Jurnal Teknologi, bil. 23, Jun 1994 hlm. 65 – 80
©Universiti Teknologi Malaysia

AN INTERACTIVE OPTIMISATION EXPERT SYSTEM FOR THE PRELIMINARY DESIGN OF OFFSHORE SUPPLY VESSELS

MOHD RAMZAN HJ. MAINAL
Fakulti Kejuruteraan Jentera
Universiti Teknologi Malaysia
Sekudai, Johor

Abstract. Over the last two decades, numerous methods of optimum design have been developed for use in preliminary ship design. Nevertheless, the methods are constraint to those who are experts in this field while those who are non-experts have difficulty in making the algorithm work for their application. This paper describes the layout of a nonlinear optimisation expert system which has the capability of providing guidance and consulting help to any designer. Some of the basic principles of interactive optimisation system are also discussed. A simple application for the design of offshore supply vessel is included to demonstrate somè of the capabilities.

1 INTRODUCTION

It has always been a challenge for designers to produce efficient and cost-effective systems without compromising their integrity. The conventional design process depends on the designer's intuition, experience and skill. Figure 1 shows the self-explanatory flowchart for a conventional design process that involves the use of information gathered from one or more trial designs together with the designer's experience and intuition.

The need for efficiency in today's competitive world have forced designers to evince greater interest in economical and better designs. With recent advances in computer technology affecting various disciplines of engineering, the design process can hardly remain untouched. Design is not only regarded as the more or less intuitively guided creation of new information, it also comprises analysis, presentation of results, simulation and optimisation. These are essential constituents of an iterative process leading to a feasible and finally optimum design. Figure 2 shows the optimum design process.

The optimum design process forces the designer to identify explicitly a set of design variables, a cost function to be minimised, and the constraint functions for the system. This rigorous formulation of the design problem helps the designer to gain a better understanding of the problem. However, the optimisation process can benefit substantially from the designer's experience and intuition. Thus, the best approach would be to have an optimum design process that is aided by the designer's interaction.

Several methods for optimum design of systems have been developed over the past two decades, and the preliminary ship design is no excemption. A computer's is harnessed speed with computational algorithms to methodically generate efficient designs which are needed in today's competitive world. Nevertheless, most of the methods work well only when used by the optimisation experts. Designers who are not optimisation experts have difficulty in

Typeset by $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX

making the algorithms and programs work for their application. This indicates that the rules used by the expert in making the program work should be captured and put in the knowledge base to provide consulting help to the designers. Such a system should be well designed for knowledge acquisition and utilisation.

This paper describes attributes of an expert system for design optimisation for the preliminary design of offshore supply vessels based on the non-linear constrained optimisation concept developed by Hooke and Jeeves [7] and Nelder and Mead [5].

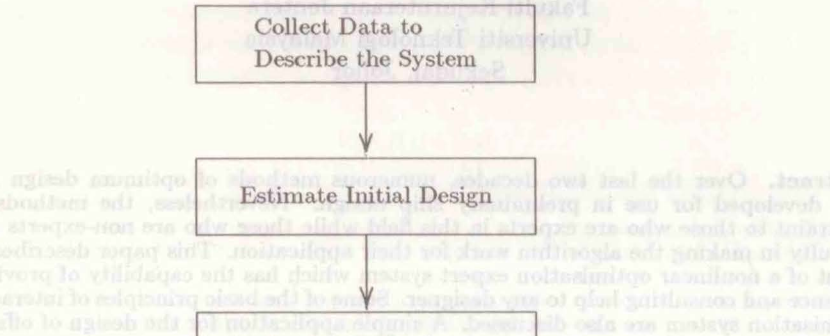
Collect Data to Describe the System
Estimate Initial Design
Analyse the System
Check Performance Criteria
Is Design Satisfactory?
yes
Stop
no
Change Design Based on Experience /Heuristics

Fig. 1 Conventional design process

2 FUNDAMENTALS OF OPTIMISATION CONCEPT

This section will briefly discuss some of the fundamental ideas and methods of mathematical programming. The discussion will focus on the basic ideas of how and why these procedures work; mathematical detail is intentionally avoided to the extent possible. Full details on the optimisation methods together with their mathematical formulation are given in numerous texts such as [6], [10] and [11].

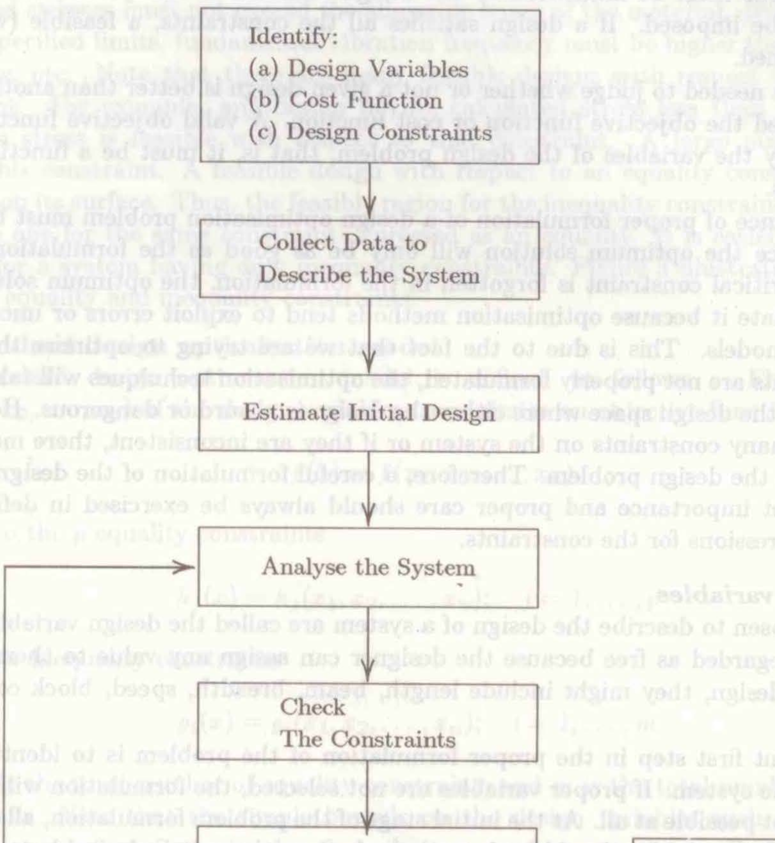

Fig. 2 Optimum design process

2.1 Problem formulation

Formulation of an optimum design problem involves transcribing a verbal description of the problem into a well-defined mathematical statement. The formulation process begins by identifying a set of variables to describe the system, called design variables. Once the variables are given numerical values, a design system is produced.

All systems are designed to perform within a given set of constraints which include limitation on resources, material failure, response of system, member size, etc.

The constraints must be influenced by the design variables of the system, because only then can they be imposed. If a design satisfies all the constraints, a feasible (workable) system is obtained.

A criterion is needed to judge whether or not a given design is better than another. This criterion is called the objective function or cost function. A valid objective function must be influenced by the variables of the design problem, that is, it must be a function of the design variables.

The importance of proper formulation of a design optimisation problem must be clearly understood since the optimum solution will only be as good as the formulation is. For example, if a critical constraint is forgotten in the formulation, the optimum solution will most likely violate it because optimisation methods tend to exploit errors or uncertainties in the design models. This is due to the fact that we are trying to optimise the system and if constraints are not properly formulated, the optimisation techniques will take designs into portion of the design space where either the design is absurd or dangerous. However, if there are too many constraints on the system or if they are inconsistent, there may not be any solution to the design problem. Therefore, a careful formulation of the design problem is of paramount importance and proper care should always be exercised in defining and developing expressions for the constraints.

2.1.1 *Design variables*

Parameters chosen to describe the design of a system are called the design variables. These variables are regarded as free because the designer can assign any value to them. In preliminary ship design, they might include length, beam, breadth, speed, block co-efficient, etc.

An important first step in the proper formulation of the problem is to identify design variables for the system. If proper variables are not selected, the formulation will be either incorrect, or not possible at all. At the initial stage of the problem formulation, all options of identifying design variables should be investigated. Sometimes it is desirable to designate more design variables than may be apparent from the statement of the problem. This gives an added flexibility in the problem formulation. Later, it is possible to assign fixed numerical value to any variable and thus eliminate it from the problem formulation. Another important factor is that all design variables should be independent of each other as far as possible. One should be able to assign numerical value to any variable independent of any other variable.

2.1.2 *Design constraints*

All restrictions placed on a design are collectively called constraint. Each constraint must be influenced by one or more design variables. Only then is it meaningful and does it have influence on optimum design. Some constraints are quite simple, such as the minimum and maximum value of the design variables while more complex ones may be indirectly influenced by the design variables. Many constraint functions have only first-order terms in the design variables which are called linear constraints, where as more general problems have nonlinear constraint functions as well. Thus, methods to treat both linear and nonlinear constraints must be developed.

In practice, design problems may have equality as well as inequality constraints. A typical example of equality constraint in preliminary ship design is a situation where a ship might be required to have a specified deadweight. A feasible design must satisfy precisely such an equality constraint. Examples of inequality constraints are the requirement that calculated stresses must not exceed the allowable stress of the material, deflection must not exceed specified limits, fundamental vibration frequency must be higher than the operating frequency, etc. Note that there are many feasible designs with respect to an inequality constraint. For example, any design having calculated stress less than or equal to the allowable stress is feasible with respect to that constraint. A large number of designs satisfy this constraint. A feasible design with respect to an equality constraint, however, must lie on its surface. Thus, the feasible region for the inequality constraints is much larger than the one for the same constraint expressed as an equality. It is easier to find feasible designs for a system having only inequality constraints. Figure 3 illustrates the difference between equality and inequality constraints.

2.2 Standard design optimisation model

The standard design optimisation model is defined as follows : Find an n-vector $x = (x_1, x_2, \ldots, x_n)$ of the design variables to minimise an objective function

$$f(x) = f(x_1, x_2, \ldots, x_n) \tag{1}$$

subject to the p equality constraints

$$h_j(x) = h_j(x_1, x_2, \ldots, x_n); \quad j = 1, \ldots, p \tag{2}$$

and the m inequality constraints

$$g_i(x) = g_i(x_1, x_2, \ldots, x_n); \quad i = 1, \ldots, m \tag{3}$$

where p is the total number of equality constraints and m is the total number of inequality constraints. Note that the simple bounds on the design variables such as $x_i \geq 0$, $i = 1, \ldots, n$, or $x_{i1} \leq x_i \leq x_{iu}$, $i = 1, \ldots, n$ where x_{i1} and x_{iu} are the smallest and the largest allowed value for x_i, are included in the equalities of Eqn. 3. In numerical methods, these constraints can be treated more efficiently in the original form without converting them to the form of Eqn. 3.

Design optimisation problems from different fields of engineering can be transcribed into the standard model. Thus, the standard model is quite general. It is important to note that once design problems from different fields are transcribed into the standard model, they all look alike.

2.3 Selection of optimisation algorithms

Search methods for use on multivariable unconstrained problems have rapidly increased in number and sophistication in recent years. While all realistic problems are constrained, an unconstrained building block is often required. The unconstrained methods are normally divided into two categories, derivative free methods and gradient methods.

The gradient methods require function and derivative evaluations while the derivative free methods require function evaluations only. In general, one would expect the gradient methods to be more effective, due to the added information provided. However, if analytical derivatives are available, the question of whether a search technique should be used at all is presented. If numerical derivative approximations are utilised, the efficiency of the gradient methods should be approximately the same as that of the derivative free methods. Gradient methods incorporating numerical derivatives would be expected to present some numerical problems in the vicinity of the optimum, that is, the approximations would become very small.

In this paper, two very distinguished methods is utilised; the Nelder and Mead algorithm and Hooke and Jeeves algorithm. Details of both algorithms are available in [5], [7], 10], [11] and [26].

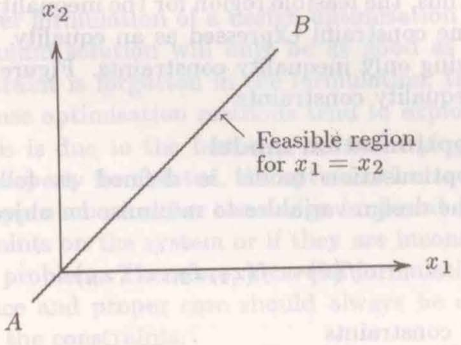

Fig. 3(a) Feasible region for Constant $x_1 = x_2$ (line A–B)

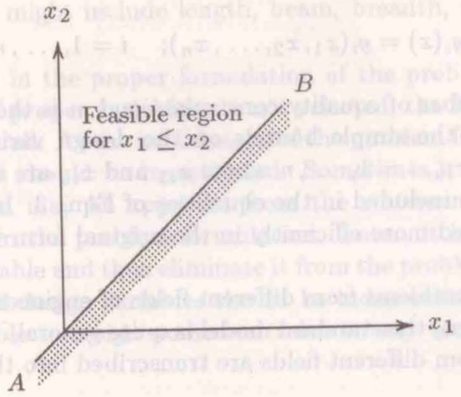

Fig. 3(b) Feasible region for Constant $x_1 \leq x_2$ (line A–B) and region above it

2.4 Transformation of constrained optimisation problem

It has been discovered that unconstrained optimisation methods can be used to solve constrained problems. The basic idea is to construct a composite function using the objective and constraint functions. It contains certain parameters, called the penalty parameters, that penalise the composite function for violation of the constraints. The larger the violation, the larger is the penalty. Once the composite function is defined for a set of penalty parameters, it is minimised using any of the unconstrained optimisation techniques. There are several varieties of penalty function methods; however in this paper, it will be restricted to the external penalty function method.

Two versions of the external penalty technique as presented by Zangwill [15], [16] use the penalty functions

$$P(\overline{x}, r_k) = F(\overline{x}) - r_k \sum_{j=1}^{p} \min(g_j(\overline{x}), 0) \tag{4}$$

and

$$P(\overline{x}, r_k) = F(\overline{x}) + r_k \sum_{j=1}^{p} [\min(g_j(\overline{x}), 0)]^2 \tag{5}$$

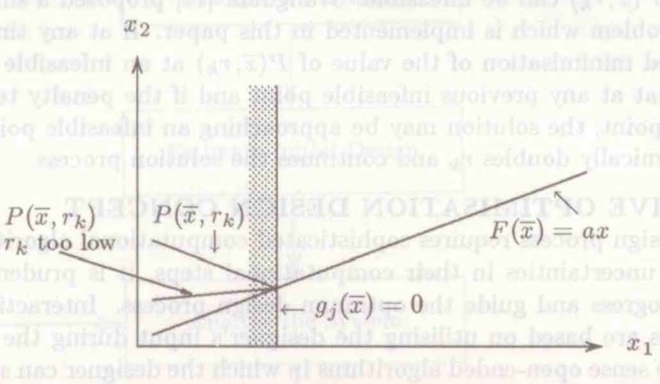

Fig. 4(a) Using Equation 4

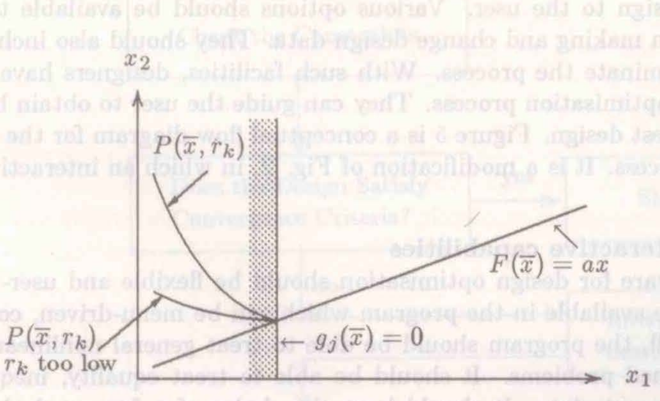

Fig. 4(b) Using Equation 5

which are illustrated in Fig. 4. These methods permit infeasible initial and intermediate points during the solution process. If the value of r_k is sufficiently large, only a single unconstrained solution for the minimum of $P(\overline{x}, r_k)$ is required. This can have significant advantage over Sequential Unconstrained Minimisation Techniques (SUMT) where at least 3 values of r_k are needed. These methods, especially Eqn. 4, suffer from ill conditioning of the surface $P(\overline{x}, r_k)$ due to the discontinuity at the constraint boundary. With this discontinuity, the unconstrained minimisation techniques which depend on the function being reasonably quadratic for best efficiency can be expected to be less effective on the external penalty technique problems. On the other hand, those methods which just compare values of the function at points in a prescribed pattern can be fully effective on external penalty technique. This is the reason why the Hooke and Jeeves direct search and Nelder and Mead simplex search have been incorporated into the external penalty technique optimisation program.

For typical ship design problems where $F(\overline{x})$ is reasonably flat and where the constraints are in the normalised form, r_k equal 1024 has usually been adequate [6]. An excessive value will sharpen the valley in $P(\overline{x}, r_k)$ at the constraint boundary and may cause a procedure like Hooke and Jeeves direct search to be less reliable. If r_k is too low, the solution to the minimisation of $P(\overline{x}, r_k)$ can be infeasible. Wangdahl [17] proposed a simple procedure to eliminate this problem which is implemented in this paper. If at any time the solution of the unconstrained minimisation of the value of $P(\overline{x}, r_k)$ at an infeasible point is found to be lower than that at any previous infeasible point and if the penalty term is larger than at this previous point, the solution may be approaching an infeasible point. If this occurs, the system dynamically doubles r_k and continues the solution process.

3 INTERACTIVE OPTIMISATION DESIGN CONCEPT

The optimum design process requires sophisticated computational algorithms. Since, most algorithms have uncertainties in their computational steps, it is prudent to interactively monitor their progress and guide the optimum design process. Interactive design optimisation algorithms are based on utilising the designer's input during the iterative process. They are in some sense open-ended algorithms in which the designer can specify what needs to be done depending on the current design conditions. They must be implemented into an interactive software having capabilities to interrupt the iterative process and report the status of the design to the user. Various options should be available to the designer to facilitate decision making and change design data. They should also include the capability to restart or terminate the process. With such facilities, designers have complete control over the design optimisation process. They can guide the user to obtain better designs and ultimately the best design. Figure 5 is a conceptual flow diagram for the interactive design optimisation process. It is a modification of Fig. 2, in which an interactive block has been added.

3.1 Desired interactive capabilities

Interactive software for design optimisation should be flexible and user-friendly. Help facilities should be available in the program which can be menu-driven, command-driven or both. First of all, the program should be able to treat general nonlinear programming as well as constrained problems. It should be able to treat equality, inequality and design variable bound constraints. It should have the choice of a few good algorithms that are robustly implemented. It should also trap user's mistakes and not abort abnormally.

3.1.1 *Interactive data preparation*

The software should have a module for interactive data preparation and editing. The commands for data entry should be explicit. Only the minimum amount of data should be required. The user should be able to edit any data that have been entered previously. The step-by-step procedure should be to display the menu for data selection and entry, or it should be possible to enter data in a simple question/answer session. The system should be set up in such a way that it is protected from any of the designer's mistakes. If data mis-

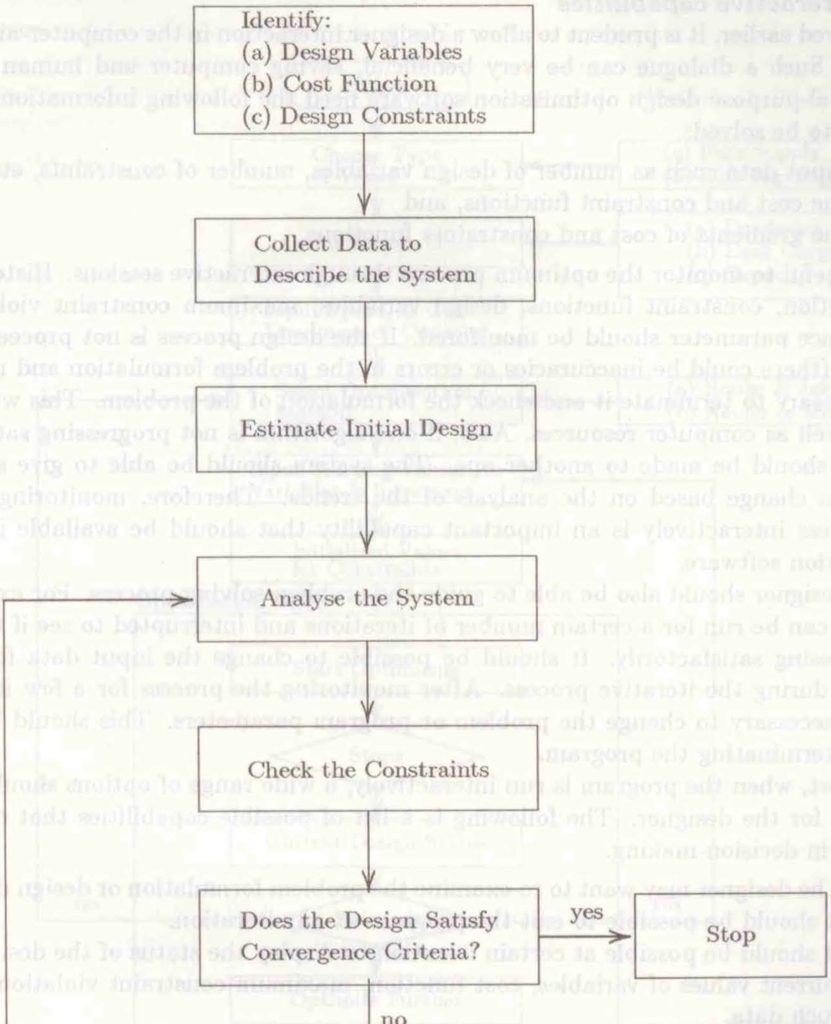

Fig. 5 Flowchart for interactive optimisation design

match is found, messages should be given in detail. The interactive input procedure should be simple so that even a beginner can follow it easily.

3.1.2 *Interactive capabilities*

As observed earlier, it is prudent to allow a designer interaction in the computer-aided design process. Such a dialogue can be very beneficial, saving computer and human resources. All general-purpose design optimisation software need the following information about the problem to be solved:

(a). input data such as number of design variables, number of constraints, etc,
(b). the cost and constraint functions, and
(c). the gradients of cost and constraints functions.

It is useful to monitor the optimum process through interactive sessions. Histories of the cost function, constraint functions, design variables, maximum constraint violation, and convergence parameter should be monitored. If the design process is not proceeding satisfactorily (there could be inaccuracies or errors in the problem formulation and modelling), it is necessary to terminate it and check the formulation of the problem. This will save human as well as computer resources. Also, if one algorithm is not progressing satisfactorily, a switch should be made to another one. The system should be able to give suggestions for design change based on the analysis of the trends. Therefore, monitoring the iterative process interactively is an important capability that should be available in a design optimisation software.

The designer should also be able to guide the problem-solving process. For example, the program can be run for a certain number of iterations and interrupted to see if the process is progressing satisfactorily. It should be possible to change the input data for a design problem during the iterative process. After monitoring the process for a few iterations it may be necessary to change the problem or program parameters. This should be possible without terminating the program.

In short, when the program is run interactively, a wide range of options should be made available for the designer. The following is a list of possible capabilities that can aid the designer in decision-making.

(a). The designer may want to re-examine the problem formulation or design data. Thus, it should be possible to exit the program at any iteration.
(b). It should be possible at certain iteration to display the status of the design, such as current values of variables, cost function, maximum constraint violation and other such data.
(c). It should be possible to change data at certain iteration, such as design variables and their limits, and other relevant data.
(d). The designer should be able to run the algorithm one iteration at a time or several iterations.
(e). It should be able to restart the program from any iteration.
(f). It is possible to change the algorithm during the interative process.

4 INTERACTIVE DESIGN OPTIMISATION 'OPTOSVD'

The preceding section essentially describes specifications for a general purpose interactive design optimisation software. Based on them, a software system can be designed and implemented. It can be observed that to implement all the flexibilities and capabilities, the software will be quite large and complex. The most modern software design and data management techniques will have to be utilised to achieve the stated goals. The entire process of software design, implementation and evaluation can be quite costly and time consuming, requiring the equivalent of several man-years.

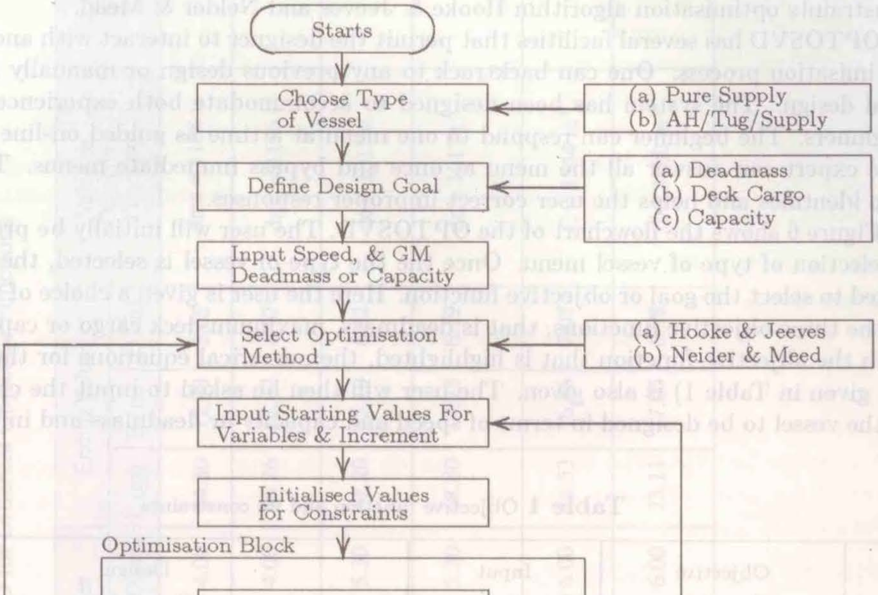

Fig. 6 Flowchart for OPTOSVD

In this section, a brief description is given of a software OPTOSVD that has some of the previously stated capabilities. OPTOSVD, which stands for Optimisation of Offshore Supply Vessels Design, is a specially written program incorporating expert system Leonardo for pure supply vessels or anchor handling/tug/ supply vessels. With the OPTOSVD program, the computer and the designer's experience can be utilised to adjust the design variables so as to improve the design objective while satisfying the constraints. It contains two nonlinear constraints optimisation algorithm Hooke & Jeeves and Nelder & Mead.

OPTOSVD has several facilities that permit the designer to interact with and control the optimisation process. One can backtrack to any previous design or manually input a new trial design. The system has been designed to accommodate both experienced users and beginners. The beginner can respond to one menu at a time as guided on-line instruction. The expert can answer all the menu at once and bypass immediate menus. The software also identifies and helps the user correct improper responses.

Figure 6 shows the flowchart of the OPTOSVD. The user will initially be prompted with a selection of type of vessel menu. Once the the type of vessel is selected, the user will be asked to select the goal or objective function. Here the user is given a choice of selecting one of the three objective functions, that is deadmass, maximum deck cargo or capacity. Along with the objective function that is highlighted, the numerical equations for the constraints (as given in Table 1) is also given. The user will then be asked to input the characteristics of the vessel to be designed in terms of speed and capacity or deadmass and initial stability.

Table 1 Objective function and its constraints

	Objective Function	Input Requirements	Design Constraints
1	Deadmass(t) (Dms=LDB/3)	Speed & Capacity	$3.75 \geq \mathrm{L/B} \leq 4.50$ $2.20 \geq \mathrm{B/D} \leq 2.70$ $(1.135 - (\mathrm{V}/3.62 * \mathrm{sqrt}(\mathrm{L}))) - 0.70 \geq 0.0$ $40 * (\mathrm{LDB}/100) - \mathrm{Capacity} \geq 0$
2	Cargo Deck(t) (CD=L*B)	Speed Deadmass	$3.75 \geq \mathrm{L/B} \leq 4.50$ $2.20 \geq \mathrm{B/D} \leq 2.70$ $(1.135 - (\mathrm{V}/3.62 * \mathrm{sqrt}(\mathrm{L}))) - 0.70 \geq 0.0$ $(\mathrm{L}/6.75 * ((\mathrm{V}+2)/\mathrm{V})\hat{}2)\hat{}3 - \mathrm{Deadmass} \geq 0.0$

3	Capacity(m^3)
	Cap=40*(LBD/100)
		Speed,
		Deadmass &
		Initial Stability
			$3.75 \geq \mathrm{L/B} \leq 4.50$
			$2.20 \geq \mathrm{B/D} \leq 2.70$
			$(\mathrm{L}/6.75 * ((\mathrm{V}+2)/\mathrm{V})\hat{}2)\hat{}3 - \mathrm{Deadmass} \geq 0.0$
			$(0.001 * \mathrm{L} * \mathrm{B}) - \mathrm{GM} \geq 0.0$

After deciding the objective function of the design, the user will be asked to select the type of optimisation algorithm from either Hooke and Jeeves or Nelder and Mead. Depending on the objective function and the type of algorithm selected, the appropriate screen will be highlighted asking the user for the starting values for the variables as well as for its incremental values.

Once the starting values of the variables and its incremental values have been installed, the system will start to execute. While the system is executing a screen will be highlighted

Table 2 Sample of Output from OPTOSVD for Different Objective Functions

Objective Function	Input Requirement	Method of Optimisation	Starting Values			EndingValues			Value of Objective Function	No. of Evaluation of Objective Function
			L (m)	B (m)	D (m)	L (m)	B (m)	D (m)		
1	V = 11.0 knots C = 1400 m^3	H & J	40.00	10.00	4.00	54.50	12.73	5.05	dms = 1168 t	112
		N & M	40.00	10.00	4.00	56.26	12.62	4.97	dms = 1176 t	198
2	V = 12.3 knots dms = 2400 t	H & J	55.00	12.00	5.30	67.20	15.10	6.42	dc = 1015 t	218
		N & M	55.00	12.00	5.30	68.80	14.78	6.19	dc = 1017 t	.304
3	V = 14.0 knots GM = 1.0 m dms = 3000 t	H & J	65.00	12.50	6.00	74.53	16.86	7.05	C = 3544 m^3	136
		N & M	65.00	12.50	6.00	73.11	17.05	6.98	C = 3480 m^3	105

to guide the user to press the appropriate key in order to stop the program. If the key is pressed, the system will stop executing and an output screen giving the current values of the objective function, variables and the number of iterations done. The user will then be asked whether to continue the program, to change the starting values of the variables or perhaps to change to another algorithm.

The system will continue depending on which phase the user has selected and proceed to optimise the objective function. Having obtained the variables that will give an optimised objective function, the user will then be asked either to continue the system to evaluate detail calculations of the design or to terminate. Table 2 summarises the output of the software OPTOSVD which is executed for several times for different objective functions and different optimisation algorithms.

5 CONCLUSION

This paper has described an interactive preliminary design software for offshore supply vessels which not only makes use of optimisation techniques, but also draws on established theory and rule-based structures.

The method illustrates that the optimisation concept could be applied to preliminary ship design, given a suitable objective, constraints and a set of variables to manipulate. The objective function and constraints chosen in this study are made simple but could be extended for quite complex problems. However, the aim of this paper is to develop an optimisation system which could be interactive and user friendly.

ACKNOWLEDGEMENT

The author would like to thank Dr. R.M. Cameron, senior lecturer in the Department of Naval Architecture & Ocean Engineering, University of Glasgow for his encouragement in preparing this paper. The author's knowledge on ship design philosophy would remain feeble without his continuous guidance.

REFERENCES

[1] Arora J.S. & Belengundu A.D., Structural Optimisation by Mathematical Programming AIAA **22** (1984).

[2] Arora J.S. & Baenziger G., *Uses of Artificial Intelligence in Design Optimisation*, vol. 54, 1986.

[3] Arora J.S. & Thanader P.B., *Computational Methods for Optimum Design of Large Complex Systems, Computational Mechanics*, vol. 1, 1986.

[4] Arora J.S., *Introduction to Optimum Design*, McGraw-Hill Book Co, 1989.

[5] Nelder J.A. & Mead R, *A Simplex Method for Function Minimisation*, vol. 7, 1965.

[6] Parson M.G., *Optimisation Methods for Use in Computer Aided Ship Design*, 1975.

[7] Hooke R & Jeeves T.A., *Direct Search Solution of Numerical and Statistical Problems*, vol. 8, 1961.

[8] Lyon T., *A Calculator-Based Preliminary Ship Design Procedure, Marine Technology*, vol. 19, 1982.

[9] Kupras L.K., *Optimisation Method and Parametric Study in Precontracted Ship Design, International Shipbuilding Progress*, vol. 23, 1976.

[10] Scales L.E., *Introduction to Non-Linear Optimisation*, Macmillan Pub Ltd, London, 1985.

[11] Himmelblau D.M., *Applied Non-Linear Programming*, McGraw-Hill Book Co, New York, 1972.

[12] Arora J.S. & Tseng C.H., *Interactive Design Optimisation, Engineering Optimisation*, vol. 13, 1988.

[13] Lyon T.D. & Mistree F., *A Computer-Based Method for the Preliminary Design of Ships*, vol. 29, 1985.

[14] Mistree F., Hughes O.F. & Phuoc, H.B., *An Optimisation Method for the Design of Large Highly Constrained Complex Systems, Engineering Optimisation*, vol. 5, 1981.

[15] Zangwill W.I., *Nonlinear Programming via Penalty Functions ,Management Science*, vol. 13, 1967.

[16] Zangwill W.I., *Nonlinear Programming - A Unified Approach*, Prentice–Hall Inc, 1969.

[17] Wangdahl G.E., *The External Penalty Function Optimisation Technique and Its Application to Ship Design, The University of Michigan, Department of Naval Architecture and Marine Engineering, Report No 129, June*, 1972.

[18] Mandel , P. & Leopold, R.,, *'Optimisation Methods Applied to Ship Design',*, vol. Vol. 74,, Trans. SNAME,, 1966.

[19] Powell, M.J.D.,, 'An Efficient Method for Finding the Minimum of a Function of Several Variables without Calculating Derivatives', **Vol. 7, No. 3, July,** (1964), Computer Journal,.

[20] Jagoda, J.,, 'Computer-Aided Multi-Level Optimisation Method Applied to Economic Ship Design', Sixth International Conference on Computer Applications in the Automation of Shipyard Operation and Ship Design, Tokyo, (1973).

[21] Nowacki, H., Brusis, F., & Swift, P.M.,, *'Tanker Preliminary Design - An Optimisation Problems with Constraints',*, vol. Vol. 78,, Trans. SNAME,, . 1970.

[22] Murphey, R.D., Sabat, D.J., & Taylor, R.J.,, *'Least Cost Ship Characteristics by Computer Techniques',*, vol. Vol. 2, No. 2, April,, Marine Technology,, 1965.

[23] Fisher, K.W.,, *'Economic Optimisation Procedures in Preliminary Ship Design (Applied to the Australian Ore Trade)',*, vol. Vol. 114,, Trans. RINA,, 1972.

[24] Kuniyasu, T.,, 'Application of Computer to Optimisation of Principal Dimensions of Ships by Parametric Study', (1968), Japan Shipbuilding, Sept.,.

[25] Box, M.J.,, 'A New Method of Constrained Optimisation and a Comparison with Other Methods', (1965-66), Computer Journal,.

[26] Kuester, J.L. & Mize, J.M.,, *'Optimisation Techniques with FORTRAN',*, McGraw Hill Book Co., New York,, 1973.

[27] Kupras, L.K.,, 'Direct Search Optimisation Technique for Constrained Objective Function Using Better Point Algorithm', (1978), Internal Report, Delft Inst. of Technology,.