathrm{ap},i} + \sum_{i=1}^{2} m_{\mathrm{tel},i} \tag{4.1}$$

where $m_{\mathrm{hc,i}}$ is the mass of the hydraulic cylinder piston rod i, $m_{\mathrm{ap},i}$ is the mass of the attachment plate i and $m_{\mathrm{tel},i}$ is the mass of the telescope i. The mass is minimised because this gives a logical direction for the optimisation process. The total mass is not very important and, therefore, the weight factor for mass is much lower than that for boom performance as seen from Table 4.5. The weighting method was as follows:

$$d_{\mathrm{p}}(\mathbf{x}) = \left[\sum_{i=1}^{2} w_i \left| \frac{f_i^0 - f_i(\mathbf{x})}{f_i^0} \right|^2 \right]^{1/2} \tag{4.3}$$

Mass minimisation guides the optimisation algorithm to search for more lightweight components when available. The large working area is the most important criterion, but mass will dominate the optimisation if the weighting factor is too large.

4.2.5 Databases

The optimisation program made use of the databases presented in appendices 6 and 13. The databases contain hydraulic cylinders and cross sections of the telescopes. There are 125 possible rectangular hollow sections and 20 hydraulic cylinders in these databases.

4.2.6 Results

The optimised height h_1 of the mounting plate was 680 mm and height h_2 of the end plate was 370 mm. The combined length l_1+l_2 of the telescopes was 3 400 mm. The total mass of the boom was 570 kg. The selected telescope profile was 250x250x12.5. Selected hydraulic cylinders were 80x56 (elements 12, 13 and 15) and 63x45 (elements 15, 16 and 17). The evolution of the optimisation FE-model is presented in Appendix 14.

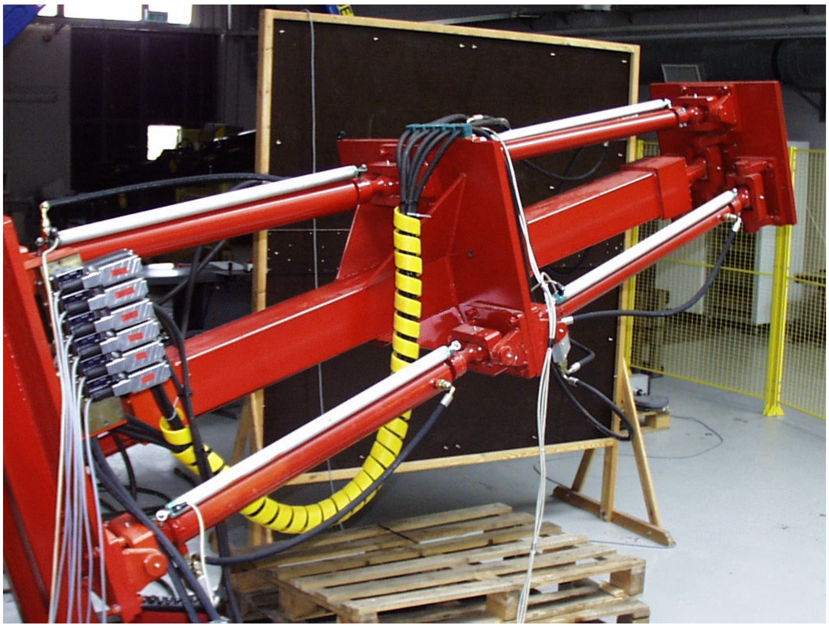


Figure 4.4 The physical prototype of the boom.

The maximum deflection constraint is activated during the optimisation. In addition, the fatigue life constraint is activated. This is a requirement, because the length of the boom is maximised and forces in welded joints are high.

A photograph of the completed prototype structure is presented in Figure 4.4. Lagstedt presents a static flexibility and kinematics study of a tripod boom with different joint combinations. Wu developed the control system and measured the performance of the boom (Kilkki, J. et al. 2002, Wu, H. et al. 2002, Lagstedt, A. et al. 1999a, Lagsted, A. et al. 1999b).

4.3 Flat roof

The second case study reported here is that of a KT-truss structure. When the design is implemented such that the loads are applied through the joints, this type of structure is suitable for very long span lengths. The number of the beam elements is low and the joints are modelled as simple. The buckling length of the upper chord is long which may lead to a relatively heavy chord as compared to the structures.

The strength of the upper chord of the KT-structure is better than that for the K-truss because the upper chord is supported at more positions. The joints of the lower chord are more difficult to fabricate for a KT-structure. Figure 4.5 shows both the K- and KT truss structure.

Limit state design concepts are used and clearly defined ultimate limit state criteria are defined for the truss members and joints. A serviceability limit state criteria for truss deflection is also defined.

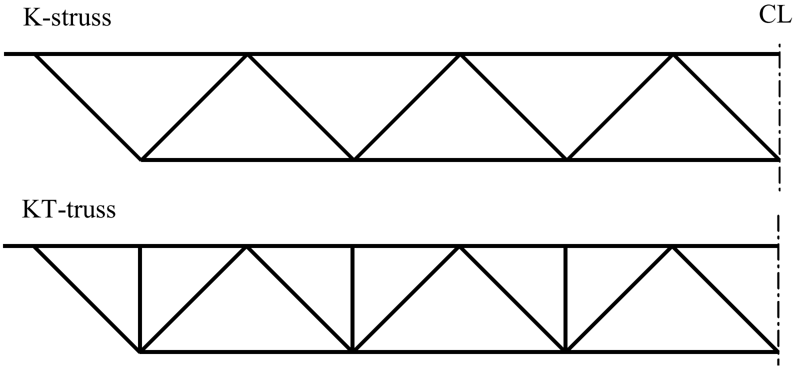


Figure 4.5 The K- and KT-structure.

The phases of the design of the truss are presented in Figure 1.3. These are:

1) Loads on the structure are clarified and the most dangerous load combinations are determined.
2) Height of the structure is decided. The type of the truss and cross member division is selected.
3) Preliminary selection of the beams. Resistance of the most heavily loaded joint is calculated..
4) Real forces of the beams are calculated.
5) Strengths of the joints are calculated.
6) Deflection of the truss is calculated.
7) Cross members and the joints of the cross members are designed.

In traditional optimisation the phases 3 to 6 should be repeated until the solution is satisfactory. In automated formulation phases 3 to 6 are included to the optimisation. The positions of the joints, the height of the structure and beam cross sections can be design variables. Constraints and objectives of the optimisation model are generated by the nodes and elements.

4.3.1 Structure

The truss structure shown in Figure 4.6 is optimised. The dimensions of the beam sections are the design variables. The slave nodes follow the z-coordinate of the master node thus ensuring that the lower chord remains straight. The symmetry line is denoted by (CL). Actually, the K-joint on symmetry line would require a symmetry K-joint constraint, which is not considered on this thesis.

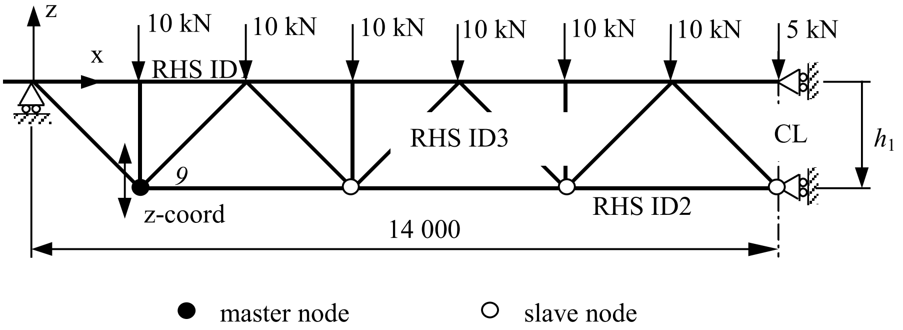


Figure 4.6 A flat roof structure. The z-coordinate of the node 9 is the design variable.

4.3.2 Constants

Constraints of the KT-truss model are presented in Table 4.6. A six gap truss is to be optimised. The span of the entire truss would be 48 000 mm so the length of the half truss that is modelled is 14 000 mm. Material in this exercise is selected as S355.

Table 4.6 Constraints of the flat roof.

Constant	Constant name	Value
Gaps	n	6
Total length	L	24 000 mm
Material	f_y	355 MPa (nominal)

4.3.3 Design variables

In this case, only the profiles of the truss element are to be optimised. Optimisation parameters are presented in Table 4.7. Only rectangular hollow section profiles are considered.

Table 4.7 Design variables of the KT flat roof truss.

Design variable	Design variable name	Value
Upper chord	RHS$_1$	$1 \leq x_1 \leq 125$
Lower chord	RHS$_2$	$1 \leq x_2 \leq 125$
Main bracing	RHS$_3$	$1 \leq x_3 \leq 125$
Height	h_1	$-6\,000 \leq x_3 \leq -\,500$

4.3.4 Constraints

The ultimate limit states for the beam elements of the truss included tensile strength, compression strength, buckling strength and bending strength. The ultimate limit states for the T-, K- and KT-joints are: yielding of the top of the chord, buckling of the chord web, cutting of the chord, breakage of the main bracing, and shearing of the chord.

Serviceability limit state constraint is the maximum deflection of the truss structure: $g_1 = L/300 - \max\{\delta_{\text{node i}}\} \geq 0$. A total of 334 inequality constraints are used in the optimisation model.

4.3.5 Objectives

The minimised objective of the optimisation is the total mass of the truss structure.

$$m_{\text{tot}} = f(\mathbf{x}) = \sum_{i}^{n} m_{\text{elem}\,i} \tag{4.4}$$

4.3.6 Databases

The optimisation program made use of the profile databases presented in Appendix 13. These are the same databases as were used for the telescope structure for the multi-redundant boom. The database contains 125 possible rectangular hollow sections.

4.3.7 Results

The resulting optimise truss structure is summarised in Table 4.8. The objective function, which was defined as minimum mass, attained a minimum value of 376 kg. The optimisation program required a total of 15 880 function evaluations.

Table 4.8 Optimised design variables in the ridge roof.

Design variable	Variable name	Value	Profile
Upper chord	RHS_1	27	RHS 100x100x3
Lower chord	RHS_2	16	RHS 80x80x4
Brace	RHS_3	6	RHS 60x60x2.5
Height	h_1	-1 306	-

The automated modelling and the integrated FE-analysis routine significantly reduced the modelling time. The modelling time of the optimisation model is almost equal to the modelling time of the FE-model. The small extra work comes form the definition of the master and slave nodes. The ultimate and serviceability constraints also have to be activated.

The automatic constraint handling in the objective function inside the DE-algorithm eliminated the setting time of penalty factors. Usually there is no clear rule as to how penalty factor should be set and it is very time consuming to try different values of the penalty factors for each constraint.

4.4 Ridge roof

The third design case is that of a ridge roof truss. Two cases where investigated. All web bars are the same profile in the A-case. The web bars may be different in the B-case. The FE-models are half models, because the loading is assumed to be symmetrical. The symmetry line is denoted by CL. The K-joint on the symmetry line requires a symmetry K-joint constraint.

4.4.1 Structure A

Truss A is a ridge roof KT-structure presented in Figure 4.7. During the automated formulation of the optimisation model (third generation program), the designer constructs a preliminary FE-model of the steel beam structure. Certain fixed dimensions of the structure determine the positions of some of the nodes. The constraints and objectives are associated with the elements and nodes. The span and number of the cross members determine the x-coordinates of the nodes in the case of the ridge roof. Joint positions for every joint type are designated by selecting the corresponding node in the FE-model. Similarly the chords and braces of each joint are specified via the corresponding beams in the FE-model. Bending and buckling constraints of the selected beams are defined. The designer does not have to know in advance which beams are under compression or tension. Constraints are activated automatically during the optimisation depending on compression and tension force of the beam element. The deflection constraint is defined by limiting the z-deflection of the node 8 to a maximum value of *span*/300. Z-coordinates of nodes 8 and 9 are selected as design variables and the upper and lower limits of the z-coordinates are defined. When the nodes 1 and 8 are selected as master nodes and 2 to 7 are selected slave nodes, nodes 2 to 7 are forced to form a straight upper chord during optimisation. Positions of nodes 2 to 7 are interpolated between nodes 1 and 8. Z-coordinates of the nodes 10 to 12 follow the Z-coordinate of master node 9.

The profile tables for the upper chord, lower chord and brace are selected. An optimisation model is formulated automatically from this data. The designer selects the population size of the DE optimisation algorithm. 3D-graphics of the model and the optimisation results are monitored during the optimisation. Profiles are selected from the table presented in the Appendix 13 which contains 125 profiles.

In conventional optimisation (first generation program) a FE-model is separated from the optimisation model. Data transfer, e.g. node coordinates, profile cross section dimensions, material properties, forces, moments and deflections, between the optimisation model and the FE-model needed to be constructed individually for each optimisation case. First, a designer should construct the FE-model and write functions for positions of the nodes. Dimensions of the beam cross sections from a profile database and material properties would then be linked to the FE-model. Forces and moments of the elements and deflections of the nodes were linked back to the optimisation model considering the orientation of each element. This procedure has the disadvantage that dozens of insecure links between FE-model and optimisation model could be produced. The optimisation model should contain 1) local strength constraints of T-, K- and KT-joints, 2) stability and compression constraints of compressed beams, 3) tension constraints of tensed beams, 4) bending constraints of bended beams 5) deflection constraint of the node 8 and 6) the total mass of the steel beam structure.

4.4.2 Design variables

The optimisation parameters are presented in Table 4.9. Only rectangular hollow section profiles are considered.

Table 4.9 Optimisation parameters of the case A.

Design variable	Variable name	$x_i^l \le x_i \le x_i^u$
Upper chord	RHS$_1$	$1 \le x_1 \le 125$
Lower chord	RHS$_2$	$1 \le x_2 \le 125$
Brace	RHS$_3$	$1 \le x_3 \le 125$
Height	H_1	$0 \le x_4 \le 6000$
Height	H_2	$-4\ 000 \le x_5 \le -500$

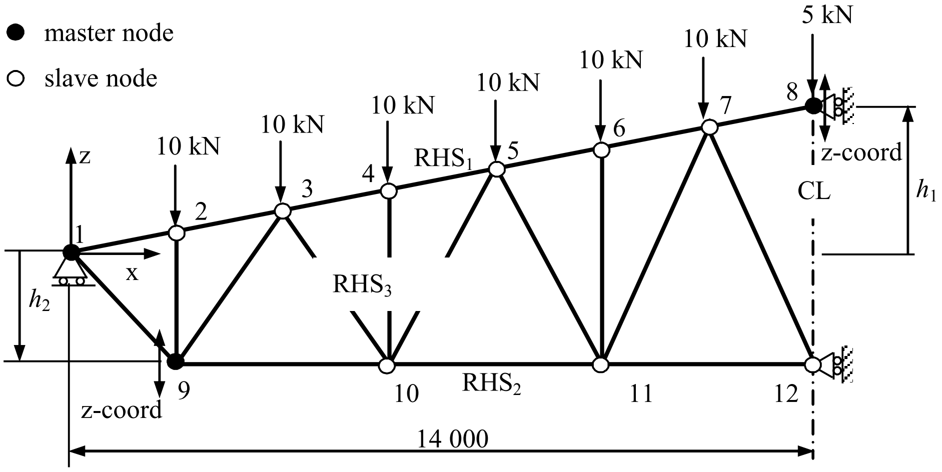


Figure 4.7 Ridge roof A.

4.4.3 Structure B

The truss B is a ridge roof KT-structure presented in Figure 4.8. The Z-coordinates of the nodes 8 and 9 (or heights h_1 and h_2) are design variables. Master nodes and slave nodes act in a fashion similar to case A. The upper and lower chord profiles and all web bars are design variables. The case B should be much more difficult than A-case, because there is much more active geometric constraints of the K- and KT-joints. T-joint constraints are calculated for nodes 1, 2, 4 and 6, K-joint constraints are calculated for nodes 3, 5, 7 and KT-joint constraints are calculated for nodes 9, 10 and 11. Lower chord and upper chord profiles are selected from the table as in case A and profiles of the braces are selected individually for every brace. This should make the optimisation case much more difficult because the geometric constraints K- and KT-joints will be activated with many profile combinations. This means, that the constraint handling method must first find some feasible solution or solutions before the real optimisation can be started.

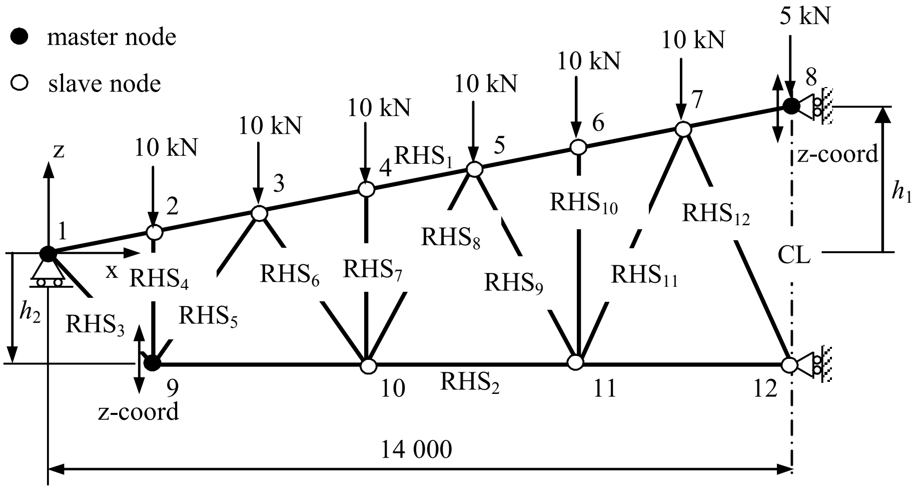


Figure 4.8 Ridge roof B.

4.4.4 Design variables

The optimisation parameters are presented in Table 4.10. Only rectangular hollow section profiles are considered. Lower chord and upper chord profiles are selected from the table like in the case A and profiles of the braces are selected individually for every brace. This should make the optimisation case much more difficult because the geometric constraints K- and KT-joints will be activated with many profile combinations. This means, that constraint handling method have to find first some feasible solution or solutions before the real optimisation can be started.

Table 4.10 Optimisation parameters of the case B.

Design variable	**Variable name**	$\mathbf{x_i^l \leq x_i \leq x_i^u}$
Upper chord	RHS$_1$	$1 \leq x_1 \leq 125$
Lower chord	RHS$_2$	$1 \leq x_2 \leq 125$
Brace	RHS$_3$	$1 \leq x_3 \leq 125$
Brace	RHS$_4$	$1 \leq x_4 \leq 125$
Brace	RHS$_5$	$1 \leq x_5 \leq 125$
Brace	RHS$_6$	$1 \leq x_6 \leq 125$
Brace	RHS$_7$	$1 \leq x_7 \leq 125$
Brace	RHS$_8$	$1 \leq x_8 \leq 125$
Brace	RHS$_9$	$1 \leq x_9 \leq 125$
Brace	RHS$_{10}$	$1 \leq x_{10} \leq 125$
Brace	RHS$_{11}$	$1 \leq x_{11} \leq 125$
Brace	RHS$_{12}$	$1 \leq x_{12} \leq 125$
Height	h_1	$0 \leq x_{13} \leq 6000$
Height	h_2	$-4000 \leq x_{14} \leq -1\ 000$

4.4.5 Constraints

Ultimate limit states for beam elements are tensile strength, compression strength, buckling strength and bending strength. The ultimate limit states for the T-, K- and KT-joints are: yielding of the top of the chord, buckling of the chord web, cutting of the chord, breakage of the main bracing, sharing of the chord.

The serviceability limit state constraint is the maximum deflection of the truss structure *L*/300.

There are total 336 inequality constraints in the optimisation model.

4.4.6 Objective

The minimised objective of the optimisation is the total mass of the truss structure.

$$m_{\text{tot}} = f(\mathbf{x}) = \sum_{i}^{n} m_{\text{elem}\,i} \tag{4.5}$$

4.4.7 Databases

Optimisation program uses databases presented in Appendix 13. Databases contain the cross sections of the telescopes. There are 125 possible rectangular hollow sections.

4.4.8 Results

The optimisation results of the case A are presented in Table 4.11 The corresponding objective function (total mass) is 528 kg. The number of function evaluations was 6 720.

The results of the case B are presented in Table 4.12. The corresponding objective function (total mass) is 483 kg. The number of function evaluations was 1 215 000.

The proposed solutions for the ridge roof truss are not necessarily here optimal. Additional tests would need to be performed to ensure that an optimal design is achieved. The thesis has put greater attention on the automated formulation method rather than the solving of a particular case. Ensuring the reliable optimal solutions would require very strict tests of the computer program. The constraint functions have been tested but the developed computer program is quite large and bugs in the structure of the program and dynamic memory handling can cause unexpected errors. Solving times were several hours because the FE-models were transferred to the FE-solver and results of FE solution were transferred back from FE-solver by text files. The dynamic link library (DLL) FE-solver was developed earlier during this study but it was not implemented on third generation program. DLL version of the FE-solver transfers the FE-model and results directly in the computer memory and optimisation times will be much shorter.

Table 4.11 Optimised design variables in the case A

Design variable	Variable name	Value	Profile
Upper chord	RHS_1	41	RHS 120x120x4
Lower chord	RHS_2	27	RHS 100x100x3
Brace	RHS_3	16	RHS 80x80x4
Height	h_1	1 751 mm	-
Height	h_2	-1 000 mm	-

Table 4.12 The optimised design variables in the case B.

Design variable	Variable name	Value	Profile
Upper chord	RHS_1	49	RHS 140x140x4
Lower chord	RHS_2	36	RHS 110x110x4
Brace	RHS_3	6	RHS 60x60x2.5
Brace	RHS_4	2	RHS 40x40x2
Brace	RHS_5	7	RHS 60x60x3
Brace	RHS_6	9	RHS 70x70x2
Brace	RHS_7	6	RHS 60x60x2.5
Brace	RHS_8	6	RHS 60x60x2.5
Brace	RHS_9	9	RHS 70x70x2
Brace	RHS_{10}	3	RHS 50x50x2
Brace	RHS_{11}	4	RHS 50x50x2.5
Brace	RHS_{12}	15	RHS 80x80x3
Height	h_1	588 mm	-
Height	h_2	-1 103 mm	-

The structure is statically determined and the member forces are independent from member sizes. T-, K- and KT-joint constraints and especially KT-joint constraints are very sensitive to the member sizes and therefore unexpected results can occur. Additional computer time may have produced slightly better solutions and the possibility of programming errors exists.

4.5 Summary

The optimisation portion of the program was first tested using eight standard test functions and optimisation problems. This included testing the differential evolution algorithm, constraint handling and also input and output vectors. This was done to ensure that the differential evolution algorithm performed properly. In all eight cases, the test values of the optimisation function and the design variables are identical or nearly identical to the best-known published values. The optimisations are not necessarily completed in the case of the ridge roofs and some further optimisation would perhaps be possible. Development of the automated formulation method was the most important aspect of this study. Solving times were several hours but it is expected that this could be much shorter with DLL version of the FE-solver, which transfers the FE-model and results directly in the computer memory.

The optimisation system, including optimisation algorithm, automated modelling routine and FE analysis programs, was then tested on several large steel structures optimisation problems. These structures were a hydraulically driven multi-redundant boom, a flat roof truss structure, and a ridge roof KT-structure. The progressive boom construction was selected as an optimisation case, because it was completely new concept and the structure of the best construction was uncertain. The optimisation of the steel structure of the boom showed that the preliminary supposition was right. Two steel structures were selected for test cases for third generation modelling and optimisation program, because these includes quite few elements but numerous geometric constraints, which make almost all solutions infeasible. This is a demanding challenge for the optimisation algorithm. The FE-model, which has constructed from 14-degree of freedom elements, is a challenge for modelling tool.

The first generation modelling and optimisation program was editable and flexible to use. It includes editable mathematical components, editable table components and linking components to external programs. In the case of steel structures, the optimisation model and FE-model need dozens or hundreds of links. In addition, the linking is not easy because each element has its own orientation and this complicates the reading of the FE-analysis results. Progress of the optimisation is uncertain because the FE-program does not monitor the FE-model during the optimisation. This program offers the possibility of multi criterion optimisation with six different multi-criteria weighting methods.

The external connection between optimisation program and FE-model was eliminated by developing a new program with a ready compiled FE-modelling tool with 3D monitoring. However, a ready-compiled FE-model is suitable for only one case and cannot be recycled for a new FE-model. This was a major motivation for developing an automated formulation.

An automated formulation of the optimisation model of the steel beam structures is the most powerful tool for optimisation of steel structures. The optimisation model is build up simultaneously with the FE-model. The user has to define some geometric constraints and optimisation parameters of the optimisation algorithm and the model is ready for solving. The FE-model, individuals, generations, fitness values and constraints are monitored during the optimisation. The user gets immediately feedback from the optimisation process and possible modifications to the optimisation model can be made quickly.

The Differential Evolution algorithm is easy to use. The population size is the only parameter that has to be changed for most optimisation cases. The new constraint handling method does not need the penalty factors or other penalty coefficients. This is a great advantage in optimisation of steel structures, because these can include hundreds of constraints.

5 DISCUSSION

In the early stages of this research project, after it was decided to link the FE analysis program to an optimisation algorithm, it was discovered that the development of a workable optimisation model was the most time consuming aspect of the entire optimisation process. For rather simple geometric cases, such as those found in textbooks, the modelling is rather straightforward. For real-life engineering structures, with perhaps dozens of elements and hundreds of constraints the task is formidable. Publications are full of solutions and applications considering "nice" optimisation models and optimisation results. The modelling process and the effort required to perform the modelling is seldom reported.

Optimisation is common in the automotive and aviation industries where the volume of the production is large or in the aerospace industry where extremely high costs for individual components are allowed. The vast majority of mechanical engineering work, however, has not implemented optimisation on a wide scale because of the lack of flexibility of most modelling tools. For many engineering companies in the Nordic regions, production runs are usually measured in hundreds of components per year. This demands that an optimisation tool cannot be designed for a specific type of component, but must flexible enough so as to be usable for a variety of structures. Smaller companies are often not aware or not interested in optimisation because there is no cheap and easy modelling and solving tools. In some cases the goal of the optimisation is only to find a feasible solution and not always, e.g., to minimize weight or maximize profit.

Optimisation methods can be employed in a variety of ordinary design situations to achieve practical and serviceable solutions. Engineers at all industrial companies are busy, and the optimisation modelling has to be easy and not time consuming. The AGIFAP Win FE-program is easy to use and relatively inexpensive when it compared to the large commercial FEM-packages. Optimisation linked to an easy-to-use FE-modelling tool should add the interest to the optimisation of the steel structures.

5.1 Modelling

5.1.1 Modelling tool

Three generations of modelling and optimisation programs were developed during the course of this research project. The first modelling tool focused to the modelling by means of components. The math components are flexible and editable. The ready component libraries could be readily used and the component were easy to copy and modify. The finite element analysis program was linked to the optimisation program by means of FE-components.

This modelling tool was suitable for small optimisation problems and could also be applied in larger more complex problems. However, some optimisation problems were found to be too time consuming and difficult to model with a reasonable degree of effort. The truss structures were one example. A truss structure includes hundreds of constraints and large amount of forces, moments and deflections that must be transferred between the FE-analysis package and the optimisation model. Mistakes during transfer are easy to make and difficult to notice and isolate.

Orientation of specific beam elements strongly affects the development of the optimisation model. Forces and moments depend on rotation angle and orientation of the beam element. For example, the very common beam element shown in Figure 5.1 has 12 degrees of freedom. The nodes have to move to new positions during the optimisation if the node coordinates are chosen as design variables.

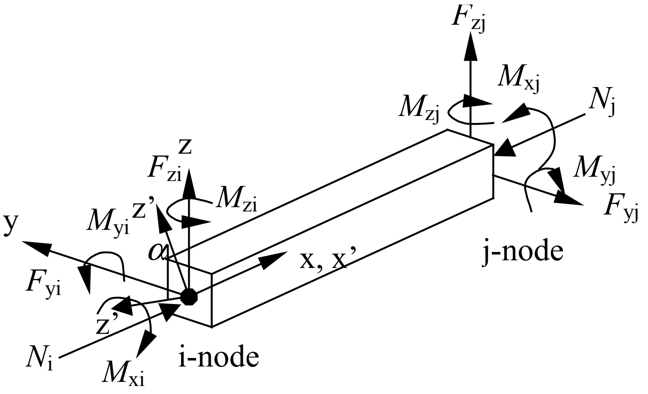


Figure 5.1 The orientation and rotation of the beam element. Coordinates x', y' and z' are the local coordinates of the element. Angle α is a rotation angle of the element.

Figure 5.2 illustrates the concept of slave nodes and how they are programmed to follow the master nodes in the finite element model. In this case, parametric formulas the coordinates of every node are needed.

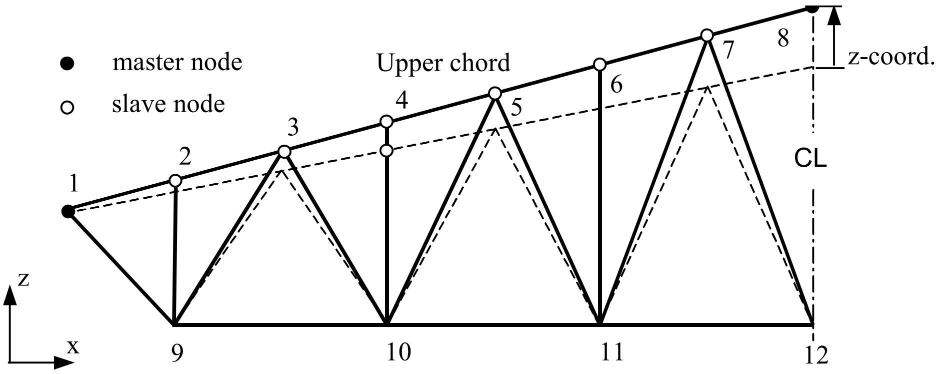


Figure 5.2 Slave nodes follow the master nodes.

5.1.2 Finite element model with optimisation model

The third and final program developed in this project has focused primarily on the automatic formulation of the optimisation model during FE-modelling. The FEM-program was modified to build the optimisation input and output vectors simultaneously with the FE-modelling. A FE-model is constructed in a conventional manner. Elements and nodes contain objective and constraint functions. Constraint functions are deflection, tension, compression, bending, fatigue, hydraulic cylinder, buckling and strengths of the T-, K- and KT-joints. The user activates the objectives, constraints and design variables during the FE modelling process. Design variables can be, for example, the coordinates of the nodes and the cross section properties of the beam elements. Automation significantly reduced the modelling time and errors. A designer does not have to write constraints, objective functions and functions of the node coordinates. All selections are made with a graphical interface of the computer program. The designer can also select tables where the beam dimensions are selected during the optimisation.

In this third-generation program, the formulation of an optimisation model that could be linked to FE analysis was both fast and easy to accomplish. This program version contained significant improvements over the second-generation program because the constraint and objective functions in the second generation were pre-compiled which caused a degree of inflexibility. Pre-compiled functions cannot be modified by the user. The program could be extended by developing more constraint and objective functions.

5.2 Optimisation

The optimisation algorithm chosen for this program was the differential evolution DE algorithm. DE-algorithm requires only a few parameters and can be easily modified by the user as required. The three parameters are: the size of the population, the differential factor and the mutation probability. A suitable differential factor can be fixed to 0.8 while a mutation probability of 0.95 is very suitable. These values have been confirmed by numerous tests made with the differential evolution algorithm (Lampinen, J. 2000). This means that population size is, in practice, the only that the user must give special attention to.

The third generation program was modified so that unnecessary FE-analyses could be avoided. FE-analysis typically takes the majority of the analysis time when objective functions are being evaluated. Most of the constraints for the optimisation model of a steel structure are geometric. A FE-analysis is useless if one of these constraints is violated. Avoiding FE analysis of models that violate one or mode of the geometric constraints greatly accelerated the optimisation process.

5.3 Optimisation of the steel structures

Two of the selected real-life test structures were plane trusses. Evaluation of 3D-structures is also possible without modifications of the program. It was judged that the optimisation of a space truss would only provide a small degree of extra information about the developed method.

Different roof structures were optimised by the developed FE-optimisation modelling tool. Optimisation models were built simultaneously with the FE-modelling. The operator activated various constraints. The designer must have sufficient experience to know generally the type of structure needed, i.e., where the K- and KT-joints should be located and that beams must be selected to connect the joints. The optimisation program itself, of course, determines the final location of the joint and the best beam profile. The selection of beams is important because other elements can join to the same node, for example with 3D-structures.

The roles of various load combinations on a structure are not within the scope of this thesis. It is assumed that the most severe load case has been defined before the optimisation of the structure. The most dangerous load combination includes the partial safety factors of the load. The partial material and joint safety factors are included to the constraints of the optimisation model. Constraint functions have some default values, which the user can modify before the optimisation.

Geometric constraints of the FE-geometry are important. This means that each time one node position changes, it may be needed to also update other node coordinates. This can be made using metric node coordinates, but this is time consuming and additionally there is a risk of the programming errors.

5.3.1 Multi-redundant boom

The optimisation of a multi-redundant boom structure was also performed as part of this research. This was done in conjunction with the Laboratory of Virtual Engineering and Mechatronics who had responsibility for developing the new boom construction and its control system. The boom is a good representative of an optimised structure because booms of this type must be functional, but also light and rigid. The boom was a new product concept and the construction was developed during the past four years.

The optimisation analysis made for the boom construction shows that the assumptions about the construction were generally correct. Other possible constructions were not optimal and therefore rejected.

The boom construction has a geometric constraint, which is difficult to take into account in the FE-optimisation modelling tool. The shape of the base plates was difficult to pre-program to the FE-program and a flexible math-component was needed. In this case, the boom structure was optimised by the intermediate phase program. The components were compiled and the shape of the boom was found without great difficulty. The coordinates of the nodes were parametric functions of the height, width and length of the boom.

5.3.2 Ridge roof B

The ridge roof case B was the most difficult optimisation problem. Each diagonal brace member was selected individually from the cross section table that contained 125 alternatives. Geometry of the K- and KT-joints were strictly constrained by the dimension ratios of the joining members. The selection of the sufficient diagonals was time consuming. The FE-analysis was not performed if the one of the geometric constraints was violated, thus saving significant computer time.

6 CONCLUSIONS

The main goal of this research project was to investigate the possibilities for automating the modelling of truss type structures in an optimisation program. During this thesis project, three modelling and optimisation programs were developed and tested with practical mechanical engineering optimisation problems. The final program made use of the observed limitations from earlier program versions. The optimisation program combines a modern evolution based optimisation algorithm, an automated modelling tool that greatly reduces the effort between different model generations, and a finite element based analysis tool.

During the optimisation of real-life engineering structures, modelling was found to be the most time consuming phase in the optimisation. This naturally led to the development of an automated means for formulating the optimisation model. In total, three generations of modelling tools were developed, each generation included improvements based on experienced gained with previous generations. The first tool was flexible but was difficult to use when large optimisation models needed to be formulated. The third modelling tool was integrated in the FE-modelling program so that the user constructs a model suitable for optimisation during the process of constructing the preliminary FE-model. Constraints and objective functions are created automatically and require only minimal user input. A FE-model is constructed in a conventional manner. Elements and nodes contain objective and constraint functions. Constraint functions are deflection, tension, compression, bending, fatigue, hydraulic cylinder, buckling and strengths of the T-, K- and KT-joints. The designer activates the objectives, constraints and design variables during the FE modelling process. The members of the T-, K- and KT-joins have to be selected by the designer.

The user must also to select the interpolated slave nodes and master nodes so that multi-element beams remain straight. Coordinates of the nodes are continuous design variables and measures of the cross sections of the beam elements are discrete variables. The designer has to select tables where the discrete beam dimensions are selected during the optimisation. Default values of the constraints like safety factors can be changed by the user. The designer also has to select tables from which the beam dimensions and material properties are chosen during the optimisation. All selections are made with the aid of the graphical interface of the computer program. This automation reduces significantly the modelling time and errors. Constraints are not editable in this version modern but limit state design concepts were used in the formulation of the constraint classes.

Three main objectives were considered when developing the automated modelling tool: flexibility, speed and reliability. The final tool is a balanced compromise of objectives. Reliability and speed are high if the constraints are compiled, but flexibility is then low minimum. High flexibility results in both poor reliability and low speed because a complex FE-model needs to be manually linked to the optimisation model. This is a very time consuming process and mistakes are common.

The differential evolution algorithm used in this program can be used nearly as a black box. In practice, the user has to select only the population size. This value can be selected after only a few tests. The third modelling tool was tested with truss structures, which include from 200 to 300 constraints. Penalty functions are not needed in the final variation of the differential evolution algorithm used here.

The optimisation algorithm is tested using common test functions used for the evolution algorithms and was found to perform very well with only minor differences between the values obtained here and published ideal values.

The optimisation system, including optimisation algorithm, automated modelling routine and FE analysis programs, was then tested on several large steel structures optimisation problems. These structures were a hydraulically driven multi-redundant boom, a flat roof truss structure, and a ridge roof KT-structure. The progressive boom construction was selected as an optimisation case, because it was completely new concept and the structure of the best construction was uncertain. The optimisation of the steel structure of the boom showed that the preliminary supposition was right. Two steel structures were selected for test cases for third generation modelling and optimisation program, because these includes quite few elements but numerous geometric constraints, which make almost all solutions infeasible. This is a demanding challenge for the optimisation algorithm. The FE-model, which has constructed from 14-degree of freedom elements, is a challenge for modelling tool. Especially with case B of the ridge roof truss, the optimisation model was easy to form. This problem was very complex in terms of element possibilities and constraints and more computer time would be required to achieve a fully optimal solution.

6.1 Recommendations for further work

The goal of this project was not to develop a commercial optimisation program, but to demonstrate the concept and usefulness of automated optimisation model formulation for steel structures. A professional software developer could probably suggest numeous improvements that would result in a more efficient code. The libraries of joint types and profiles, for example, could also be expanded to make the optimisation tool more useable in a variety of design tasks. The current FE-model was written for FE-solver using text files. A future version could make use of DLL-functions which is a much more rapid method of data transmission.

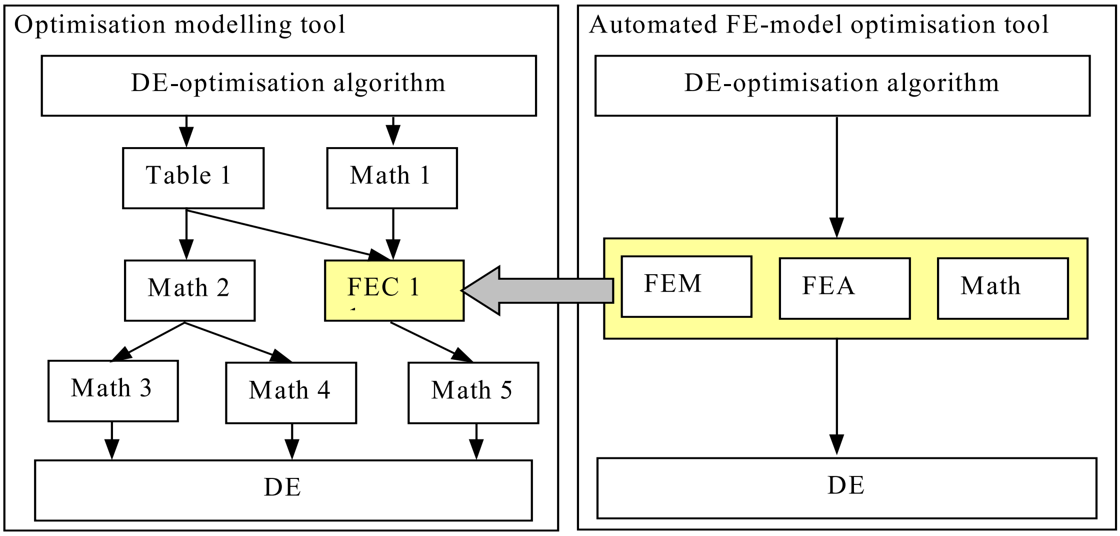

Optimisation modelling tool
DE-optimisation algorithm
Table 1
Math 1
Math 2
FEC 1
Math 3
Math 4
Math 5
DE
Automated FE-model optimisation tool
DE-optimisation algorithm
FEM
FEA
Math
DE

Figure 6.1 The linking of the optimisation modelling tool and the automated FE-optimisation tool.

The flexibility and reliability of the entire optimisation modelling tool could probably be improved by combining some features from the first-generation tool with the final program. The left side of Figure 6.1 shows the structure of the first generation program while the right side of this figure presents the third generation program. Flexibility could be gained by formulating the third generation program as one element in the first generation program. It is expected that this process would preserve the program reliability. The user can link the variable vector of the FE-component to the flexible math components and write specialised objective or constraint functions. Very complex user defined geometric constraints are possible because the user can link the math component to the design variable vector of the FE-model.

REFERENCES

Adeli, H. 1994. Advances in Design Optimization. London: Chapman & Hall. 573 p.

AGIFAP Version 5.51 User Guide. 1995. Lappeenranta: Lappeenranta University of Technology. 84 p.

B7 1996. Ympäristöministeriö, Asunto- ja rakennusosasto 1996. Finnish code of building regulations B7. 51 p. (in Finnish)

Coello, C. 2000. Treating constraints as objectives for single-objective evolutionary otpimization. In: Engineering Optimization Vol. 32 pp. 275-308.

Cook, R. D. 1981. Concepts and Applications of Finite Element Analysis. Second Edition. New York: John Wiley & Sons. 537 p.

Dowling, J. D., Knowles, P. R. and Owens, G. W. 1988. Structural Steel Design. London: Butterworths. 399 p.

Dulikravich, G. S., Martin, T. J., Dennis, B.H. and Foster, N.F. 1999. Multidisciplinary Hybrid Constrained GA Optimization. In: Proceedings of EUROGEN99 - Short Course on Evolutionary Algorithms in Engineering and Computer Science. Reports of the Department of Mathematical Information Technoloty Series A. Collections No. A 2/1999. University of Jyväskylä. pp. 233 - 259.

Erbatur, F., Hasancebi, O., Tutuncu, I. and Kilic, H. 2000. Optimal design of planar and space structures with genetic algorithms. In: Computers and structures 75, Elsevier Science Ltd. pp. 209-224.

Eschenauer, H., Koski, J., Osyczka, A.1990. Multicriteria Design Optimization. Berlin: Springer-Verlag. 482 p.

Eskelinen, H. 1999. Tuning the design proceures for laser processed microwave mechanics. Acta Universitatis Lappeenrantaensis 87. Diss. Lappeenranta University of Technology. 172 p.

Farkas, J. & Jármai, K. 1997. Analysis and Optimum Design of Metal Structures. Rotterdam: A.A. Balkema. 347 p.

Farkas, J. 1984. Optimum Design of Metal Structures. New York: Horwood Halsted Press. 222 p.

Floudas, C. A. and Pardalos, P.M. 1987. A Collection of Test Problems for Constrainded Global Optimization Algorithms. Lecture Notes in Computer Science, vol. 455. Heidelberg: Springer-Verlag.

Fuchs, M. B. and Farhi, E. F. 2001. Shape of tiffest controlled structures under unknown loads. In: Computers and structures 79. pp. 1661-1670.

Gil, L. and Andreu, A. 2001. Shape and cross-section optimisation of a truss structure. In: Computers and Structures 79. pp. 681-689.

Gill, P., Murray, W., Wright, M. 1981. Practical optimization. London: Academic Press. 401 p.

Giraud-Moreau, L. and Lafon, P. 2002. A comparison of evolutionary algorithms for mechanical design components. In: Engineering Opitmization. Vol. 34. Taylor&Francis. pp. 307-322.

Goldberg, D.E. 1989. Genetic Algorithms in Search. In: Optimisation and Machine Learning. Addison-Wesley. 412 p.

Gorlen, K. E., Orlow, S. M. and Plexico, P. S. 1990. Data abstraction and object-oriented programming in C++. Chichester: Wiley. 402 p.

Haataja, J. 1995. Solving of the optimisation problem. Espoo: CSC - Tieteellinen laskenta Oy. 2. painos. 232 p. (in Finnish)

Hayalioglu, M. S. 2000. Optimum design of geometrically non-linear elastic-plastic steel frames via genetic algorithm. In: Computers and Structures 77. pp. 527-538.

Himmelblau, D. 1972. Applied Nonlinear Programming. New York: McGraw-Hill.

Hobbacher, A. 1996. Fatigue Design of Welded Joints and Components: Recommendations of IIW Joint Working Group XIII-XV/A. IIW Doc. XIII-1539-96/XV-845-96. Cambridge: Abington. 127 p.

Hock, W. and Schittkowski, K. 1981. Test Examples for Nonlinear Programming Codes. Lecture Notes in Economics and Mathematical Systems, vol. 187. Heidelberg: Springer-Verlag.

Ignizio, J. 1999. Illusions of Optimality. In: Engineering Optimization No 6. pp. 749-761.

Innocenti, C., Parenti-Castelli, V. 1993. Echelon form solution of direct kinematics for the general fully parallel spherical wrist. In: Mech. Mach. Theory. Vol. 28, No 4, pp. 553-561.

Jármai, K. 1989. Single- and multicriteria optimization as a tool of decision support system. In: Computers in Industry, Elsevier Applied Science Publishers, Vol. 11, No. 3 pp. 249-266.

Jármai, K. and Farkas, J. 1999. Cost calculation and optimisation of welded steel structures. In: Journal of Constructional Steel Research. Elsevier. pp. 115-135.

Kameshki, E. S. and Saka, M. P. 2001. Optimum design of nonlinear steel frames with semi-rigid connections using a genetic algorithm. In: Computers and Structures 79. Pergamon. pp. 1593-1604.

Kilkki, J. 2002. Formulation of the optimisation model for the differential evolution optimisation algorithm. In: Koneensuunnittelun XXVII kansallinen symposiumi LTKK:ssa 6. ja 7. kesäkuuta 2002. Raportti 12. Konetekniikan osasto, Lappeenrannan teknillinen korkeakoulu, Finland. pp. 77-81. (in Finnish)

Kilkki, J., Wu, H. and Handroos, H. 2002. Applying the Differential Evolution Algorithm to the Optimisation of Redundant Parallel Manipulator (2001). In: Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems. Edited by Giannakoglou, K. C. et al., Proceedings of the EUROGEN2001 Conference, Athens, Greece, September 19-21, 2001. Barcelona, Spain: Artes Graficas Torres S.A. pp. 223-228.

Kilkki, J., Lampinen, J. and Martikka, H. 2001. Applying the differential evolution algorithm to the optimisation of cross sections of steel columns. In: Computer aided optimum design of structures VII, Edited by Hernandez, S. & Brebbia, C. A. UK, Southmapton: Wit Press. pp. 63-72.

Kilkki, J. 2001. Modernisation of the AGIFAP. In: Hitsattujen Rakenteiden Optimointi -foorumin Teemapäivät 2001, Toimittanut Poutiainen, I. Raportti 11. Konetekniikan osasto, Lappeenrannan teknillinen korkeakoulu, Finland. pp. 143-149. (in Finnish)

Kilkki, J. 2000. Optimisation of the metal structures by evolution algorithms. In: Viidennet teräsrakenteiden tutkimus- ja kehityspäivät 18.-19.1.2000. Toim. Kesti, J. ja Mäkeläinen, P. TKK-TER-13. Espoo, Finland. pp. 237-244. (in Finnish)

Koziel, S. and Michalewicz, Z. 1999. Evolutionary Algorithms, Homorphous Mappings, and Constrained Parameter Optimization. In: Evolutionary Computation 7(1).

Lagstedt, A., Kilkki, J. and Handroos, H. 1999a. The flexibility behaviour of a spatial parallel mechanism. Research report 28. Department of Mechanical Engineering, Lappeenranta University of Technology, Lappeenranta, Finland. 37 p.

Lagstedt, A., Kilkki, J. and Handroos, H. 1999b. Spatial parallel mechanism based on new generation working machine boom. In: Proceedings of the 4th JHPS International Symposium of Fluid Power, Ed. S.Yokota. Tokyo, 15-17, Nov, pp. 513-518.

Lampinen, J. 2002. A Constraint Handling Approach for the Differential Evolution Algorithm. In: The 2002 IEEE World Congress on Computational Intelligence - WCCI 2002, CEC2002, Honolulu, Hawaii, May 12-17, 2002. pp. 1468 - 1473.

Lampinen, J. and Zelinka, I. 2000. On stagnation of the differential evolution algorithm. In: Osmera, P. (ed.). Proceedings of MENDEL 2000, 6th International Conference on Soft Computin, June 7.-9. 2000. Brno University of Technology, Szech Republic, pp.76-83.

Lampinen, J. and Zelinka, I. 1999a. Mechanical Engineering Design Optimization by Differential Evolution. In: Advanced topics in computer science series. Edited by Corne, D, Dorigo, M. and Glover Fred. pp. 127-146.

Lampinen, J. 1999b. Differential evolution - new naturally parallel approach for engineering design optimization. In: Developments in computational mechanics with high performance computing. Edited by Topping B. H. V. Civil-comp press. Edinburgh, UK. pp. 217-228.

Lippman, S. B. 2000. C++, Vol 1. Jyväskylä: Gummerus Kirjapaino Oy. 1209 p.

Lämmer, L., Meissner, U. and Petersen, M. 2001. Object-oriented integration of construction and simulation models. In: Computers and Structrues 79. p. 2143-2149.

MacGinley, T. J. and Ang, T. C. 1987. Structural Steelwork: Design to Limit State Theory. London: Butterworths. 359 p.

Manickarajah D., Xie Y. M. and Steven G. P. 2000. Optimum design of frames with multiple constraints using an evolutionary method. In: Computers and Structures 74. pp. 731-741.

Mannesmann Rexroth 1996. Hydraulic cylinders and accessories. Hydraulic catalogues on the range of products supplied by the Mannesmann Rexroth group companies. Lohr am Main: Mannesmann Rexroth GmbH.

Meriläinen, V. 2000. Application of discrete optimisation to virtual design of the mechatronic machine. Diplomityö. Lappeenrannan teknillinen korkeakoulu, Finland. 62 p. (in Finnish)

Michalewicz, Z., Deb, K., Schmidt, M. and Stidsen, T. 1999. Evolutionary Algorithms for Engineering Applications. In: Evolutionary algorithms in engineering and computer science. John Wiley & Sons, Ltd. pp. 73 - 94.

Miettinen, K., Mäkelä, M. and Toivanen, J. 1999. Proceedings of EUROGEN99 - Short Course on Evolutionary Algorithms in Engineering and Computer Science. Reports of the Department of Mathematical Information Technoloty Series A. Collections No. A 2/1999. University of Jyväskylä. 187 p.

Miettinen, K., Mäkelä, M., Toivanen, J. and Périaux, J. 1999. Evolutionary algorithms in engineering and computer science. Chichester: John Wiley & Sons, Ltd. 483 p.

Mikkola, A. 1997. Studies on Fatigue Damage in a Hydraulically driven boom system using virtual prototype simulations. Research papers 61. Diss. Lappeenranta University of Technology, Finland. 80 p.

Miles, J. C., Sisk, G. M. and Moore, C. J. 1999. The conceptual design of commercial buildings using a genetic algorithm. In: Computers and Structures 79. Pergamon. pp. 1583-1592.

Ohsaki, M., Fujisawa, K., Katoh, N. and Kanno, Y. 1999. Semi-definite programming for topology optimisation of trusses under multiple eigenvalue constraints. In: Comput. Methods Appl. Mech. Engrg. 180. pp. 203-217.

Pahl, G., and Beitz, W. 1996. Engineering Design, a systematic approach. London: Springer. 544 p.

Price, K. V. 1999. An Introduction to Differential Evolution. New Ideas in Optimization, Editedt by Corne, D., Dorigo, M. and Glover, F. New York: McGraw-Hill. pp. 79-108.

Rafiq, M.Y. 1988. Genetic algorithms in optimal design and detailing of reinforced concrete biaxial columns supported by a declarative approach for cpacity checking. In: Computers and Structures 69, Pergamon Press. pp. 443-457.

Rao, S. S. 1978. Optimization theory and applications. Indian Institute of tecnology, Kanpur: Wiley Eastern Limited.

Rautaruukki steel products, Designer's guide. Keuruu: Otava. 1998. 345 p.

Rong, J. H., Xie, Y. M. and Yang, X.Y. 2000. An improved method for evolutionary structural optimisation against buckling. In: Computers and Structures 79. pp. 253-263.

Rozvany, G. I. N. 1992. Shape and layout optimization of structural systems and optimality criteria methods. Courses and lectures - No. 325. Wien: Springer-Verlag. 496 p.

Salajegheh, E. 2000. Optimum design of steel space frames with frequency constraints using three point Rayleigh quotient approximation. In: Journal of Constructional Steel Research 54. Elsevier. pp. 305-313.

Sarma, K. and Adeli, H. 2000. Cost optimization of steel structures. In: Engineering Optimization, Vol. 32, pp. 777-802.

Shrestha, S. M. and Ghaboussi J. 1998. Evolution of Optimum Structural Shapes Using Genetic Algorithm. In: Journal of Structural Engineering. pp. 1331 - 1338.

Siddall, J. N. 1982. Optimal Engineering Design, Principles and applications. New York: Marcel Dekker Inc. 523 p.

Sopanen, J. and Mikkola, A. 2000. Study of sub-harmonic vibration of a tube roll using simulation model. In: 2nd Triennial International Symposium on Multi-body Dynamics: Monitoring & Simulation Techniques, Bradford, UK. June 26 - 27, 2000. pp. 57-68.

Standard SFS 2378. Welding. Load capacity of welded joints in fatigue loaded steel structures. Federation of the Finnish Metal and Engineering Industries, Standards Department. 1992. 51 p.

Stocki, R., Kolanek, K., Jendo, S. and Kleiber M. 2001. Study on discrete optimization techniques in reliability-based optimisation of truss structures. In: Compures and Structures 79. Pergamon. pp. 2235-2247.

Storn, R. and Price, K. 1995. Differential Evolution - A simple and efficient adaptive scheme for global optimization over continuous spaces. TR-95-012. ftp.icsi.berkeley.edu/pub/techreports/1995/tr-95-012.ps.gz

Taipale, J. 1999. Economic optimisation of the structures. Teknillisen korkeakoulun mekaniikan laboratorion julkaisuja. Espoo, Finland. 154 p.

Takada, T., Kohama, Y. and Miyamura, A. 2001. Optimization of shear wall allocation in 3D frames by branch-and-bound method. In: Computer aided optimum design of structures VII, Edited by S. Hernandez & C.A. Brebbia. Wit Press. pp. 63-72.

Tanskanen, P. 2000. The evolutionary structural optimization (ESO) method: Theoretical aspects and the modified evolutionary structural optimization (MESO) method. Diss. Lappeenranta University of Technology. 67 p.

Templeman, A. and Blachut, J. 1997. Engineering Optimization. Special Issue. Euromech Colloquium 345. The Future of Structural Optimization. Gordon and Breach Science Publishers.

Tong, W. H. 2000. An optimization procedure for truss structures with discrete design variables and dynamic constraints. In: Computers and Structures 79. pp. 155-162.

Viitanen, S. 1997. Some New Global Optimization Algorithms. Diss. Department of Computer Science. Åbo Akademi University. 106 p.

Wu, H. 2002. Analysis, Design and Control of a hydraulically driven parallel robot manipulator. Acta Universitatis Lappeenrantaensis 88. Diss. Lappeenranta University of Technology. 80 p.

Wu, H., Handroos, H. and Kilkki, J. 2000. On the design of a large scale redundant parallel manipulator. In: 17th International Symposium on Automation and Robotics in Construction, Taipei, Taiwan. pp. 545-550.

Yu, L. 2000. An object-oriented modular framework for implementing the finite element method. In: Computers & Structures 79, Elsevier Science Ltd. pp. 919-928.

Table 1 Yield strength of the hot rolled steel plate (B7 1996).

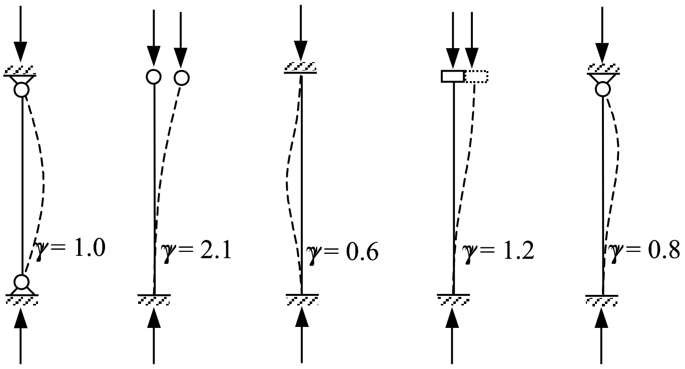
Thickness (mm)	R_{eh} **N/mm^2 (Hot rolled steel plates)**		
	S235	**S275**	**S355**
t ≤ 16	235	275	355
16 < t ≤ 40	225	265	345
40 < t ≤ 63	215	255	335
63 < t ≤ 80	215	245	325
80 < t ≤ 100	215	235	315
100 < t ≤ 150	215	225	295
150 < t ≤ 200	195	215	285
200 < t ≤ 250	175	205	275

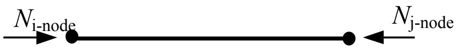
Table 2 Yield strength of the hot rolled steel plate (B7 1996).

Thickness (mm)	R_{eh} **N/mm^2 (Hot rolled steel plates)**		
	S275	**S355**	**S420**
t ≤ 16	275	355	420
16 < t ≤ 40	265	345	400
40 < t ≤ 63	255	335	390
63 < t ≤ 80	245	325	370
80 < t ≤ 100	235	315	360
100 < t ≤ 150	225	295	340

APPENDIX 2 THE BUCKLING RESISTANCE OF A COLUMN

The buckling resistance of a column (B7 1996).





$$g(\mathbf{x})_1 = N_{\mathrm{Rc}} - N \geq 0$$

$$g(\mathbf{x})_2 = 3.5 - \overline{\lambda_{\mathrm{k}}} \geq 0$$

$$N_{\mathrm{Rc}} = A \cdot f_{\mathrm{ck}}$$

$$f_{\mathrm{ck}} = \begin{cases} \left(\beta - \sqrt{\beta^2 - 1/\,\overline{\lambda}_{\mathrm{k}}^2}\right) \cdot f_{\mathrm{y}} & \text{when} \quad \overline{\lambda}_{\mathrm{k}} > 0.2 \\ f_{\mathrm{y}} \quad \text{when} \quad \overline{\lambda}_{\mathrm{k}} \leq 0.2 \end{cases}$$

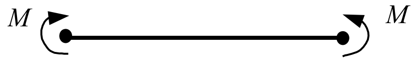
$$\beta = \frac{1 + \alpha\left(\overline{\lambda}_{\mathrm{k}} - 0.2\right) + \overline{\lambda}_{\mathrm{k}}^2}{2\overline{\lambda}_{\mathrm{k}}^2}$$

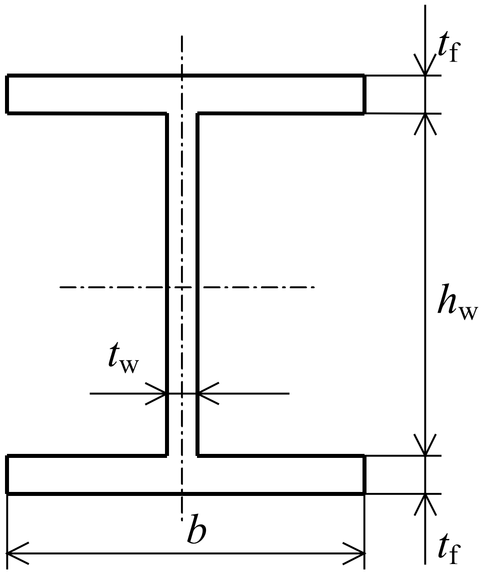
$$\overline{\lambda}_{\mathrm{k}} = \frac{L_{\mathrm{c}}}{i \cdot \pi} \cdot \sqrt{f_{\mathrm{y}}/E}$$

$$L_{\mathrm{c}} = L \cdot \gamma$$

Factor α depends on cross section and residual stresses.

The bending resistance is calculated as follows in the case of the double symmetric box beam or I-beam in direction of the z-axis ($I_y > I_z$) (B7 1996).





$$g(\mathbf{x})_1 = M_R - \left|M_{i-node}\right| \geq 0$$
$$g(\mathbf{x})_2 = M_R - \left|M_{j-node}\right| \geq 0$$

$$h = 2 \cdot t_f + h_w$$
$$b_f = \tfrac{1}{2}(b - t_w)$$
$$A = b \cdot h - b_f \cdot h_w$$
$$e_z = \tfrac{1}{2} h$$
$$e_f = \tfrac{1}{2}(h_w + t_f)$$
$$I_z = \tfrac{1}{12}\left(b \cdot h^3 \cdot b_f \cdot h_w{}^3\right)$$
$$W = \frac{b \cdot h^3 - b_f \cdot h_w{}^3}{6 \cdot h}$$
$$W_p = 2 \cdot b \cdot t_f \cdot e_f + \tfrac{1}{4} h_w{}^2 \cdot t_w$$

Design strength:

$$f_d = \min\{f_{dw},\ f_{df}\}$$

$$N_\mathrm{p} = f_\mathrm{y} \cdot A$$

The cross-section class of the web:

$$PL_\mathrm{w} = \begin{cases} 1, \text{ when } \begin{cases} \dfrac{h_\mathrm{w}}{t_\mathrm{w}} \leq 2{,}40 \cdot \left(1 - 1{,}40\dfrac{N}{N_\mathrm{p}}\right) \cdot \sqrt{\dfrac{E}{f_\mathrm{yw}}} \text{ and } \dfrac{N}{N_\mathrm{p}} < 0{,}39 \\ \dfrac{h_\mathrm{w}}{t_\mathrm{w}} \leq 1{,}10 \cdot \sqrt{\dfrac{E}{f_\mathrm{yw}}} \text{ and } \dfrac{N}{N_\mathrm{p}} \geq 0{,}39 \end{cases} \\ 2, \text{ when } \begin{cases} \dfrac{h_\mathrm{w}}{t_\mathrm{w}} \leq 3{,}00 \cdot \left(1 - 1{,}60\dfrac{N}{N_\mathrm{p}}\right) \cdot \sqrt{\dfrac{E}{f_\mathrm{yw}}} \text{ and } \dfrac{N}{N_\mathrm{p}} < 0{,}125 \\ \dfrac{h_\mathrm{w}}{t_\mathrm{w}} \leq 2{,}57 \cdot \left(1 - 0{,}53\dfrac{N}{N_\mathrm{p}}\right) \cdot \sqrt{\dfrac{E}{f_\mathrm{yw}}} \text{ and } \dfrac{N}{N_\mathrm{p}} \geq 0{,}125 \end{cases} \\ 4, \text{ otherwise} \end{cases}$$

The cross-section class of the flange:

$$PL_\mathrm{f} = \begin{cases} 4, \; jos \; \dfrac{b_\mathrm{f}}{t_\mathrm{f}} > 0.44 \sqrt{\dfrac{E}{f_\mathrm{yf}}} \\ 3, \; jos \; \dfrac{b_\mathrm{f}}{t_\mathrm{f}} \leq 0.44 \sqrt{\dfrac{E}{f_\mathrm{yf}}} \\ 2, \; jos \; \dfrac{b_\mathrm{f}}{t_\mathrm{f}} \leq 0.36 \sqrt{\dfrac{E}{f_\mathrm{yf}}} \\ 1, \; jos \; \dfrac{b_\mathrm{f}}{t_\mathrm{f}} \leq 0.30 \sqrt{\dfrac{E}{f_\mathrm{yf}}} \end{cases}$$

The cross-section class of the beam:

$$PL = \max\left\{PL_\mathrm{f},\; PL_\mathrm{w}\right\}$$

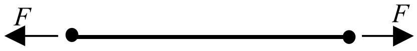
The bending resistance of the beam:

$$\eta = \begin{cases} \dfrac{W_\mathrm{e}}{W}, & \text{when } \; PL = 4 \\ 1, & \text{when } \; PL = 3 \\ \dfrac{W_\mathrm{p}}{W}, & \text{when } \; PL = \{1,\ 2\} \end{cases}$$

have to be $\eta \leq 1.2$

$$M_\mathrm{R} = \eta \cdot f_\mathrm{d} \cdot W$$

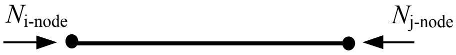
Tension resistance (Rautaruukki steel products designers guide 1998).



$$g(\mathbf{x})_1 = N_{\mathrm{Rt}} - N \geq 0$$

$$N_{\mathrm{Rt}} = f_{\mathrm{d}} \cdot A$$

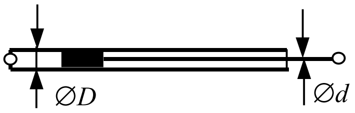
Compression resistance (Rautaruukki steel products designers guide 1998).



$$g(\mathbf{x})_1 = N_{\text{Rt}} + N \geq 0$$

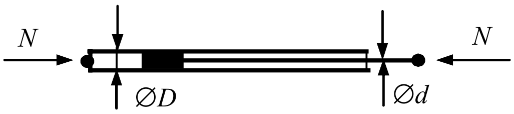
$$N_{\text{Rt}} = f_{\text{d}} \cdot A$$

The table of the hydraulic cylinders (Mannesmann Rexroth 1996).



ID	D	d
1	25	12
2	25	18
3	32	14
4	32	22
5	40	18
6	40	28
7	50	22
8	50	36
9	63	28
10	63	45
11	80	36
12	80	56
13	100	45
14	100	70
15	125	56
16	125	90
17	160	70
18	160	110
19	200	90
20	200	140

The buckling of the hydraulic cylinder and the adequate force (Mannesmann Rexroth 1996).



$$N_\mathrm{R} = \begin{cases} \dfrac{\pi^2 \cdot E}{3.5 \cdot l^2} \cdot \dfrac{\pi \cdot d^4}{64}, & \text{when} \quad \lambda_\mathrm{g} > \lambda \\ \\ \dfrac{d^2 \cdot \pi \cdot (314 - \lambda)}{14}, & \text{when} \quad \lambda_\mathrm{g} \le \lambda \end{cases}$$

where

$\lambda = 4l/d$ and $\lambda_\mathrm{g} = \pi\sqrt{E/f_\mathrm{y}}$

The forces of the hydraulic cylinders:

$$F_\mathrm{c} = 0.25 \cdot D^2 \cdot p \cdot \eta$$
$$F_\mathrm{t} = 0.25 \cdot \left(D^2 - d^2\right) \cdot p \cdot \eta$$

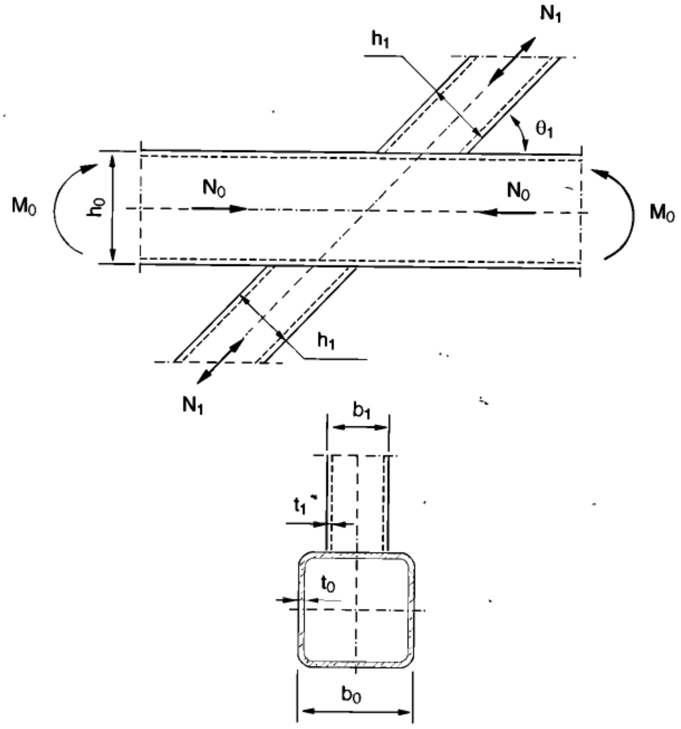
The constraints:

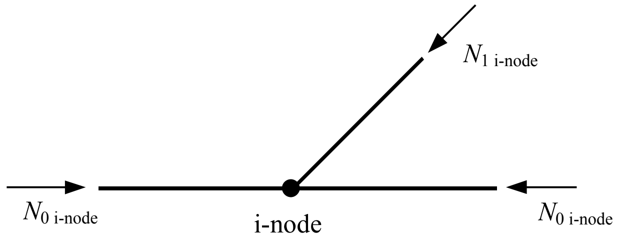
$$g(\mathbf{x})_1 = N_\mathrm{Rc} - N \ge 0$$

$$g(\mathbf{x})_2 = N_\mathrm{Fc} - N \ge 0, \quad \text{if} \quad N \ge 0$$

$$g\left(\mathbf{x}\right)_3 = N_\mathrm{Ft} - |N|, \quad \text{if} \quad N < 0$$

The calculation of the T- and Y-joint of the trusses. Trusses are rectangular hollow sections (Rautaruukki steel products designers guide 1998).





The yielding of the upper flange of the chord, $\beta \leq 0.85$. Beam 1.

$$g(\mathbf{x})_1 = N_{1,\mathrm{Rd}} - |N_1| \geq 0$$

$$N_{1,\mathrm{Rd}} = \frac{k_\mathrm{n} \cdot f_{\mathrm{y0}} \cdot t_0^2}{(1-\beta)\sin\theta_1}\left(\frac{2\eta}{\sin\theta_1} + 4\sqrt{1-\beta}\right)\frac{1.1}{\gamma_{\mathrm{mj}} \cdot \gamma_{\mathrm{m0}}} \quad \text{if} \quad \beta \leq 0.85$$

$$k_\mathrm{n} = \begin{cases} 1 \quad \text{if} \quad N_0 > 0 \\ 1.3 - (0.4/\beta)n \quad \text{if} \quad N_0 \leq 0 \end{cases}$$

$$n = \frac{\gamma_{\mathrm{m0}} \cdot \gamma_{\mathrm{mj}}}{1.1}\left(\frac{N_{0,\mathrm{Sd}}}{A_0 \cdot f_{\mathrm{y0}}} + \frac{M_{0,\mathrm{Sd}}}{W_0 \cdot f_{\mathrm{y0}}}\right)$$

Common parameters

$$\beta = b_1/b_0$$
$$\eta = h_1/b_0$$

Yielding of the upper flange of the chord, when 0.85 < β < 1.0

$$g(\mathbf{x})_2 = N_{1,\mathrm{Rd2}} - |N_2| \geq 0$$

$$N_{1,\mathrm{Rd2}} = N_{1,\mathrm{Rd}\,\beta=0.85} + \frac{N_{1,\mathrm{Rd}\,\beta=1.0} - N_{1,\mathrm{Rd}\,\beta=0.85}}{0.15}(\beta - 0.85) \quad \text{if} \quad 0.85 < \beta \leq 1$$

$$N_{1,\mathrm{Rd}\,\beta=0.85} = \frac{k_\mathrm{n} \cdot f_{\mathrm{y0}} \cdot t_0^2}{(1-\beta)\sin\theta_1}\left(\frac{2\eta}{\sin\theta_1} + 4\sqrt{1-0.85}\right)\frac{1.1}{\gamma_{\mathrm{mj}} \cdot \gamma_{\mathrm{m0}}}$$

$$N_{1,\mathrm{Rd}\,\beta=1.0} = \frac{f_\mathrm{b} \cdot t_0}{\sin\theta_1}\left(\frac{2h_1}{\sin\theta_1} + 10t_0\right)\frac{1.1}{\gamma_{\mathrm{m0}} \cdot \gamma_{\mathrm{mj}}}$$

$$f_\mathrm{b} = \begin{cases} f_{\mathrm{y0}} & \mathit{if} \quad N_0 \geq 0 \quad \text{(tension)} \\ \chi \cdot f_{\mathrm{y0}} & \mathit{if} \quad N_0 < 0 \quad \text{(compression)} \end{cases}$$

$$\chi = \frac{1}{\phi + \sqrt{\phi^2 - \lambda^2}}$$

$$\phi = 0.5 \cdot \left[1 + \alpha \cdot (\lambda - 0.2) + \lambda^2\right], \quad \text{where} \quad \alpha = 0.45$$

$$\lambda = 3.46\left(\frac{h_0}{t_0} - 2\right)\sqrt{\frac{f_{\mathrm{y0}}}{E \cdot \sin(\theta_1)}}\frac{1}{\pi}$$

The buckling of the chord web, when β = 1,0.

$$g(\mathbf{x})_3 = N_{1,\mathrm{Rd3}} - |N_3| \geq 0$$

$$N_{1,\mathrm{Rd3}} = \frac{f_\mathrm{d} \cdot t_0}{\sin\theta_1}\left(\frac{2h_1}{\sin\theta_1} + 10t_0\right)\frac{1.1}{\gamma_{\mathrm{m0}} \cdot \gamma_{\mathrm{mj}}}$$

$$f_\mathrm{b} = \begin{cases} f_{\mathrm{y0}} & \mathit{if} \quad N_0 \geq 0 \quad \text{(tension)} \\ \chi \cdot f_{\mathrm{y0}} & \mathit{if} \quad N_0 < 0 \quad \text{(compression)} \end{cases}$$

$$\chi = \frac{1}{\phi + \sqrt{\phi^2 - \lambda^2}}$$

$$\phi = 0.5 \cdot \left[1 + \alpha \cdot (\lambda - 0.2) + \lambda^2\right], \quad \text{where} \quad \alpha = 0.45$$

$$\lambda = 3.46 \cdot \left(\frac{h_0}{t_0} - 2\right) \sqrt{\frac{f_{\text{y0}}}{E \cdot \sin(\theta_1)}} \frac{1}{\pi}$$

The cutting of the chord flange, when $0.85 \leq \beta \leq 1\text{-}(1/\gamma)$.

$$N_{\text{1Rd4}} = \frac{f_{\text{y0}} \cdot t_0}{\sqrt{3} \cdot \sin(\theta_1)} \cdot \left(\frac{2 \cdot h_1}{\sin(\theta_1)} + 2 \cdot b_{\text{ep}}\right) \cdot \frac{1.1}{\gamma_{\text{mj}} \cdot \gamma_{\text{m0}}}$$

The braking of the diagonal, when $\beta \geq 0.85$

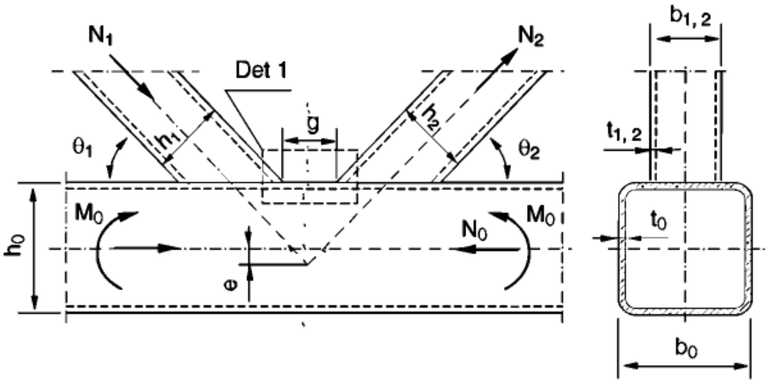
$$N_{\text{1,Rd5}} = f_{\text{y1}} \cdot t_1 \cdot (2 \cdot h_1 - 4 \cdot t_1 + 2 \cdot b_{\text{eff}}) \cdot \frac{1.1}{\gamma_{\text{mj}} \cdot \gamma_{\text{m0}}}$$

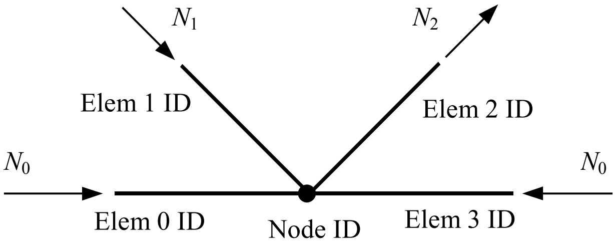
$$b_{\text{eff}} = \frac{10 \cdot b_1 \cdot t_0^2 \cdot f_{\text{y0}}}{b_0 \cdot t_1 \cdot f_{\text{y1}}}$$

Geometric constraints

$$\underline{g}(\mathbf{x})_6 = \frac{b_1 + h_1}{t_1} \geq 25$$
$$\underline{g}(\mathbf{x})_7 = b_1/b_0 \geq 0.25$$
$$\underline{g}(\mathbf{x})_8 = h_1/b_0 \geq 0.25$$
$$\underline{g}(\mathbf{x})_9 = h_1/b_1 \geq 0.5$$
$$\underline{g}(\mathbf{x})_{10} = 2 \geq h_1/b_1$$
$$\underline{g}(\mathbf{x})_{11} = 35 \geq b_1/t_1$$
$$\underline{g}(\mathbf{x})_{12} = 35 \geq h_1/t_1$$
$$g(\mathbf{x})_{13} = 1{,}25\sqrt{\frac{E}{f_{\text{y1}}}} \geq \frac{b_1}{t_1} \quad \text{if} \quad N_1 < 0 \text{ (compression)}$$
$$g(\mathbf{x})_{14} = 1{,}25\sqrt{\frac{E}{f_{\text{y1}}}} \geq \frac{h_1}{t_1} \quad \text{if} \quad N_1 < 0 \text{ (compression)}$$
$$\underline{g}(\mathbf{x})_{15} = \frac{b_0 + h_0}{t_0} \geq 25$$
$$\underline{g}(\mathbf{x})_{16} = h_0/b_0 \geq 0.5$$
$$\underline{g}(\mathbf{x})_{17} = 2 \geq h_0/b_0$$
$$\underline{g}(\mathbf{x})_{18} = 35 \geq b_0/t_0$$
$$\underline{g}(\mathbf{x})_{19} = 35 \geq h_0/t_0$$

The strength calculation of the K-joint of the trusses. Trusses are rectangular hollow sections (Rautaruukki steel products designers guide 1998).





Common parameters

$$\beta = \frac{\sum_{i=1}^{m} b_i}{m \cdot b_0}, \qquad \eta = \frac{h_i}{b_0}, \qquad \gamma = \frac{b_0}{2t_0}$$

$$e = \left(\frac{h_1}{2\sin(\theta_1)} + \frac{h_2}{2\sin(\theta_2)} + g \right) \frac{\sin(\theta_1) \cdot \sin(\theta_2)}{\sin(\theta_1 + \theta_2)} - \frac{h_0}{2}$$

Yielding of the chord flange. Beam 1

$$g(\mathbf{x})_1 = |N_1| \leq N_{1,\mathrm{Rd1}}$$

$$N_{1,\mathrm{Rd}} = 8.9 \frac{f_{y0} \cdot t_0^2}{\sin(\theta_1)} \left(\frac{\sum_{i=1}^{m} b_i + \sum_{i=1}^{m} h_i}{2m \cdot b_0} \right) k_{\mathrm{n}} \sqrt{\gamma} \frac{1.1}{\gamma_{\mathrm{mj}} \cdot \gamma_{\mathrm{m0}}}$$

$$k_{\mathrm{n}} = \begin{cases} 1, & \text{if} \quad N_0 > 0 \\ 1.3 - (0.4/\beta)n, & \text{if} \quad N \leq 0 \end{cases}, \quad \text{however} \quad k_{\mathrm{n}} \leq 1$$

$$n = \frac{\gamma_{\mathrm{m0}} \cdot \gamma_{\mathrm{mj}}}{1.1}\left(\frac{N_{0,\mathrm{Sd}}}{A_0 \cdot f_{\mathrm{y0}}} + \frac{M_{0,\mathrm{Sd}}}{W_0 \cdot f_{\mathrm{y0}}}\right)$$

Yielding of the chord flange. Beam 2

$$g(\mathbf{x})_2 = |N_2| \le N_{2,\mathrm{Rd1}}$$

$$N_{2,\mathrm{Rd1}} = 8.9\frac{f_{\mathrm{y0}} \cdot t_0^2}{\sin(\theta_2)}\left(\frac{\sum_{i=1}^{m} b_i + \sum_{i=1}^{m} h_i}{2m \cdot b_0}\right) k_{\mathrm{n}}\sqrt{\gamma}\frac{1.1}{\gamma_{\mathrm{mj}} \cdot \gamma_{\mathrm{m0}}}$$

$$k_{\mathrm{n}} = \begin{cases} 1, & \text{if} \quad N_0 > 0 \\ 1.3 - (0.4/\beta)n, & \text{if} \quad N_0 \le 0 \end{cases}, \quad \text{however} \quad k_{\mathrm{n}} \le 1$$

$$n = \frac{\gamma_{\mathrm{m0}} \cdot \gamma_{\mathrm{mj}}}{1.1}\left(\frac{N_{0,\mathrm{Sd}}}{A_0 \cdot f_{\mathrm{y0}}} + \frac{M_{0,\mathrm{Sd}}}{W_0 \cdot f_{\mathrm{y0}}}\right)$$

The cutting of the chord, beam 1.

$$g(\mathbf{x})_3 = |N_1| \le N_{1,\mathrm{Rd3}}$$

$$N_{1,\mathrm{Rd3}} = \frac{f_{y0} \cdot A}{\sqrt{3}\sin(\theta_1)}\frac{1.1}{\gamma_{\mathrm{mj}} \cdot \gamma_{\mathrm{m0}}}$$

The cutting of the chord, beam 2.

$$g(\mathbf{x})_4 = |N_2| \le N_{2,\mathrm{Rd3}}$$

$$N_{2,\mathrm{Rd3}} = \frac{f_{y0} \cdot A}{\sqrt{3}\sin(\theta_2)}\frac{1.1}{\gamma_{\mathrm{mj}} \cdot \gamma_{\mathrm{m0}}}$$

Cutting of the chord, when $V_{\mathrm{Sd}} \ge 0.5\ V_{\mathrm{pl,Rd}}$.

$$g(\mathbf{x})_5 = |N_0| \le N_{0,\mathrm{Rd3}}$$

$$N_{0,\mathrm{Rd3}} = \begin{cases} \left[A_0 - A_{\mathrm{v}}\left(\frac{2V_{\mathrm{Sd}}}{V_{\mathrm{pl.Rd}}} - 1\right)^2\right]\frac{f_{\mathrm{y0}}}{\gamma_{\mathrm{m0}}}, & \text{if} \quad V_{\mathrm{Sd}} > 0.5\,V_{\mathrm{pl,Rd}} \\ A_0\frac{f_{\mathrm{y0}}}{\gamma_{\mathrm{m0}}}, \quad \text{if} \quad V_{\mathrm{Sd}} \le 0.5\,V_{\mathrm{pl.Rd}} \end{cases}$$

$$A_{\mathrm{v}} = (2h_0 + \alpha \cdot b_0)t_0$$

$$\alpha = \sqrt{\frac{1}{1+\frac{4g^2}{3t_0^2}}}$$

$$V_{\text{pl,Rd}} = \frac{f_{\text{y0}} \cdot A_{\text{v}}}{\sqrt{3} \cdot \gamma_{\text{m0}}}$$

Diagonal brake. Beam 1.

$$g(\mathbf{x})_6 = \left|N_1\right| \leq N_{1,\text{Rd4}}$$

$$N_{1.\text{Rd4}} = f_{\text{y1}} \cdot t_1 \left(2h_1 - 4t_1 + b_1 + b_{\text{eff}}\right) \frac{1.1}{\gamma_{\text{mj}} \cdot \gamma_{\text{m0}}}$$

$$b_{\text{eff}} = \frac{10b_1 \cdot t_0^2 \cdot f_{\text{y0}}}{b_0 \cdot t_1 \cdot f_{\text{y1}}}, \text{ however } b_{\text{eff}} \leq b_1$$

Diagonal brake. Beam 2.

$$g(\mathbf{x})_7 = \left|N_2\right| \leq N_{2,\text{Rd4}}$$

$$N_{2,\text{Rd4}} = f_{\text{y2}} \cdot t_2 \left(2h_2 - 4t_2 + b_2 + b_{\text{eff}}\right) \frac{1.1}{\gamma_{\text{mj}} \cdot \gamma_{\text{m0}}}$$

$$b_{\text{eff}} = \frac{10b_2 \cdot t_0^2 \cdot f_{\text{y0}}}{b_0 \cdot t_2 \cdot f_{\text{y2}}}, \text{ however } b_{\text{eff}} \leq b_2$$

The cutting of the chord flange, when $\beta \leq$ 1-(1-γ). Beam 1.

$$g(\mathbf{x})_8 = \left|N_1\right| \leq N_{1,\text{Rd5}}$$

$$N_{1.\text{Rd5}} = \frac{f_{\text{y0}} \cdot t_0}{\sqrt{3}\sin\left(\theta\right)_1} \left(\frac{2h_1}{\sin\left(\theta_1\right)} + b_1 + b_{\text{ep}}\right) \frac{1.1}{\gamma_{\text{mj}} \cdot \gamma_{\text{m0}}}$$

$$b_{\text{ep}} = \frac{10t_0 \cdot b_1}{b_0}$$

The cutting of the chord flange, when $\beta \leq$ 1-(1-γ). Beam 1.

$$g(\mathbf{x})_9 = \left|N_2\right| \leq N_{2,\text{Rd5}}$$

$$N_{2.\text{Rd5}} = \frac{f_{\text{y0}} \cdot t_0}{\sqrt{3}\sin\left(\theta_2\right)} \left(\frac{2h_1}{\sin\left(\theta_2\right)} + b_2 + b_{\text{ep}}\right) \frac{1.1}{\gamma_{\text{mj}} \cdot \gamma_{\text{m0}}}$$

$$b_{\text{ep}} = \frac{10t_0 \cdot b_2}{b_0}$$

Other constraints are:

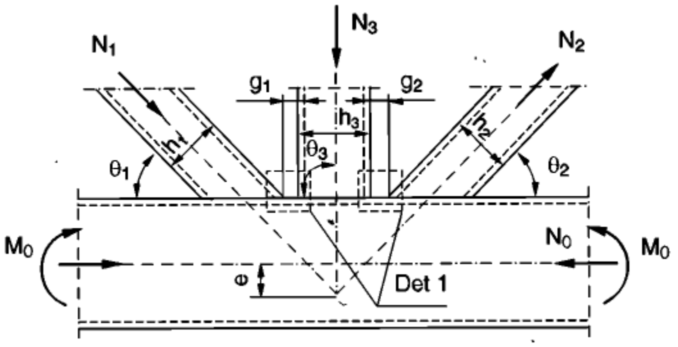
$$g(\mathbf{x})_{10} = 1.25\sqrt{E/f_{\mathrm{y1}}} - b_1/t_1 \geq 0, \quad \text{if} \quad N_1 < 0 \quad \text{(compression)}$$

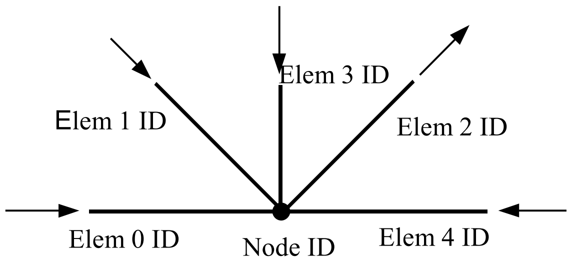
$$g(\mathbf{x})_{11} = 1.25\sqrt{E/f_{\mathrm{y2}}} - h_2/t_2 \geq 0, \quad \text{if} \quad N_2 < 0 \quad \text{(compression)}$$

Geometric constraints are:

$$\underline{g}(\mathbf{x})_{12} = \eta - 0.1 + b_0/100t_0 \geq 0$$
$$\underline{g}(\mathbf{x})_{13} = \beta - 0.1 + b_0/100t_0 \geq 0$$
$$\underline{g}(\mathbf{x})_{14} = \beta - 0.35 \geq 0$$
$$\underline{g}(\mathbf{x})_{15} = h_1/b_1 - 0.5 \geq 0$$
$$\underline{g}(\mathbf{x})_{16} = h_2/b_2 - 0.5 \geq 0$$
$$\underline{g}(\mathbf{x})_{17} = 2 - h_1/b_1 \geq 0$$
$$\underline{g}(\mathbf{x})_{18} = 2 - h_2/b_2 \geq 0$$
$$\underline{g}(\mathbf{x})_{19} = 25 - (b_1 + h_1)/t_1 \geq 0$$
$$\underline{g}(\mathbf{x})_{20} = 25 - (b_2 + h_2)/t_2 \geq 0$$
$$\underline{g}(\mathbf{x})_{21} = 35 - b_1/t_1 \geq 0$$
$$\underline{g}(\mathbf{x})_{22} = 35 - b_2/t_2 \geq 0$$
$$\underline{g}(\mathbf{x})_{23} = 35 - h_1/t_1 \geq 0$$
$$\underline{g}(\mathbf{x})_{24} = 35 - h_2/t_2 \geq 0$$
$$\underline{g}(\mathbf{x})_{25} = 2 - h_0/b_0 \geq 0$$
$$\underline{g}(\mathbf{x})_{26} = h_0/b_0 - 0.5 \geq 0$$
$$\underline{g}(\mathbf{x})_{27} = 35 - b_0/t_0 \geq 0$$
$$\underline{g}(\mathbf{x})_{28} = 35 - h_0/t_0 \geq 0$$
$$\underline{g}(\mathbf{x})_{29} = 25 - (b_0 + h_0)/t_0 \geq 0$$
$$\underline{g}(\mathbf{x})_{30} = g/b_0 - 0.5(1 - \beta) \geq 0$$
$$\underline{g}(\mathbf{x})_{31} = 1.5(1 - \beta) - g/b_0 \geq 0$$
$$\underline{g}(\mathbf{x})_{32} = g - (t_1 + t_2) \geq 0$$
$$\underline{g}(\mathbf{x})_{33} = 1 - k_{\mathrm{n}} \geq 0$$

The calculation of the KT-joint of the trusses. Trusses are rectangular hollow sections (Rautaruukki steel products designers guide 1998).





Eccentricity and gaps of the joint.

$$g_1 = \left(e + \frac{h_0}{2}\right) \cdot \left(\frac{\sin(\theta_1 + \theta_3)}{\sin(\theta_1) \cdot \sin(\theta_3)}\right) - \left(\frac{h_1}{2 \cdot \sin(\theta_1)} + \frac{h_3}{2 \cdot \sin(\theta_3)}\right)$$

$$g_2 = \left(e + \frac{h_0}{2}\right) \cdot \left(\frac{\sin(\theta_2 + \theta_3)}{\sin(\theta_2) \cdot \sin(\theta_3)}\right) - \left(\frac{h_2}{2 \cdot \sin(\theta_2)} + \frac{h_3}{2 \cdot \sin(\theta_3)}\right)$$

where

$$e = \max(e_1, e_2)$$

and, *where*

$$e_1 = \left(\frac{h_1}{2 \cdot \sin(\theta_1)} + \frac{h_3}{2 \cdot \sin(\theta_3)} + g_{\min}\right) \cdot \frac{\sin(\theta_1 + \theta_3)}{\sin(\theta_1) \cdot \sin(\theta_3)} - \frac{h_0}{2}$$

$$e_2 = \left(\frac{h_2}{2 \cdot \sin(\theta_2)} + \frac{h_3}{2 \cdot \sin(\theta_3)} + g_{\min}\right) \cdot \frac{\sin(\theta_2 + \theta_3)}{\sin(\theta_2) \cdot \sin(\theta_3)} - \frac{h_0}{2}$$

Common parameters

$$\gamma = \frac{b_0}{2t_0}$$

$$\beta = \frac{\sum_{i=1}^{m} b_i}{m \cdot b_0}$$

$$\eta = \frac{h_i}{b_0}$$

Yield of the chord flange. Beam 1

$$g(\mathbf{x})_1 = N_{1,\mathrm{Rd1}} - |N_1| \geq 0$$

$$N_{1,\mathrm{Rd1}} = 8.9 \frac{f_{\mathrm{y0}} \cdot t_0^2}{\sin\theta_1} \left(\frac{\sum_{i=1}^{m} b_i + \sum_{i=1}^{m} h_i}{2m \cdot b_0} \right) k_\mathrm{n} \sqrt{\gamma} \frac{1.1}{\gamma_{\mathrm{mj}} \cdot \gamma_{\mathrm{m0}}}$$

$$k_\mathrm{n} = \begin{cases} 1, & \text{if} \quad N_0 > 0 \\ 1.3 - (0.4/\beta)n, & \text{if} \quad N_0 \leq 0 \end{cases}, \quad \text{however} \quad k_\mathrm{n} \leq 1$$

$$n = \frac{\gamma_{\mathrm{m0}} \cdot \gamma_{\mathrm{mj}}}{1.1} \left(\frac{N_{0,\mathrm{Sd}}}{A_0 \cdot f_{\mathrm{y0}}} + \frac{M_{0,\mathrm{Sd}}}{W_0 \cdot f_{\mathrm{y0}}} \right)$$

Yield of the chord flange. Beam 2

$$g(\mathbf{x})_2 = N_{2,\mathrm{Rd1}} - |N_2| \geq 0$$

$$N_{2,\mathrm{Rd1}} = 8.9 \frac{f_{\mathrm{y0}} \cdot t_0^2}{\sin\theta_2} \left(\frac{\sum_{i=1}^{m} b_i + \sum_{i=1}^{m} h_i}{2m \cdot b_0} \right) k_\mathrm{n} \sqrt{\gamma} \frac{1.1}{\gamma_{\mathrm{mj}} \cdot \gamma_{\mathrm{m0}}}$$

$$k_\mathrm{n} = \begin{cases} 1, & \text{if} \quad N_0 > 0 \\ 1.3 - (0.4/\beta)n, & \text{if} \quad N_0 \leq 0 \end{cases}, \quad \text{however} \quad k_\mathrm{n} \leq 1$$

$$n = \frac{\gamma_{\mathrm{m0}} \cdot \gamma_{\mathrm{mj}}}{1.1} \left(\frac{N_{0,\mathrm{Sd}}}{A_0 \cdot f_{\mathrm{y0}}} + \frac{M_{0,\mathrm{Sd}}}{W_0 \cdot f_{\mathrm{y0}}} \right)$$

Yield of the chord flange. Beam 3

$$g(\mathbf{x})_3 = N_{3,\mathrm{Rd1}} - |N_3| \geq 0$$

$$N_{3,\mathrm{Rd1}} = 8.9 \frac{f_{\mathrm{y}0} \cdot t_0^2}{\sin\theta_3} \left(\frac{\sum_{i=1}^{m} b_i + \sum_{i=1}^{m} h_i}{2m \cdot b_0} \right) k_{\mathrm{n}} \sqrt{\gamma} \frac{1.1}{\gamma_{\mathrm{mj}} \cdot \gamma_{\mathrm{m0}}}$$

$$k_{\mathrm{n}} = \begin{cases} 1, & \text{if} \quad N_0 > 0 \\ 1.3 - (0.4/\beta)n, & \text{if} \quad N_0 \leq 0 \end{cases}, \quad \text{however} \quad k_{\mathrm{n}} \leq 1$$

$$n = \frac{\gamma_{\mathrm{m0}} \cdot \gamma_{\mathrm{mj}}}{1.1} \left(\frac{N_{0,\mathrm{Sd}}}{A_0 \cdot f_{\mathrm{y0}}} + \frac{M_{0,\mathrm{Sd}}}{W_0 \cdot f_{\mathrm{y0}}} \right)$$

The cut of the chord. Beam 1

$$g(\mathbf{x})_4 = N_{1,\mathrm{Rd2}} - |N_1| \geq 0$$

$$N_{1,\mathrm{Rd2}} = \frac{f_{y0} \cdot A}{\sqrt{3} \sin\theta_1} \frac{1.1}{\gamma_{\mathrm{mj}} \cdot \gamma_{\mathrm{m0}}}$$

The cut of the chord. Beam 2

$$g(\mathbf{x})_5 = N_{2,\mathrm{Rd2}} - |N_2| \geq 0$$

$$N_{2,\mathrm{Rd2}} = \frac{f_{y0} \cdot A}{\sqrt{3} \sin\theta_2} \frac{1.1}{\gamma_{\mathrm{mj}} \cdot \gamma_{\mathrm{m0}}}$$

The cut of the chord. Beam 3

$$g(\mathbf{x})_6 = N_{3,\mathrm{Rd2}} - |N_3| \geq 0$$

$$N_{3,\mathrm{Rd2}} = \frac{f_{\mathrm{y}0} \cdot A}{\sqrt{3} \sin\theta_3} \frac{1.1}{\gamma_{\mathrm{mj}} \cdot \gamma_{\mathrm{m0}}}$$

The cut of the chord, Beam 0

$$g(\mathbf{x})_7 = N_{0,\mathrm{Rd2}} - |N_0| \geq 0$$

$$N_{0,\mathrm{Rd2}} = \begin{cases} \left[A_0 - A_\mathrm{v} \left(\dfrac{2\,V_\mathrm{Sd}}{V_\mathrm{pl,Rd}} - 1 \right)^2 \right] \dfrac{f_\mathrm{y0}}{\gamma_\mathrm{m0}}, & \text{if} \quad V_\mathrm{Sd} > 0.5\,V_\mathrm{pl,Rd} \\ A_0 \dfrac{f_\mathrm{y0}}{\gamma_\mathrm{m0}}, & \text{if} \quad V_\mathrm{Sd} \leq 0.5\,V_\mathrm{pl,Rd} \end{cases}$$

$$A_\mathrm{v} = \left(2h_0 + \alpha \cdot b_0 \right) t_0$$

$$\alpha = \sqrt{\frac{1}{1 + \dfrac{4g^2}{3t_0^2}}}$$

$$V_\mathrm{pl,Rd} = \frac{f_\mathrm{y0} \cdot A_\mathrm{v}}{\sqrt{3} \cdot \gamma_\mathrm{m0}}$$

Diagonal break. Beam 1

$$g(\mathbf{x})_8 = N_{1,\mathrm{Rd3}} - |N_1| \geq 0$$

$$N_{1.\mathrm{Rd3}} = f_\mathrm{y1} \cdot t_1 \left(2h_1 - 4t_1 + b_1 + b_\mathrm{eff} \right) \frac{1.1}{\gamma_\mathrm{mj} \cdot \gamma_\mathrm{m0}}$$

$$b_\mathrm{eff} = \frac{10 b_1 \cdot t_0^2 \cdot f_\mathrm{y0}}{b_0 \cdot t_1 \cdot f_\mathrm{y1}}, \quad \text{however} \quad b_\mathrm{eff} \leq b_1$$

Diagonal break. Beam 2

$$g(\mathbf{x})_9 = N_{2,\mathrm{Rd3}} - |N_2| \geq 0$$

$$N_{2,\mathrm{Rd3}} = f_\mathrm{y2} \cdot t_2 \left(2h_2 - 4t_2 + b_2 + b_\mathrm{eff} \right) \frac{1.1}{\gamma_\mathrm{mj} \cdot \gamma_\mathrm{m0}}$$

$$b_\mathrm{eff} = \frac{10 \cdot b_2 \cdot t_0^2 \cdot f_\mathrm{y0}}{b_0 \cdot t_2 \cdot f_\mathrm{y2}}, \quad \text{however} \quad b_\mathrm{eff} \leq b_2$$

Diagonal break. Beam 3

$$g(\mathbf{x})_{10} = N_{3,\mathrm{Rd3}} - |N_3| \geq 0$$

$$N_{3,\mathrm{Rd3}} = f_\mathrm{y3} \cdot t_3 \left(2h_3 - 4t_3 + b_3 + b_\mathrm{eff} \right) \frac{1.1}{\gamma_\mathrm{mj} \cdot \gamma_\mathrm{m0}}$$

$$b_{eff} = \frac{10 \cdot b_3 \cdot t_0^2 \cdot f_\mathrm{y0}}{b_0 \cdot t_3 \cdot f_\mathrm{y3}}, \quad \text{however} \quad b_\mathrm{eff} \leq b_3$$

Cutting of the chord flange. Beam 1

$$g(\mathbf{x})_{11} = N_{1,\mathrm{Rd4}} - |N_1| \geq 0$$

$$N_{1.\mathrm{Rd4}} = \frac{f_{\mathrm{y0}} \cdot t_0}{\sqrt{3}\sin\theta_1}\left(\frac{2h_1}{\sin\theta_1} + b_1 + b_{\mathrm{ep}}\right)\frac{1.1}{\gamma_{\mathrm{mj}} \cdot \gamma_{\mathrm{m0}}}$$

$$b_{\mathrm{ep}} = \frac{10 \cdot t_0 \cdot b_1}{b_0}$$

Cutting of the chord flange. Beam 2

$$g(\mathbf{x})_{12} = N_{2,\mathrm{Rd4}} - |N_2| \geq 0$$

$$N_{2.\mathrm{Rd4}} = \frac{f_{\mathrm{y0}} \cdot t_0}{\sqrt{3}\sin(\theta_2)}\left(\frac{2h_2}{\sin(\theta_2)} + b_2 + b_{\mathrm{ep}}\right)\frac{1.1}{\gamma_{\mathrm{mj}} \cdot \gamma_{\mathrm{m0}}}$$

$$b_{\mathrm{ep}} = \frac{10 \cdot t_0 \cdot b_2}{b_0}$$

Cutting of the chord flange. Beam 3

$$g(\mathbf{x})_{13} = N_{3,\mathrm{Rd4}} - |N_3| \geq 0$$

$$N_{3,\mathrm{Rd4}} = \frac{f_{\mathrm{y0}} \cdot t_0}{\sqrt{3}\sin(\theta_3)}\left(\frac{2h_2}{\sin(\theta_3)} + b_3 + b_{\mathrm{ep}}\right)\frac{1.1}{\gamma_{\mathrm{mj}} \cdot \gamma_{\mathrm{m0}}}$$

$$b_{\mathrm{ep}} = \frac{10 \cdot t_0 \cdot b_3}{b_0}$$

Geometric constraints.

$$\underline{g}(\mathbf{x})_{14} = \eta - 0.1 + b_0/100t_0 \geq 0$$
$$\underline{g}(\mathbf{x})_{15} = \beta - 0.1 + b_0/100t_0 \geq 0$$
$$\underline{g}(\mathbf{x})_{16} = \beta - 0.35 \geq 0$$
$$\underline{g}(\mathbf{x})_{17} = h_1/b_1 - 0.5 \geq 0$$
$$\underline{g}(\mathbf{x})_{18} = h_2/b_2 - 0.5 \geq 0$$
$$\underline{g}(\mathbf{x})_{19} = h_3/b_3 - 0.5 \geq 0$$

$$\underline{g}(\mathbf{x})_{20} = 2 - h_1/b_1 \geq 0$$
$$\underline{g}(\mathbf{x})_{21} = 2 - h_2/b_2 \geq 0$$
$$\underline{g}(\mathbf{x})_{22} = 2 - h_3/b_3 \geq 0$$
$$\underline{g}(\mathbf{x})_{23} = 25 - (b_1 + h_1)/t_1 \geq 0$$
$$\underline{g}(\mathbf{x})_{24} = 25 - (b_2 + h_2)/t_2 \geq 0$$
$$\underline{g}(\mathbf{x})_{25} = 25 - (b_3 + h_3)/t_3 \geq 0$$
$$\underline{g}(\mathbf{x})_{26} = 35 - b_1/t_1 \geq 0$$
$$\underline{g}(\mathbf{x})_{27} = 35 - b_2/t_2 \geq 0$$
$$\underline{g}(\mathbf{x})_{28} = 35 - b_3/t_3 \geq 0$$
$$\underline{g}(\mathbf{x})_{29} = 35 - h_1/t_1 \geq 0$$
$$\underline{g}(\mathbf{x})_{30} = 35 - h_2/t_2 \geq 0$$
$$\underline{g}(\mathbf{x})_{31} = 35 - h_3/t_3 \geq 0$$
$$g(\mathbf{x})_{32} = 1.25\sqrt{E/f_{\mathrm{y}1}} - b_1/t_1 \geq 0, \quad \text{if} \quad N_1 < 0$$
$$g(\mathbf{x})_{33} = 1.25\sqrt{E/f_{\mathrm{y}2}} - b_2/t_2 \geq 0, \quad \text{if} \quad N_2 < 0$$
$$g(\mathbf{x})_{34} = 1.25\sqrt{E/f_{\mathrm{y}3}} - b_3/t_3 \geq 0, \quad \text{if} \quad N_3 < 0$$
$$\underline{g}(\mathbf{x})_{35} = 2 - h_0/b_0 \geq 0$$
$$\underline{g}(\mathbf{x})_{36} = h_0/b_0 - 0.5 \geq 0$$
$$\underline{g}(\mathbf{x})_{37} = 35 - b_0/t_0 \geq 0$$
$$\underline{g}(\mathbf{x})_{38} = 35 - h_0/t_0 \geq 0$$
$$\underline{g}(\mathbf{x})_{39} = 25 - (b_0 + h_0)/t_0 \geq 0$$
$$\underline{g}(\mathbf{x})_{40} = g/b_0 - 0.5(1 - \beta) \geq 0$$
$$\underline{g}(\mathbf{x})_{41} = 1.5(1 - \beta) - g/b_0 \geq 0$$
$$\underline{g}(\mathbf{x})_{42} = g - (t_1 + t_2) \geq 0$$
$$\underline{g}(\mathbf{x})_{43} = 1 - k_n \geq 0$$
$$\underline{g}(\mathbf{x})_{44} = b_1 - b_{\mathrm{eff}} \geq 0$$
$$\underline{g}(\mathbf{x})_{45} = b_2 - b_{\mathrm{eff}} \geq 0$$
$$\underline{g}(\mathbf{x})_{46} = b_3 - b_{\mathrm{eff}} \geq 0$$
$$\underline{g}(\mathbf{x})_{47} = b_1 - \frac{10 \cdot t_0 \cdot b_1}{b_0} \geq 0$$
$$\underline{g}(\mathbf{x})_{48} = b_2 - \frac{10 \cdot t_0 \cdot b_2}{b_0} \geq 0$$
$$\underline{g}(\mathbf{x})_{49} = b_2 - \frac{10 \cdot t_0 \cdot b_2}{b_0} \geq 0$$

Test function 1

The function is also known as Rastrigin's function:

$$f(x) = nA + \sum_{i=1}^{n} (x_i^2 - A\cos(2\pi x_i))$$

with additional box constraints

$\forall i \in [1, n], \quad x_i \in [-5.12, 5.12]$ and $A = 10$

Test function 2

The function is presented by Floudas (Floudas, C.A. et al. 1987).

Minimize

$$\mathrm{f}(\mathbf{x}) = 5\sum_{i=1}^{4} x_i - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i$$

subject to constraints

$$\mathrm{g}_1(\mathbf{x}) = 2x_1 + 2x_2 + x_{10} + x_{11} - 10 \le 0$$
$$\mathrm{g}_2(\mathbf{x}) = 2x_1 + 2x_3 + x_{10} + x_{12} - 10 \le 0$$
$$\mathrm{g}_3(\mathbf{x}) = 2x_1 + 2x_3 + x_{11} + x_{12} - 10 \le 0$$
$$\mathrm{g}_4(\mathbf{x}) = -8x_1 + x_{10} \le 0$$
$$\mathrm{g}_5(\mathbf{x}) = -8x_2 + x_{11} \le 0$$
$$\mathrm{g}_6(\mathbf{x}) = -8x_3 + x_{12} \le 0$$
$$\mathrm{g}_7(\mathbf{x}) = -2x_4 - x_5 + x_{10} \le 0$$
$$\mathrm{g}_8(\mathbf{x}) = -2x_6 - x_7 + x_{11} \le 0$$
$$\mathrm{g}_9(\mathbf{x}) = -2x_8 - x_9 + x_{12} \le 0$$

and subject to boundary constraints

$$0 \le x_i \le 1, \quad i = 1, ..., 9$$
$$0 \le x_i \le 100, \quad i = 10, 11, 12$$
$$0 \le x_{13} \le 1$$

The global optimum is at $\mathbf{x}^* = (1,1,1,1,1,1,1,1,1,3,3,3,1)$

and

$$f(\mathbf{x}^*) = -15$$

Test function 3

The problem is no. 113 presented by Hock (Hock, W. et al. 1981).

Minimize

$f(\mathbf{x})=(x_1-10)^2-5(x_2-12)^2+3(x_4-11)^2+10x_5^6+7x_6^2+x_7^4-4x_6x_7-10x_6-8x_7$

subject to constraints

$g_1(\mathbf{x})=-127+2x_1^2+3x_2^4+x_3+4x_4^2+5x_5\leq 0$

$g_2(\mathbf{x})=-287+7x_1+3x_2+10x_3^2+x_4+x_5\leq 0$

$g_3(\mathbf{x})=-196+23x_1+x_2^2+6x_6^2+8x_7\leq 0$

$g_4(\mathbf{x})=4x_1^2+x_2^2-3x_1x_2+2x_3^2+5x_6-11x_7\leq 0$

and subject to boundary constraints

$-10\leq x_i\leq 10,\quad i=1,...,7$

The numerically obtained best known solution is at

$\mathbf{x}^*=(2.330499,\quad 1.951372,-0.4775414,\quad 4.365726,\quad -0.6244870,\quad 1.038131,\quad 1.594227)$

and

$f(\mathbf{x})=680.6300573$

Test function 4

The problem is no. 116 presented by Hock (Hock, W. et al. 1981).

$f(\mathbf{x})=x_1+x_2+x_3$

subject to constraints

$g_1(\mathbf{x})=1-0.0025(x_4+x_6)\geq 0$

$g_2(\mathbf{x})=1-0.0025(x_5+x_7-x_4)\geq 0$

$g_3(\mathbf{x})=1-0.01(x_8-x_5)\geq 0$

$g_4(\mathbf{x})=x_1x_6-833.332582\,x_4-100\,x_1+83333.333\geq 0$

$g_5(\mathbf{x})=x_2x_7-1250x_5-x_2x_4+1250x_4\geq 0$

$g_6(\mathbf{x})=x_3x_8-1250000-x_3x_5+2500x_5\geq 0$

and subject to boundary constraints

$100\leq x_i\leq 10000,\quad i=1$

$1000\leq x_i\leq 10000,\quad i=2,3$

$10\leq x_i\leq 1000,\quad i=4,...,8$

The numerically obtainded bes known solution is at

$\mathbf{x}^*=(579.3040,\quad 1359.975,\quad 5109.970,\quad 182.0175,$
$\quad 295.6012,\quad 217.9825,\quad 286.4163,\quad 395.6012)$

and

$f(\mathbf{x}^*)=7049.248021$

Test function 5

The function is presented by Koziel (Koziel, S. et al. 1999).

Minimize

$$f(\mathbf{x}) = \frac{\sin^3(2\pi x_1) \cdot \sin(2\pi x_2)}{x_1^3(x_1 + x_2)}$$

subject to

$$g_1(\mathbf{x}) = x_1^2 - x_2 + 1 \le 0$$

$$g_2(\mathbf{x}) = 1 - x_1 + (x_2 - 4)^2 \le 0$$

and subject to boundary constraints

$$0 \le x_1 \le 10$$

$$0 \le x_2 \le 10$$

The global optimum is

$$f\left(\mathbf{x}^*\right) = 0.095825$$

Test function 6

The function is presented by Floudas (Floudas, C.A. 1987).

Minimize

$$f(\mathbf{x}) = (x_1 - 10)^3 + (x_2 - 20)^3$$

subject to constraints

$$g_1(\mathbf{x}) = -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \le 0$$

$$g_2(x) = (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \le 0$$

and subject to boundary constraints

$$13 \le x_1 \le 100$$

$$0 \le x_2 \le 100$$

The known global optimum is at

$$\mathbf{x}^* = (14.095, \quad 0.84296)$$

and

$$f(\mathbf{x}) = -6961.81381$$

Test function 7

The function is problem no. 100 presented by Hock (Hock, W. et al. 1981).

Minimize

$$f(\mathbf{x}) = x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 + (x_5 - 3)^2 + 2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45$$

subject to constraints

$$g_1(\mathbf{x}) = 105 - 4x_1 - 5x_2 + 3x_7 - 9x_8 \ge 0$$
$$g_2(\mathbf{x}) = -10x_1 + 8x_2 + 17x_7 - 2x_8 \ge 0$$
$$g_3(\mathbf{x}) = 8x_1 + 2x_2 - 5x_9 + 2x_{10} + 12 \ge 0$$
$$g_4(\mathbf{x}) = -3(x_1 - 2)^2 - 4(x_2 - 3)^2 - 2x_3^2 + 7x_4 + 120 \ge 0$$
$$g_5(\mathbf{x}) = -5x_1^2 - 8x_2 - (x_3 - 6)^2 + 2x_4 + 40 \ge 0$$
$$g_6(\mathbf{x}) = -x_1^2 - 2(x_2 - 2)^2 + 2x_1x_2 - 14x_5 + 6x_6 \ge 0$$
$$g_7(\mathbf{x}) = -0.5(x_1 - 8)^2 - 2(x_2 - 4)^2 - 3x_5^2 + x_6 + 30 \ge 0$$
$$g_8(\mathbf{x}) = 3x_1 - 6x_2 - 12(x_9 - 8)^2 + 7x_{10} \ge 0$$

and subject to boundary constraints

$$-10 \le x_i \le 10, \quad i = 1,...,10$$

The numerically obtained best known solution reported is at

$\mathbf{x}^*$=(2.171996, 2.363683, 8.773926, 5.095984, 0.9906548, 1.430574, 1.321644, 9.828726, 8.280092, 8.375927)

and

$$f(\mathbf{x}^*) = 24.3062091$$

Test function 8

The function is presented by Himmelblau (Himmelblau, D. 1997).

Minimize

$f(\mathbf{x}) = 5.3578547x_3^2 + 0.8356891x_1x_5 + 37.293239x_1 - 40792.141$
subject to constraints
$g_1(\mathbf{x}) = 85.334407 + 0.0056858x_2x_5 + 0.00026x_1x_4 - 0.0022053x_3x_5 - 92 \leq 0$
$g_2(\mathbf{x}) = -85.334407 - 0.0056858x_2x_5 - 0.00026x_1x_4 + 0.0022053x_3x_5 \leq 0$
$g_3(\mathbf{x}) = 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 + 0.0021813x_3^2 - 110 \leq 0$
$g_4(\mathbf{x}) = -80.51249 - 0.0071317x_2x_5 - 0.0029955x_1x_2 - 0.0021813x_3^2 + 90 \leq 0$
$g_5(\mathbf{x}) = 9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 + 0.0019085x_3x_4 - 25 \leq 0$
$g_6(\mathbf{x}) = -9.300961 - 0.0047026x_3x_5 - 0.0012547x_1x_3 - 0.0019085x_3x_4 + 20 \leq 0$
and subject to boundary constraints
$78 \leq x_1 \leq 102$
$33 \leq x_2 \leq 45$
$27 \leq x_i \leq 45, \quad i = 3,4,5$
The global optimum at $\mathbf{x}^* = (78.000, \quad 33.000, \quad 27.071, \quad 45.000, \quad 44.969)$
and
$f(\mathbf{x}) = -31025.5602$

The fatigue constraints of the multi-redundant boom are calculated according the IIW Fatigue Recommendations (Hobbacher, A. 1996).

$$\Delta\sigma_{\mathrm{S,d}} = \Delta\sigma_{\mathrm{S,k}} \cdot \gamma_{\mathrm{F}} \le \Delta\sigma_{\mathrm{R,d}} = \frac{\Delta\sigma_{\mathrm{R,k}}}{\gamma_{\mathrm{m}}}$$

$$\Delta\sigma_{\mathrm{R,k}} = \Delta\sigma_{\mathrm{FAT}} \cdot \left(\frac{2\cdot 10^6}{N}\right)^{\frac{1}{m}}$$

where $\Delta\sigma_{\mathrm{eq}}$ is an equivalent stress range, $\Delta\sigma_{\mathrm{S,d}}$ is a design value of stress range caused by actions and $\Delta\sigma_{\mathrm{R,k}}$ is the characteristic stress range at the required number of stress cycles.

$$g_{13} = \Delta\sigma_{R,d} - \Delta\sigma_{S,d,13}$$
$$g_{14} = \Delta\sigma_{R,d} - \Delta\sigma_{S,d,14}$$

where

$$\Delta\sigma_{R,d} = \frac{\Delta\sigma_{R,k}}{\gamma_M}$$

$$\Delta\sigma_{S,d,13} = \Delta\sigma_{S,k,13} \cdot \gamma_F$$

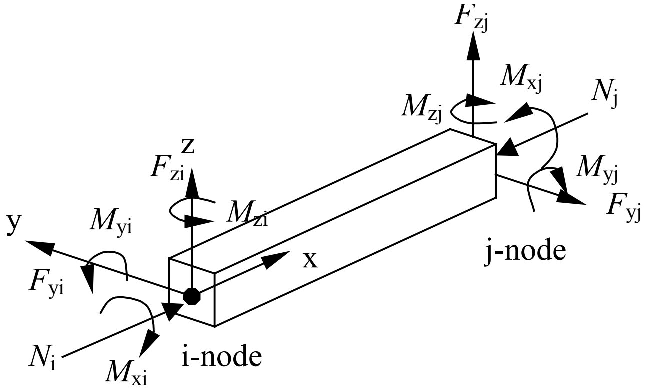
where

$$\Delta\sigma_{S,d,13} = \frac{\left|M_{y,elem,13}\right|}{W_{y,13}} + \frac{\left|M_{z,elem,13}\right|}{W_{z,13}}$$

$$\Delta\sigma_{S,d,14} = \Delta\sigma_{S,k,14} \cdot \gamma_F$$

where

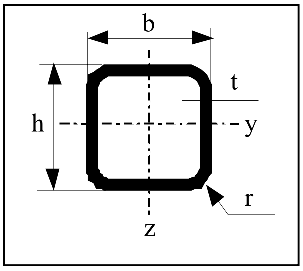
$$\Delta\sigma_{S,d,14} = \frac{\left|M_{y,elem,14}\right|}{W_{y,14}} + \frac{\left|M_{z,elem,14}\right|}{W_{z,14}}$$



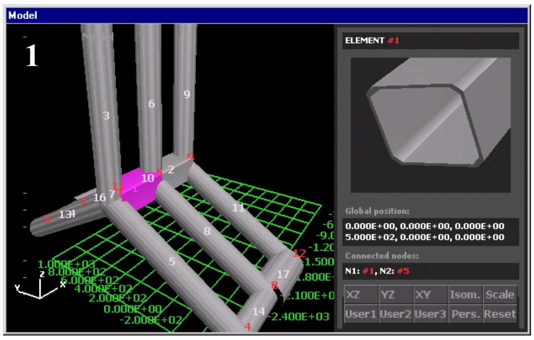
APPENDIX 13 CROSS-SECTIONS OF RECTANGULAR BEAMS

(Rautaruukki steel products designers guide 1998)

ID	H	B	T	R
1	40	40	2	4
2	40	40	2.5	5
3	50	50	2	4
4	50	50	2.5	5
5	50	50	3	6
6	60	60	2.5	5
7	60	60	3	6
8	60	60	4	8
9	70	70	2	4
10	70	70	2.5	5
11	70	70	3	6
12	70	70	4	8
13	80	80	2	4
14	80	80	2.5	5
15	80	80	3	6
16	80	80	4	8
17	80	80	5	10
18	90	90	2	4
19	90	90	2.5	5
20	90	90	3	6
21	90	90	4	8
22	90	90	5	10
23	90	90	6	12
24	90	90	6.3	15.8
25	100	100	2	4
26	100	100	2.5	5
27	100	100	3	6
28	100	100	4	8
29	100	100	5	10
30	100	100	6	12
31	100	100	6.3	15.8
32	100	100	7.1	17.8
33	100	100	8	20
34	110	110	2.5	5
35	110	110	3	6
36	110	110	4	8
37	110	110	5	10
38	110	110	6	12
39	110	110	6.3	15.8
40	120	120	3	6
41	120	120	4	8
42	120	120	5	10
43	120	120	5.6	11.2
44	120	120	6	12
45	120	120	6.3	15.8
46	120	120	7.1	17.8
47	120	120	8	20
48	120	120	8.8	22
49	140	140	4	8
50	140	140	5	10
51	140	140	5.6	11.2
52	140	140	6	12
53	140	140	6.3	15.8
54	140	140	7.1	17.8
55	140	140	8	20
56	140	140	8.8	22
57	140	140	10	25
58	150	150	4	8
59	150	150	5	10
60	150	150	6	12
61	150	150	6.3	15.8
62	150	150	7.1	17.8
63	150	150	8	20
64	150	150	8.8	22
65	150	150	10	25
66	160	160	4	8
67	160	160	5	10
68	160	160	6	12
69	160	160	6.3	15.8
70	160	160	7.1	17.8
71	160	160	8	20
72	160	160	8.8	22
73	160	160	10	25
74	160	160	12	36
75	160	160	12.5	37.5
76	180	180	5	10
77	180	180	6	12
78	180	180	6.3	15.8
79	180	180	7.1	17.8
80	180	180	8	20
81	180	180	8.8	22
82	180	180	10	25
83	180	180	12	36
84	180	180	12.5	37.5
85	200	200	5	10
86	200	200	6	12
87	200	200	6.3	15.8
88	200	200	7.1	17.8
89	200	200	8	20
90	200	200	8.8	22
91	200	200	10	25
92	200	200	12	36
93	200	200	12.5	37.5
94	220	220	6	12
95	220	220	6.3	15.8
96	220	220	7.1	17.8
97	220	220	8	20
98	220	220	8.8	22
99	220	220	10	25
100	220	220	12	36
101	220	220	12.5	37.5
102	250	250	6	12
103	250	250	6.3	15.8
104	250	250	7.1	17.8
105	250	250	8	20
106	250	250	8.8	22
107	250	250	10	25
108	250	250	12	36
109	250	250	12.5	37.5
110	260	260	6	12
111	260	260	6.3	15.8
112	260	260	7.1	17.8
113	260	260	8	20
114	260	260	8.8	22
115	260	260	10	25
116	260	260	11	33
117	260	260	12.5	37.5
118	300	300	6	12
119	300	300	6.3	15.8
120	300	300	7.1	17.8
121	300	300	8	20
122	300	300	8.8	22
123	300	300	10	25
124	300	300	12	36
125	300	300	12.5	37.5



The evolution of the boom FE-model during the optimisation. Pictures are screen captures.



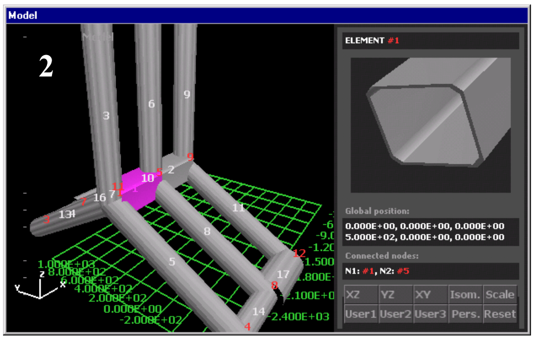
ELEMENT #1

Global position:

0.000E+00, 0.000E+00, 0.000E+00
5.000E+02, 0.000E+00, 0.000E+00

Connected nodes:

N1: #1, N2: #5



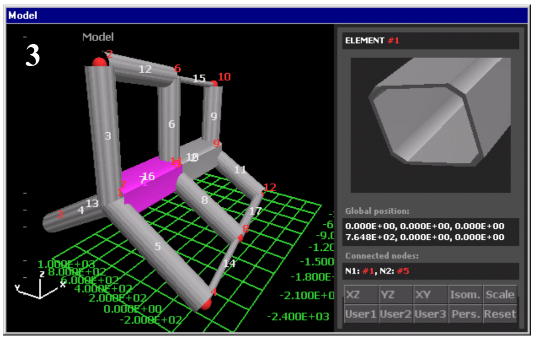
ELEMENT #1

Global position:

0.000E+00, 0.000E+00, 0.000E+00
5.000E+02, 0.000E+00, 0.000E+00

Connected nodes:

N1: #1, N2: #5



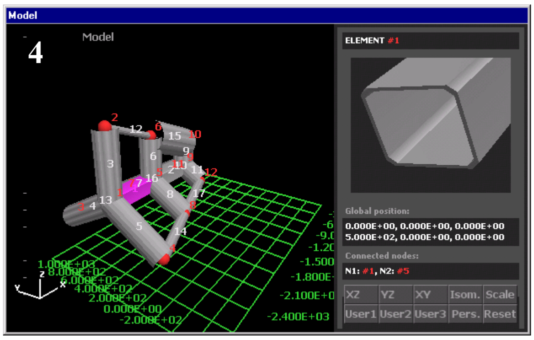
ELEMENT #1

Global position:

0.000E+00, 0.000E+00, 0.000E+00
7.648E+02, 0.000E+00, 0.000E+00

Connected nodes:

N1: #1, N2: #5



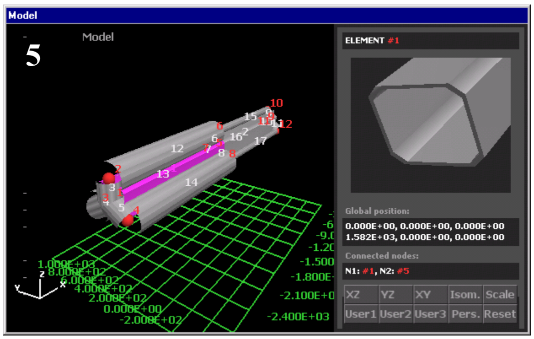
ELEMENT #1

Global position:

0.000E+00, 0.000E+00, 0.000E+00
5.000E+02, 0.000E+00, 0.000E+00

Connected nodes:

N1: #1, N2: #5



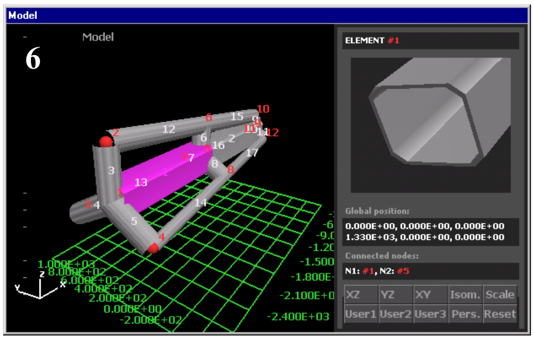
ELEMENT #1

Global position:

0.000E+00, 0.000E+00, 0.000E+00
1.582E+03, 0.000E+00, 0.000E+00

Connected nodes:

N1: #1, N2: #5



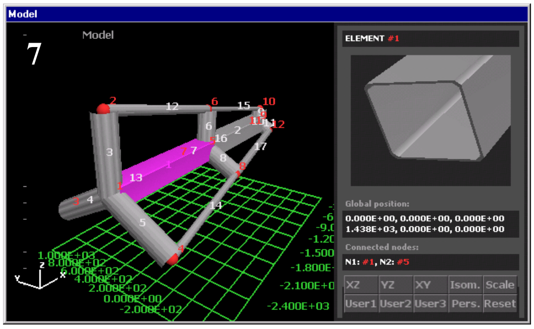
ELEMENT #1

Global position:

0.000E+00, 0.000E+00, 0.000E+00
1.330E+03, 0.000E+00, 0.000E+00

Connected nodes:

N1: #1, N2: #5



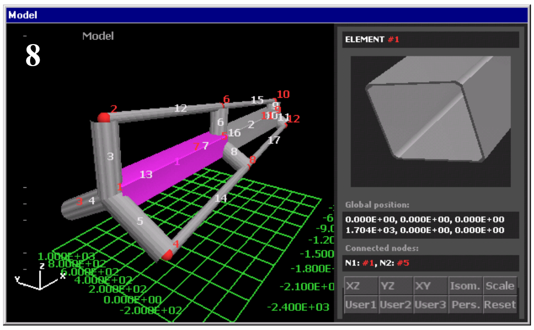
ELEMENT #1

Global position:

0.000E+00, 0.000E+00, 0.000E+00
1.438E+03, 0.000E+00, 0.000E+00

Connected nodes:

N1: #1, N2: #5



ELEMENT #1

Global position:

0.000E+00, 0.000E+00, 0.000E+00
1.704E+03, 0.000E+00, 0.000E+00

Connected nodes:

N1: #1, N2: #5

ACTA UNIVERSITATIS LAPPEENRANTAENSIS

93. NERG, JANNE. Numerical modelling and design of static induction heating coils. 2000. 86 s. Diss.

94. VARTIAINEN, MIKA. Welding time models for cost calculations in the early stages of the design process. 2000. 89 s., liitt. Diss.

95. JERNSTRÖM, EEVA. Assessing the technical competitiveness of printing papers. 2000. 159 s., liitt. Diss.

96. VESTERINEN, PETRI. On effort estimation in software projects. 2000. U.s. Diss.

97. LUUKKO, JULIUS. Direct torque control of permanent magnet synchronous machinesanalysis and implementation. 2000. 172 s. Diss.

98. JOKINEN, ARTO. Lobbying as a part of business management. 2000. 244 s. Diss.

99. JÄÄSKELÄINEN, EDUARD. The role of surfactant properties of extractants in hydrometallurgical liquid-liquid extraction processes. 2000. U.s. Diss.

100. Proceedings of 3rd Finnish-French Colloquium on Nuclear Power Plant Safety. 2000. 118 s.

101. TANSKANEN, PASI. The evolutionary structural optimization (ESO) method: theoretical aspects and the modified evolutionary structural optimization (MESO) method. 2000. 67 s., liitt. Diss.

102. JERNSTRÖM, PETTERI. The effects of real-time control of welding parameters on weld quality in plasma arc keyhole welding. 2000. 69 s., liitt. Diss.

103. KAARNA, ARTO. Multispectral image compression using the wavelet transform. 2000. U.s. Diss.

104. KOTONEN, ULLA. Rahavirta-analyysit, erityisesti kassavirtalaskelma, kunnan talouden ohjauksen apuvälineenä. 2000. 209 s., liitt. Väitösk.

105. VARIS, JUHA. A novel procedure for establishing clinching parameters for high strength steel sheet. 2000. 84 s., liitt. Diss.

106. PÄTÄRI, EERO. Essays on portfolio performance measurement. 2000. 201 s. Diss.

107. SANDSTRÖM, JAANA. Cost information in engineering design – potentials and limitations of activity-based costing. 2001. 143 s., liitt. Diss.

108. TOIVANEN, JOUKO. Balanced Scorecardin implementointi ja käytön nykytila Suomessa. 2001. 216 s. Väitösk.

109. PESONEN, MAUNO. Applying AHP and A´WOT to strategic planning and decision making: case studies in forestry and forest industry. 2001. U.s. Diss.

110. Proceedings of Fifth International Seminar on Horizontal Steam Generators. Ed. by Juhani Vihavainen. 2001. 255 s.

111. LAINE, PERTTI. Kohti vesiensuojelun aikaa: veden laadun muutokset eteläisellä Saimaalla. 2001. 264 s. Väitösk.

112. SILVENTOINEN, PERTTI. Electromagnetic compatibility and EMC-measurements in DC-voltage link converters. 2001. 115 s. Diss.

113. TERVONEN, ANTERO. Laadun kehittäminen suomalaisissa yrityksissä. 2001. 206 s. Väitösk.

114. SALMINEN, ANTTI. The effects of filler wire feed on the efficiency, parameters and tolerances of laser welding. 2001. 82 s., liitt. Diss.

115. HORTTANAINEN, MIKA. Propagation of the ignition front against airflow in packed beds of wood particles. 2001. U.s. Diss.

116. IKONEN, JOUNI. Improving distributed simulation in a workstation environment. 2001. U.s. Diss.

117. WU, HUAPENG. Analysis, design and control of a hydraulically driven parallel robot manipulator. 2001. U.s. Diss.

118. REUNANEN, ARTTU. Experimental and numerical analysis of different volutes in a centrifugal compressor. 2001. 150 s. Diss.

119. TAAVITSAINEN, VELI-MATTI. Strategies for combining soft and hard modelling in some physicochemical problems. 2001. U.s. Diss.

120. SAVOLAINEN, RAIJA. The use of branched ketene dimers in solving the deposit problems related to the internal sizing of uncoated fine paper. 2001. U.s. Diss.

121. SARAVIRTA, ALI. Project success through effective decisions: case studies on project goal setting, success evaluation and managerial decision making. 2001. 286 s. Diss.

122. BLOMQVIST, KIRSIMARJA. Partnering in the dynamic environment: the role of trust in asymmetric technology partnership formation. 2002. 296 s., liitt. Diss.

123. KARVONEN, VESA. Development of fiber recovery process. 2002. U.s. Diss.

124. KÄYHKÖ, JARI. The influence of process conditions on the deresination efficiency in mechanical pulp washing. 2002. 87 s., liitt. Diss.

125. SAVOLAINEN, PEKKA. Modeling of non-isothermal vapor membrane separation with thermodynamic models and generalized mass transfer equations. 2002. 179 s. Diss.

126. KÄRKKÄINEN, HANNU. Customer need assessment: Challenges and tools for product innovation in business-to-business organizations. 2002. U. s. Diss.

127. HÄMÄLÄINEN, MARKKU. Spray coating technique as a surface treatment for woodcontaining paper grades. 2002. 121 s. Diss.

128. RANTA, TAPIO. Logging residues from regeneration fellings for biofuel production – a GIS-based availability and supply cost analysis. 2002. 182 s. Diss.

129. KUOSA, MAUNU. Numerical and experimental modelling of gas flow and heat transfer in the air gap of an electric machine. 2002. 97 s. Diss.

130. LAITINEN, NIINA. Development of a ceramic membrane filtration equipment and its applicability for different wastewaters. 2002. U. s. Diss.

131. SUNDQVIST, SANNA. Market orientation in the international context: Antecedents, consequences and applicability. 2002. U. s. Diss.

132. TORKKELI, MARKO. Technology selection and group decision support systems: Case studies on supporting strategic technology selection processes. 2002. U.s. Diss.

134. HEIKKILÄ, TANJA. Permanent magnet synchronous motor for industrial inverter applications – analysis and design. 2002. 109 s. Diss.

135. HUTTUNEN, PENTTI. Data-parallel computation in parallel and distributed environments. 2002. U.s. Diss.

137. JUHANTILA, OLLI-PEKKA. Establishing intercompany relationships: Motives and methods for successful collaborative engagement. 2002. 281 s. Diss.