The Development of a Unique Algorithm

for the

Solution of HVAC System Design Optimisation Problems.

Geoffrey C. Lambert.

Thesis submitted in accordance with the requirements of the University of Liverpool for the degree of Doctor in Philosophy.

December 1992.

ABSTRACT.

Heating, ventilating and air-conditioning (HVAC), system design optimisation problems can be solved through the application of direct search methods. This thesis develops two such optimisation methods for use with HVAC system design; the complex method and a penalty function method with pattern search. The search methods have different approaches to the problem, the complex method rejects infeasible points, whereas the penalty function attempts to prevent constraint violation. Both algorithms where developed to successfully solve small scale HVAC system design optimisation problems, which displayed the main characteristics of such problems.

Limitations of the complex method around constraint functions, and its relatively slow search speed, prevented the further development of this type of search for large scale HVAC design problems. Conclusion of the development of the penalty function method with pattern search lead to significant improvements in the performance of the algorithm.

The formulation of a unique algorithm for solving HVAC system design optimisation problems is described within this thesis, and it is recommended that an algorithm which treats problem variables and constraint functions more effectively to direct the search toward the solution be developed fully. The reduction in size of the optimisation, simplification of the system simulation and use of genetic algorithms to assist the search are all recommended future developments.

ACKNOWLEDGEMENTS.

The author would like to thank Dr. J.A.Wright for his advice and assistance and never ending encouragement throughout the course of this research. Thanks also to the many other research students in the Dept. of Building Engineering and Architecture who made life a little more interesting through the day. The author would also like to thank the staff of Ferguson Williams Ltd., in particular Alan Jeston for his expert contribution to the figures and diagrams.

A special thankyou to Brendan and John for their friendship and their much needed distractions from work.

My final thanks go to Brian and Diana, and last but by no means least to my best friend Nicky, and to Louis.

CONTENTS. PAGE

CHAPTER 7. THE DEVELOPMENT AND TESTING OF THE COMPLEX ALGORITHM.

CHAPTER 1.

COMPUTER AIDED DESIGN OF HVAC SYSTEMS.

Traditionally the design process of heating, ventilation and air-conditioning (HVAC), systems has been carried out manually by the design engineer, all be it recently with the assistance of software packages to aid such areas as building design load calculations and to model the performance of the chosen system. The introduction of such software may allow the design engineer to evaluate the performance of several alternative schemes. However the process in essence remains manual and will produce a 'workable' design. This thesis investigates the use of computers in the 'optimum' design of HVAC systems.

1.1 Current Trends in the Computer Design of HVAC Systems.

In the past twenty years a multitude of software packages have been developed and marketed to assist the design process for the engineer. The level of complexity offered by these design tools varies greatly, from simple manual methods designed to assist in the calculation of maximum loads on the system, and the subsequent sizing of components within that system, to detailed component based methods which simulate the HVAC system, the equipment control system, the building shell and space and the dynamic interaction among these systems.

Of the manual simulation tools available probably the best known is HEVACOMP (HEVACOMP, 1988), this allows the design engineer to interactively build a database of information with regard to the building fabric and space. From this database load calculations can be carried out and heat losses, heat gains, energy consumption and lighting design can be found for the building. Additionally the HEVASTAR Building Services Package (HEVASTAR, 1988), allows the design engineer to size pipes and ducts, and heating and cooling system components. Although this is not strictly a simulation software package, because it does not reproduce the performance of the system, the software greatly enhances the productivity of the design engineer, allowing more time to explore alternative component selections in the given system configuration and alternative system configurations. It is however unable to simulate the dynamic performance or to investigate part-load performance of the system.

A second level of HVAC system simulation tools which are component based in their approach to the design problem emerged in the 1980's. These software tools allow greater flexibility in the modelling of systems and allow part-load performance to be evaluated. With this technique systems are represented by forming a network of component models based on the design engineers schematic diagrams. Flexibility to specify the components within a given design allows the engineer to move away from the rigidity of specific system configurations such as VAV, dual-duct, fan-coil etc. SPATS (Murray, 1984) is one such component based simulation software package. The software networks the component models by linking the input and output parameters of the components. A set of simultaneous equations formed from the component performance equations is then

solved for a given plant operating point. The SPATS simulation is steady-state and hence does not take into account warm-up time and temperature distribution delays for the system. These time constants for the system are however significantly less than the time constants associated with the actual building fabric model and analysis has been justified on this basis.

Dynamic component based simulations are available (Clark et al, 1985). HVACSIM$^+$, is one such simulation package. Broadly, HVACSIM$^+$ is similar in concept to SPATS in that the component models are networked together to form a system. This simulation package, however, includes a nonlinear dynamic simulation that can model time delays and hysteresis effects on the system. Case studies using HVACSIM$^+$ have taken place (Park et al, 1989) which has proven it's capability to deal with large system applications. HVACSIM$^+$, however, remains primarily a research tool.

The more commercially available simulation software packages such as TAS (Gough, 1986), and APACHE (Oscar Faber Partnership), generally use component based simulation but have simpler models. The obvious advantage of these packages, however, is the enhanced user interfaces adopted.

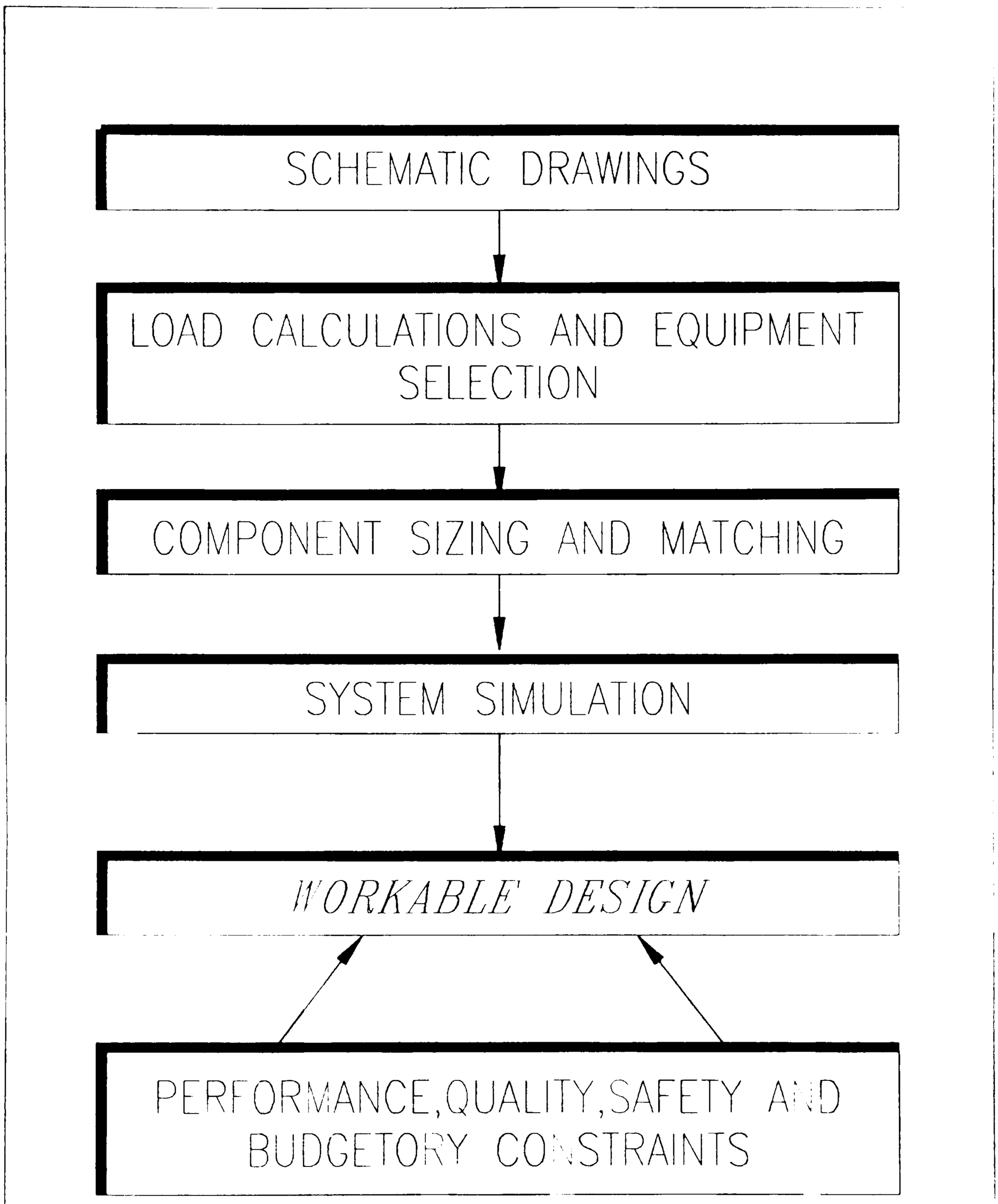
The use of current simulation software in the design of HVAC systems helps increase productivity and is a useful tool to check part-load performance of the system. They allow greater flexibility to the design engineer to explore more system configurations, but rarely lead to major improvements, i.e. they generally produce a 'workable' design as opposed

to a 'optimum' design solution.

1.2 Workable Design.

This manual design process can be generalised. At the initial stage of the manual design process, an appropriate system configuration is selected; the selection is usually based upon the design engineers experience. Drawings are produced, design loads calculated and the components are sized to meet the design loads of the system. Simulation of the system configuration with the sized components follows, this process provides information on the performance and operating point of the system. Provided the system meets the requirements of its' purpose, i.e. it provides adequate heating and cooling, is within the budgetary constraints imposed, and meets all safety and quality standards then the design engineer can justify the initial system selection with the production of a 'workable' design (Stoecker, 1989). In summary, a 'workable' system performs the assigned tasks within any imposed constraints.

Figure 1.1 illustrates the the manual design process that leads to a 'workable' design solution.



The Manual Design Process

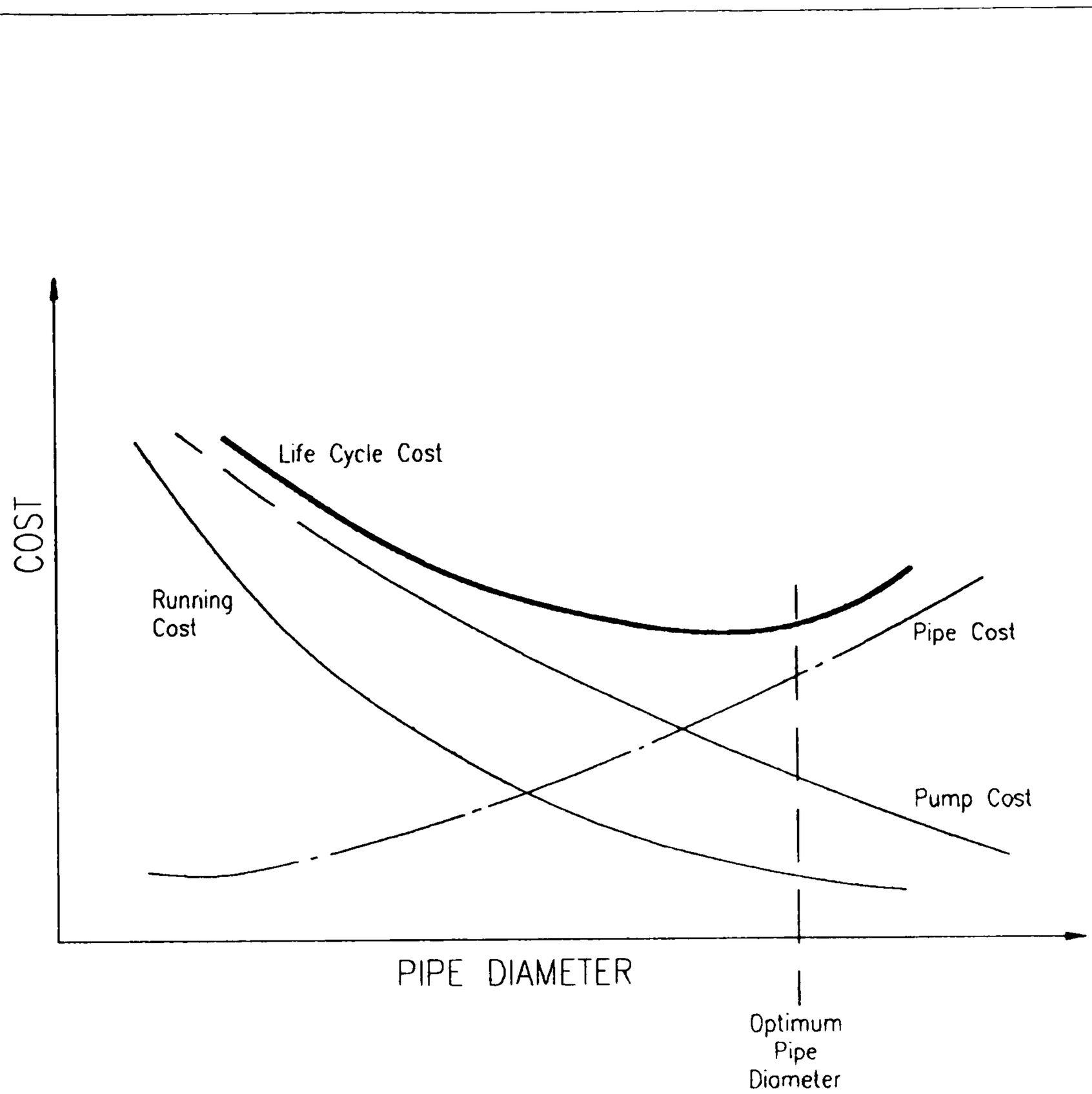
Figure 1.1

1.3. Optimum Design.

The concept of optimum system design is best illustrated by example. Suppose that a pump and pipework is installed to pump water from a basement tank to a tank on the roof. The approach in producing a workable design might be:

1) Allow a nominal water velocity of 1.5m/s.
2) Size the pipe diameter from the required volume flow rate and the water velocity.
3) Calculate the head loss in the system.
4) Size the pump from the head loss and the volume flow rate.

In an optimum system design, the system (pipe diameter and pipe size) is sized to meet a specified criterion, termed the objective function, in this case water velocity. A typical criterion that a design engineer might be interested in optimising could be life-cycle cost, which in turn is a function of first cost, maintenance cost and pumping cost. As the pipe diameter increases so to does the first cost, but due to lower head loss the running and first cost of the pump decreases. Assuming that the life-cycle cost is the sum of all individual costs it can be seen from Figure 1.2 that there is an optimum pipe diameter to give minimum life-cycle cost.



Life Cycle Cost of a Pump Scheme

Figure 1.2

In summary, the optimum system is the 'best' of all workable systems when measured against a criterion. The advantages of optimum design are that it allows the engineer to explore all variations in component size and design conditions within any given system configuration. To manually perform an optimisation using a simulation package would be time prohibitive, the development of computer based software to assist in the search of the possible permutations for a given system configuration would greatly enhance this process. The formulation and solution of HVAC optimisation problems is based on numerical optimisation methods combined with system simulation techniques.

1.4 The Formulation of HVAC System Optimum Design Problems.

Any optimisation is specified by three elements.

1) The problem variables.

2) The objective function.

3) The problem constraints.

Large scale optimisation problems are solved by an algorithmic search for the optimum. The mathematical formulation of an optimisation problem can be described in terms of the problem variables, the problem constraints and the objective function. The constraint and objective functions are related to the performance of the system and therefore, in HVAC system optimisation, a system performance simulation is included in the problem formulation.

1.4.1. The Problem Variables.

Components within a HVAC system can be described by a set of quantities, some of which may be fixed at the outset of the design, but others which are variable and are called the problem variables. It is the values of these variables that an optimisation algorithm will assess and change until a combination is found which gives the optimum value of the objective function of the problem. Examples of problem variables are cooling coil dimensions such as width and height or condenser water flow temperature. The problem variables can be continuous or discrete in nature and are denoted in the formulation of an optimisation problem as follows:

x_i , where i = 1,2,3,.....,n.

or in vector form:

$\underline{X} = (x_1, x_2, x_3, \ldots\ldots\ldots\ldots x_n)$.

where n is the number of problem variables.

1.4.2. The Objective Function.

Conventional design procedures may aim to find an acceptable or workable design which merely satisfies the functional and other requirements of the

design problem. In general there will be more than one possible design which is acceptable and it is the purpose of optimisation to find the best of these acceptable designs. This criterion when expressed as a function of the problem variables is termed the objective function. The objective function can be expressed mathematically as follows:

$f(\underline{X})$, where $\underline{X}$ denotes the problem variables in vector form.

In HVAC design, optimising the objective function involves finding values of the problem variables so as to give a minimum or maximum value of the objective function. Normally a minimum value is the requirement because the objective function is cost related, for example minimum first cost or life-cycle cost.

1.4.3. The Problem Constraints.

In practical HVAC design problems constraints are usually present, for instance air flow across a cooling coil can range from zero and an upper limit after which carry-over of the condensing water on the coil surface occurs. Such constraints on design solutions need to be included within an optimisation problem. Constraints can range in their complexities (Rao, 1987) but are generally summarised by two distinct types, linear constraints and non-linear constraints.

Variables which have a restriction on their value are said to be 'simply bounded' this is a specific form of linear constraint but one which occurs

often in HVAC design optimisation, because component selection within a given system is restricted by limitations on physical dimensions. Generally a linear constraint can be defined as a function which remains linear in more than one variable and can take the form of equality, inequality or range constraints as follows:

equality constraints:	$g_i(\underline{X}) = b_i$	$i = 1,2,.....,m_1$
inequality constraints:	$g_i(\underline{X}) \leq b_i$	$i = m_1+1,.....,m_2$
	$g_i(\underline{X}) \geq b_i$	$i = m_2+1,.....,m_3$
range constraints:	$lb_j \leq g_i(\underline{X}) \leq ub_j$	$i = m_3+1,.....,m_4$
		$j = 1,2,.....,m_4-m_3$

Each g_i is a linear function and b_i, lb_j and ub_j are scalar constants.

A non-linear constraint is a function which is non-linear in one or more of the problem variables and again can take the form of equality, inequality or range constraints as follows:

equality constants:	$c_i(\underline{X}) = 0$	$i = m_4+1,.....,m_5$
inequality constraints:	$c_i(\underline{X}) \leq 0$	$i = m_5+1,.....,m_6$
	$c_i(\underline{X}) \geq 0$	$i = m_6+1,.....,m_7$
range constraints:	$lb_j \leq c_i(\underline{X}) \leq ub_j$	$i = m_7+1,.....,m_8$
		$j = 1,2,.....,m_8-m_7$

Each c_i is a non-linear function and lb_j and ub_j are scalar constants.

1.4.4 The Statement and Characteristics of Optimisation Problems.

Having reviewed the elements of an optimisation problem, it can be stated as follows:

Find $\underline{X}$, which minimises $f(\underline{X})$.

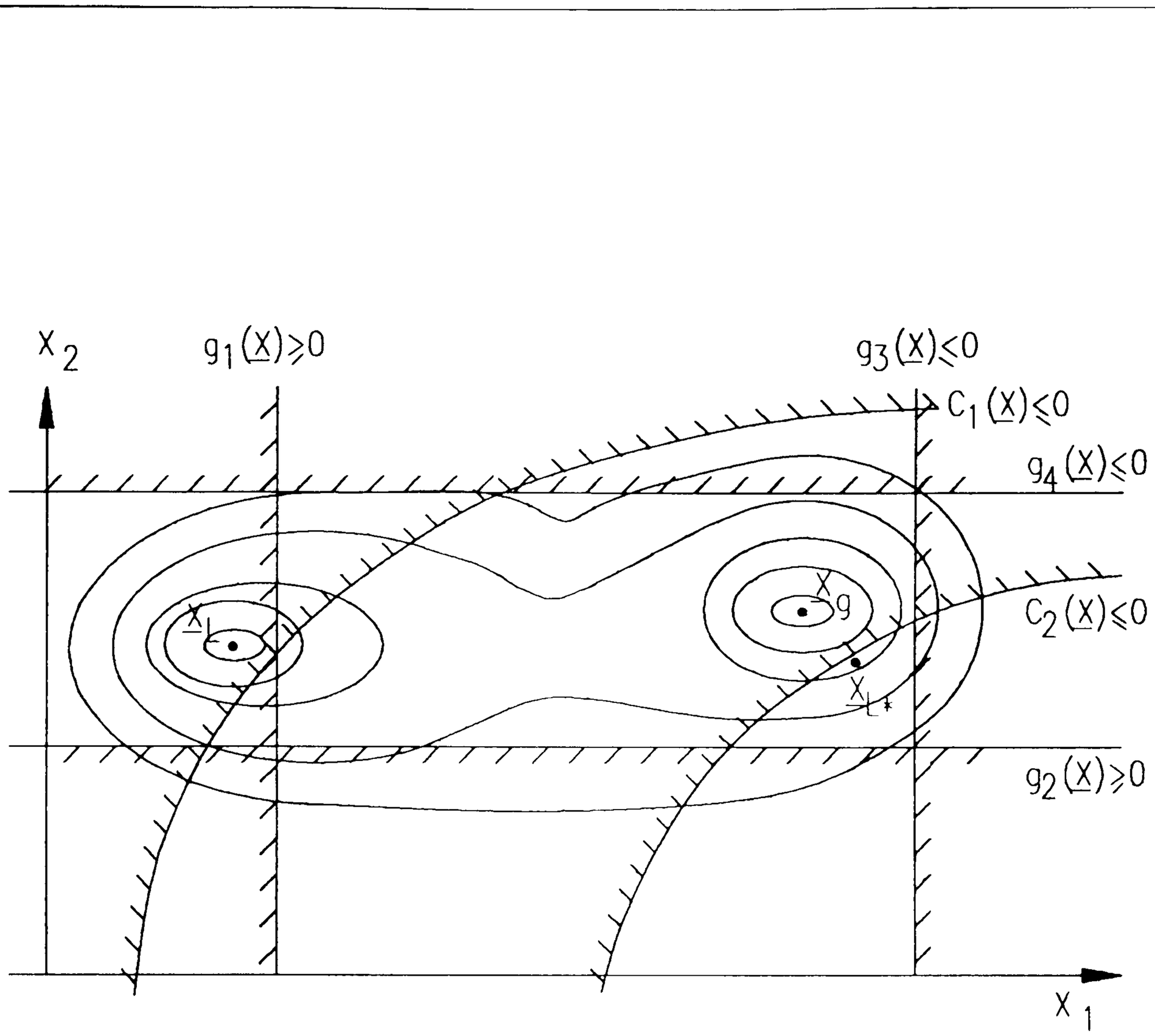
subject to $g_i(\underline{X}) \leq 0$, where i = 1,2,.....,m.

$c_j(\underline{X}) = 0$, where j = 1,2,.....,n.

where $\underline{X}$ are the problem variables denoted in vector form, $f(\underline{X})$ is the objective function and $g_i(\underline{X})$ and $c_j(\underline{X})$ are examples of linear inequality and non-linear equality constraints respectively.

An example two variable optimisation problem is illustrated in Figure 1.3. The contours denote positions of equal value for the objective function. The hatched side of the constraints g_1, g_2, g_3 and g_4 and c_1 denote the infeasible region. $\underline{X}_L$ represents a local minimum which in this case is outside of the feasible region, $\underline{X}_g$ represents the global minimum and as such is the point which the optimisation algorithm will seek.

Constraints can sometimes remove the global minimum $\underline{X}_g$ from the feasible region as illustrated by the additional constraint $c_2(\underline{X}) \leq 0$ shown by a dotted line. In this case the optimisation would seek the position $\underline{X}_L$, which represents the minimum point within the feasible region, lying against the new non-linear constraint.



An Example Two-dimensional Optimisation Problem

Figure 1.3

Optimisation problems are solved using algorithms which will search the objective function by accepting preferred movements (lower values of the objective function for minimisation and higher values for maximisation) that the given algorithm produces. The search algorithm has built-in methods to deal with positions which fall outside of the feasible region. Iteration of the algorithm's methodology will lead the search to the feasible optimum solution. A review of the search methods available is documented in Chapter 4. of this thesis.

1.4.5 The Problem Formulation and System Simulation.

Evaluation of energy related objective functions and system performance constraints requires the simulation of the system performance. The most appropriate form of simulation is a component based simulation, not only because it gives flexibility in defining the system configuration, but also in that it is suited to the formulation of the optimisation problem.

All of the problem variables are derived from the individual components in the system. The majority of the constraint functions are related to the design limits of the components. System cost of energy related objective functions can be formulated from the individual cost and energy use of the components.

Hence, Hanby and Wright, (1989) identified that system optimisation problems can be formed from a component based procedure that has four sub models for each component, a performance model, a cost model, an

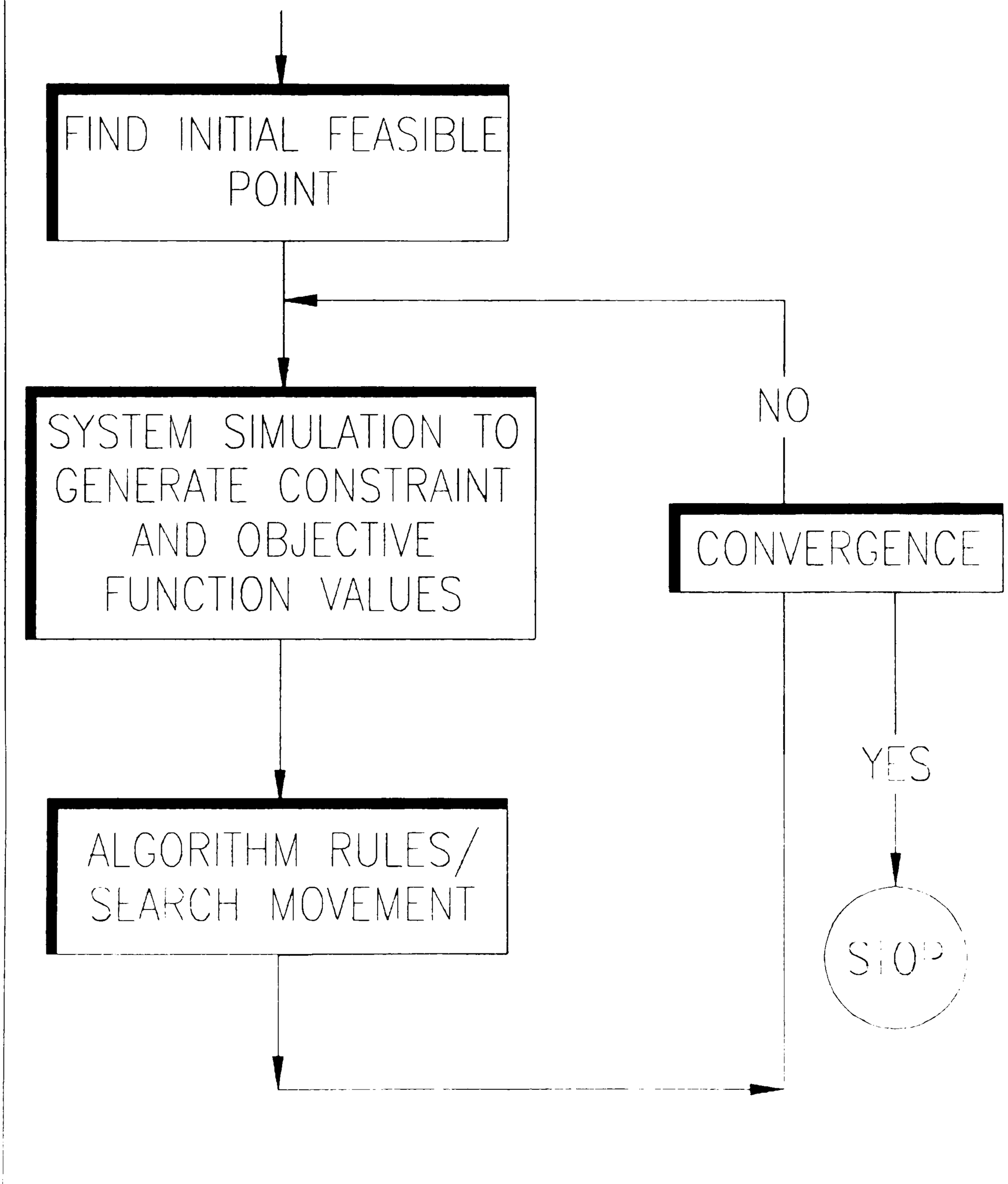
energy model and a constraint model.

The performance model reproduces the performance of the component for use in the system simulation. It includes data for different sizes of components. The cost and energy models produce data for use by the objective function and the constraint model is used for the determination of validity for solution points.

The formulation of HVAC system design problems is well established, but the solution methodology for solving such problems is not, this thesis concentrates on this aspect.

1.5 The Solution of HVAC Optimisation Problems.

The sequence of operation of any algorithm too optimise a HVAC design problem is first to find an initial start point which satisfies all problem constraints and therefore is within the feasible region, simulation of the initial system follows and values of the constraint and objective functions are found. The algorithm will employ rules to produce a new set of search point values, these rules are dependent on the type of optimisation algorithm used, but generally will assess the direction in which to move and the distance of that move. Repetition of this process coupled with a check for convergence complete the operation. Figure 1.4 illustrates this process.



A Generalised Optimisation Algorithm Procedure

Figure 1.4

Two classes of search method exist; direct search methods are heuristic in character and base the next set of search point values on the comparison of objective function value of the new position with that of the previous search position, whereas derivative methods are mathematical in character and employ the derivatives of the objective function to find a search direction.

The use of a homogeneous optimisation algorithm to solve for all optimisation problems would prove inefficient because of the differences in the nature of individual problems. For maximum efficiency an algorithm must be tailored to the individual problem in question. The characteristics of the problem therefore need to be examined fully before deciding on the method of optimisation to be used.

The formulation of HVAC optimisation problems has been developed (Wright, 1986), but to date there has been little work on the development of an optimisation algorithm. The optimisation methods implemented, although able to find solutions, are slow and lack robustness. This has impaired the integration of optimisation methods into the HVAC system design process. The objective of this thesis is therefore to investigate the development of an algorithm that is efficient in solving HVAC system optimisation problems.

CHAPTER 2.

THE DEVELOPMENT OF AN OPTIMISATION ALGORITHM.

Optimisation theory is a well documented subject, (Rao, 1987). Many optimisation methods have been developed ranging in sophistication. These methods have been developed to solve problems ranging from the simplest unconstrained deterministic continuous value problems to highly constrained discrete-value problems. In the development of many of these methods, however, the computer time required to evaluate the objective and constraint functions has not been considered important. The application of many optimisation methods to HVAC system design problems would prove prohibitive because the time required to evaluate objective and constraint functions is high in comparison to the time required by the optimisation algorithm. Each time that an objective or constraint function is evaluated, the current system performance must be simulated. This alone leads to the inefficient use of generalised optimisation methods in HVAC system design because the overall computer time required in performing the simulation function becomes prohibitive and some of the benefits of the optimisation process are lost in this expense. There is a need therefore, to develop of an algorithm that matches the characteristics of HVAC optimisation problems.

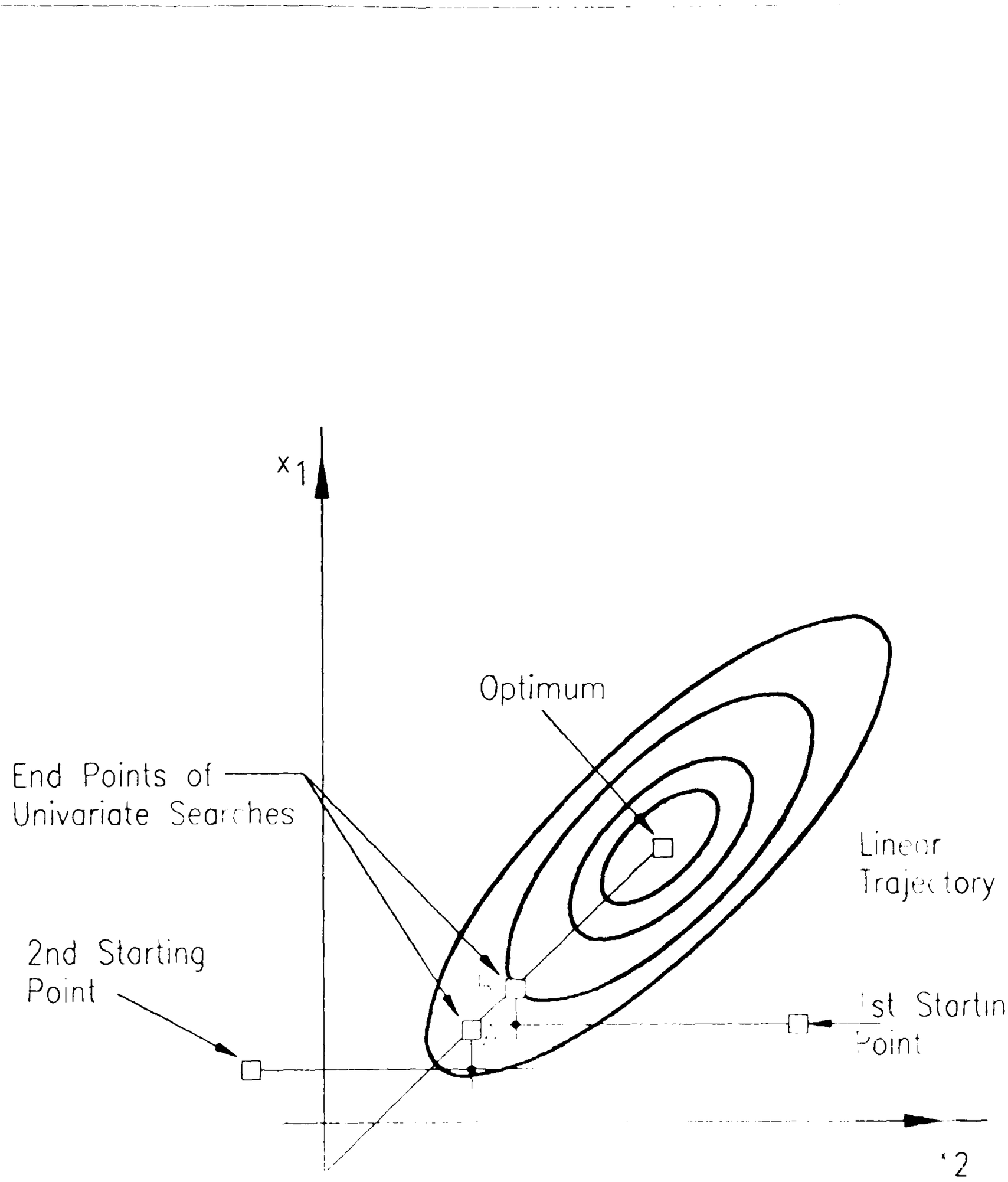
2.1 Previous Work in HVAC System Optimisation.

A review of the literature available on work previously conducted in the field of HVAC system design optimisation is scarce. Only two significant studies have taken place to date.

Leah (1983), conducted research into the optimised design of a chilled water system. A steady state simulation was employed and the objective was to minimise life cycle cost of the system. Leah used a direct search method of optimisation which seeks to find the principle axis of the objective function and then imposes a quadratic approximation to speed the search towards the minimum value. The basic univariate search ideology used searches along each variable in turn until a minimum value of that variable is straddled by the current search point, at this stage a quadratic approximation is made to the last three search points to produce a parabola the minimum of which approximates the minimum of the objective function.

Two modifications were made to improve the speed and efficiency of the search. First, a linear trajectory modification was developed which employs two start points that are minimised for the given variables. By linking the two minimum values a principle linear search direction is developed. Figure 2.1, illustrates this method. This method was found to be useful for objective functions which exhibited a straight valley shape, however if the objective function deviated from that shape then it became less efficient. This method is known as the Q1D method.

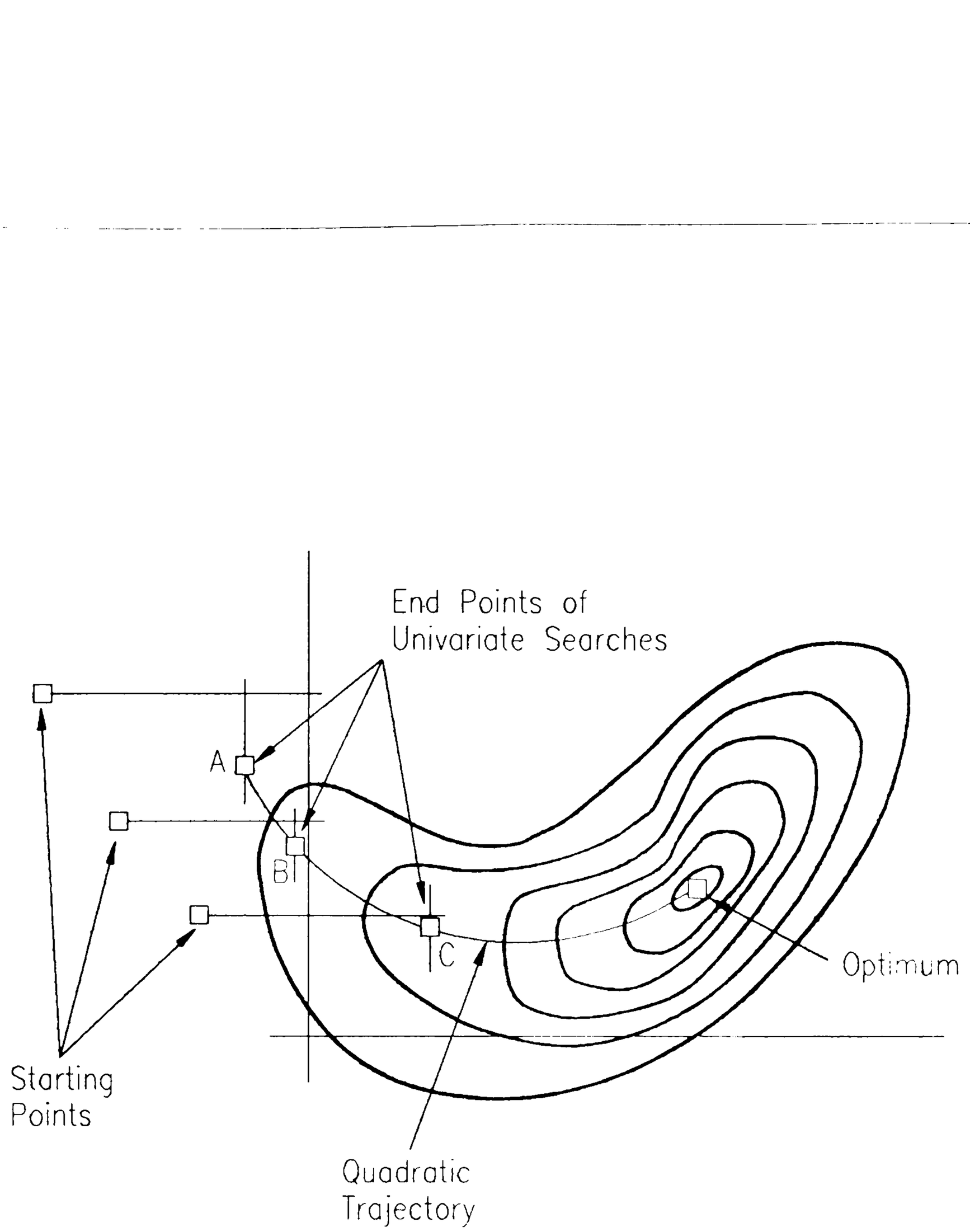
DAMAGED TEXT IN ORIGINAL



The Q1D Search Method
(Leah, 1983)

Figure 7.

A second modification was developed to overcome the inefficiency of the Q1D method where three start points are minimised for each variable. By linking these three minima a quadratic trajectory of the search was developed, as illustrated in Figure 2.2. This method is known as the Q1Q method. In each of the modifications a quadratic approximation is used to find the minimum along the given trajectory.



The Q1Q Search Method
(Leah, 1983)

Figure 2.2

The quadratic approximation and the quadratic trajectory are related but different. The quadratic approximation is the process used to find the minimum along a given path, whereas the quadratic trajectory is merely a way of defining a curvilinear path of the search over the objective function.

The optimisation routines developed by Leah were not fully automated. The design engineer needed to select the type of search required, either Q1D or Q1Q from previous knowledge obtained from the univariate minimisation of the selected starting points. Some further development to automate and control this selection process is required.

Throughout the work Leah concentrated on the solution algorithm and did not investigative the characteristics of HVAC system design problems. The example problem in the research used only continuous variables and had very limited facility to deal with constraint functions. The work overall found that different methodologies are necessary for the varying complexities of the example problems objective function and suggested that further investigation be conducted to produce a fully automated algorithm which would assess the level of complexity of the objective function and switch the search methodology accordingly.

It was suggested that a pattern search method be developed to run under conditions where the Q1D and Q1Q methods failed to be efficient. This has been investigated in this thesis. Development of a penalty function was also considered worthy by Leah and again has been investigated in this thesis.

The second source of literature that exists is the work by Wright (1986). A different view point of the overall problem was considered within this work, where the formulation and characteristics of the problem are developed. The work formulates numerous objective functions which would be used in the appraisal of solutions and describes the use of constraint functions in restricting the solution to a practicable design. The definition of the characteristics of solutions to HVAC system design problems is documented so that a unique algorithm can be developed which complements these characteristics.

Preliminary algorithm development of two optimisation methods was carried out, an exhaustive search method and a pattern search method. The system simulation used throughout the work, called SPATS, was steady state and component based, where models of manufactured components were developed incorporating operating parameters such as controller set points, flow rates and temperatures. The example models used in the development of the algorithms used mixed discrete and continuous variables and were fully constrained.

The simple exhaustive search method was simple used as a 'bench mark' to evaluate the pattern search method against in terms of accuracy and number of objective function evaluations. The pattern search method employed was a modified Hooke Jeeves search (Hooke and Jeeves, 1960), which by performing 'exploratory' and 'pattern' moves could evaluate the direction and distance, respectively, of the search position. The method was applied to a number of small models and was found to perform well over

most objective function surfaces. The search however was found to lack robustness near constraints and the work concluded that further development of the algorithm was required to cope fully with HVAC system design optimisation.

It is concluded that the review of literature specifically focusing on HVAC design optimisation is scarce, both the work by Leah (1983), and Wright (1986), however have positive attributes. Leah concentrated on the solution algorithm alone, whereas Wright dealt more with the formulation and characteristics of the problem. A combination of the two approaches would seem to be an advantageous way to proceed where the characteristics of the problem are found and a robust direct search solution algorithm developed. Both authors suggest that a pattern search algorithm would be fruitful and also both recommend the use of penalty functions to constrain the optimisation problem. These factors are investigated within this thesis.

2.2. Research Objectives.

The most efficient algorithm is one that responds to or matches the characteristics of the optimisation problem being solved. The objective of the research is to develop an efficient algorithm for solving HVAC system optimisation problems. The method of achieving this aim is to :

1. Investigate the characteristics HVAC optimisation problems.
2. Review the optimisation techniques available.

3. Implementation and test selected algorithms against specific assessment criteria.
4. Formulate an ideal algorithm.

2.2.1. Investigation of Problem Characteristics.

In section 1.3 an optimisation problem was described as being formed by three elements, problem variables, the objective function, and problem constraints. Investigation of the characteristics of each of these elements of the optimisation problem is necessary to assess the type of problem involved. An optimisation problem will display the specific characteristics associated with the problem variables, objective function and problem constraints, which describe the problem in quantifiable terms, and are unique to the problem posed. Other characteristics, associated with the solution point and mode of operation of the algorithm being used, may also exist. In HVAC system design problems one such characteristic is that the solution point lies on or near a constraint function. With knowledge of all of the characteristics of the optimisation problem then an algorithm can be assessed against and matched too these characteristics.

2.2.2. A Review of Available Optimisation Techniques.

A review of available optimisation techniques coupled with knowledge of the characteristics of the optimisation problem, will enable the selection of methods that possess positive attributes which match the problem

characteristics. This selection procedure enables large areas of optimisation methodology to be eliminated, and condenses useful methods to a few, hence directing the research to promising areas.

2.2.3. Implementation and Testing of Selected Algorithms against Specific Assessment Criteria.

The selected algorithms will be tested on example models that possess the characteristics of HVAC design optimisation problems. The algorithms are assessed against specific criteria which remain the same throughout this experimental phase. The selected criteria are accuracy, speed of search, and numerical stability and robustness. Accuracy is an assessment of the algorithms ability to find objective function, constraint function, and variable values at each stage of the search. All of these factors can effect the ability of the algorithm to find the optimum solution. Search speed is measured by the number of system simulation calls as this is the dominant factor in the overall computer time. Robustness and numerical stability is a subjective measure of the algorithms ability to negotiate the characteristics of the optimisation problem.

2.2.4. Final Formulation of an Ideal Algorithm.

From the results of the algorithm testing and the results of any subsequent modifications to the algorithms, analysis of the advantages of each method of optimisation can be made. This analysis will lead too the methodology

required to proceed with an idealised algorithm.

CHAPTER 3.

HVAC SYSTEM DESIGN OPTIMISATION CHARACTERISTICS.

Every numerical optimisation problem can be broken down into the three elements of the problem, the problem variables, constraints and the objective function. Each of these elements possesses characteristics which can be used to assess the overall type of optimisation problem. Selection of an algorithm to solve the optimisation problem is simplified with knowledge of the characteristics.

In addition to the general characteristics displayed by the elements of the problem, other characteristics evolve by analysis of the problem solution. These characteristics are specific to the problem type but are equally as important to the selection of an algorithm to solve the problem.

In order to fully investigate optimisation algorithms to solve HVAC system design problems, all of the characteristics of the problem must be defined. Efficient matching of the algorithm to the problem type can then be performed.

3.1. HVAC System Design Problem Variables.

The problem variables are the parameters normally used to describe the selection of HVAC components. These represent the physical size or operating point of the component, or may be associated with the capacity

of the component. For example, the parameters used to specify the selection of a centrifugal fan are impeller diameter and running speed, the impeller diameter representing the physical size of the component and the running speed its operating point. Conversely a boiler is described by its maximum rating which relates to the capacity of the component. A problem variable such as maximum boiler rating forms an indication of both physical size and operating point of the component. A final group of problem variables are the fluid property variables which affect the choice of components and therefore the optimum solution. In practice these variables generally appear as the set points of the equipment controls, for example, the flow water temperature of a boiler is one such problem variable.

3.1.1 The Characteristics of Problem Variables.

The most important characteristic of problem variables in HVAC system design is that the majority are discrete in their nature. The discrete nature of the problem variables exists because of the way that most components are manufactured and marketed. A centrifugal fan for instance is manufactured and marketed having fixed impeller diameters. Similarly a steam boiler is manufactured in discrete maximum rating intervals of for instance 1000kg of steam/hour. Continuous problem variables are rare and those that do occur, such as boiler water flow temperature, can be approximated into discrete intervals. This would avoid the need to devise an algorithm for mixed discrete-continuous problem variables.

3.2 HVAC System Design Problem Constraints.

The most important factor imposed upon a HVAC system design problem is that the end result, i.e. the optimum solution to the problem, operates within all of the design limitations, under all load conditions. Other design constraints arise from Codes of Practice, restrictions on the component configuration and the dimensional restrictions on the components and on the fluid variable values. Section 1.4.3. details the mathematical form of constraint functions possible in numerical optimisation. Table 3.1, gives further definition to constraint functions found in HVAC system design.

MATHEMATICAL CHARACTERISTIC.	MATHEMATICAL FORM.	DESIGN FUNCTION.	SIMULATION LINK.
Simple bound. Smooth nonlinear function. Sparse nonlinear function.	Equality constraint, $G(x) = 0.0$ Inquality constraint, $C(x) \geq 0.0$ Range constraint, $LB \leq C(x) \leq UB$	Variable bound. Fluid limit. Component configuation. Component performance limit. System constraint.	Unlinked. Linked. Weak linkage.

Constraint classification. Table 3.1

The mathematical characteristic and form are self explanatory and have been covered in section 1.4.3 of this thesis, the most commonly occurring constraint function in HVAC system design problems are smooth nonlinear functions or simple bounds for the variable limits.

The design function of most constraints is again self explanatory. The variable bounds constrain the range of the problem variables, for example in the selection of a steam boiler, the problem variable upon which the selection takes place may be the maximum rating of the boiler, this is constrained by the manufacturers lower and upper bounds on the component range. Typically a 'wet-back' shell type boiler comes in a range of maximum ratings of 1000kg of steam/hour to 15000kg of steam/hour. The variable bound constraint is therefore :

1000 kg of steam/hour ≤ maximum rating ≤ 15000 kg of steam/hour.

Fluid limits can take the form of either a simple bound on the problem variables, such as the limiting value of chilled water temperature; or inequality constraint functions that limit fluid velocity. The face velocity, for example, of a cooling coil is often limited to a maximum value so as to remove the possibility of moisture carry-over into the conditioned air. Face velocity can be seen to be a function of two problem variables, coil width and coil height, as follows :

$$\text{Face velocity} = \frac{\text{Air volume flow rate}}{\text{Coil width x Coil height.}} = \text{m/s.}$$

a typical performance constraint function therefore would be:

0.0 m/s ≤ Face velocity ≤ 2.5 m/s.

From this example it can be seen that the constraint function effects both the problem variables of coil width and coil height. It is non-linear in its

nature.

Component configuration constraint functions occur, for instance, because the size and configuration is often limited by the availability of space in a plant room, in designing an air handling unit (AHU), sufficient space must be available to allow the installation and maintenance of the components in the unit. The size and number of fans therefore may be limited for a given size of AHU.

A sparse configuration constraint appears occasionally in HVAC system design problems and is the result of a design configuration restriction. Continuing with the example of the selection of a steam boiler, if the flue of the boiler is specified to be at the rear of the boiler, then the boiler selected would have to have an odd number of smoke tube passes, giving rise to a sparse constraint as follows:

Fractional part of (Passes/ (2 circuits)) = 1/2.

This type of constraint must be considered when selecting and building an algorithm.

The performance constraint function can occur because of a limit on the tested performance of a component. A series of fans, for instance, are tested and produce a family of performance curves the limits to this family of performance curves form an envelope outside of which data on the fans performance is not known. The limits of the performance can be formed into performance constraints.

System constraints arise from the global limits of the system design, a maximum capital expenditure, for example, may be imposed upon a given system design which forms such a constraint.

The linkage to the simulation is an indication of the dependence of the constraint function on the system simulation. Some constraint functions cannot be evaluated without simulating the performance of the system, whilst others can. For instance, evaluation of a fluid velocity constraint is based on the simulated mass flow rate, whereas a limiting dimension of the component can be checked without reference to any simulated variable.

3.2.1. The Characteristics of Problem Constraint Functions.

The types of constraints encountered and described above have vastly different characteristics and as such must be approached in different ways by search algorithms. The variable bounds are linear inequality range constraints and are functions of only one problem variable and as such pose few problems because of their simple form. The fluid limit constraint functions are often non-linear inequality constraints and can be functions of more than one variable. Optimisation algorithms can have difficulty in traversing or negotiating these constraints, particularly when movement of the search is restricted by the discrete interval between variable values.

Sparse configuration constraint functions produce characteristics which are particularly difficult to deal with. These constraints can produce bands or ridges of acceptable or feasible solutions surrounded by bands or ridges

of unacceptable or infeasible solutions. The abnormalities which arise due to sparse configuration constraint functions make investigation of other aspects of the overall design optimisation problem difficult to interpret, because most search algorithms require continuous feasible areas to maintain stability. Sparse constraints, however, occur infrequently in HVAC system design problems and therefore the majority of problems could be solved by an algorithm tailored to the common characteristics of the problem, therefore for clarity these types of constraint function will be omitted in the example models used in this research. The formulation of an ideal algorithm, however, must eventually have capability to deal with such constraint function characteristics and the sparse constraint is considered during the final stages of development of the unique algorithm.

3.3. HVAC System Design Objective Functions.

The objective functions for HVAC system design is typically that which the end user of the system requires. It is the measure by which different system designs and system configurations can be assessed. Wright (1986), identified six such objective functions which HVAC system designers used as design comparators, these are as follows:

1. Net energy consumption of the system.
2. Primary energy consumption of the system.
3. Capital cost of the system.
4. Annual operating cost of the system.
5. Net present value of the system.

6. Payback period of the system.

Although this list is by no means exhaustive it is considered comprehensive enough to draw conclusions as to the characteristics of objective functions in HVAC system design problems.

The objective function is considered to be the function which is most dominant to the design. Two objective functions can be imposed but one is usually dominant over the other, which subsequently acts similarly to a constraint function, for example, the end user may ask for the HVAC system design to be the cheapest with respect to capital cost and to have a net energy consumption of less than a certain value, in this case, capital cost is the objective function and net energy consumption acts as a constraint function. Conversely if the end users requirement was to have the lowest net energy consumption possible and to be within a certain capital cost limitation then the two functions would reverse there roles, with net energy consumption becoming the objective function and capital cost acting as a constraint function. This type of constraint function is in the form of a system constraint.

3.3.1. The Characteristics of Objective Functions In HVAC System Design.

The objective functions used in HVAC system design are predominantly cost or energy related, and as such the optimum solution is always a minimum value of the objective function. There are few objective functions which require maximisation, perhaps the only significant one in addition to those described by Wright (1986), is that of overall system

efficiency although this rarely used as the predominate consideration in the overall design of the system.

All the objective functions implemented in the work by Wright (1986), with the exception of discounted payback period, had in general optimum solutions which tend towards the bounds of the problem variables. Solutions for energy systems consumption and operating cost tend towards the upper bounds of the problem variables because larger sized components are generally more efficient than smaller components which incur larger system losses through friction etc. Conversely, the solutions when capital cost is the objective function tend towards the lower bounds of the problem variables because the smaller the component size the cheaper the capital cost of that component. Although the net present value objective function is a combination of both capital cost and operating cost, solutions tend towards the largest component sizes because operating cost tends to be the dominant factor. Discounted payback period is the only objective function which may have solutions in the mid-range of the problem variables.

Discontinuities in the objective function are common in HVAC system design problems, for example, with an objective function of capital cost, a change in manufacturing technique can cause a discontinuity. If, for instance a boiler is selected from a range of available boiler sizes, in the range 1000 kg of steam/hour to 20000 kg of steam/hour. The manufacturer may only be able to produce up to 15000 kg of steam/hour from a single furnace boiler, after which a twin furnace boiler must be employed. This change in manufacture will cause a discontinuity in the capital cost objective function. In general HVAC system design objective functions can

be described as discontinuous, non-linear functions.

3.4. Problem Characteristics of HVAC System Simulation.

The system simulation is run in order to allow evaluation of the energy related objective functions and of the constraints. Numerical instability can occur for energy related objective functions when a change in the value of a variable produces only a small change in the value of the objective function, a small change is subject to any rounding error which the simulation may impose upon it, causing instability in the objective function. This instability must be avoided because false local minimums can occur in the objective function though an insensitive simulation.

The system simulation should be conducted not just at peak load conditions, which is often the parameter by which the component is selected, but at the various loads that can be expected on the system throughout the year and throughout the day. In practice the extreme load conditions should be simulated. Constraint functions have to be formulated to ensure that a selected component not only can cope with the peak load but also the minimum expected load. For example, if a boiler plant had a peak load of 2000 KW, and a minimum load requirement of 200 KW, then one single 2000 KW output boiler would not meet the full range of load conditions because the turndown ratio is to high. 2 x 1000 KW boilers would be required to meet the minimum load assuming a turndown ratio of 5 : 1.

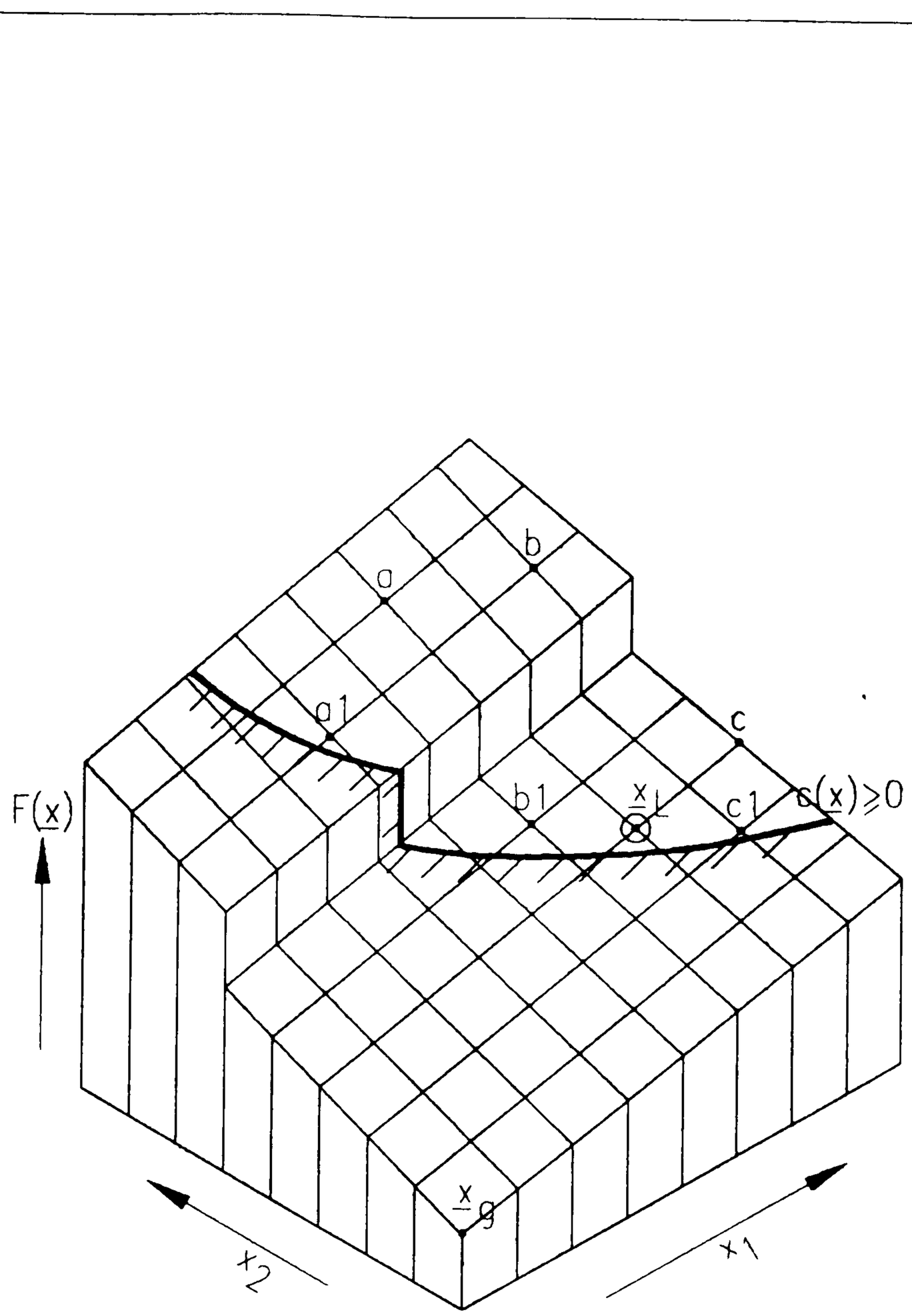
Finally, it is important to remember that simulation of a system is a time

consuming operation, and it is the most influential factor when considering overall computing time in solving HVAC optimisation problems. Analysis of the problem variables, constraints and objective functions, against those required for simulation will show those which are independent of the simulation process. An algorithm should be able to recognise this characteristic so that an objective or constraint function that is independent of the variables involved in simulation is calculated without calling the simulation routine, hence saving computer time.

3.5 Characteristics of Solutions of HVAC System Design Problems.

In section 3.3.1 it was established that because objective functions are either energy or cost related that the optimum solution tended towards the upper or lower bounds of the problem variables. When looking for a global optimum solution for HVAC system design optimisation problems then the solution would be relatively easy to find, employing a solution algorithm which uses simplistic methodology. Introduction of performance constraint functions, discrete variables and discontinuities to the problem, however, adds to its complexity, and the solution algorithm employed equally needs to be more complex to cope with these characteristics.

The main characteristic of solutions to HVAC system design problems is that they lie on one or more constraint functions, or in the case of discrete variables, close to one or more constraint functions. This characteristic becomes apparent if a simple two dimensional example is considered, as illustrated in Figure 3.2.



Example Problem

Figure 3.2

The problem is to minimise the objective function which is represented by the surface of the grid. The problem variables x_1 and x_2 are discrete, the discrete intervals are represented by the grid lines. The objective function is linear and discontinuous in the x_2 direction and has a global optimum solution $\underline{X}_g$, which any numerical optimisation algorithm will initially try to reach. The problem is however, constrained by the non-linear, inequality constraint function $c(x) \geq 0$ thus removing the global optimum solution from the feasible region and producing a local optimum solution $\underline{X}_L$ which for obvious reasons lies near the constraint that divides the feasible region from the global optimum. Depending on the initial starting position of the search the solution algorithm may encounter the constraint function $c(x) \geq 0$ at any position, for example as for the paths aa_1, bb_1, cc_1. On encountering the constraint the solution algorithm would have to have the ability to traverse the constraint function to the optimum solution X_L. Particular methodology is required to perform this task and is particularly important in the selection of solution algorithms.

3.6. The Choice of Search Method.

There are two categories of search method available for numerical optimisation these are direct search methods and gradient search methods. The overall characteristics of the problem dictates the category that can be used.

Direct search methods are heuristic in character basing their search on a comparison of objective function values. Gradient based methods are mathematical in character basing their search strategy on the derivatives of

the objective functions.

The most influential reason for adopting direct search methods to solve HVAC system design optimisation problems, is the behaviour of derivative techniques when used with discrete variables and discontinuous objective functions. As the partial derivatives of the objective functions are unavailable, the implementation of gradient methods would require calculation of the derivatives by numerical methods. These estimates are frequently upset by numerical difficulties, such as rounding errors, which will effect the values of variables, constraint and objective functions, and may hinder convergence of the algorithm, also the discrete interval between variables, limits the interval over which the gradients are calculated, and discontinuities in the objective function will cause further instability. With these problems in mind gradient methods are best avoided. A final point in favour of direct search methods is that because they tend to repeat identical arithmetic operations, it is easier to understand the characteristics of the problem and to assess the ability of solution algorithms to cope with them.

CHAPTER 4.

OPTIMISATION TECHNIQUES.

The two categories of search method, direct and gradient methods, have been discussed briefly in section 3.6. Gradient methods which select the direction of search by the use of partial derivatives of the objective function with respect to the problem variables. These methods, however, are unstable when used with discrete variable problems and discontinuous objective functions and are therefore not suitable for solving HVAC optimisation problems. Gradient based methods are therefore not considered further in this thesis but, a comprehensive assessment of these techniques can be found in such texts as Rao, (1987).

Unlike gradient based methods, direct search methods are stable with discrete variables and are not affected by discontinuous objective functions, an assessment of these techniques with respect to the characteristics of the problem of HVAC system design follows.

4.1 Unconstrained Direct Search Methods.

Direct search methods are those which do not require the evaluation of the partial derivatives of the objective function, but instead rely solely on values of the objective function, and information gained from earlier iterations to obtain a search direction. Although HVAC system design problems are fully constrained a review of unconstrained methods

follows, because essentially the methodology of these techniques does not alter except for the fact that constraint violation handling rules are added to constrain the problem. The more applicable constrained methods are assessed further in section 4.2.

Unconstrained direct search methods can be divided into three classes, tabulation methods, sequential methods and linear methods.

4.1.1 Tabulation Methods.

These methods make the assumption that the optimum solution, $(x_1,x_2,.....,x_n)$ lies within a given region,

$$X_i \le x_i \le X_i+d_i, \quad i=1,2,.....,n. \qquad [4.1]$$

where the X_i and d_i are known.

One basic method of approximately locating the minimum is to evaluate the function at the nodes of a grid covering the region given by the inequalities [4.1]. For example the range d_i of the ith variable x_i can be divided into r_i equal sub-intervals, where r_i is chosen to give acceptable spacing d_i/r_i of the grid lines for this variable. The use of this strategy requires the objective function to be evaluated at the following points.

$$(r_1 + 1), (r_2 + 1),.....(r_n +1) \qquad [4.2]$$

The smallest function value found is taken as the minimum.

Often with HVAC system optimisation X_i and d_i are known, but a well defined feasible region is not known, constraint functions complicate the feasible region and without standard methodology for constraint violation, such as simple rejection of infeasible points, these methods are rendered useless for the needs of the problem. An exhaustive search is one form of tabulation method, it uses the whole range of each variable so that d_i becomes the full range of the variables x_i, it is useful to act as a gauge of efficiency for other search methods as it will explore all possible combinations of the problem variables and acts as a bench mark from which to assess other search methods. An example of other tabulation methods of direct search are random search methods which uses random numbers generated in the range 0 to 1 to find randomly selected solution points. After a sufficiently large number of such solution points have been found and their objective function values determined the solution point with the lowest objective function value is taken as the minimum solution.

4.1.2 Sequential Methods.

Sequential methods are those methods which investigate the objective function by performing objective function evaluations at the vertices of a geometric configuration in the space of the problem variables.

Factorial designs are one such sequential method. The objective function is evaluated the the vertices of a geometric shape such as a cube. The search continues to move the shape in the direction of the lowest objective

function vertex. For a cube, eight objective function evaluations would be required, one at each vertex, for each movement of the cube. This is a major disadvantage because of the time involved in simulating the system performance at each evaluation. The number of objective function evaluations can be reduced with fractional factorial designs, but the number of evaluations still remains excessive. Both methods are described fully in (Davies 1954).

A second and far more promising sequential method is the Simplex method devised by Spendley, Hext and Himsworth (1962). The objective function is evaluated at n+1 mutually equidistant points in the space of the n problem variables, these points forming the vertices of a regular shape called the simplex. Therefore a regular simplex in two dimensions consists of an equilateral triangle, whilst that in three dimensions consists of a tetrahedron.

The method is initiated by setting up a regular simplex in the space of the problem variables, followed by the evaluation of the objective function at the vertices. The basic iteration then proceeds according to the following rules:

1. Find the vertex with the worst objective function value and reflect this vertex in the centroid of the remaining vertices, thus forming a new simplex.

2. Evaluate the new vertex and continue again with step 1.

If the new vertex happens to be the vertex of the worst objective function

value in the new simplex, this search procedure would cease to progress, and oscillation between the last two simplexes would occur. To prevent such an occurrence, a further rule is introduced:

3. If, at any stage, the vertex selected by step 1 is the most recently introduced vertex of the current simplex, reflect the vertex with the next worst objective function value instead.

A typical performance of this basic simplex method is illustrated in Figure 4.1. A two variable problem is shown, the task is to find the minimum value represented by the contour lines. With these simple rules and additional criteria for convergence the simplex method by Spendley, Hext and Himsworth operates.

At any iteration of this method only one evaluation of the objective function is made, and hence only one system simulation would be necessary reducing computer time to a minimum. The characteristics of HVAC system design optimisation, however, are more complicated than the simple example shown in Figure 4.1 where a smooth concave objective function is encountered. The objective function can be discontinuous and not necessarily concave in shape. The simplex method is, however, worthy of further investigation because the methodology used can be converted successfully into a useful constrained method.

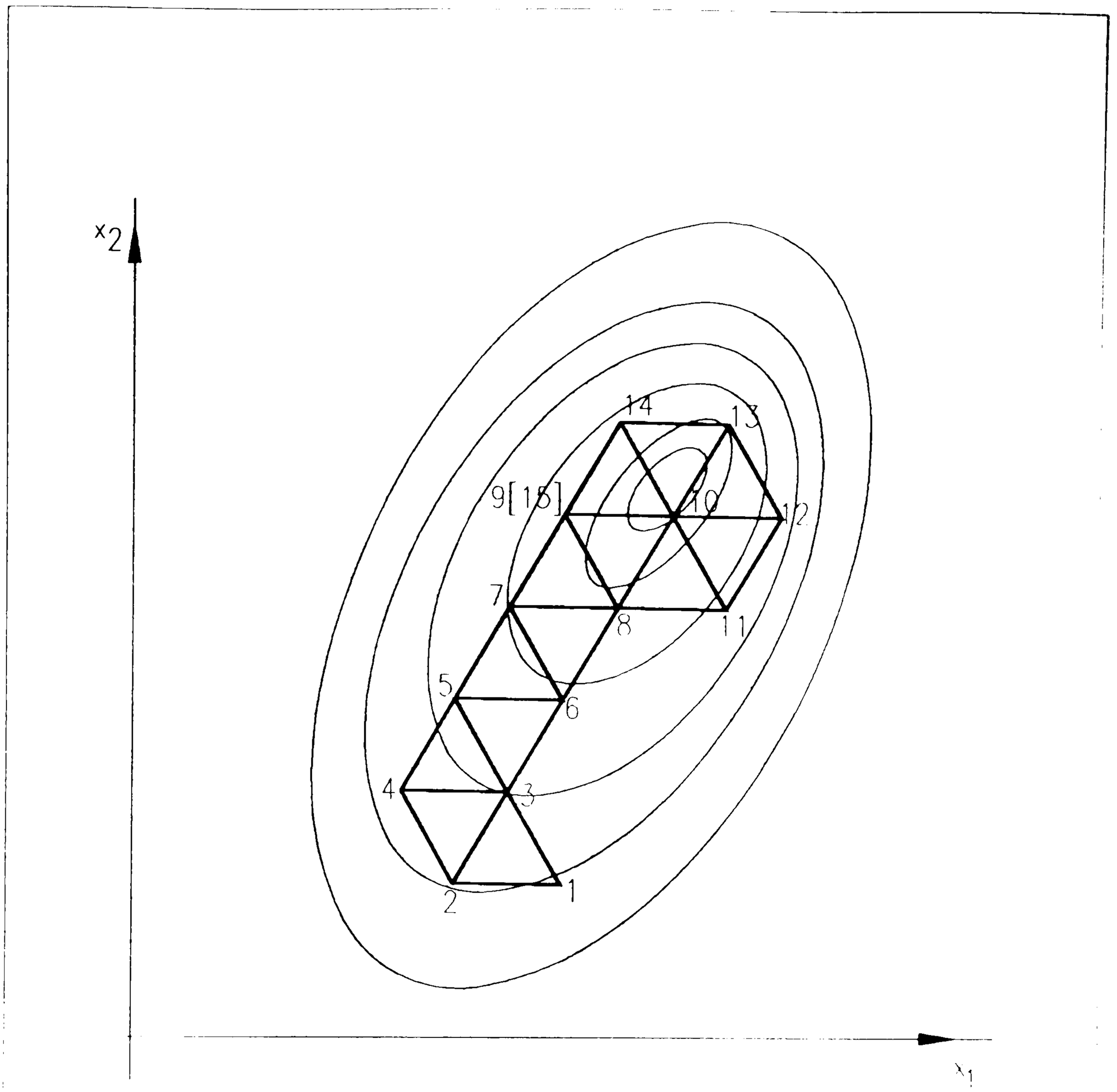


Figure 4.1

The simplex method by Spendley, Hext and Himsworth proceeds at a prescribed rate dependent upon the size of the regular simplex and as such is not sensitive to the changing environment of the objective function of HVAC system design problems. Modifications to this method were developed firstly by Campey and Nichols (1961) and to a greater extent by Nelder and Mead (1965) allowing more flexibility in the simplex design, and forming the most efficient of all the current sequential techniques. In this variation there are three basic operations, namely reflection, expansion and contraction. The general iteration of this method by Nelder and Mead can be described as follows. Let the vertices of the simplex be V_i and the corresponding objective function values be f_i, where i = 0,1,.....,n. Let g,h, and s be respectively the subscripts of the vertices for the greatest, next largest and smallest objective function values, V is the centroid of all vertices excluding V_g. This initial set up in two dimensions is illustrated in Figure 4.2.

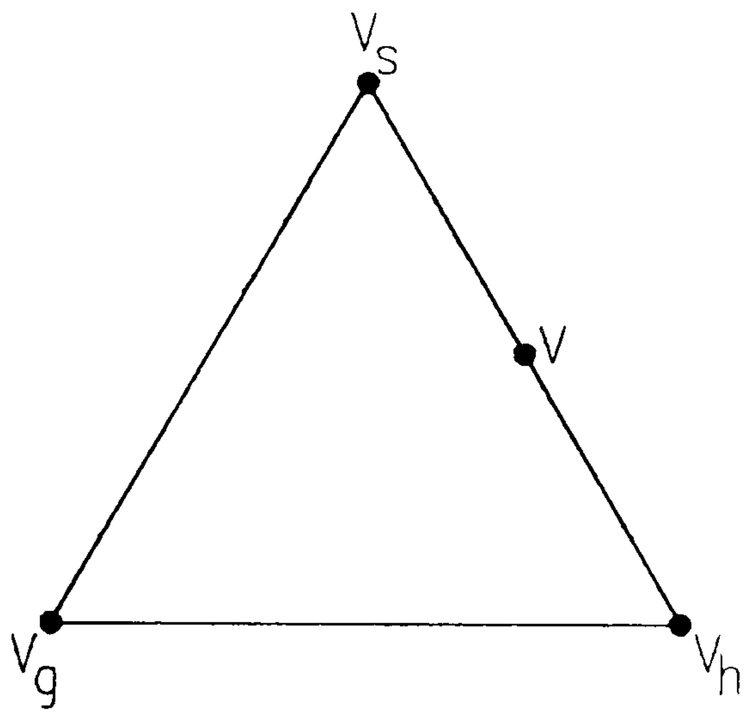
Initially V_g is reflected in V to give a new vertex V_r, where,

$$V_r = (1+å)\ V - åV_g$$

in which å is a positive constant called the reflection coefficient. V_r is therefore on the line joining V_g and V, on the side of V opposite to V_g, such that,

$$\frac{[V_rV]}{[VV_g]} = å$$

where for example $[V_rV]$ denotes the distance from V_r to V.



The Initial Simplex for the Nelder and Mead Method

Figure 4.2

If $f_h > f_r > f_s$, then V_r replaces V_g and the basic iteration continues.

If the reflection produces a new minimum, that is $f_r < f_s$, then it seems worthwhile to investigate whether a further step in this direction will also be successful. Therefore a new point, V_e, on the extended line $V_g V V_r$ is calculated, where

$$V_e = çV_r + (1-ç)\, V$$

in which the expansion coefficient, ç, is given by

$$ç = \frac{[VeV]}{[V_rV]} > 1$$

If the new objective function value f_e is less than f_s, then the expansion has been successful and V_g is replaced by V_e. Otherwise the expansion is considered to have failed, and V_g is simple replaced by V_r.

If reflection gives a point for which $f_r < f_g$, then V_r replaces V_g, but not otherwise. A new point V_c is calculated on the line joining V_g and V as follows:

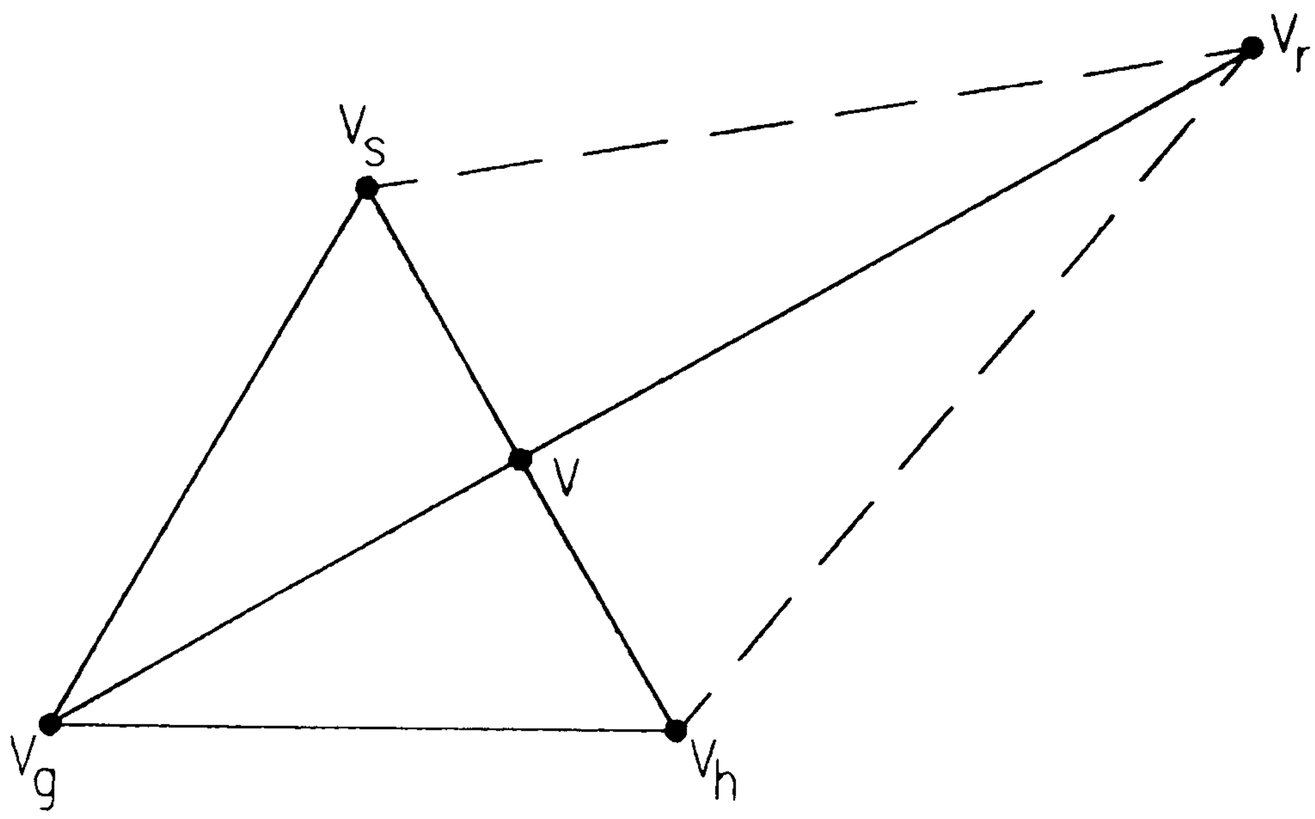
$$V_c = ßV_g + (1-ß)V$$

where the contraction coefficient ß is given by

$$ß = \frac{[VcV]}{[V_gV]} \qquad 0 < ß < 1.$$

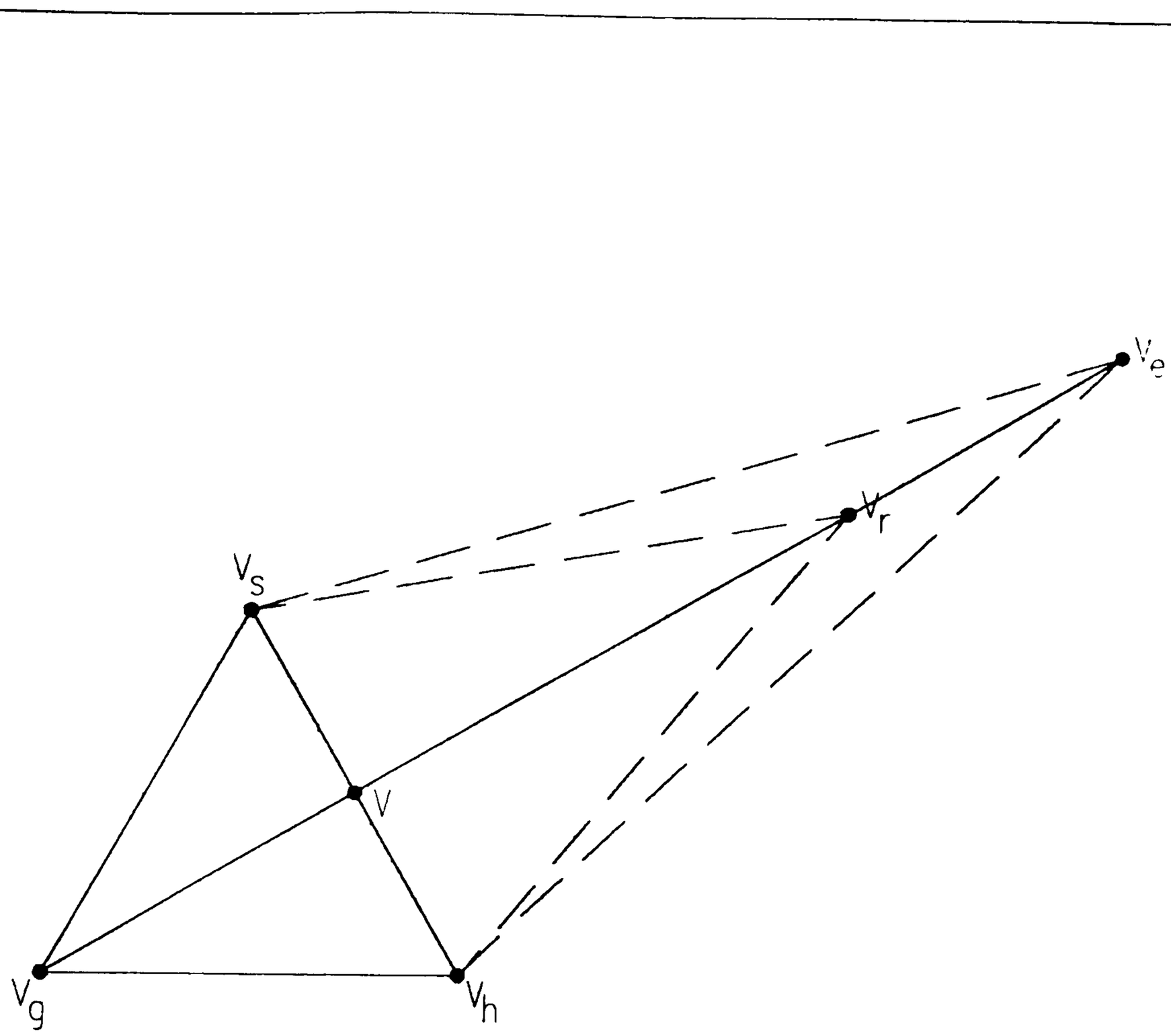
If $f_c < f_g$, the contraction is considered successful, V_c replaces V_g and the basic process is recommenced. If the contraction is not successful, a new configuration is set up by halving the distance from V_s of all the vertices of the simplex before the basic process is continued, hence this acts as a convergence rule. Additional convergence rules complete the process. The basic steps of reflection, expansion and contraction are illustrated in Figures 4.3, 4.4, and 4.5.

This method by Nelder and Mead is considered of interest to HVAC system design problems because it is able to adapt to the changes of the objective function by the expansion and contraction steps and which also allows the search speed to vary.



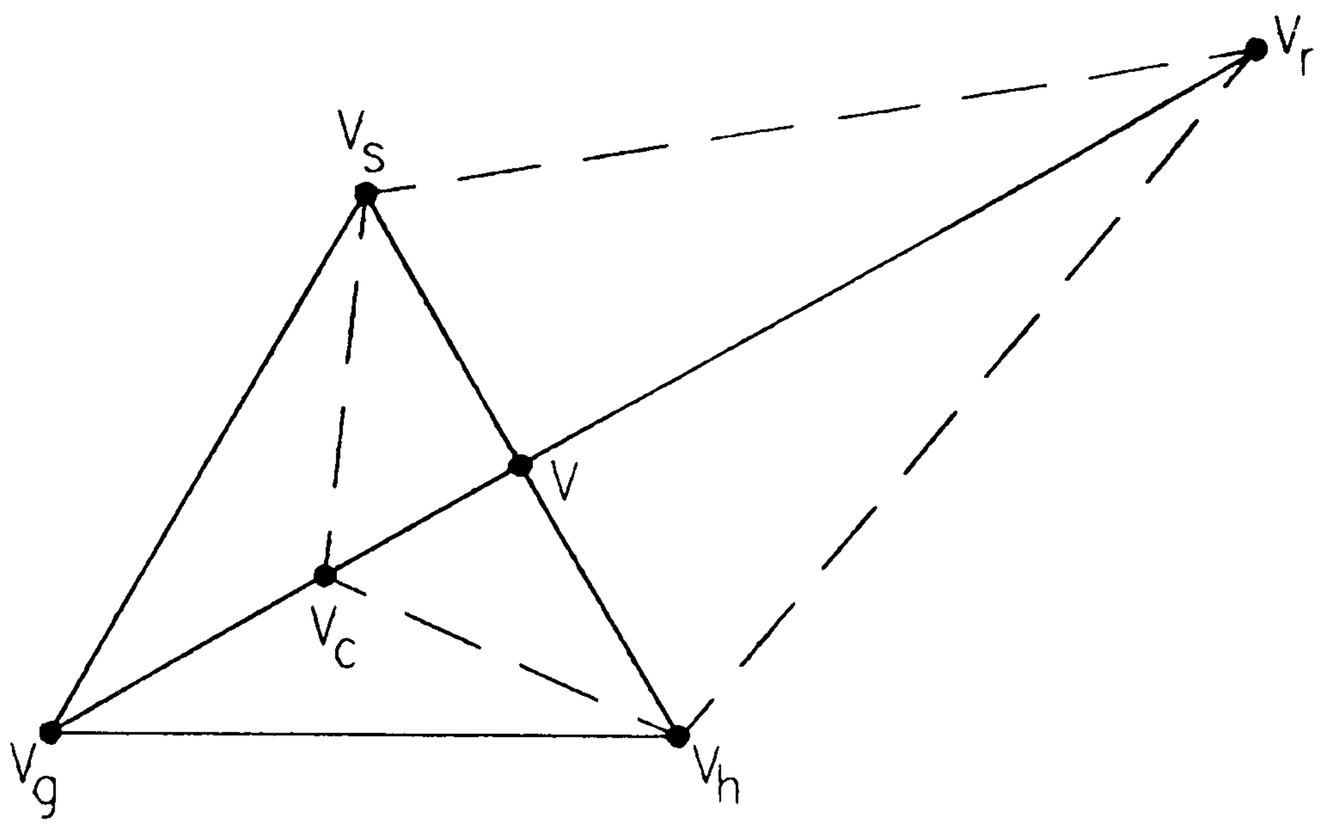
The Reflection Step, V.

Figure 4.3



The Expansion Step, V_e

Figure 4.4



The Contraction Step. V_c.
if $fc < fg$

Figure 4.5

4.1.3. Linear Methods.

Linear method are those techniques in which a set of direction vectors are used throughout the search. Explorations are made along these directions, called axial probes, and the action taken in directing the search is governed by the outcome obtained.

These methods range in there complexity from a simple alternating variable search method which evaluates each variable direction in turn, to more complex methods such as Rosenbrock's rotating coordinate method (1960) and Powell's method (1964) which is based upon conjugate directions.

A method previously investigated (Wright, 1986) and implemented with a certain amount of success for HVAC system design problems, is the Hooke and Jeeves (1960) pattern search method.

This method like many other linear methods attempts to align a search direction with the principal axis of the objective function. In this attempt two strategies, known as exploratory and pattern moves, are used alternately.

The initial starting point within the feasible region forms the first base point b_1, at which the objective function value is f_1. In the exploratory move each variable is considered in turn, an axial probe of a given distance is made in the positive direction of the variable. If a smaller objective function results from this step then the point is accepted and becomes the current point from which the search considers the next

variable. If a greater objective function value is obtained, however, then the point is rejected and an axial probe in the negative variable direction is made from the base point b_1. Again only if a smaller objective function value is obtained will the point be accepted and become the current point, otherwise it is rejected, in either case the next variable is then considered in the same manner. The exploratory move is completed when all variables in turn have been considered. The final current point becomes the new base point b_2, with the corresponding objective function value f_2.

The exploratory move results in the search moving from b_1 to b_2. The pattern move is designed to assess whether there is advantage in continuing in the same direction. The pattern move is made to a point $2b_2 - b_1$, effectively doubling the exploratory move in the same direction. Assessment of the new current point is not compared with the previous base point but a set of exploratory moves are performed from this point firstly, to yield a new base point b_3, at which the objective function value is f_3.

If $f_3 > f_2$ the pattern move plus additional exploratory moves is deemed to be a failure and the whole procedure is recommenced from the base point b_2. If, however, $f_3 \leq f_2$ then the direction from b_2 to b_3 is considered worthy of further investigation, and a new pattern move is made to the point $2b_3 - b_2$, and the process continues with a set of exploratory moves starting from this point, as before.

The above rules are applied until a set of exploratory moves about a base point (as opposed to a point obtained from a pattern move) all fail. At this point either a minimum has been reached or a reduction in the axial probe

lengths is necessary for the search to continue. Convergence is assumed when the step-length of the axial probes reach a preassigned value for each of the problem variables.

The caution of repeated exploratory moves lends itself to the solution of HVAC system design problems as although the direction of the optimum can be well defined, too rapid a progress can result in difficulties when solutions are rejected though the violation of constraints.

4.2. Constrained Optimisation using Direct Search Methods.

The general review of direct search optimisation methods so far has not considered the constrained problem. The optimum solution of HVAC system design problems often lies on or near constraint functions and so careful consideration and handling of constraint violation by any of the previously documented direct search method is necessary. The search method also requires the ability to traverse along a constraint as an optimum solution may be some distance from the position where the search method first encounters the constraint.

Two methods of constraint handling are considered in general optimisation theory, one method replaces the violated point back inside the feasible region and rejects the violation, the other method places a penalty upon the objective function discouraging the search from violating the constraints. Each constraining technique is considered, with the particular unconstrained direct search methods recommended from section 4.1.

4.2.1. Constrained Optimisation by Rejection of Infeasible Solutions.

Simple rejection of infeasible points has proved to be an unreliable constraint handling technique when used on HVAC system design optimisation problems (Wright 1986), a more rigorous method of constraint handling is required. One such method can be applied to the sequential simplex method by Nelder and Mead (1965), described in section 4.1.2. The method was developed by Box (1965) and called the Complex method.

The Complex method uses a geometric figure of $k > n+1$ trial points (called the simplex) where n is the number of problem variables. Given an initial feasible solution point, the remaining k-1 points are generated to randomly lie within the bounds of the problem variables. If a variable bound is violated then the violated point is moved to just inside of that variable bound. If the remaining constraint functions, however, are violated then the trial point is successfully moved halfway back towards the centroid of the existing points in the simplex (of which there is at least one, the initial point), until the point becomes feasible.

Once the simplex of the feasible solution points has been generated, the objective function is evaluated at each point and the worst solution point determined (that with the greatest value). Movement of the search is similar to that of the simplex method, however, extra rules are applied to each new trial point to constrain the problem.

1. If the trial point violates a variable bound, the point is relocated just inside the violated bound.

2. If a constraint function is violated, a new point is constructed by successfully moving the point halfway back towards the centroid of the remaining points until it is feasible.

3. When the new trial point satisfies all of the constraints, the objective function value is evaluated and process repeated unless the trial point is the worst point of the new simplex, in which case a move halfway back towards the centroid of the remaining points is made until it is no longer the worst point.

The work by Wright (1986) concentrated on the Hooke and Jeeves method and used only simple rejection of infeasible points which subsequently proved to be unreliable. The Complex method is a different type of direct search method to that used by Wright and employs more sophisticated constraint handling. The method appears to be suited to the characteristics of HVAC system design problems and will be pursued during this research.

4.2.2. Constrained Optimisation by Penalty Function Method.

There are two penalty function methods commonly documented in standard texts (Rao, 1987), these being the method by Rosenbrock, (1960) and the method by Carroll, (1961). The Rosenbrock method uses the technique of implementing a boundary zone inside all variable bounds and constraint functions, and should trial points enter this boundary zone then a penalty is applied to the objective function making the trial point less desirable to the search. The method by Carroll, called the Created

Response Surface method can be used successfully with many linear direct search methods and may well assist in the use of such methods when applied to HVAC system design problems. The work by Wright indicates that the linear direct search Hooke and Jeeves method used would benefit from more robust handling of constraint functions and application of Carrolls method is thought to be more advantageous than the Rosenbrock method because it is less complicated in its set up and it avoids the problems involved with the pseudo constraints created by the boundary zones.

The penalty function method developed by Carroll imposes a penalty on the objective function in the following form:

$$f^*(x) = f(x) + r\sum W_i/C_i$$

where Wi is the individual weighting given to each of the i constraints, C_i is the value of the constraint function when in the form,

$$C_i(x) > 0.0$$

and r is the weighting given to the penalty in relation to the objective function f(x), and $f^*(x)$ is the modified objective function value.

As a constraint boundary is approached the value of the constraint tends to zero. The penalty, and therefore the modified objective function, is then increased making any trial point close to the constraint boundary undesirable. It is therefore hoped that the search will not cross the

constraint boundary and will remain in the feasible solution region. Each time the search finds an optimum solution for the given modified objective function $f^*(x)$, the penalty weighting r, is reduced, and the search re-run starting from the previous optimum solution point. Eventually the penalty weighting r, is reduced to zero and has no influence on the objective function, the search therefore will operate on the true objective function to find the optimum, constraint violation, however, has been avoided throughout the search.

4.3. Selection of Direct Search Technique.

Consideration of all the existing direct search methods has shown that few are suitable for HVAC system design problems, of the possible exceptions two methods appear to complement the characteristics of such problems. The Complex method, because its search procedure based upon the simplex method is robust in the feasible region and is adaptable to the different objective function environments. The additional rules that the method incorporates for constraint violation means that the method is reliable about constraint function boundaries. The penalty function method is purpose built for avoiding the difficulties of constraint violation and also regulates the speed at which the search can approach the constraint functions, which proved to be particularly critical for HVAC system design problems when using a linear search method on its own (Wright, 1986). The use of a linear direct search method, such as the Hooke and Jeeves pattern search is recommended to operate with the penalty function method applied to it.

CHAPTER 5.

THE APPRAISAL OF HVAC SYSTEM DESIGN OPTIMISATION ALGORITHMS.

The appraisal of algorithms is a well documented. (Amstral and Poirters, 1989). There are two main reasons why analysis of algorithms should take place. Firstly, it allows the algorithm design to be improved where possible, and secondly appraisal allows the different algorithms which perform the same task to be compared. Traditional methods of algorithm appraisal (Sedgewick, 1991) concentrate on the size of the problem to be solved, from which accurate mathematical formulas are used to provide analysis of the 'average case', the amount of time an algorithm might be expected to take on 'typical' input data, and the 'worst case', the amount of time an algorithm would take on the worst possible input data configuration. This type of analysis, whilst possessing merit, would not give any indication of the behaviour of a HVAC system design optimisation algorithm.

There is a need to develop other methods of appraisal that assess the behaviour and performance of the search algorithms. Monitoring of the algorithms performance throughout the search is required, not just an overall assessment of computer time taken which is the approach adopted by traditional methods.

5.1 Useful Criteria with which to Assess HVAC System Design Optimisation Algorithms.

Much of the previous work in optimisation of HVAC system design use analysis which is based upon the speed of the search to the optimum solution. This indeed is a critical factor in the overall acceptability of the algorithm to the problem (Leah, 1983). However, this should not be the limit to the appraisal of a search algorithm. Many other factors are important when looking at the overall acceptability and applicability of an algorithm. The algorithm must show good speed of search but must also be able to deal with the rigours of the variable, constraint and objective function characteristics of the overall problem. A fast search algorithm is of no use alone if it cannot adequately negotiate all characteristics. Many algorithms are designed to cope with specific types of problem, for instance, the work by Leah (1983), dealt solely with concave shaped objective functions which were smooth and continuous in character, we know, however, that many other objective function characteristics can occur and algorithms such as the quadratic search method employed by Leah (1983), would fail given some of these alternative characteristics. Appraisal criteria of the algorithms performance for all problem characteristics must be developed.

The chosen criteria developed for appraisal of HVAC system design problems are:

Accuracy of the search.

Speed of the search.

Numerical stability of the search.

Robustness of the search.

Accuracy and speed of search are both quantitative forms of algorithm appraisal, where as, numerical stability and robustness give a qualitative review of how the search algorithm is proceeding.

5.2. Quantitative Appraisal Criteria.

Accuracy and the speed of the search algorithm are both quantitative appraisal criteria which enable direct appraisal of different algorithms that perform the same task.

5.2.1. Accuracy of the Search Algorithm.

There are obvious reasons for the appraisal of any optimisation algorithm by accuracy of the search as it proceeds and accuracy at the optimum solution point. There is little reason for using an algorithm for optimisation which in effect does not reach the true optimum solution point because of its inaccuracy.

Every solution point found by an optimisation algorithm for HVAC system design, results in constraint and objective functions being evaluated. The constraint function values give the algorithm information about the validity of that solution point and the objective function value provides information for the next iteration of the search algorithm. Accuracy of these function values is therefore critical for the search to proceed correctly.

It is critical that the variable values found by the algorithm throughout the search procedure are accurate. Should a variable have a large influence on the value of the objective function then inaccuracy in the variable value will cause an inaccurate objective function value. This inaccuracy subsequently may cause the objective function to distort away from the true function shape, and any incorrect information being fed back into the algorithm from this function value, at any stage of the search, may cause the next search iteration to proceed in a direction which is not actually defined by the true objective function value. This inaccurate search move could result in a false optimum solution point being found by the algorithm. This is obviously a worst case scenario, but at the very least the efficiency of the performance of the search algorithm would be impaired.

HVAC system design optimisation problems can have discrete and continuous variables. The discrete variable values throughout the search are easily marshalled for accuracy, by rounding to the nearest discrete interval for each variable. This process of rounding to a discrete interval can be performed by additional subroutines incorporated into the main algorithm flow. Although an algorithm may essentially be built to run with continuous variables only, this transition for discrete variables has been performed successfully throughout this research. Integer formats are used in all algorithms for the discrete variable values to ensure that no computational inaccuracies occur. Double precision formats are used for all continuous variables to ensure the maximum computational accuracy for these variables.

HVAC system components when installed into a system will not give the exact performance anticipated, there will be a difference between the

expected laboratory performance and the actual installed performance of a component. Two boilers of the same size (thermal rating), for example, may not have exactly the same performance characteristics due to practical tolerances on the quality of manufacturing of the boilers. A manufacturer will specify a guaranteed performance for a component, ensuring that the performance is greater than or equal to that specification. Simulation algorithms, however, generally produce ideal solutions, which in the case of HVAC system design is not entirely necessary, a tolerance associated with the manufacturers performance guarantees should be included within the simulation algorithm to give a range of acceptable solutions.

Objective functions are another area where absolute accuracy may not be necessary. One objective function commonly associated with HVAC system design optimisation problems is that of annual energy consumption. Many factors influence energy consumption some of which are not associated with the system components themselves. The annual climatic conditions, for example, is one such factor which will change from year to year, and will directly effect the annual energy consumption of any HVAC system. As stated previously, component performance is not 'ideal' and tolerance of this performance will effect an energy related objective function.

HVAC system design optimisation algorithms should accommodate acceptable tolerances within the objective functions to cater for such installed system inaccuracies. It is the aim of such algorithms to produce a system which is practical, and therefore needs to have these inaccuracies built into it.

5.2.2. Speed of the Search Algorithm.

The speed of an optimisation algorithm to find the optimum solution is a critical factor which dominates its efficiency. The essence of optimisation of HVAC system design problems is to reduce the time involved in finding the best possible solution point, it is therefore essential that the speed of the optimisation search algorithm is included as an appraisal criteria.

Each time that an algorithm calls the objective function or checks a constraint function then simulation of the current system design is carried out to provide necessary information for the function values to be calculated. Simulation of the system is the most time consuming process involved in the evaluation of a new solution point and as such directly effects the overall speed of the search.

As the system simulation routine remain standard for all optimisation algorithms used in the course of this research, a direct comparison of the search speed can be made by assessment of the number of objective and constraint function calls made by the optimisation algorithm. The algorithms used in this research all have counters attached to the objective and constraint function routines to provide information on the number of calls made to each during the course of the search. Direct comparison can be made between each algorithm using this information.

The speed of the search algorithm for HVAC system design problems will be determined by the total number of calls made to the system simulation throughout the course of the search. The algorithm performance must be

such that the use of a automated optimisation design tool enhances the time spent within a design office.

This research has looked at a small scale HVAC system model, and it is not anticipated that the speed of the search of the algorithms developed so far would be adequate for large HVAC systems yet. However, as a data base of different component models is developed, and rules in the form of an expert system are added, then it is anticipated that the speed of the system optimisation will increase to an acceptable practical design level.

5.3. Qualitative Appraisal Criteria.

Numerical stability and robustness are both qualitative measures of assessing the search algorithms. No mathematical or statistical measure can be placed upon these assessments of the algorithm, they measure the ability of the search algorithm to proceed in a stable and efficient manner to the optimum solution. They form an integral part of the assessment of HVAC system design optimisation problems at the initial stage of their development.

5.3.1. Numerical Stability of the Search Algorithm.

Numerical stability of the search algorithm has been chosen as an assessment criteria because of the rigorous nature of the search environment when considering HVAC system design optimisation problems. Constraint functions which form the boundaries of the

environment, discrete variables and objective function discontinuities collectively form a search environment which is imposing for any search algorithm to negotiate. Monitoring of the search algorithm as it proceeds though the optimisation function is necessary to ensure that correct and efficient progress is being made towards the optimum solution at all times.

Numerical stability of the search algorithm is a qualitative assessment and ensures that the search continues in an orderly fashion throughout. By continuously recording the present search with the previous search positions it can be determined weather the search algorithm is behaving in a stable manner. Oscillation between search positions is an example of numerically unstable search. Similarly movement of the search in an erratic manner, such as, increasingly distant search positions in opposite directions is another indication that the search has lost stability.

Numerical stability is often a function of the accuracy of the search positions found. This can readily be envisaged using the example where the objective function surface has a shallow gradient in relation to the problem variables, hence changes in the variable values produce a relatively small change in the objective function value. Instability occurs if the value of the change in the objective function is influenced to a greater extent by the accuracy of the computation than the actual change in the objective function value. Improvement in numerical stability can be made, given that the instability is a result of the computational inaccuracy. Scaling of the solution points may be one such improvement, where, the objective function value is multiplied by say an order of magnitude before computation hence reducing the effect of rounding errors. This would,

however, be detrimental to the speed of the search algorithm as additional subroutines would have to be incorporated to perform and manage such a task in a stable manner. Alternatively the increment by which a variable value is changed could be made dependent upon a set minimum objective function change.

For the purposes of this research it is sufficient to monitor the numerical stability of the search and to improve the search algorithm, if necessary, as instabilities arise.

5.3.2. Robustness of the Search Algorithm.

The robustness of the search algorithm differs from the assessment of numerical stability in that it has no mathematical or numerical link, it is purely a qualitative assessment of the search algorithms progress around the characteristics of the optimisation problem.

HVAC system design optimisation problems have particular characteristics which can cause a given search to fail before reaching the optimum solution. It is known that the optimum solution often lies on or near a constraint function and as such the search algorithm must be strong when assessing constraint violation and correcting the search position when constraint violation occurs. This coupled with the fact that the optimum may not be aligned with the principle axis, or steepest gradient, of the objective function means that the direct search algorithm may have to, not only be able to cope with constraint violations sensibly, but also to traverse a constraint function until the optimum solution is found. The

process of traversing a constraint is one which often presents itself within HVAC system design optimisation problems. Wright (1986), found that failure of his pattern search algorithm was due to the inability of the search to traverse a constraint function, the failure in this instance was also linked to the optimisation problem having discrete variables, another common characteristic of HVAC system design problems.

The robustness of a search algorithm is also linked with the repeatability of the search algorithm, giving a measure of the search algorithms ability to find the same optimum solution if started from the same initial point. Failure to reach the same optimum solution would quite obviously be detrimental to the optimisation process giving no confidence in the search procedure.

The repeatability factor can also be extended to ensure that the same optimum solution is found regardless of the initial starting point.

It is proposed that robustness and repeatability are monitored throughout the search algorithm testing and corrected when necessary.

CHAPTER 6.

THE EXAMPLE OPTIMISATION PROBLEM.

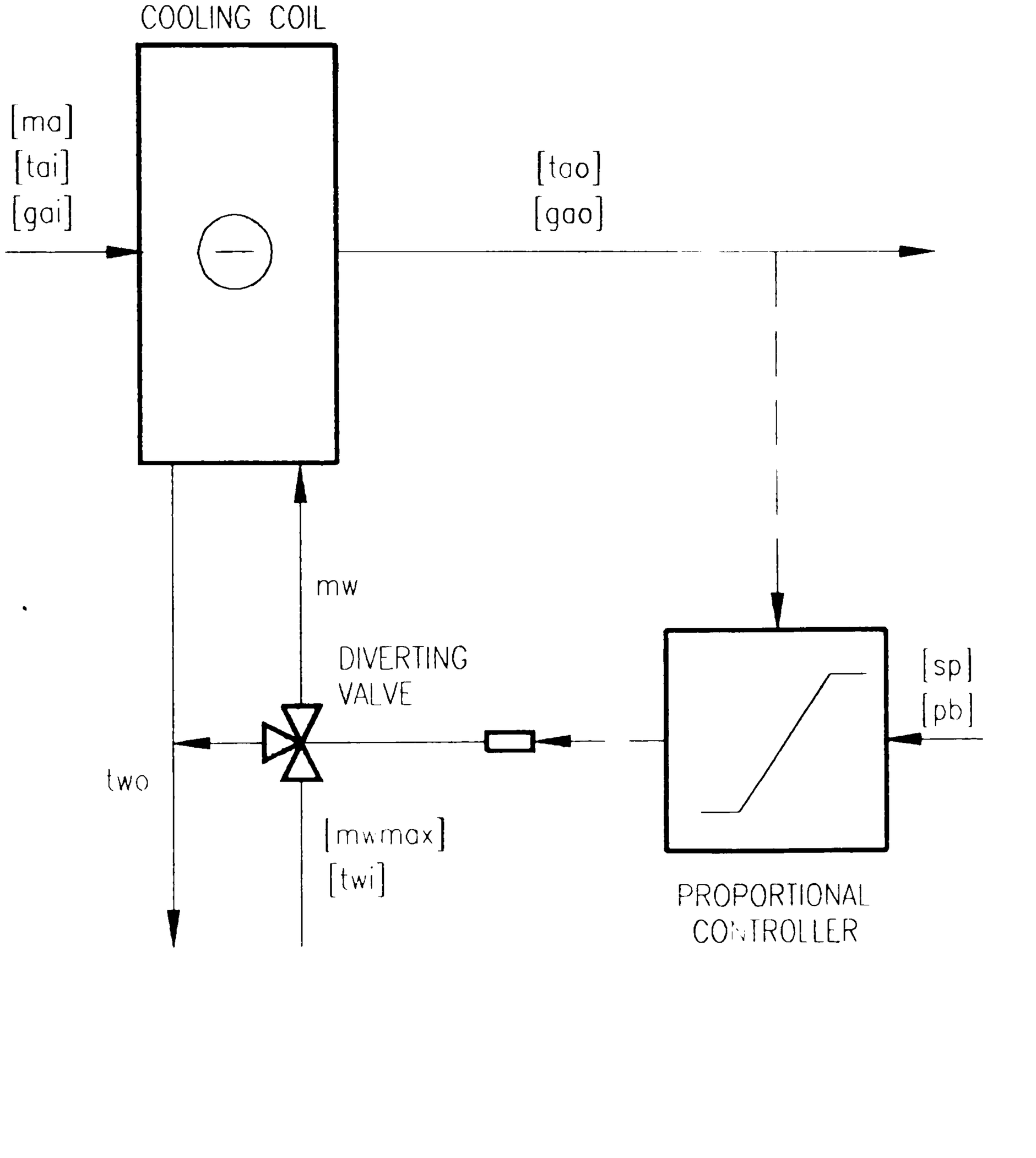
Development of a rigorous optimisation problem to test the direct search algorithms was undertaken. A standardised HVAC system problem was considered to be essential so that a direct comparison of the algorithm performance could be made.

The characteristics of HVAC system design optimisation problems are well defined (Chapter 3). The dominant characteristics has been built into the example problem and therefore the merits of the search algorithms can be assessed against these specific characteristics.

The example problem is the capital cost optimisation of a cooling coil.

6.1 The Cooling Coil Problem.

A cooling coil component model was chosen and developed as the example problem for algorithm testing. It was chosen because it is representative of a typical HVAC component that displays in simple form the characteristics of HVAC system design problems. The model includes the proportional control of the air temperature leaving the coil (tao). Figure 6.1 illustrates the cooling coil component model in schematic form.



The Cooling Coil Model

Figure 5.1

A steady state simulation of the cooling coil problem was developed and linked to the optimisation algorithm routines. This simulation is based on British Standard test methods (BS 5141), and predicts the temperature of the air leaving the coil (tao), the control variable and the moisture content of the air leaving the coil (gao). The water mass flow rate (mw) though the coil and the water return temperature (two) are also calculated by the steady state simulation.

The water inlet condition (twi) to the coil, the set point temperature (sp), and the proportional temperature band (pb) are input giving the desired operating point and controller setting in which the coil will operate. For simplicity the hydraulic performance of the system is excluded from the problem. The control valve is assumed to have a simple linear characteristic.

For simplicity of testing the algorithms, the operating point of the cooling coil has been fixed in the simulation at peak conditions. The simulation of seasonality effects on the load of the cooling coil can be performed producing load conditions for the coil throughout the year, however for the purpose of optimisation algorithm testing it is not considered to be significant, since the underlying, dominant, characteristics of the optimisation problem will not change with cooling load.

The optimisation problem expressed by the cooling coil problem can be defined in terms of the variables, constraints, objective function and system load.

6.2 The Example Problem Variables.

The example problem variables are formed from the physical dimensions of the coil and from the coils operating variables.

Five variables exist in the example problem are as follows :

1) Cooling coil width.

2) Cooling coil height.

3) Number of rows in the coil.

4) Number of water circuits in the coil.

5) Maximum water mass flow rate (mwmax).

The dimension variables are all discrete in nature. The cooling coil height, for instance, is manufactured in 50.0mm increments. Conversely the operating variable of maximum water mass flow rate is a continuous variable. By assigning a discrete interval for the maximum water mass flow rate and rounding the calculated value of this variable to the nearest discrete interval allowable by the constraint functions the variable is converted to a discrete variable. The algorithm tested against the example problem have been set up to deal with discrete variables only. This is justified in that the majority of the problem variables are discrete.

6.3 The Example Problem Constraints.

The constraints on the example model optimisation are formed from the feasible range of each of the problem variables, by design limitations, and

from the desired operating point and controller settings of the coil.

Chapter 3 of this thesis categorises the constraint functions as one of the following: variable bounds, performance constraints or sparse configuration constraints. Variable bounds and performance constraints are present in the example problem but the sparse configuration constraint is not present because of the overriding difficulties involved with this form of constraint. The handling of sparse constraint is considered in Chapter 9.

6.3.1. Example Problem Variable Bounds.

The variable bound constraints and discrete intervals of each variable are as follows:

1) 0.05m ≤ cooling coil width ≤ 2.00m.

2) 0.05m ≤ cooling coil height ≤ 2.00m.

3) 2 ≤ number of coil rows ≤ 10.

4) 1 ≤ number of coil circuits ≤ 40.

5) 0.0kg/s ≤ maximum water mass flow rate ≤ 10.0kg/s.

with the following discrete intervals respectively :

1) 0.05m.

2) 0.05m.

3) 1 coil row.

4) 1 coil circuit.

5) 0.25 kg/s.

6.3.2 Example Problem Performance Constraints.

The performance constraints present in the example problem are as follows:

6) 0.0m/s ≤ Air face velocity ≤ 2.5m/s.

7) 0.0m/s ≤ Water velocity ≤ 1.8m/s.

8) Set point (sp)+ (proportional band(pb)/2.0)-air outlet temperature (tao) ≥ 0.0 C.

Constraints 6 and 7 prevent excessive fluid velocities. The air face velocity constraint simulates the design limit after which carry over of moisture which forms on the cooling coil surface is transmitted into the exiting air.

Constraint 8 is the design limit of the proportional controller this ensures that the controlled variable (tao) is within the given allowable margin determined by the proportional band (pb).

6.4 The Example Problem Objective Function.

The objective function chosen for the example problem is capital cost of the cooling coil. This is a typical objective used in any design function and HVAC design is no exception.

In a competitive market manufacturers are reluctant to release cost data, particularly the mathematical form which governs the scheduling of their price ranges. It is also the authors experience, that the presentation of cost

data is very diverse from manufacturer to manufacturer. These two facts together make the formulation of generalised cost data impossible. For the purposes of this research cost data form a specific manufacturer of HVAC components has been used. A least squares polynomial curvefit of the manufacturers price list was performed to formulate the cost function.

In the example used, the manufacturers cost data, presented, was a function of the physical dimensions of cooling coil width and cooling coil height. A third variable, that of number of rows in the coil, mean that a family of cost curves were produced, one for each discrete interval of coil rows, thus complicating the overall cost function. Future software development may allow alternative and more efficient methods for formulating the cost function, such as, defining one data file for each range of components or price list.

The algorithms to be tested are structured in such a way as to calculate the cost objective function after the simulation of the system thus ensuring the component is correctly sized before the costs are evaluated.

The cost function values are defined in pounds sterling (£), and the full objective cost function is presented below:

For cooling coil height ≤ 1.4m,

Capital cost (£), F = A1 + (A2 x coil width) + (A3 x coil height) +

(A4 x coil width x coil height)

or for cooling coil height > 1.4m,

Capital cost (£), F = 2 (A1 + (A2 x coil width) + (A3 x coil height/2.0) +

(A4 x coil width x coil height/2.0))

where, the values of A1, A2, A3, and A4 are tabulated of the discrete intervals of coil rows as follows:

No of coil rows.	A1	A2	A3	A4
2	92.45	2.74	143.38	83.66
3	94.45	18.91	180.92	98.73
4	99.89	21.23	220.71	121.94
5	89.22	29.60	263.79	138.64
6	78.56	37.97	306.86	155.34
7	91.72	40.66	310.85	184.53
8	104.87	43.34	314.84	213.72
9	94.31	26.60	392.72	248.14
10	83.75	9.87	470.61	282.57

As can be observed by the objective function there is a break point at the cooling coil height of 1.4m. This break represents a manufacturing change or discontinuity. The manufacturer produces cooling coils up to 1.4m in height after which to obtain the desired size of coil two coils of equal size are used.

6.5 Example Problem System Loads and Simulation.

As previously stated the system load on the cooling coil is defined by the condition of the air entering the coil and the desired operating point of the coil. An advantage of using a system simulation in design, is that the performance of the system and the design constraints can be evaluated for all load conditions. However, a single peak load condition has been taken to simplify the assessment of the optimisation algorithms. The peak load conditions are as follows:

1) Air entering the coil temperature (tai) = 27°C

2) Air entering the coil moisture content (gai) = 0.00145 kg/kg

3) Air mass flow rate (ma) = 6.5 kg/s

4) Water inlet temperature (twi) = 8.0°C

5) Controller set point (sp) = 12°C

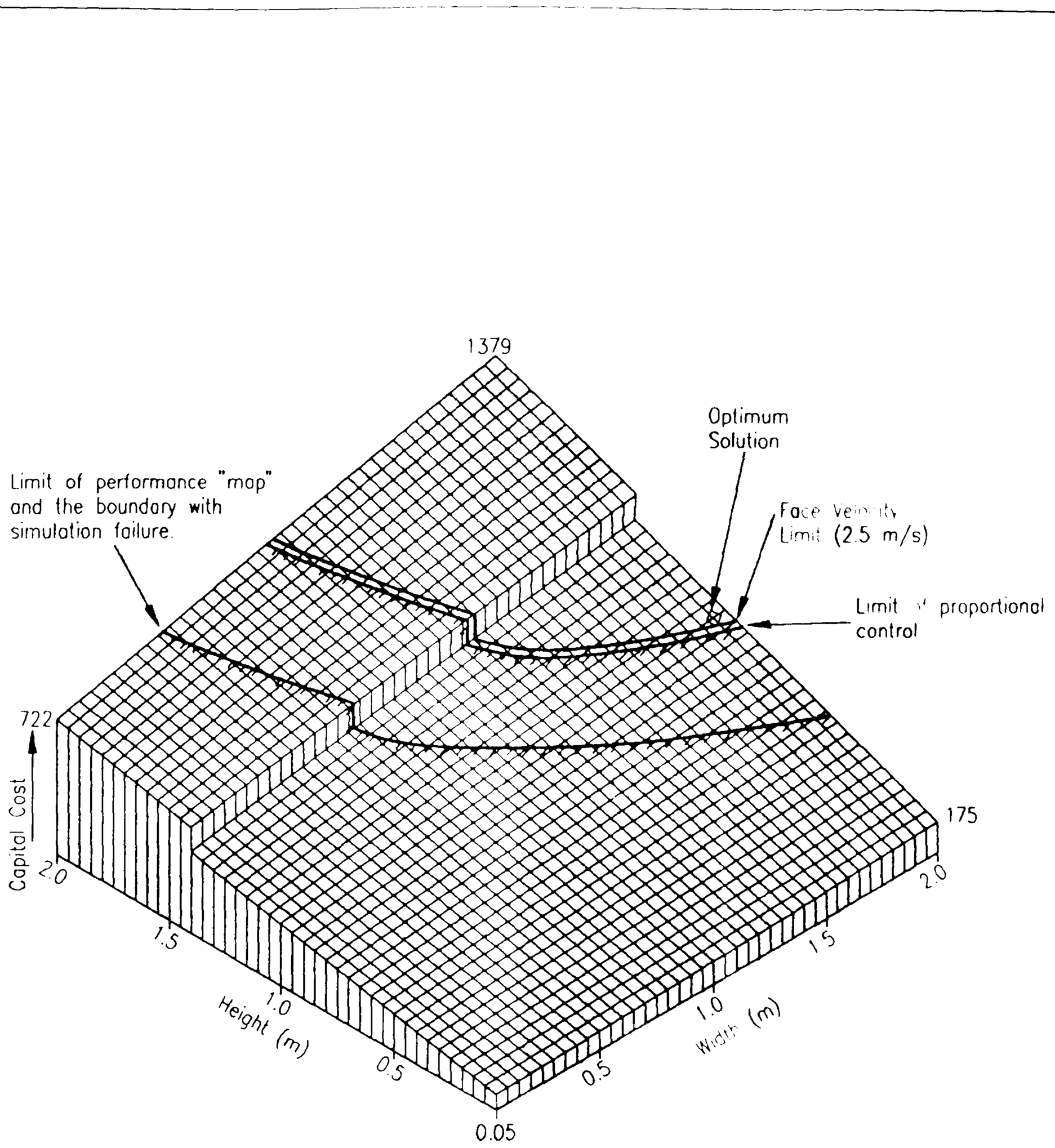
6) Controller proportional band (pb) = 1°C

It can been observed from these conditions that this would represent a hot summer day when the cooling coil would be required to operate at a peak load.

6.6 The Characteristics of the Example Problem.

Chapter 3 of this thesis describes in detail the characteristics of HVAC system design optimisation problems, and it is the purpose of the chosen cooling coil example problem to display these characteristics to assess the optimisation algorithms against them.

It is difficult to describe the characteristics of any optimisation problem in more than two variables. Figure 6.2 illustrates the example problem of the cooling coil with two variable dimensions of cooling coil width and cooling coil height. The remaining variables present in the problem have, for the purpose of this illustration, been fixed at their optimum values. The surface of the problem displayed in Figure 6.2 represents the objective function to be optimised, namely, capital cost of the cooling coil.



Cooling Coil Problem Characteristics

Figure 6.2

The region of feasible solutions is constrained by the upper bounds of the cooling coil width and height and by the upper limit of the performance constraint of face velocity. It can be seen that the optimum solution for the variables of coil width and height lies at a position close to the coil face velocity constraint and is close to the control constraint set by the proportional band. This is a common characteristic of HVAC system design optimisation problems, when capital cost is an objective function, as the solution is the system that is just large enough to be operational. The optimisation algorithms used to solve the example problem begin their search from an initial feasible solution and use a series of trial points to find the optimum. Since the unconstrained optimum would be for a coil width and height of 0.05m, giving the lowest possible capital cost, the search will move towards this solution until the face velocity constraint is encountered. The search must then traverse the face velocity constraint towards the optimum solution at a coil width of 1.95m and a coil height of 1.00m. The example problem therefore presents what would be a typical environment for a HVAC system design optimisation problem.

The optimum solution for the example problem is presented in Table 6.3, this was determined by inspection:

Problem Variable	Optimum Solution
Coil width	1.95m.
Coil height	1.00m.
Number of rows	5 rows.
Number of circuits	30 circuits.
Maximum water mass flow rate.	9.75 kg/s

Optimum Solution. Table 6.3

Figure 6.2 also presents the limit to which performance data of the simulation is available. Beyond this limit the reliability of the simulated coil is in doubt, (in fact no performance data is available since in this problem the simulation fails to find a solution).

As already stated, there is a discontinuity in the objective function, and this can clearly be seen in Figure 6.2, at a coil height of 1.4m. It represents a limit in the manufacturers product range. The maximum height of a single coil block is 1.4m, coil heights above this value are manufactured form two or more coil blocks. The algorithms ensure that multi-coil block systems are produced with coil blocks of equal sizes, thus causing an increase in cost. For example, a required coil of 1.8m in height is manufactured from two equal coils each of 0.9m in height. There maybe instances in practice where the splitting of components into equal sizes is not the most desirable arrangement. It could be argued that one large coil

and one small coil would give better performance over the full range of load conditions throughout a year than two of equal size, however, for the purpose of this research this has not been developed as an option, but is noted as a possible practical scenario and is one which could be accommodated within the structure of the future development of the optimisation algorithms.

The predominant characteristic of HVAC system design problem variables is that they are discrete. The example problem chosen employs discrete variables for coil width, coil height, number of coil rows and number of coil circuits. The fifth variable of maximum water mass flow rate is strictly continuous in nature, but within the problem it has been allocated a discrete characteristic by approximating any value found to the nearest preset discrete interval, the discrete intervals chosen for maximum water mass flow rate is 0.25 kg/s.

6.7 Algorithm Testing Against the Example Problem.

The preceding sections of this chapter have set out the example problem used for the testing of optimisation algorithms. Chapter 4 of this thesis identified two optimisation search methods which possessed attributes which would suit HVAC system design optimisation problems. These methods where, the complex method which uses the rejection of infeasible solutions to constrain the optimisation problem, and a penalty function method which imposes a penalty upon the objective function to ensure that violation of the constraints is not possible.

Development of these methods into algorithms specifically for HVAC system design optimisation problems, and the testing against the cooling coil example problem is described in the following chapters 7 and 8.

The testing of the algorithms will be conducted in two phases. A simplified two-dimensional model for the cooling coil was developed for the initial testing, using the variables of coil width and height. The remaining variables were fixed at there optimum values. The optimisation is made faster with fewer variables yet does not lose the dominant characteristics of the problem. The full example problem was then tested against the algorithm upon satisfactory conclusion of the two-dimensional model testing.

CHAPTER 7.

THE DEVELOPMENT AND TESTING OF THE COMPLEX ALGORITHM.

The complex method by Box (1965), is a direct search optimisation method. It has been selected for development within HVAC system design optimisation problems because the search procedure adopted by this method is robust within a feasible region of solutions, and is adaptable to different objective function environments, hence, lending itself to the rigours of HVAC system design optimisation. The additional rules incorporated in the complex method for constraint handling, gives it a degree of reliability around constraint functions. This is particularly important as one of the characteristics of HVAC optimisation is that the optimum solution lie near a constraint function. In summary the complex method is a direct search method which can be readily used with discrete variables, it complements the characteristics of HVAC system design optimisation problems and is strong within a constrained environment.

7.1 The Complex Method.

The simplex method for unconstrained optimisation, devised by Spendley, Hext and Himsworth (1962), and the modifications to this method by Nelder and Mead (1965), are described fully in section 4.1.2 of this thesis and form the basis for the complex method. The complex method is an adaptation of the simplex method which effectively applies

additional rules to constrain the optimisation.

The complex method searches for the maximum value of a function, f ($\underline{X}$) subject to m constraints of the form $lb_k < C_k < ub_k$, (To find a minimum, -f is maximised). It is assumed that an initial point x_1, satisfies all the m constraints is available.

In this method, k > n+1 points are used, of which one is the initial feasible point, and where n, is the number of problem variables. The further (k-1) points required to set up the initial configuration of the simplex are obtained one at a time by the use of pseudo-random numbers and the bounds of the problem variables. i.e. $x_i = lb_i + r_i\ (ub_i - lb_i)$ where r_i is the pseudo-random number distributed over the interval (0,1).

A point so selected will satisfy the problem variable bounds, but need not satisfy all the problem constraint functions. If a problem constraint is violated, the trial point is moved halfway towards the centroid of those points already selected (where the given initial point is included). Ultimately a satisfactory point within the feasible region will be found. Proceeding in this way, (k-1) points are found which satisfy all constraints.

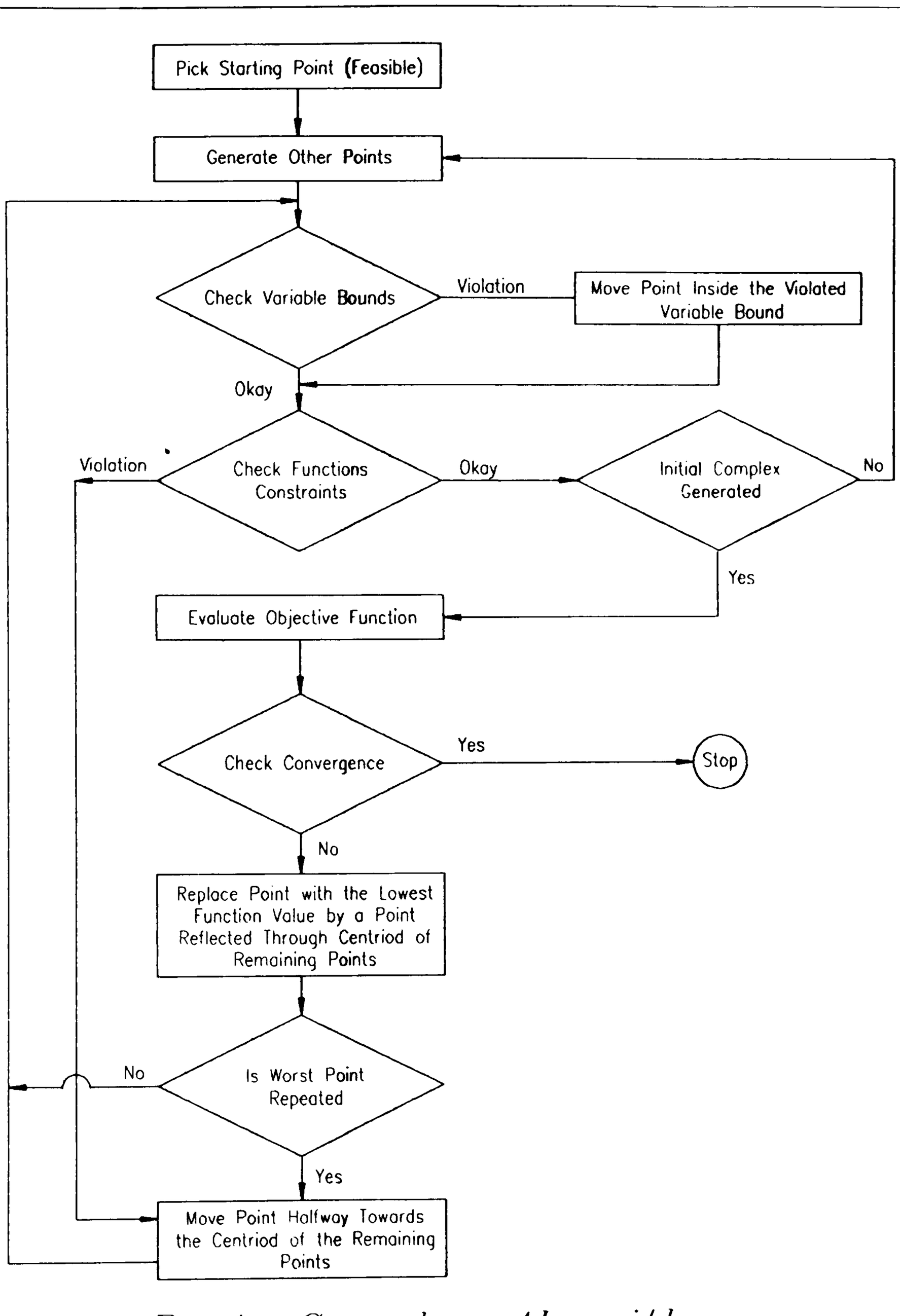
The objective function f, is evaluated at each selected point, or vertex of the simplex configuration. The vertex with the worst objective function value is replaced by reflecting it about the centroid of the remaining points in the simplex. Although the reflected point is collinear with the rejected point and the centroid of the remaining vertices, the distance of the reflected point from the centroid is greater than that of the rejected point by a factor α (> 1). This encourages the search to move in the

direction of the optimum. If the reflected point becomes the worst point, it is moved halfway back towards the centroid of the remaining points to give a new trial point. The above procedure is repeated unless a problem constraint is violated.

If a trial point does not satisfy a problem variable bound constraint, the variable is reset to a value just inside the appropriate violated bound. If a problem constraint is violated, the trial point is moved halfway towards the centroid of the remaining points. Ultimately an acceptable point is found. Thus as long as the complex method has not collapsed into the centroid, progress will continue.

The advantage of over-reflection by a factor $\alpha > 1$ enables rapid progress to be made when the initial point is remote from the optimum solution. The contracting move halfway back towards the centroid will, however prevent the search method from becoming unstable. A value of $\alpha = 1.5$ was chosen to keep the same ratio of search movement when the complex is expanding and contracting halfway back towards the centroid.

Figure 7.1 shows a simplified logic diagram of the developed algorithm for the complex method in its basic form.



Basic Complex Algorithm
Flow Diagram

Figure 7.1

7.2 Modifications to the Complex Method for Use with HVAC System Design Optimisation Problems.

The complex method of direct search optimisation by Box (1965), was developed for use with continuous variables. It is known that the characteristics of HVAC system design optimisation problems are such that discrete problem variables are predominant, and the basic complex method was therefore modified to incorporate discrete variables.

An additional procedure was developed and incorporated into the complex algorithm to deal with discrete variables. This procedure moves the initial feasible point and all subsequent trial points to the nearest discrete value for each of the problem variables. For example, the height of a cooling coil can be manufactured to 0.05m. Hence a search point of 1.357m would be truncated to 1.35m, whereas a point of 1.942m would be increased to 1.95m. With this modification incorporated, the vertices of the configuration are always maintained on discrete solution points. There are three instances where this will occur, when the initial search point is generated, when a new trial point is produced, and when a constraint function is violated and the trial point is re-sited within the feasible region.

The second modification to the basic complex algorithm was to incorporate stopping criteria. Stopping criteria for discrete variables must be different that used by continuous variables, where control of the algorithm can be governed by a preset distance between the centroid of the simplex and the problem variables; convergence is deemed to have occurred when the distance is satisfied by each variable. For discrete

variables, the stopping criteria was that the search would stop when for five consecutive solutions the variables were within one discrete interval of each other.

7.3 The Two Dimensional Example Problem.

The initial testing of the complex method was based on a simplified two dimensional capital cost optimisation of a cooling coil (Chapter 6). The search variables were taken as the coil width and height. Since the number of water circuits and the water mass flow rate have no direct effect on the capital cost objective function, they were eliminated from the problem. The number of coil rows was fixed at two. It was considered more advantageous to the research to perform the initial assessment of the algorithm against a simplified problem, and to extent the optimisation problem given the results of the initial testing. Although simple, the problem has the main characteristics of larger HVAC optimisation problems, namely discrete variables, non linear constraints and a discontinuous objective function. The variable bounds for coil width and coil height were extended by 0.5m to increase the feasible region and therefore allow the complex method algorithm to perform in a larger domain, this allowed a better assessment of the numerical stability of the algorithm.

7.3.1 Testing of the Basic Complex Algorithm with Discrete Problem Variables against the Two Dimensional Example Problem.

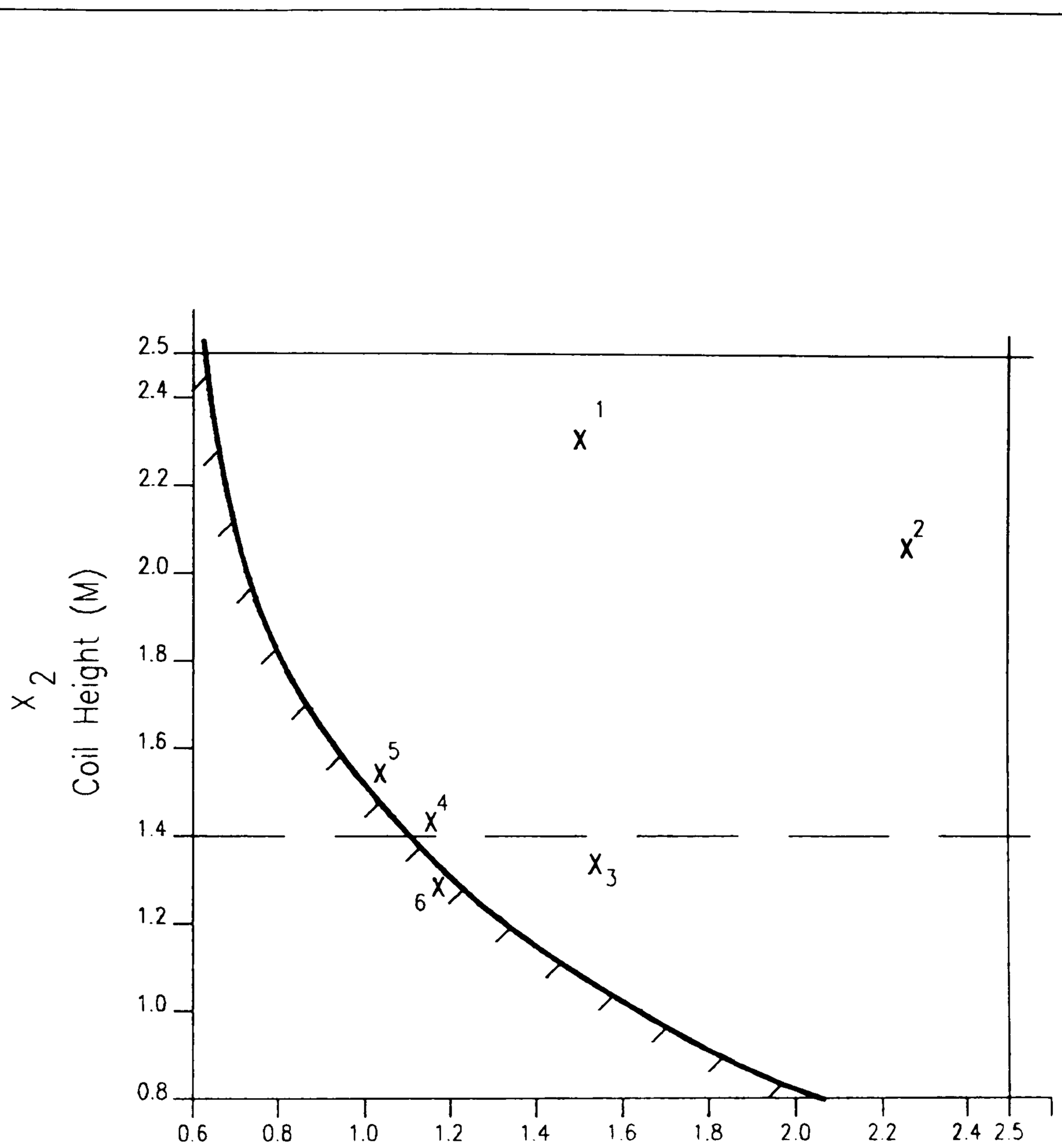
A typical set of results for the basic complex algorithm with discrete with discrete problem variables tested against the two-dimensional example problem are presented in Table 7.2.

Four trial points were used for the vertices of each complex configuration. The initial point (1.5, 2.3) was selected manually and was known to be within the constrained feasible region. The remaining starting points for the 1st complex configuration were selected by the psuedo-random number generator. It can be seen that one of the generated starting points, (0.6, 0.8), violated the coil face velocity constraint and was corrected by the algorithm back to within the feasible region at a position (1.2, 1.4). The objective function values for the 1st complex configuration were calculated and the 'best' and 'worst' points found as the lowest and highest objective function values respectively.

Correct progress of the complex algorithm was made to the 2nd complex configuration by rejection of the worst point, and generation of a new trial point again with constraint violation correction. At the end of the 3rd complex configuration it became apparent that the search was not proceeding correctly since a new trial point was not found by the algorithm.

STARTING POINTS		CORRECTED POINTS		FUNCTION VALUE (£)	HIGHEST/LOWEST
X1	X2	X1	X2		
1.5	2.3			811	
2.3	2.0			869	HIGH
1.5	1.3			446	LOW
0.6	0.8	1.2	1.4	532	
2nd COMPLEX					
1.5	2.3			881	HIGH
0.2	1.2	0.8	1.4		
		1.1	1.5	544	
1.5	1.3			446	LOW
1.2	1.4			532	
3rd COMPLEX					
1.0	0.2	1.1	0.8		
		1.2	1.1		
		1.2	1.2		
		1.2	1.3		
		1.2	1.3		
ALGORITHM FAILS					

The Basic Complex Algorithm Results Table 7 2



Basic Complex Algorithm
Search Path

Figure 7.3

Figure 7.3 gives an illustration of the search progress and it is clear from this why the algorithm fails. At position (6), the complex algorithm finds a position (1.2, 1.3) this position is still in violation of the face velocity constraint yet the search cannot move to a position within the feasible region. Upon investigation it was clear why the algorithm failed at this point. The 2nd complex configuration has vertices at the following positions with the respective objective function values :

VERTEX	X1	X2	OBJECTIVE FUNCTION VALUE (£).
VERTEX 1	1.5	2.3	881
VERTEX 2	1.1	1.5	544
VERTEX 3	1.5	1.3	446
VERTEX 4	1.2	1.4	532

The point with the worst objective function is at the vertex 1, the centroid of the remaining vertices is therefore at a position (1.2666, 1.4). The over-reflection performed by the complex algorithm gives a new trial point at a position (1.0, 0.2), well outside the feasible region. Movement halfway back towards the centroid starts at position (1.1, 0.8), again this is in violation of the constraint function,and therefore a second move towards the centroid takes the trial point to the position (1.2, 1.1). This process continues until the position (1.2, 1.3), is reached. At this point the algorithm fails, since the discrete variable prevents a move back towards

the centroid ie a subsequent move halfway back towards the centroid is calculated by the algorithm. The calculated point is developed as follows :

(((1.2666 - 1.2)/2) + 1.2), (((1.4 - 1.3)/2) + 1.3),

giving a calculated position of (1.2333, 1.35), because discrete problem variables are present the algorithm adjusts the position of the trial point to the nearest discrete point this being back at the position of (1.2, 1.3). The result is clear, the algorithm is unable to move from this position and fails.

There are some positive conclusions to be drawn from the initial testing of the basic algorithm. The numerical stability and the robustness of the algorithm over the objective function surface prior to encountering the face velocity constraint were both good. The algorithms search path passed over the discontinuity in the objective function without encountering a problem, and the problem variable bounds again presented no difficulty to the algorithm.

The search path was such that the basic complex algorithm was moving towards the global unconstrained optimum solution at a coil width = 0.1 m and a coil height = 0.1 m but was prevented from reaching this by encountering the face velocity constraint at a position some distance from the optimum solution, in this example at a position of coil width = 2.3 m and coil height = 0.7 m. The algorithm failed to cope with the directional change required to start the traverse of the face velocity constraint towards this optimum solution, this was largely due to the fact that discrete variables were present within the example problem.

7.4 The Modified Complex Algorithm.

From the initial test of the basic complex algorithm incorporating discrete problem variables, it was concluded that a modified algorithm needed to be developed to deal with the specific problem of failure of the algorithm against constraint functions. A new type of search move was required to firstly move a trial point which is in violation of a constraint function back into the feasible region and secondly, if possible, to change the direction of the search path towards the constrained optimum solution, ie. forcing the search to start moving along the constraint function in the direction of the optimum.

The nature of the failure of the basic complex algorithm suggested that a single discrete interval movement in one problem variable direction would be sufficient to overcome the failure, as the mid-point between the violated point and the centroid of the remaining points was less than half a discrete interval away from the violated point in any problem variable direction. Rounding of the mid-point back to the violated position would therefore be overcome. Additional routines were incorporated into complex algorithm to perform such a move.

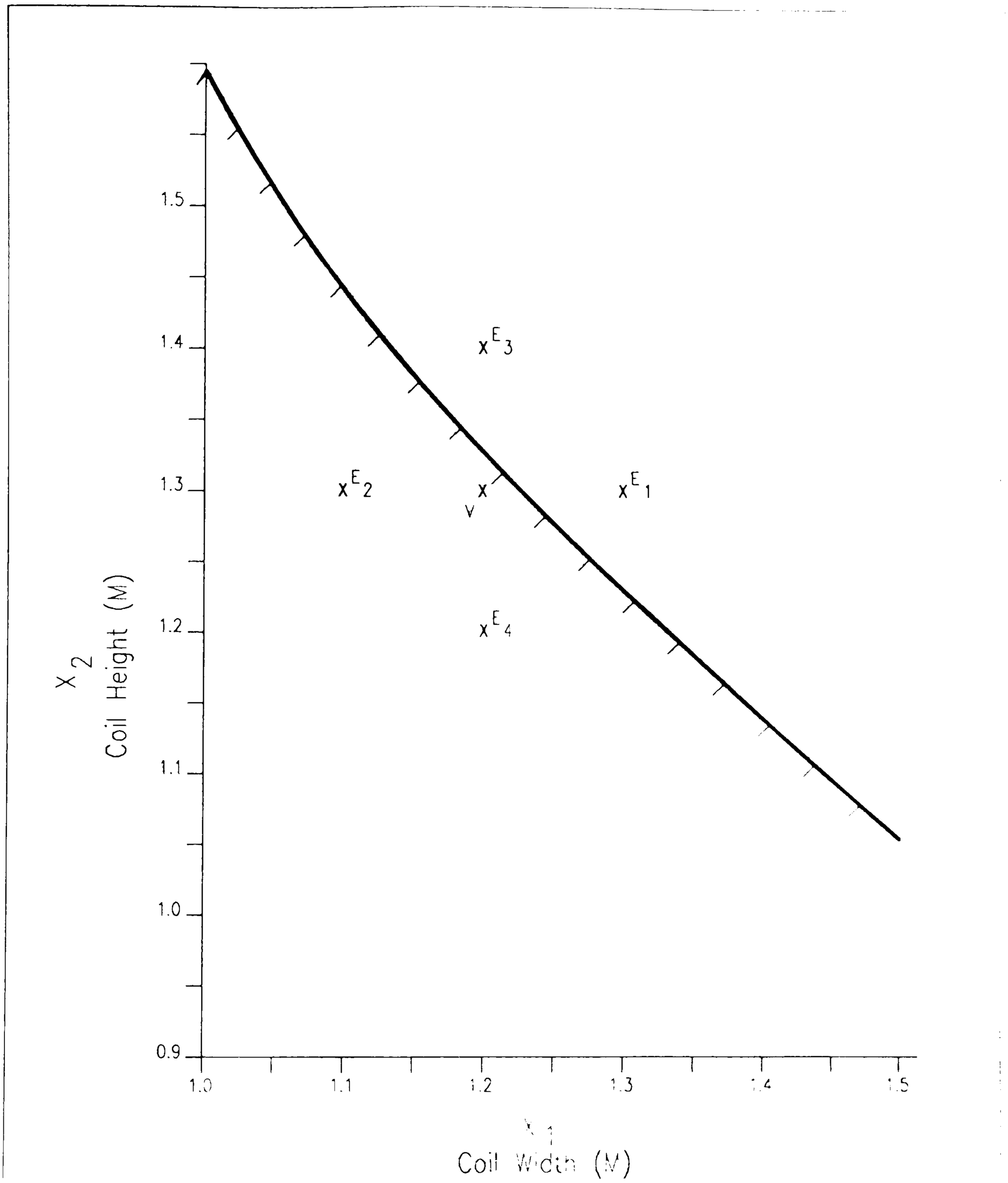
7.4.1. The Exploratory Search Move.

An exploratory search move was chosen to deal with the failure of the basic complex algorithm. An exploratory search forms part of the Hooke and Jeeves (1960), pattern search which has already been used with some success by Wright (1986), with HVAC system design optimisation

problems. This type of search technique lends itself specifically to discrete variable problems, using a discrete variable interval as the unit of any move. It was therefore considered to be an appropriate modifying move.

The exploratory move evaluates each problem variable in turn, one discrete interval away from the violated point in both a positive and negative direction along the problem variable axis. Constraint functions are then checked against the exploratory trial points; violation of a constraint function results in the rejection of the trial point. The objective function of the optimisation problem is then evaluated at each of the feasible exploratory positions. The exploratory point which is within the feasible region and has the best objective function value is selected as the new search point. The algorithm at this stage then reverts back to the basic complex algorithm and the search continues.

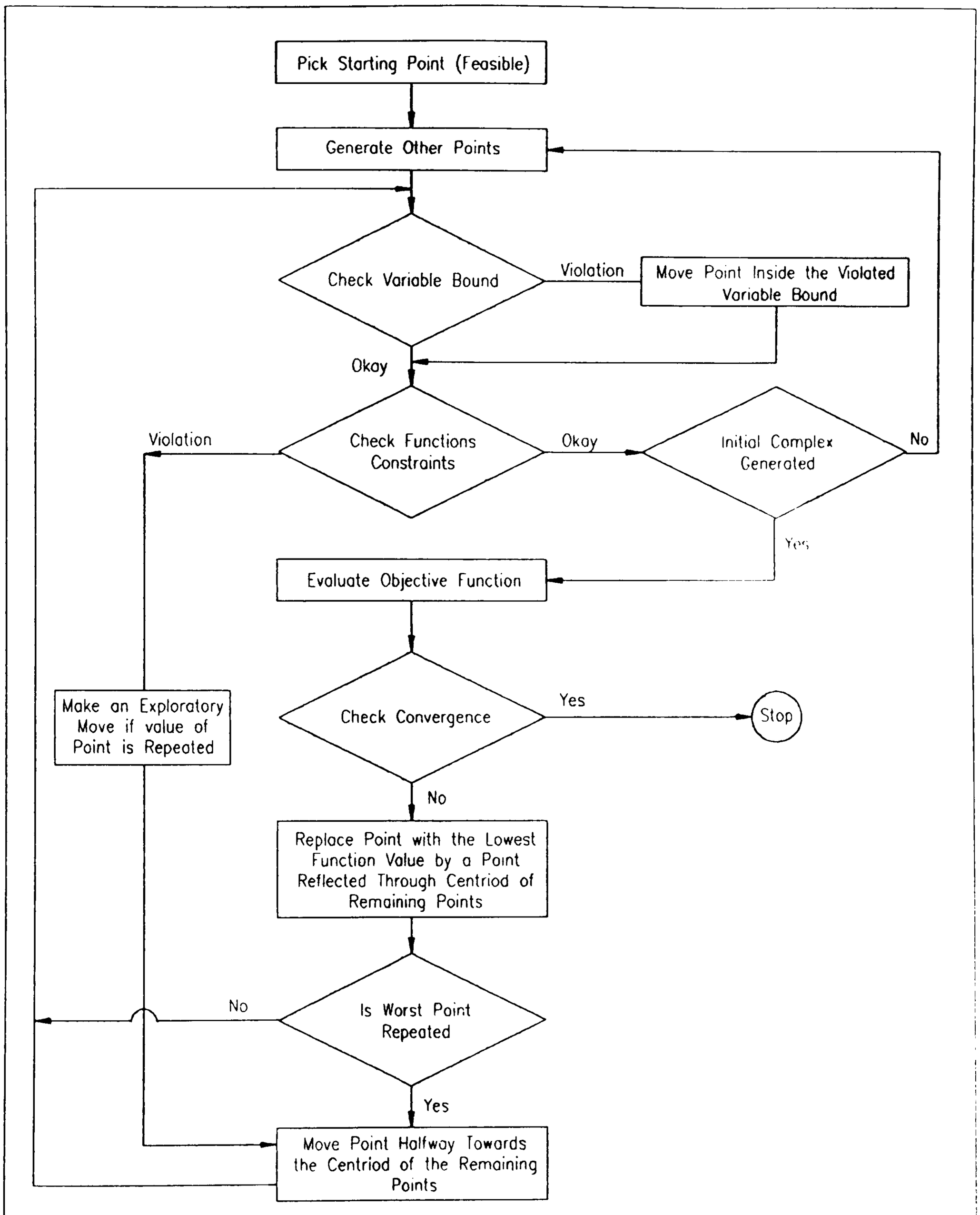
The new exploratory search move is explained using the example detailed in section 7.3.1. The failed point V, at a position $X_1 = 1.2$, $X_2 = 1.3$, where X_1 represents the coil width and X_2 represents the coil height. At this point of failure the pattern search probes, firstly in the X_1 direction, finding exploratory positions at E_1 and E_2 of $X_1 = 1.1$, $X_2 = 1.3$, and $X_1 = 1.3$, $X_2 = 1.3$. The X_1 value is then re-set to the value at V, and exploratory positions are found in the X_2 direction, at E_3 and E_4 of $X_1 = 1.2$, $X_2 = 1.2$ and $X_1 = 1.2$, $X_2 = 1.4$. Figure 7.4 illustrates this procedure, where E_1, E_2, E_3, E_4 represent the exploratory trial points.



The Exploratory Search Mov

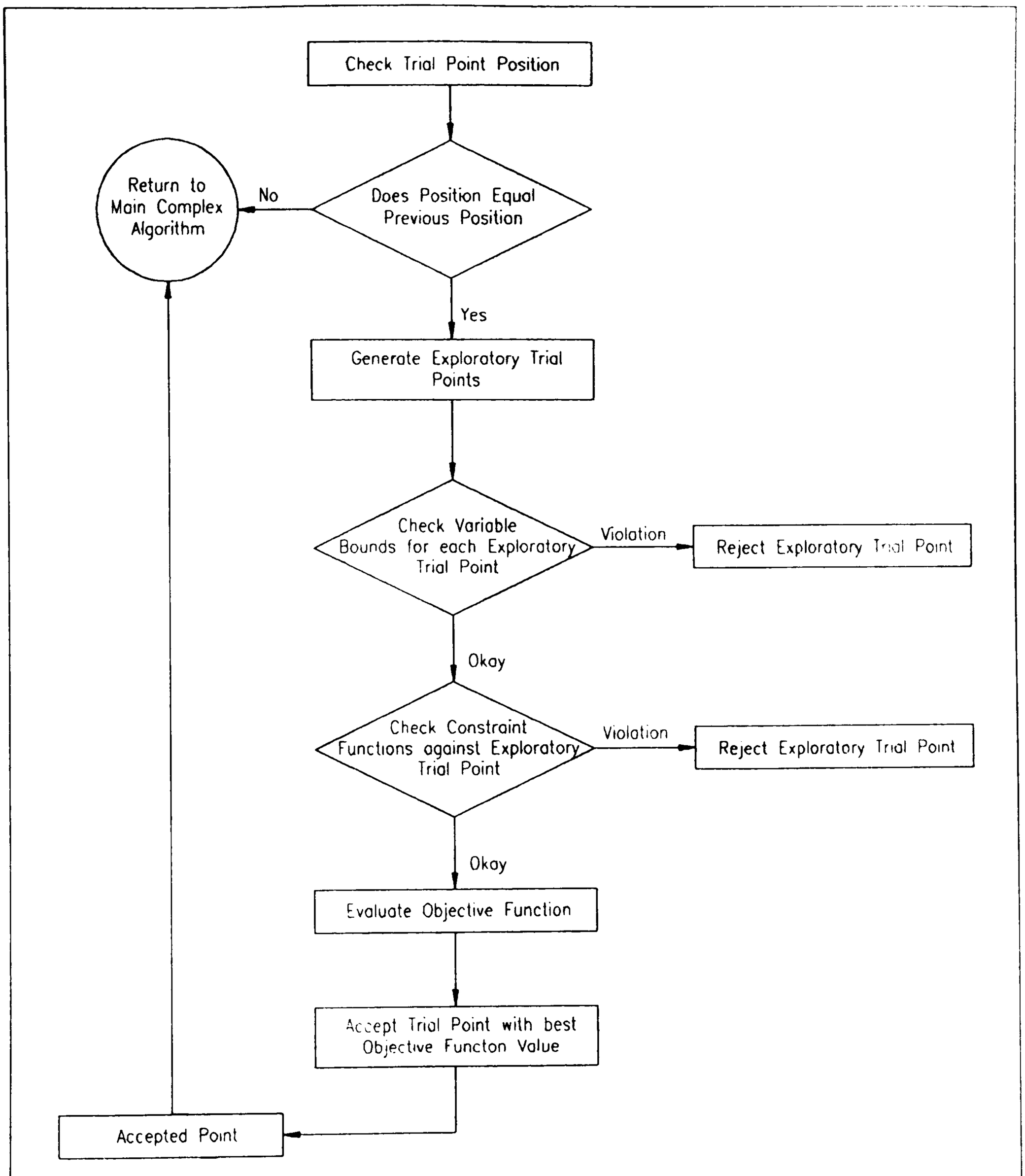
Figure 7.4

When a constraint function is checked against the four exploratory trial points it is found that E_2 and E_4 violate the constraint, and are therefore rejected. The objective function values for the remaining exploratory trial points E_1 and E_3 are calculated at £423 and £437 respectively. E_1 is therefore accepted as the new trial point. Reversion back to the basic complex algorithm completes the modification. The new logic diagram in Figure 7.5 incorporates the position of the modification within it, under the name of Exploratory, and Figure 7.6 shows the logic diagram of the exploratory modification.



Complex Algorithm Flow Diagram with Exporatory Move Positioning

igure 7.5



Flow Diagram of the Exploratory Move

Figure 7.6

7.4.2 Testing of the Modified Complex Algorithm with Discrete Problem Variables against the Two Dimensional Example Model.

A typical set of results for the modified complex algorithm incorporating the exploratory search move and tested against the two dimensional example model (Section 7.3) are presented in Table 7.7.

Four trial points were used for the vertices of each complex configuration. The initial point at (1.5, 2.3), and the generated other vertices for the 1st complex configuration were kept the same as those used for the basic complex algorithm test, so that a direct comparison could be made.

The basic complex algorithm was seen to fail whilst producing the 3rd complex configuration, the results of the modified complex algorithm show that the cause of that failure has been successfully rectified by the inclusion of the exploratory move. The exploratory move produced a new trial point at a position (1.3, 1.3), satisfying the previously violated constraint function. The modified algorithm reverted back to the basic complex method to produce the remaining trial points of the 3rd complex configuration and continued.

STARTING POINTS		CORRECTED POINTS		FUNCTION VALUE (£)	HIGHEST/LOWEST
X1	X2	X1	X2		
1.5	2.3			811	
2.3	2.0			869	HIGH
1.5	1.3			446	LOW
0.6	0.8	1.2	1.4	532	
2nd COMPLEX					
1.5	2.3			811	HIGH
0.2	1.2	0.8	1.4		
		1.1	1.5	544	
1.5	1.3			446	LOW
1.2	1.4			532	
3rd COMPLEX					
1.0	0.2	1.1	0.8		
		1.2	1.1		
		1.2	1.2		
		1.2	1.3		
		EXPLORATORY MOVE			
		1.3	1.3	423	LOW
1.1	1.5			544	HIGH
1.5	1.3			446	
1.2	1.4			532	
4th COMPLEX					
1.3	1.3			423	
1.6	1.1			401	LOW
1.5	1.3			446	
1.2	1.4			532	HIGH
5th COMPLEX					
1.3	1.3			423	
1.6	1.1			401	
1.5	1.3			446	HIGH
1.8	1.0			391	LOW
6th COMPLEX					
1.3	0.8			423	HIGH
1.6	1.1			401	
1.7	0.9	1.6	1.1	401	
1.8	1.0			391	LOW
7th COMPLEX					
2.1	0.8			353	LOW
1.6	1.1			401	HIGH
1.6	1.1			401	
1.8	1.0			391	
8th COMPLEX					
2.1	0.8			353	LOW
1.9	0.8	EXPLORATORY MOVE			
		1.9	0.9	369	
1.6	1.1			401	HIGH
1.8	1.0			391	
9th COMPLEX					
2.1	0.8			353	LOW
1.9	0.9			369	
2.4	0.6	2.2	0.7		
		2.1	0.8	353	
1.8	1.0			391	HIGH

The Modified Complex Algorithm Results Table 7 7
page 1 of 2

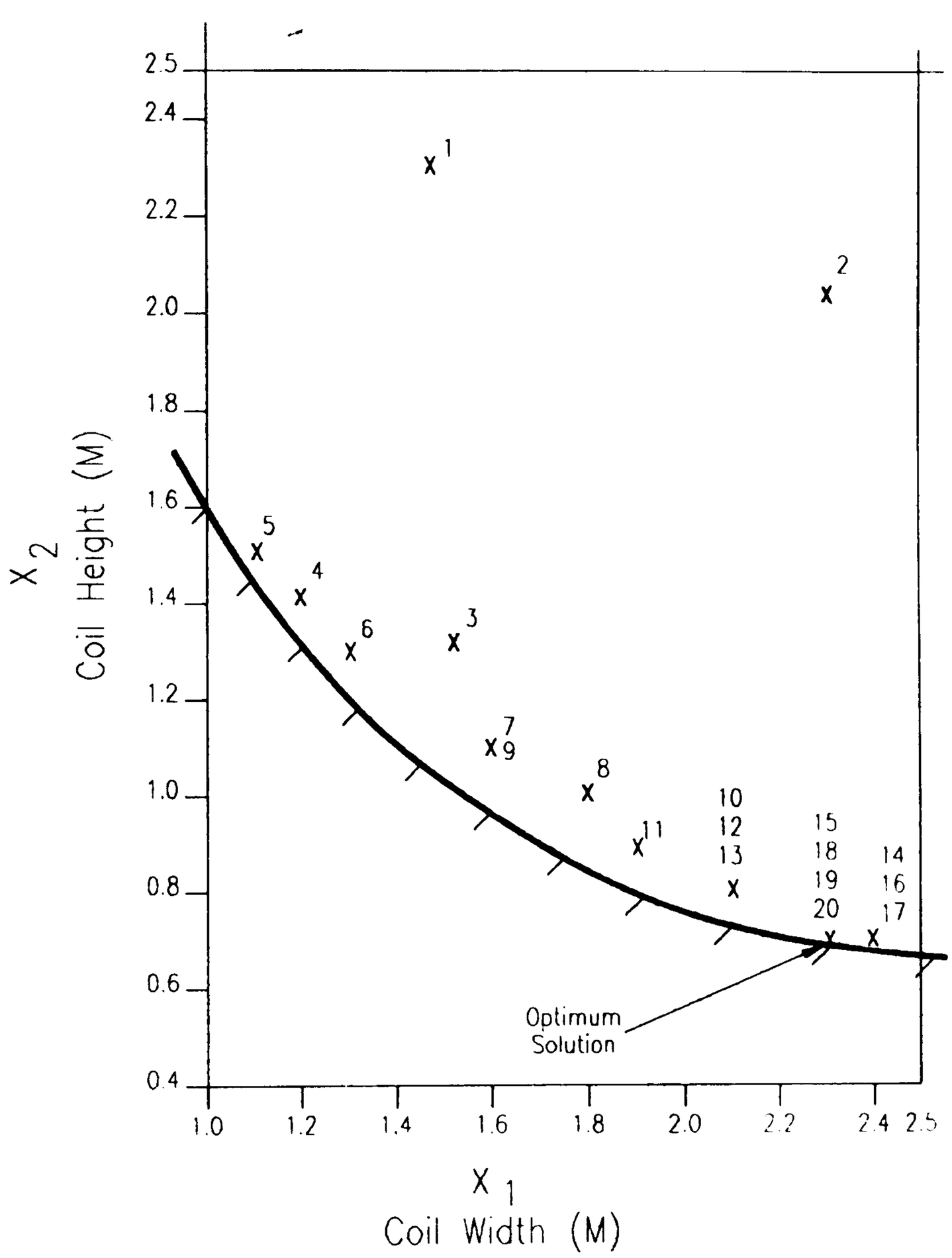
STARTING POINTS		CORRECTED POINTS		FUNCTION VALUE (£)	HIGHEST/LOWEST
X1	X2	X1	X2		
10th COMPLEX					
2.1	0.8			353	LOW
1.9	0.9			369	HIGH
2.1	0.8			353	
2.3	0.6	2.2	0.7		
		2.1	0.8	353	
11th COMPLEX					
2.1	0.8			353	HIGH
2.4	0.7			339	LOW
2.1	0.8			353	
2.1	0.8			353	
12th COMPLEX					
2.3	0.7			333	LOW
2.4	0.7			339	
2.1	0.8			353	HIGH
2.1	0.8			353	
13th COMPLEX					
2.3	0.7			333	LOW
2.4	0.7			339	
2.5	0.6	2.4	0.7	339	
2.1	0.8			353	HIGH
14th COMPLEX					
2.3	0.7			333	LOW
2.4	0.7			339	HIGH
2.4	0.7			339	
2.5	0.6	2.4	0.6		
		EXPLORATORY MOVE			
		2.4	0.7	339	
15th COMPLEX					
2.3	0.7			333	LOW
2.3	0.7			333	
2.4	0.7			339	HIGH
2.4	0.7			339	
16th COMPLEX					
2.3	0.7			333	LOW
2.3	0.7			333	
2.2	0.7	2.3	0.7	333	
2.4	0.7			339	HIGH
17th COMPLEX					
2.3	0.7			333	LOW
2.3	0.7			333	
2.3	0.7			333	
2.2	0.7	EXPLORATORY MOVE			
		2.3	0.7	333	HIGH
OPTIMUM FOUND					

The Modified Complex Algorithm Results Table 7.7
page 2 of 2

The results of the modified complex algorithm tested against the example problem were successful. The algorithm accurately found the optimum solution at a position (2.3, 0.7), at the end of the 17^{th} complex configuration. During the search, it can be seen that three more exploratory moves were made by the algorithm, in each case these exploratory moves successfully found a new trial point within the feasible region. These exploratory moves occurred during the 8^{th} complex configuration, 14^{th} complex configuration and the 17^{th} complex configuration.

The full search path that the modified complex algorithm took is illustrated in Figure 7.8 with the optimum solution being found after 20 search moves.

The modified complex algorithm was particularly successful in two ways. Firstly it provided a method by which a violated trial point could be moved back within the feasible region so that the optimisation could continue, and secondly it provided a method by which the search direction could change to direct the search along the constraint function and towards the optimum solution.



Path of the Modified Complex Search

Figure 7.8

7.5 Accuracy of the Modified Complex Algorithm.

From the results presented in Table 7.7 it can be seen that the modified complex algorithm was accurate throughout the search.

The problem variable values found throughout the search were primarily controlled by the modifications made to the basic complex algorithm for the handling of discrete problem variables. The problem variable values were always at a discrete search position showing that the modification was accurately finding solution points.

The constraint and objective function values found throughout the search were also accurate. There was no instance when an inaccuracy in either a constraint or an objective function caused the algorithm to become numerically unstable thus delaying the search in finding the optimum solution.

The algorithm accurately found the optimum solution, giving the correct lowest objective function value.

7.6 The Speed of the Modified Complex Algorithm.

Chapter 5 of this thesis identifies that the most time consuming factor for a HVAC system design optimisation algorithm is the simulation of the system at a given new solution point. The simulation of the system is performed each time a constraint or objective function is evaluated by the algorithm, and therefore constraint or objective function evaluations are

used to measure the overall speed of the modified complex algorithm.

The modified complex algorithm makes either a constraint function call or an objective function call for each trial point in each complex configuration. For newly generated trial points both the constraint and objective functions are evaluated, however, in this instance only one call to the system simulation is made. In the case where a new trail point violates a constant function and a corrective step is made by the algorithm one addition call is made for each corrective step. When the exploratory move is used by the algorithm then two function calls per variable are made. The path of the modified complex search algorithm shown in Table 7.7 gives the following breakdown of function call evaluations shown in Table 7.9 overleaf.

A comparison of the speed of the modified complex algorithm can be made against an exhaustive search which would evaluate the objective function at every solution point within the problem variable bounds. In the example problem 625 solution points are present, the modified complex algorithm made 14.5% of the possible objective function calls.

From Figure 7.8 it can be seen that the majority of the search path was spent negotiating the face velocity constraint function and this is significant in the number of total function evaluations made. The influence of the face velocity constraint provides a barrier which directly effects the shape of the complex configuration. The natural movement of the search toward the global optimum has the effect of confining the search path to a narrow band of solution points along the constraint, thus slowing the search speed to the 'optimum' solution.

TYPE OF TRAIL MOVE	CONSTRAINT/OBJECTIVE FUNCTION EVALUATIONS
NORMAL COMPLEX CONFIGUATION TRAIL POINTS	68
COMPLEX CONFIGUATION CORRECTIVE TRIAL POINTS	15
PATTERN MOVE TRIAL POINTS	8
TRIAL POINT TOTAL	91

Breakdown of Function Evaluations for the Complex Algorithm. Table 7 9

7.7 The Numerical Stability of the Modified Complex Algorithm.

The modified complex algorithm was entirely numerically stable. The use of discrete problem variables greatly assisted the stability, ensuring that instability due to rounding errors in the objective function evaluation was eliminated. There were occasions when the search path, on initial inspection, appeared to move away from the 'optimum' solution, this can be seen in Figure 7.8. Between solution points 4 and 5, 8 and 9, 10 and 11, and 15 and 16 the search moves to a position further away from the optimum solution, however, upon investigation it was found that this was a correct function of the algorithm in each case eliminating the worst solution point of the previous complex configuration and replacing it with a new solution point conforming to the rules applied by the algorithm. A new solution point does not necessarily have to be the 'best' solution point of the new complex configuration, merely it has to be better than the replaced solution point.

The numerical stability of the constraint and objective function evaluations again were entirely satisfactory.

7.8 The Robustness of the Modified Complex Algorithm.

The failure of the basic complex algorithm incorporating discrete problem variables (Section 7.3.1) was due to lack of robustness of the algorithm when encountering the face velocity constraint. The robustness of the modified complex algorithm was therefore monitored carefully.

The modified complex algorithm was found to be robust under all circumstances throughout the search. The failure of the basic complex algorithm being successfully corrected by the inclusion of the exploratory move into the algorithm.

It was noted however that the modified complex algorithm progressed slower as it moved closer to the optimum solution. This was found not to be significant, and the algorithm can be concluded to be fully robust against the example problem.

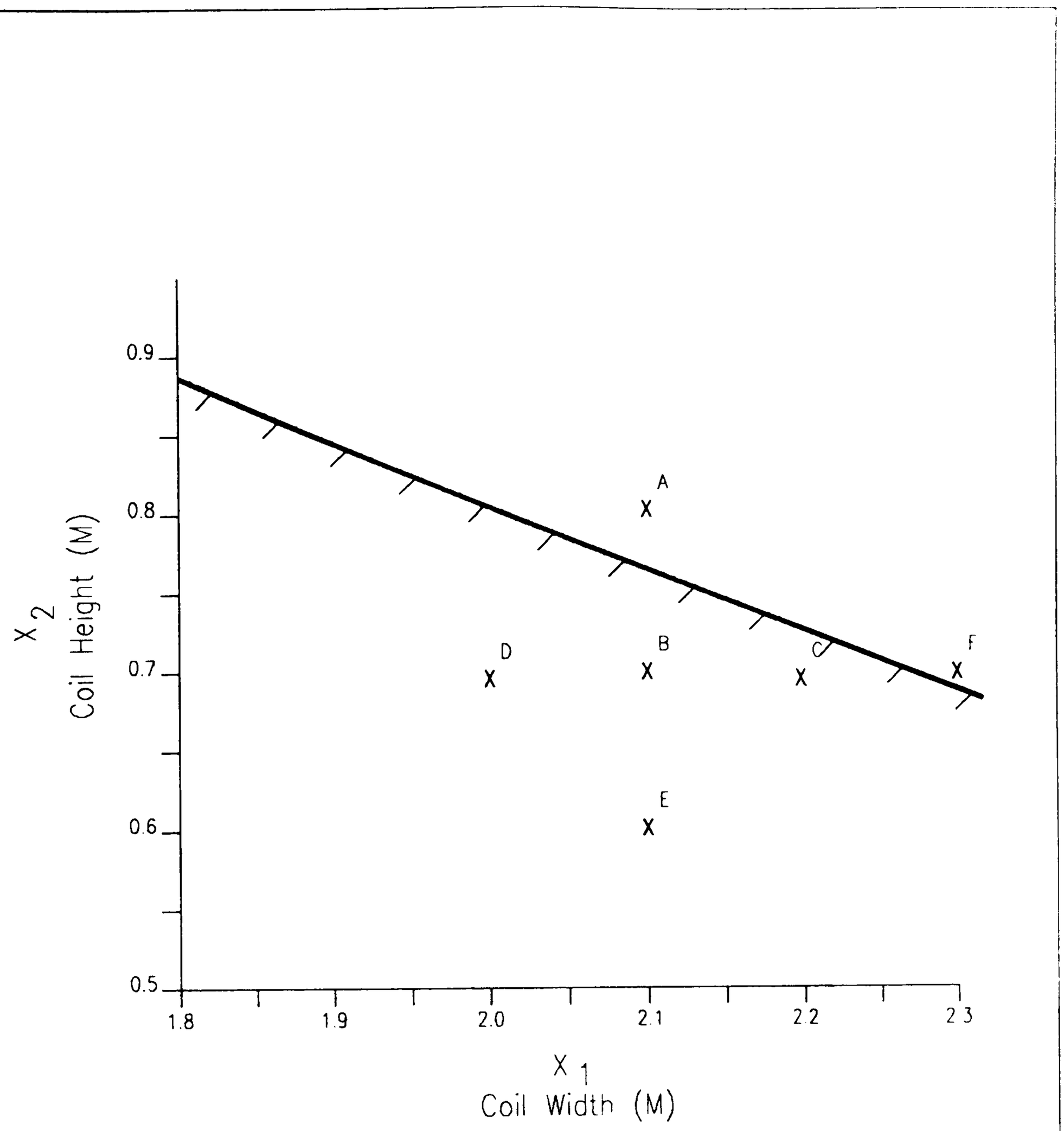
During the evaluation of the modified complex algorithm, a weakness in the algorithm was discovered. This occurred where the exploratory move modification was unable to find a feasible new solution point hence causing the failure of the algorithm.

7.9 Failure of the Exploratory Move.

The failure of the exploratory move within the modified complex algorithm can occur when a constraint function boundary is aligned or parallel with a variable axis.

Figure 7.10 illustrates an example when this failure will occur. From the trial point A, at position (2.1, 0.8) a new trial point B, is found at position (2.1, 0.7). Assuming that a normal complex correction of the constraint violation is unsuccessful a exploratory move would take place from the trial point, B. A single discrete interval exploratory probe in each of the problem variables, in a positive and negative direction would yield

potential new trial points at C, D , E and also at the old trial point A. It can be seen that these exploratory trial point would not correct the search algorithm back into the feasible region, since positions C, D, and E are infeasible. Selection back to position A, would cause failure of the algorithm. A second exploratory move could be incorporated into the algorithm which makes exploratory probes of two discrete intervals in each problem variable direction, thus finding the feasible solution at position F, but again the same problem would occur if the constraint function gradient aligned itself further with a problem variable axis, in this example the coil width (X1) axis. The problem would merely be deferred and not rectified.



Failure of the Exploratory Search Move

Figure 7.10

7.10 Conclusions of the Modified Complex Algorithm Development and Testing.

The modified complex algorithm with the inclusion of the exploratory search move successfully found the 'optimum' solution of the two-dimension cooling coil example model. During the development and testing of the algorithm significant advances were made, improving the robustness of the algorithm to an extent where the discrete variable characteristic of HVAC system design optimisation problems was addressed fully. The algorithm was found to be accurate and numerically stable in all instances throughout the optimisation and again, particularly in relation to the specific characteristics of HVAC system design optimisation problems, ie. around constraint functions, and objective function discontinuities.

Weaknesses of the modified complex algorithm were highlighted through the development and testing against the example model. It was determined that the speed of the algorithm was dependent on two factors. Firstly, the number of problem variables that are present; and secondly the interaction of discrete variables and problem constraint functions.

As more problem variables are introduced to optimisation the search speed will be affected. In a full multi-component HVAC system the number of problem variables will significantly increase. It is considered that the objective of this research, that of reducing the time spent in a design office to produce a workable design would not be met using the modified complex algorithm in its existing form.

Constraints imposed upon a multi-component HVAC system will also be increased from the example model. The interaction of discrete problem variables with constraint functions have shown that this significantly reduces the speed of modified complex algorithm.

CHAPTER 8.

THE DEVELOPMENT AND TESTING OF THE PENALTY FUNCTION ALGORITHM.

The objective of any optimisation method is to find the values of the problem variables which optimise the objective function subject to the non-violation of constraints at the optimum solution. The Complex Method detailed in Chapter 7 allowed temporary violation of constraint functions, which in most instances were not critical to the optimum solution being found. The Complex Method, however, was found to have a risk of not finding the optimum solution under certain circumstances of constraint function violation, a risk which was heightened because constraint functions are normally operative on or near the optimum solution.

If, however, the constraint functions are never allowed to be violated during an optimisation, then the resulting optimum solution is sure to be a feasible one.

The Penalty Function Method by Carroll (1960), attempts to prevent constraint functions from being violated and was considered to be an appropriate method for addressing the weaknesses found in the Complex Method.

8.1 The Penalty Function Method.

The created response surface penalty function method by Carroll (1960) converts a constrained optimisation problem into a series of unconstrained optimisation problems, but positively avoids violation of any given constraint functions by imposing a penalty on the objective function, which becomes increasingly severe as a constraint function is approached.

The created response objective function can be written in the following form:

$$F^*(X) = F(X) + r \sum_{i=1}^{i=m} W_i / C_i$$

where m, is the total number of constraints on the problem including the variable bounds. W_i is the individual penalty term or weighting given to each of ith constraints, C_i is the value of the ith constraint function when in the form, $C_i > 0.0$, and r is the overall weighting given to the penalty in relation to the 'true' objective function F(X).

This method of imposing a penalty to an objective function as a constraint is approached can be easily visualised by an example. As a constraint C_i is approached the value of that constraint tends to zero, therefore the value of the created response objective function $F^*(X)$, increases making the solution point less attractive when minimising the optimisation problem. A constraint will tend not be violated because solution points near constraints are not attractive in terms of created response objective function.

The penalty function method by Carroll (1960), sets up the optimisation environment but does not employ a search method. A separate search method is required to operate within the created response surface environment. The search method finds the optimum for the modified objective function. This may not be the same as the global, non-penalty function optimum. This problem is overcome by successively reducing the overall weighting r, of the penalty function. When an optimum is found a reduction in the value of r, the overall penalty weighting, takes place hence reducing the severity of the penalty on the objective function. The optimum solution found from the previous search now becomes the new starting position for the current optimisation. The selected search will find the 2nd optimum solution within this new created response environment. Each optimum solution found becomes the new starting position for the next search. Successive reductions in the value of r, allows the created response objective function $F^*(X)$, to tend towards the 'true' objective function, F(X). When r = 0 the penalty on the optimisation problem is completely removed allowing the 'true' optimum solution to be found by the chosen search method.

8.2 The Pattern Search Algorithm.

A pattern search algorithm was developed for use with the Penalty Function Method as the driving optimisation method. The method is based upon the Hooke and Jeeves (1960), direct search method. It was chosen because of previous use of the method within HVAC system design

optimisation (Section 4.2.1) by Wright (1986), and also because it can be readily adapted to deal with discrete variable problems.

The procedure of the pattern search algorithm is characterised by two operations, exploratory moves and pattern moves. Exploratory moves attempt to locate the direction of the optimum solution by examining the local behaviour of the objective function. Pattern moves utilise this information and make an accelerated step towards the optimum solution. Both types of moves are made relative to the set of problem variable coordinates, $(X_1, X_2,....X_n)$ termed base points, where there are n, problem variables. Exploratory moves are made relative to a temporary base point, Tb, whilst pattern moves are made relative to a base point representing the current search solution point SP. Both types of move are made in units relating to the discrete intervals of the problem variables.

Exploratory moves probe along each problem variable direction in turn. A coordinate is increased by a fixed step length, k_i, and the value of the objective function compared with that at the temporary base, Tb. If the objective function value is lower, the position is retained to produce a new temporary base. Where the increased function coordinate produces a higher objective function value, the original coordinate is reduced by the same step-length k_i, and the objective function value comparison is repeated. Failure to improve the objective function value in either the positive or negative variable direction leaves a temporary base unchanged. The next problem variable is selected and exploratory move procedure repeated. When each problem variable direction has been explored, the pattern search compares the objective function values at the temporary and solution base points. If the temporary base point has the lower

objective function value an accelerated pattern move is made from the solution base towards and beyond the temporary base. This is in two stages, firstly, a new temporary base $\underline{Tb}^{(j+1)}$, is created at the distance equal to the increment between the two base points, and in the same direction as the existing temporary base $\underline{Tb}^{(j)}$ from the solution base:

$$\underline{Tb}^{(j+1)} = 2 \times \underline{Tb}^{(j)} - \underline{SP}^{(j)} \qquad \text{Equation 8.1.}$$

The second stage of a pattern move is to set the solution point to the original temporary base:

$$\underline{SP} = \underline{Tb}^{(j)} \qquad \text{Equation 8.2.}$$

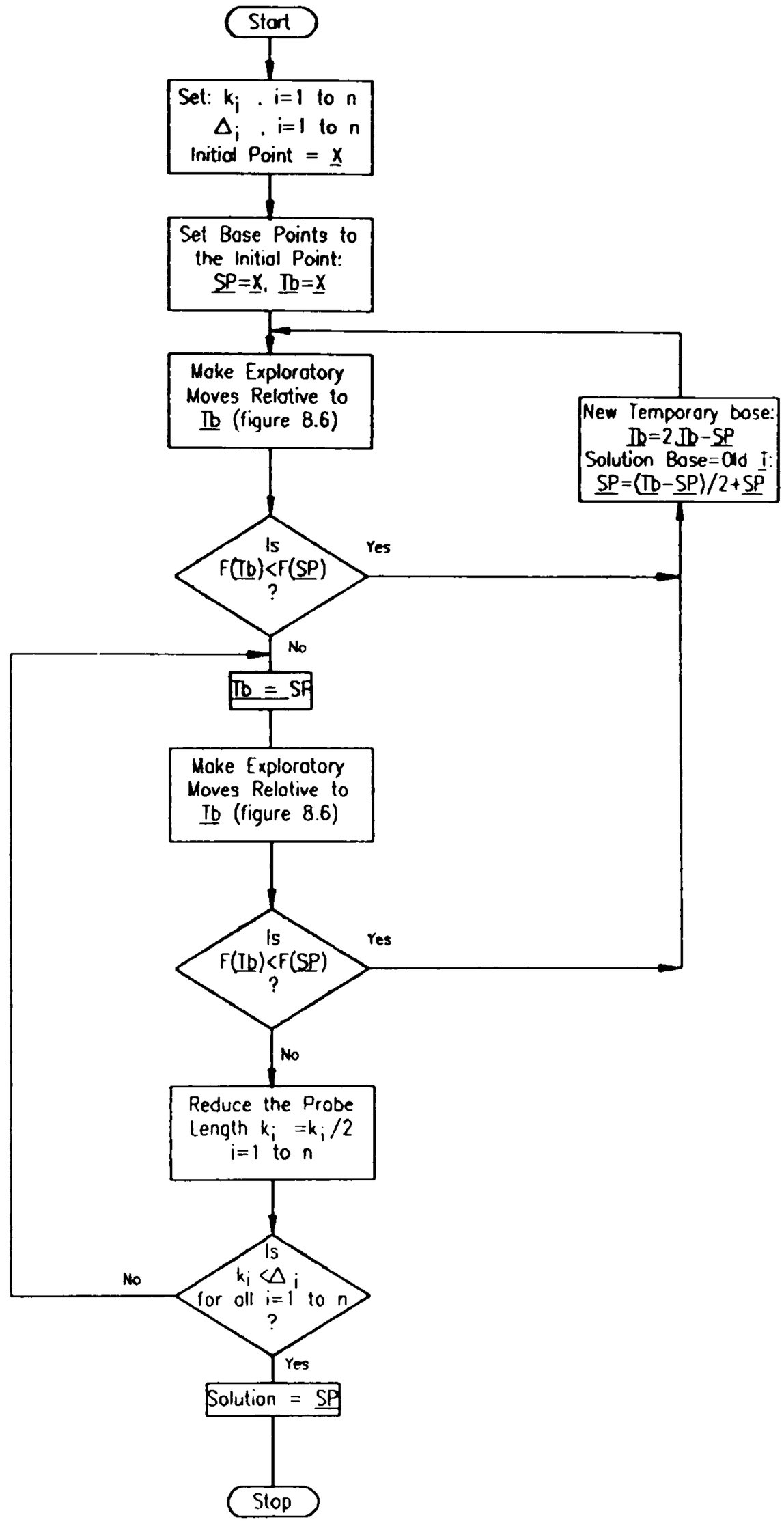
The pattern search begins from a given initial solution point, which is known to be within the feasible region. This solution point is assigned as the initial temporary and solution base. Exploratory moves are made relative to this initial base and the point arrived at used to make a pattern move. The procedure continues to alternate between exploratory and pattern moves until the point reached from a set of exploratory moves has a higher or equal objective function value than the current solution base. When this occurs the temporary base is set to the solution base and the search restarted with a set of exploratory moves. Failure here to locate a new search direction results in a reduction in the probe step length and a repeat of the exploratory moves. The search continues in this manner until a new search direction is found or the step-length falls below a predefined minimum, at which point the search is stopped. The

minimum interval for discrete variables is the discrete interval between values.

This unconstrained version of the pattern search algorithm is illustrated in Figures 8.5 and 8.6. The notation is for single base points Tb and SP which are 'overwritten' in each pattern move. This changes the format of the equations (8.1) and (8.2) to that of equations (8.3) and (8.4) respectively :

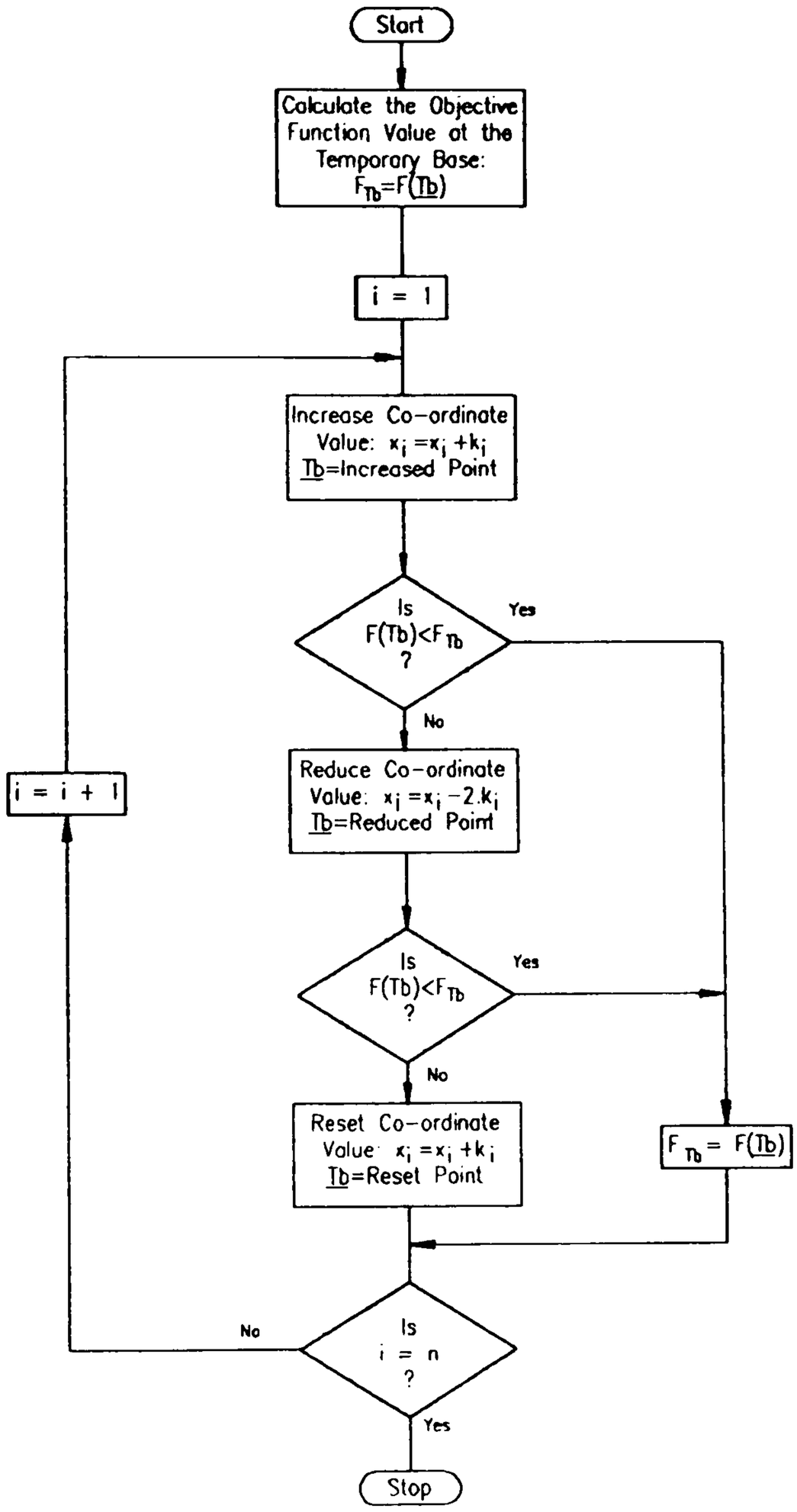
$$Tb = 2.Tb - SP \qquad \text{Equation 8.3.}$$

$$SP = (Tb - SP)/2 + SP \qquad \text{Equation 8.4.}$$



Pattern Search : Pattern Moves

Figure 8.5



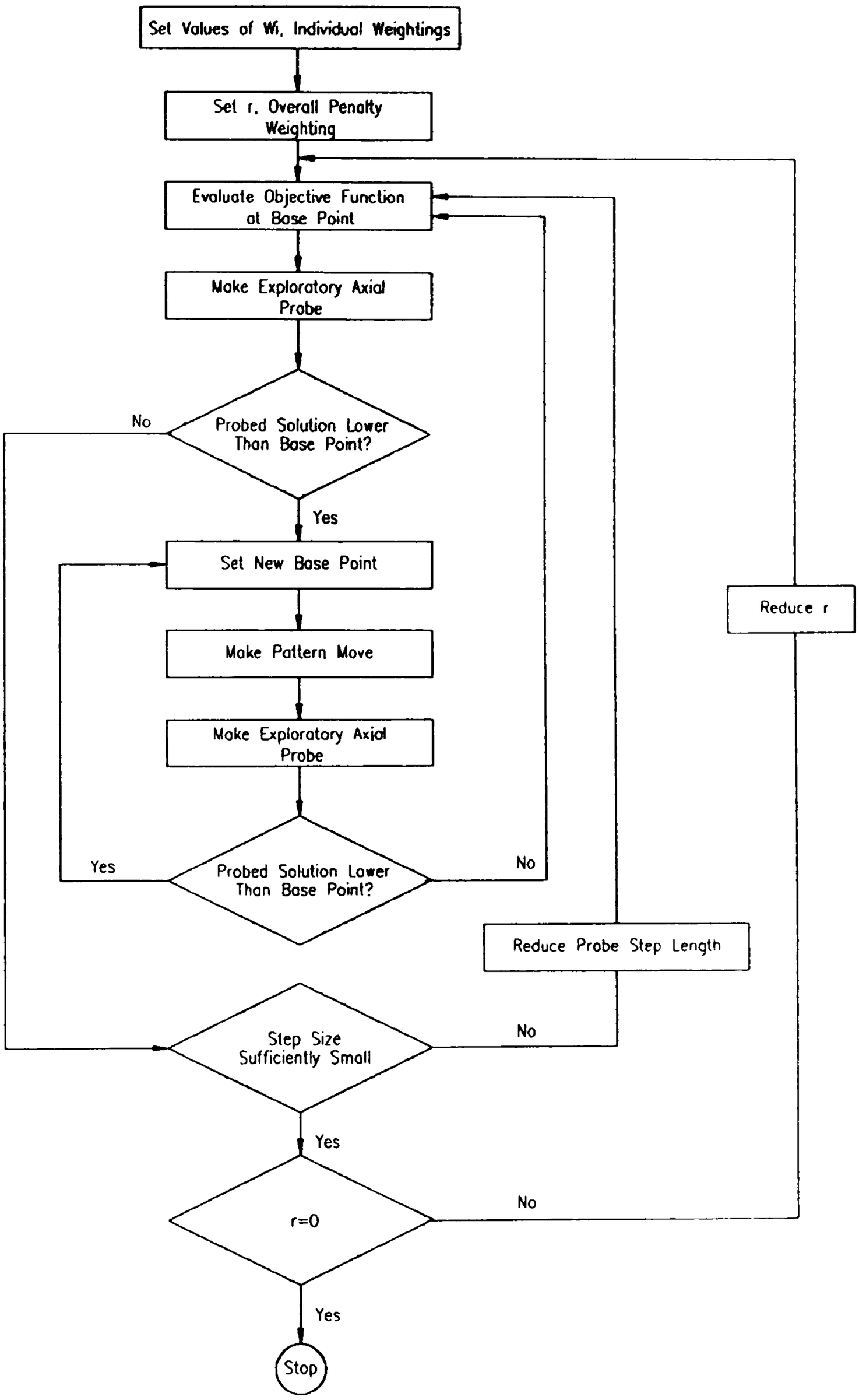
Pattern Search : Exploratory Moves

Figure 8.6

8.3 The Pattern Search - Penalty Function Optimisation Method.

In sections 8.1 and 8.2 of this thesis the penalty function method and the pattern search are fully explained. The two elements are placed together to give the pattern search - penalty function algorithm, with the pattern search driving the search procedure and the penalty function controlling the search within the feasible region. Figure 8.7 illustrates the logic of the two elements relative to each other.

The basic pattern search algorithm was modified to accept discrete variables, and because constraints could be violated if a probe length of a pattern search is too large, simple rejection of infeasible points was introduced into the algorithm. The concluding algorithm was tested against the simplified two-dimensional problem for the cooling coil model detailed in section 7.3.



Flow Diagram of Penalty Function with Pattern Search Algorithm

Figure 8.7

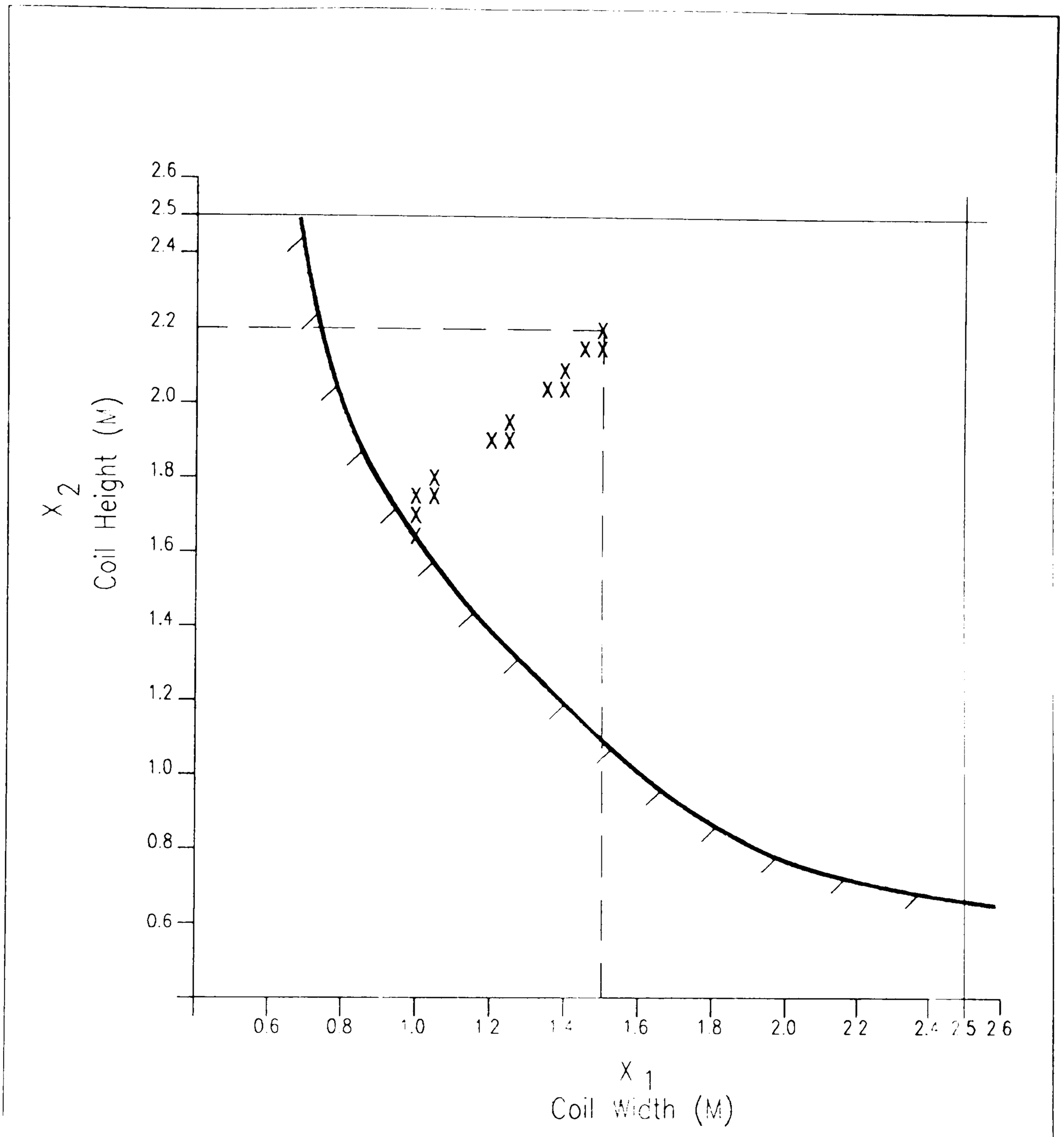
8.4 Testing of the Pattern Search Algorithm without Imposed Penalty.

Because the penalty function optimisation algorithm requires an additional search algorithm to drive the search within the penalty controlled environment it is difficult to assess the advantages of the optimisation approach. To clarify the assessment the pattern search method was run against the two - dimensional cooling coil example problem without penalties imposed. Performance of the pattern search algorithm was assessed to give a good understanding of the algorithms behaviour and the potential problems that the algorithm may encounter. Table 8.8 traces the search progress and is illustrated in two - dimensions in Figure 8.9.

From the chosen feasible initial starting position of $X_1 = 2.2$, $X_2 = 1.5$, the search path aligned itself with that of the steepest gradient of the objective function. In similarity to the complex algorithm the pattern search algorithm principally seeks to find the 'global' optimum solution at $X_1 = 0.05$, $X_2 = 0.05$, and not the constrained optimum solution. The search progresses via pattern and exploratory moves, and accelerates until the constraint function is encountered. At this stage the search attempts to move away from the constraint by changing direction, however, this fails. Figure 8.10 illustrates the reason for the failure of the algorithm. At the point A, (1.6,1.0) an exploratory move is performed by the algorithm, all points in all variable directions are rejected. The points E and D are rejected because of constraint violation, and points B and C are rejected because they do not show an improvement in the objective function value. The point F (1.55, 1.05), a feasible solution point with an improved objective function value, is unable to be reached by the algorithm.

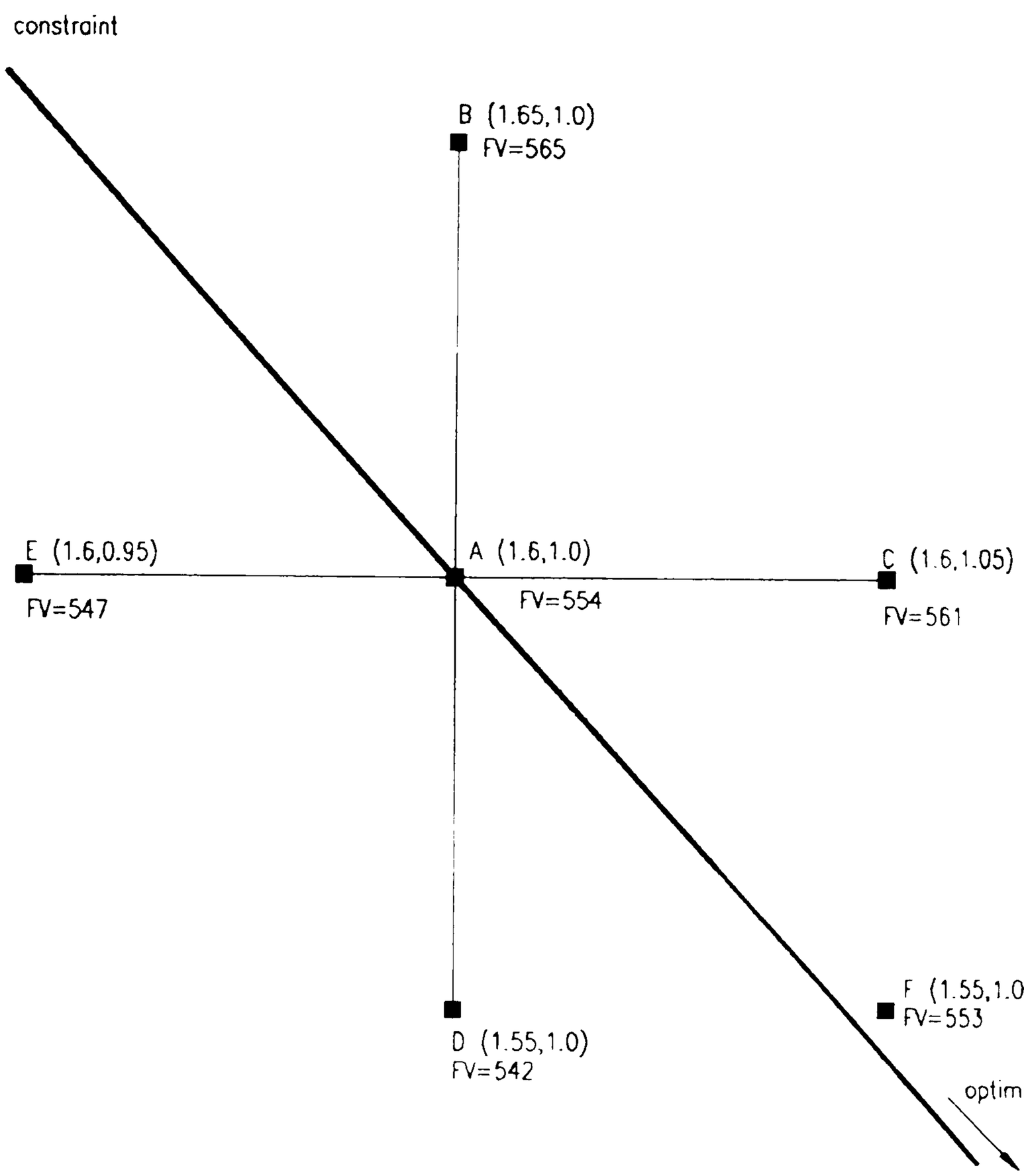
X1	X2	OBJECTIVE FUNCTION VALUE (£)	ACCEPT/ REJECT MOVE	MOVE TYPE
2.20	1.50	786	ACCEPT	STARTING POINT
2.25	1.50	798	REJECT	EXPLORATORY
2.15	1.50	771	ACCEPT	EXPLORATORY
2.15	1.55	780	REJECT	EXPLORATORY
2.15	1.45	762	ACCEPT	EXPLORATORY
2.10	1.40	740	ACCEPT	PATTERN
2.05	1.40	727	ACCEPT	EXPLORATORY
2.05	1.35	718	ACCEPT	EXPLORATORY
1.95	1.25	675	ACCEPT	PATTERN
1.90	1.25	663	ACCEPT	EXPLORATORY
1.90	1.20	655	ACCEPT	EXPLORATORY
1.75	1.05	595	ACCEPT	PATTERN
1.70	1.05	584	ACCEPT	EXPLORATORY
1.70	1.00	576	ACCEPT	EXPLORATORY
1.50	1.80	505	REJECT	PATTERN
1.65	1.00	565	ACCEPT	EXPLORATORY
1.65	0.95	558	REJECT	EXPLORATORY
1.65	1.05	572	REJECT	EXPLORATORY
1.60	1.00	554	ACCEPT	PATTERN
1.55	1.00	542	REJECT	EXPLORATORY
1.65	1.00	565	REJECT	EXPLORATORY
1.60	0.95	547	REJECT	EXPLORATORY
1.60	1.05	561	REJECT	EXPLORATORY
1.55	1.00	542	REJECT	PATTERN

Pattern Search Algorithm Results. Table 8 8



Example Problem using Hooke Jeeves Pattern Search

Figure 3.9



Detail of Failure of Hooke Jeeves Algorithm

Figure 8.10

The pattern search was found to be accurate throughout the progress of the search, it displayed numerical stability and the search speed was acceptable as the algorithm was able to accelerate through use of the pattern moves. The robustness of the search was good whilst the local objective function characteristics remained relatively uncomplicated, but failure was due to a lack of robustness around constraint functions. Clearly the penalty function algorithm will improve the robustness of the pattern search as it will effectively remove the constraint functions from the search environment by increasing the function value of search points close to constraint boundaries.

The overall penalty multiplier r, is critical if the penalty function method is to be successful as this is the factor which will primarily dictate the extent to which the pattern search can approach the constraint functions and hence it is the controlling factor for the search.

8.5. Testing of the Pattern Search - Penalty Function Algorithm against the Two - Dimensional Example Problem.

The testing of the pattern search - penalty function algorithm was conducted against the same cooling coil example problem, as used for the testing and development of the complex algorithm described in Chapter 7.

8.5.1. The Penalty Function Environment.

For the pattern search - penalty function algorithm testing an initial feasible point was chosen at a position X_1 = coil width = 2.2m, X_2 = coil height = 1.5m. To ensure that the penalties imposed on each constraint function and variable bounds did not distort the problem significantly the penalty weightings imposed were made equal to each other. The variable bounds and constraint functions are all treated in the same way. To find the values of the constraints at the starting position the constraint values are expressed in the form $C_i(X) > 0$, where C_1 relates to the lower bound of X_1, C_2 relates to the upper bound of X_1, C_3 relates to the lower bound of X_2, C_4 relates to the upper bound of X_2, C_5 relates to the lower bound of the constraint function of coil air face velocity and C_6 relates to the upper bound of the constraint function of air face velocity. The variable bounds are lb_1 =0.0, lb_2 = 0.0, ub_1 = 2.5, and ub_2 = 2.5, and the bounds on the face velocity constraint are $lb_{(face\ velocity)}$ = 0.0, and $ub_{(face\ velocity)}$ = 2.5. The values of C_i for the starting position are therefore :

$C_1 = X_1 - lb_1 = 2.2$

$C_2 = ub_1 - X_1 = 0.3$

$C_3 = X_2 - lb_2 = 1.5$

$C_4 = ub_2 - X_2 = 1.0$

$C_5 = 4/(X_1)(X_2) - lb_{(face\ velocity)} = 1.2121$

$C_6 = ub_{(face\ velocity)} - 4/(X_1)(X_2) = 1.2879$

The individual weighting W_i for the constraints are set such that the penalty terms, W_i/C_i are equal. The W_i values are set as follows :

$W_1 = 2.2$

$W_2 = 0.3$

$W_3 = 1.5$

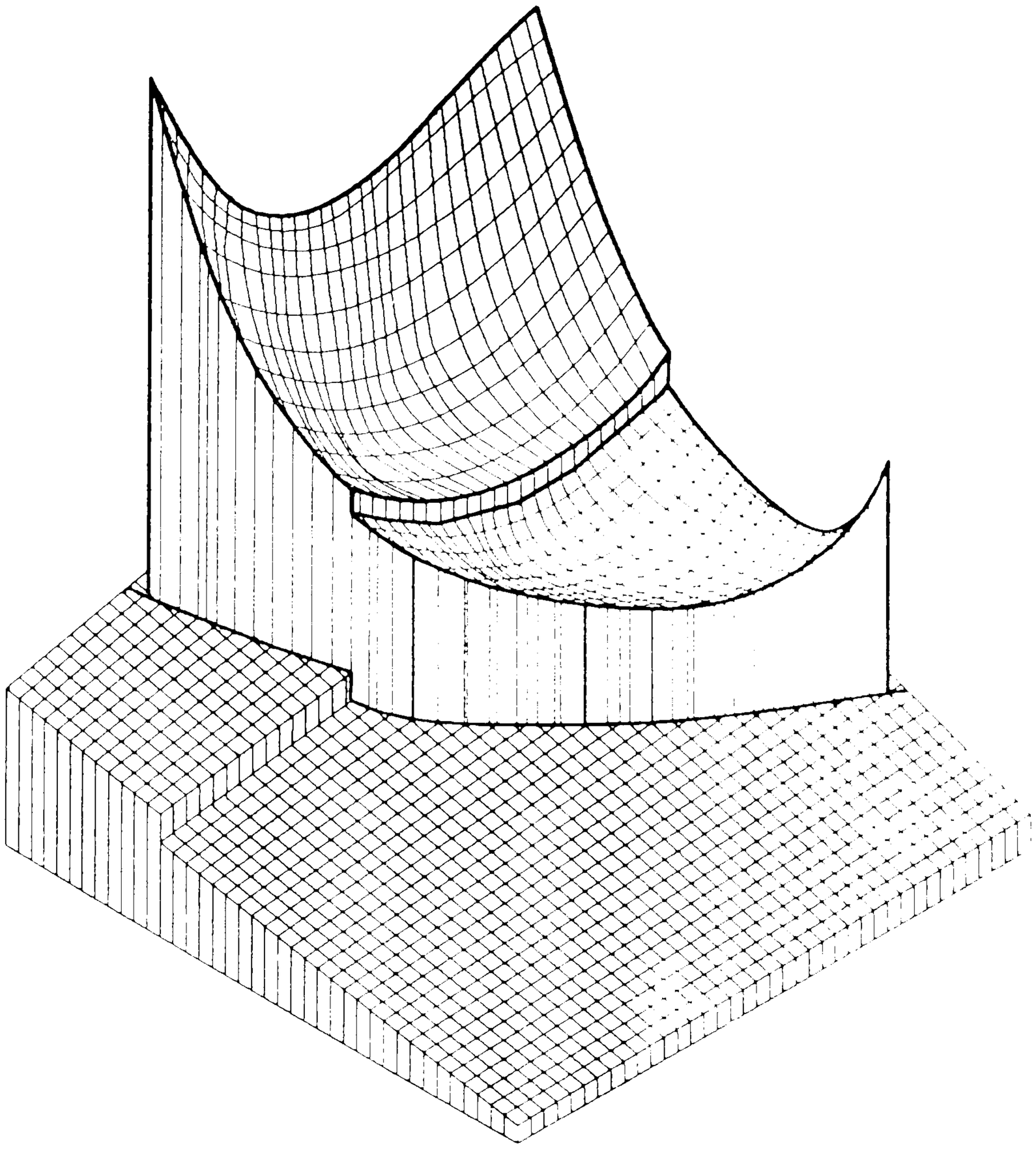
$W_4 = 1.0$

$W_5 = 1.2121$

$W_6 = 1.2879$

thus setting the individual penalty terms $W_i/C_i = 1$.

Figure 8.11 illustrates the effect of the penalty weightings on the objective function surface. The overall penalty multiplier r, is set to a value r = 100 to give reasonable definition to the diagram. The edges of the penalty are 'clipped' in the diagram as they tend to infinitely large values which distort the resolution of the diagram for the rest of the objective function surface.



Three Dimensional View of the Constrained Problem

Figure

8.5.2. An Investigation of the Penalty Multiplier.

As previously stated in section 8.4, the the penalty multiplier r, is considered to be critical to the success of the penalty function algorithm when used with a pattern search, as this factor dictates the extent to which a constraint function can be approached, a known area of the feasible region which the pattern search shows weakness.

Previous work by Fiacco and McCormick (1964), investigated the initial value of the penalty multiplier r, and the subsequent reduction in the value of r to zero. In respect to the research in this thesis the findings of this work where inconclusive, suggesting that the initial value of r was dependent on the starting position of the search and that the reduction in the value of r was not an influencing factor on the overall speed of the search. The optimum value of r is therefore specific to the given optimisation problem.

To investigate of the selection of the penalty multiplier r, an exhaustive search of the value of r was carried out to establish it's optimum value. The criteria used to assess the effectiveness of r, was the position of the search finishing position relative to the 'true' optimum solution, and the number of objective function calls made. Tables 8.12, 8.13, and 8.14 show the exhaustive search results for the optimum values of r_i until $r_i = 0$.

STARTING POSITION X1 = 2.20, X2 = 1.50				
r1	X1	X2	OBJECTIVE FUNCTION VALUE (£)	TOTAL FUNCTION CALLS
100	1.20	1.95	1070	42
50	1.05	2.05	754	61
25	0.95	2.10	583	48
20	0.90	2.15	544	48
14	0.85	2.20	496	56
12	**0.80**	**2.30**	**480**	**47**
11	0.85	2.20	470	48
10	0.85	2.20	462	51

The Optimum Value of the Penalty Multiplier, r1. Table 8.12

STARTING POSITION X1 = 0.80, X2 = 2.30				
r2	X1	X2	OBJECTIVE FUNCTION VALUE (£)	TOTAL FUNCTION CALLS
12	0.80	2.30	480	5
10	0.80	2.30	461	5
8	0.80	2.25	442	12
6	0.80	2.25	423	10
5	0.75	2.35	412	9
4	**0.75**	**2.35**	**400**	**9**
3	0.75	2.30	388	10
2	0.75	2.30	376	10

The Optimum Value of the Penalty Multiplier, r2. Table 8.13

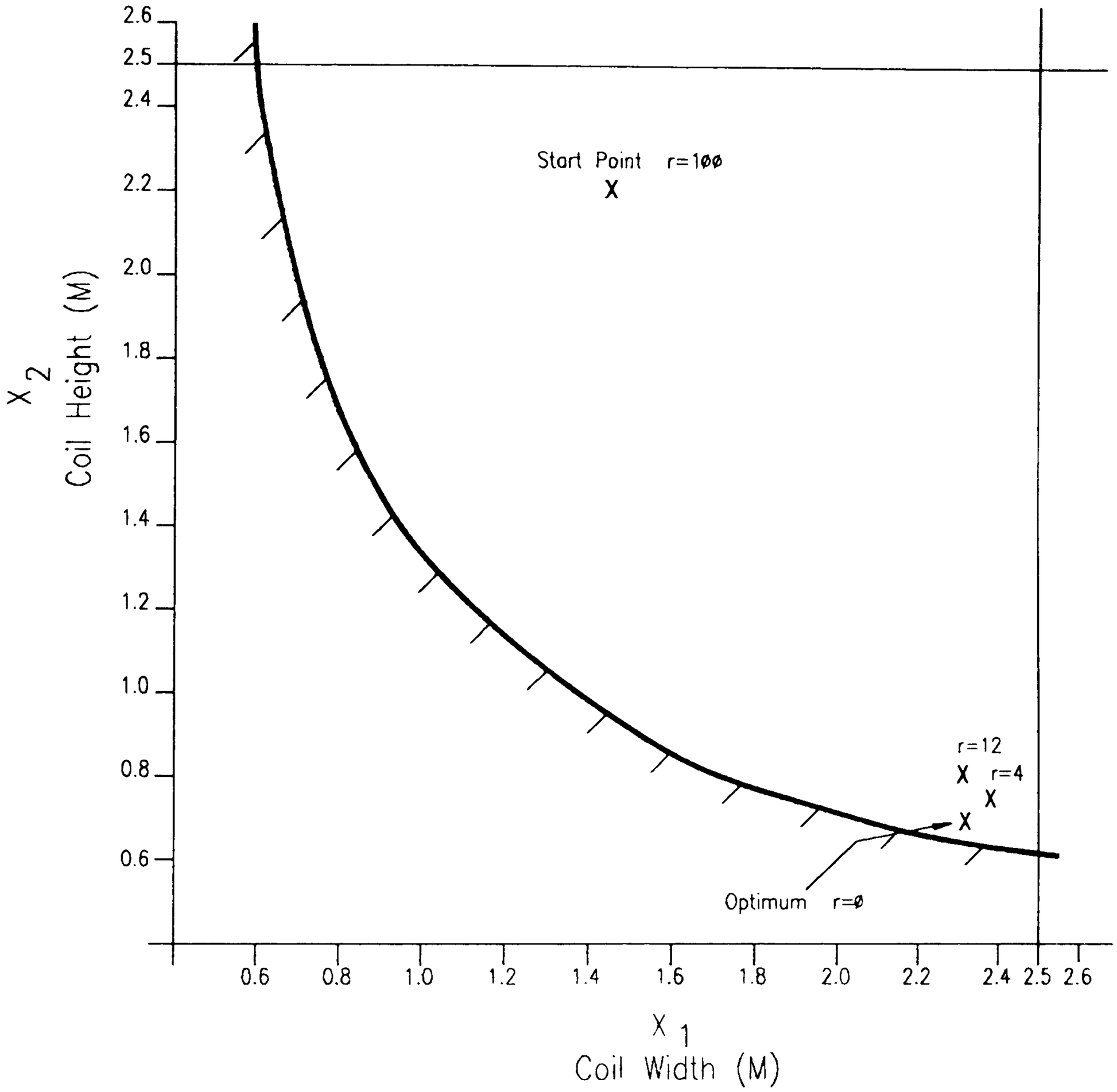
STARTING POSITION X1 = 0.75, X2 = 2.35				
r3	X1	X2	OBJECTIVE FUNCTION VALUE (£)	TOTAL FUNCTION CALLS
4	0.75	2.35	400	5
3	0.75	2.30	388	10
2	0.75	2.30	376	10
1	0.70	2.40	358	9
0	0.70	2.30	334	15

The Optimum Value of the Penalty Multiplier, r3. Table 8.14

Figure 8.15 shows the path that the search took for each successive value of the penalty multiplier. It can be seen that with correct selection of the value of r, for each successive search, allowed the optimum solution to be found in a stable and fast manner. The total number of function calls made throughout the course of the search was 71, representing a significant improvement in search speed in comparison to the performance of the complex algorithm against the same example problem. The selected penalty multiplier values ensured that at no stage throughout the search was a constraint function violated, hence removing the weakness experienced by the pattern search algorithm when operating in the uncontrolled environment detailed in section 8.4.

It was found, however, that the tolerance on the selection of the penalty multiplier value was particularly low. Selection of the correct value for r was in fact critical to the continued success of the algorithm. Too small a reduction in the value of r, caused the search to move slowly, whereas too large a reduction caused instability of the search and effective failure of the algorithm without finding the optimum solution. The conclusion drawn from the assessment of the effectiveness of the penalty multiplier value was that unless an optimum value r, was used at each stage of the search then the overall effect of the penalty distorted the 'true' objective function surface to such an extent that the natural characteristics of the 'true' objective function were lost to the imposed penalty.

Further development of the penalty weightings was therefore considered to be necessary.



Optimum Path for the Reduction of r, Penalty Multiplier

Figure 8.15

8.6. Active and Inactive Penalty Weightings.

The penalty function algorithm described and tested in the previous sections employed penalty weightings W_i for all of the constraints; namely the lower and upper bounds of both problem variables, and the upper and lower constraints of the face velocity. Hence, the shape of the objective function is significantly changed across the region of feasible solutions.

Further development of the penalty function algorithm was performed to identify the constraint penalties that influenced the search direction and those that did not. Each constraint was selected in turn as the basis for the penalty function, whilst the remaining constraints were excluded from the penalty, . For example, the constraint C_1, relating to the lower bound of cooling coil width had a penalty weighting W_1 imposed upon it, whilst the remaining constraints C_2 to C_6 were given no penalty, the created response objective function therefore becomes:

$$F^*(X) = F(X) + r\,(W_1/C_1)$$

Similarly the second run considered the constraint C_2 to have an active penalty weighting W_2, and the constraints C_1, C_3, C_4, C_5, C_6 to have no penalty weightings attached.

The results of imposing an active penalty by this method are shown in Table 8.16.

ACTIVE PENALTY WEIGHTING	STARTING POSITION		SOLUTION FOUND		OBJECTIVE FUNCTION VALUE (£)	TOTAL FUNCTION CALLS
	X1	X2	X1	X2		
W1	2.20	1.50	1.60	1.00	576	28
W2	2.20	1.50	1.60	1.00	576	28
W3	2.20	1.50	1.60	1.00	576	28
W4	2.20	1.50	1.60	1.00	576	28
W5	2.20	1.50	1.60	1.00	576	28
W6	2.20	1.50	0.75	2.50	399	48

Active Penalty Weightings – Results. Table 8.16

For each individual run the overall penalty multiplier r was equal to 10. The active penalty weightings W_1 to W_5 all produced the same solution point of $X_1 = 1.6$, $X_2 = 1.0$, from the initial starting position of $X_1 = 2.2$, $X_2 = 1.5$. This is the same solution found by the pattern search method without a penalty function attached. In each case the search failed against the face velocity constraint, giving no improvement to the basic pattern search method.

When W_6 was made the active penalty weighting, relating to the upper bound of the face velocity constraint, a different final solution was found. By making the approach towards the upper limit of the face velocity constraint less attractive, the problem of the algorithm failing against this constraint, as described in section 8.4, was eliminated.

The solution found by applying a penalty weighting to the upper limit of the face velocity constraint is $X_1 = 0.75$, $X_2 = 2.50$, this is not the 'true' optimum, but is close to the 'true' optimum. By setting the penalty multiplier r = 0, and hence removing all penalties, the algorithm quickly finds the 'true' optimum using this position as the new starting point.

8.7 Accuracy of the Penalty Function Algorithm with Active Penalty Weightings.

The results of the penalty function algorithm with active penalty weightings were accurate throughout the testing. The development of the algorithm was such that discrete problem variable values were found at each trial point.

The constraint and objective function values found throughout the testing were also accurate, ensuring that the search remained numerically stable. The nature of the example problem and the monitoring of the search algorithm ensured that accuracy was an inherent feature throughout.

8.8 Speed of the Penalty Function Algorithm with Active Penalty Weightings.

Throughout the testing of the penalty function algorithm the speed of the search was monitored via the number of objective function or constraint function calls. The effective elimination of constraint functions from the example problem by implementation of the active penalty weightings meant that the search did not spend time correcting violated points. It was considered that this was a useful factor for increasing the overall speed of the search. It can be seen from Table 8.16 that the successful search made when $W_{6,}$ was the active penalty weighting found the 1^{st} optimum solution in 48 function calls and took a further 17 function calls to find the 'true' optimum once the penalty multiplier r, was reduced to zero, giving a total of 65 function calls.

Two comparisons can be made to assess the speed of the penalty function algorithm, firstly against an exhaustive search evaluating each discrete solution position, and secondly against the speed of the complex method algorithm. Throughout the testing of the penalty function algorithm the discrete interval for the problem variables of coil width and coil height

were 0.05m, whereas during the complex algorithm testing these discrete intervals were 0.1m. The variable bounds remained the same with lower bounds = 0.0m and upper bounds = 2.5m. The penalty function algorithm was therefore tested on an example problem with 4 times as many potential solution points. The comparison of search speed is presented in Table 8.17 with percentage figures attached to enable comparison.

A significant improvement in the search speed can be seen between the complex algorithm and the penalty function algorithm, the total number of function calls has reduced from 91 to 65 whilst the number of solution points has increased, reflecting a superior lower percentage figure of function calls made to the total made by the exhaustive search.

SEARCH ALGORITHM	EXAMPLE PROBLEM SOLUTION POINTS	NUMBER OF SEARCH FUNCTION CALLS	PERCENTAGE (%)
EXHAUSTIVE SEARCH	*625/2500	*625/2500	100
COMPLEX SEARCH	625	91	14.5
PENALTY FUNCTION SEARCH	2500	65	2.6

(* Denotes the number of solution points availiable for each example).

Search Speed Comparison. Table 8.17

8.9 The Numerical Stability of the Penalty Function Algorithm with Active Penalty Weightings.

In similarity to the complex algorithm the penalty function algorithm was found to be entirely numerically stable. The use of discrete variables ensured that instability due to rounding errors in the objective function evaluation was eliminated, the discrete positions being far enough apart to make rounding errors insignificant in relation to the change in objective function value between two discrete points.

The numerical stability of the objective and constraint function evaluations again were entirely satisfactory.

The cause of the numerical instability encountered when all penalty weightings were active, had been found to be due to the selection of the value of r, the penalty multiplier. The pre-selection of active and inactive penalty weightings imposed upon the constraints successfully cured this instability.

8.10. The Robustness of the Penalty Function Algorithm with Active Penalty Weightings.

The failure of the pattern search without penalty weightings attached detailed in section 8.4 was found to be due to a lack of algorithm robustness around constraint functions. The introduction of a penalty function enhanced the algorithm robustness but limitations still exist. The pattern search was found to be sensitive to small changes in the penalty function

multiplier ,r, and hence the robustness of the algorithm was still considered to be inadequate.

Introduction of active and inactive penalty weightings successfully improved the robustness of the algorithm to an acceptable level. The correct choice of the active penalty weighting was however found to be critical for a successful search to take place. From Table 8.16 it can readily be seen that an incorrect choice of active penalty weighting will result in the algorithm failing against the face velocity constraint. The penalty weightings W_1, W_2, W_3, W_4, and W_5 when active all resulted in failure of the algorithm. Only the selection of W_6 as the active penalty weighting resulted in a successful search which was able to negotiate the characteristics of the example problem, in particular the constraint functions. The robustness of the algorithm against the particular problem characteristics therefore still remains somewhat inadequate.

8.11 The Full Cooling Coil Example Problem with Active and Inactive Penalty Weightings.

The results of the testing of the penalty function algorithm against the two - dimensional cooling coil example problem, using active and inactive penalty weightings showed significant improvement to the complex and penalty function algorithm in its earlier form. In particular the speed of the search was improved greatly. Although there still exists the problem of correct selection of the active penalty weightings it is important to establish that this algorithm can be usefully applied to larger scale optimisation problems where the optimisation of the problem is run in conjunction with a component simulation. The full scale example problem described in

Chapter 6 of this thesis was therefore used and the penalty function algorithm using active and inactive penalty weightings tested against it.

The penalty weighting W, attached to each constraint C, within the full scale example problem were as follows :

W_1 - the weighting related to the upper bound of the coil rows.

W_2 - the weighting related to the lower bound of the coil rows.

W_3 - the weighting related to the upper bound of the coil width.

W_4 - the weighting related to the lower bound of the coil width.

W_5 - the weighting related to the upper bound of the coil height.

W_6 - the weighting related to the lower bound of the coil height.

W_7 - the weighting related to the upper bound of the coil circuits.

W_8 - the weighting related to the lower bound of the coil circuits.

W_9 - the weighting related to the upper bound of the coil maximum water mass flow rate.

W_{10} - the weighting related to the lower bound of the coil maximum water mass flow rate.

W_{11} - the weighting related to the upper bound of the coil face velocity constraint.

W_{12} - the weighting related to the lower bound of the coil face velocity constraint.

W_{13} - the weighting related to the upper bound of the coil water velocity constraint.

W_{14} - the weighting related to the lower bound of the coil water velocity constraint.

W_{15} - the weighting related to the upper bound of the proportional control constraint.

With the added knowledge gained from the testing of the two-dimensional problem many of the penalty weightings when active are known to have no advantageous effect on the search. The penalty weighting which proved to be successful when active was that related to the upper bound of the face velocity constraint. In this problem the penalty weighting relating to this constraint is W11. The effect of the water velocity constraint and the control constraint are unknown. The penalty function algorithm with active penalty weightings was therefore run against the full cooling coil example problem with the penalty weighting of W_{11}, W_{13}, and W_{15} active.

8.12. The Results of the Penalty Function Algorithm with Active Penalty Weightings using the Full Scale Cooling Coil Optimisation Problem.

The penalty weightings W_{11}, W_{13}, and W_{15} were made active in turn and the penalty function algorithm run against the example problem. Additional runs for the combination of the three penalty weighting being active were completed. The overall penalty multiplier r, was set at an initial value of 10, and subsequently reduced to a value of zero once a solution point was found. The second run of the algorithm therefore is against the 'true' objective function without penalty imposed. An initial starting point was chosen as follows:

X_1 = number of cooling coil rows = 6.

X_2 = cooling coil width = 1.6m.

X_3 = cooling coil height = 1.8m.

X_4 = number of cooling coil circuits = 34.

X_5 = maximum water mass flow rate = 9.0 m/s.

This starting position was used for all runs. Table 8.18 gives an account of the results obtained.

Initial inspection of the results give rise to certain conclusions. When assessed against the objective function value and the proximity the the final solution to the optimum solution, the algorithm performs best when W_{15}, the control constraint weighting is active. Penalty weightings W_{11} and W_{13} alone and in combination produce identical results which are less accurate in terms of the objective function value and the proximity of the final solution to the optimum.

	STARTING POSITION					SOLUTION FOUND, r = 10.					SOLUTION FOUND, r = 0.						
ACTIVE PENALTY	X1	X2	X3	X4	X5	X1	X2	X3	X4	X5	X1	X2	X3	X4	X5	OBJECTIVE FUNCTION VALUE (£)	TOTAL FUNCTION CALLS
W11	6	1.60	1.80	34	9.0	5	1.35	1.80	34	9.0	5	1.35	1.80	34	9.0	836	75
W13	6	1.60	1.80	34	9.0	5	1.35	1.80	34	9.0	5	1.35	1.80	34	9.0	836	75
W15	6	1.60	1.80	34	9.0	5	1.15	1.95	28	9.0	5	1.15	1.70	28	9.0	714	119
W11,W13	6	1.60	1.80	34	9.0	5	1.35	1.80	34	9.0	5	1.35	1.80	34	9.0	836	75
W11,W15	6	1.60	1.80	34	9.0	5	1.05	2.00	28	9.0	5	1.05	1.90	28	9.0	699	113
W13,W15	6	1.60	1.80	34	9.0	5	1.20	1.95	30	9.0	5	1.10	1.95	30	9.0	734	90
W11,W13,W15	6	1.60	1.80	34	9.0	5	1.20	1.95	30	9.0	5	1.10	1.95	30	9.0	734	90

Active Penalty Weighting Results – 5 Variable Problem. Table 8.18

The reason for the significance of the control constraint is not entirely clear, but appears to be due to a combination of effects. Firstly, this constraint is outside the face velocity constraint and as such when its weighting is made active the search can approach the face velocity constraint to a greater extent than when the face velocity constraint itself is active, hence a subtle change in the imposed shape of the objective function surface around the face velocity constraint produces a different search path. Full inspection of the search path upholds this theory, in that a comparison of the temporary bases reached by the algorithm when W_{11} is active against when W_{15} is active yield different search paths. The the active constraint influences the early part of the search and therefore the final solution point found. The second reason is that the algorithm examines each problem variable in turn and in the predefined order in which they are presented in the initial setting up of the problem. This limits the flexibility of the search path. It may be the case that the change in the value of one variable means that a more advantageous search position is not obtainable when a subsequent problem variable is examined. This can be visualised using an arbitrary example, say, at a temporary base solution of (5, 1.6, 1.8, 34, 9.0) the X_2 variable is examined and a move from $X_2 = 1.6$ to $X_2 = 1.8$ is accepted, a better solution may have been where X2 remains equal to 1.6 and X_3 is moved from $X_3 = 1.8$ to $X_3 = 2.0$. With the X_2 variable always being examined ahead of the X_3 variable the opportunity to find that better solution position is missed. The order of the problem variables therefore will have an effect on the progress of the search. As the number of problem variables increases the probability of this scenario happening is also increased. The control of the algorithm is seriously undermined by this effect.

Another weakness of the algorithm is also apparent from the use of the full scale cooling coil example problem. Variables such as maximum water mass flow rate have no direct influence on the value of the objective function, yet changes in the variable value have an indirect effect by allowing more attractive positions of the other problem variables to be found, that do influence the objective function. The results show that in all of the runs of the penalty function algorithm the value of the variable of maximum water mass flow rate did not change because a better objective function value was not resultant. Larger scale HVAC systems will have more problem variables of this kind and the algorithm as it stands would not have the flexibility to cope with such problems.

8.13. Penalty Function Algorithm Conclusions.

Clearly the penalty function algorithm with active constraints showed a significant improvement in performance when compared to the complex algorithm. The adoption of the algorithm when applied to the full scale cooling coil example problem showed its limitations, and areas where the algorithms approach needs strengthening.

The use of active and inactive penalty weightings proved to be a successful method by which the penalty function algorithm could find the optimum solution for small scale problems. The choice of the active penalty weighting through knowledge of the problem could be determined for small scale problems, when used with the larger scale example problem this choice became less apparent. Exhaustive algorithm runs for large scale problems to find the influential penalty weightings would obviously

detract from the objective of the optimisation algorithm to find the optimum solution at an acceptable speed.

The use of the pattern search as the driving direct search method with the penalty function algorithm proved to have a degree of success. The pattern search when used alone failed around constraint functions, introduction of a penalty function avoided such failure and enhanced the algorithms robustness. Limitations of the pattern search as the choice of direct search method became apparent when the penalty function algorithm was tested against the full scale cooling coil problem. The performance of the algorithm could be influenced by the initial definition of the problem. The choice of variable order and the methodology of the pattern search algorithm has an influence on the search path, making potential solution points outside of the scope of the algorithm.

A final conclusion to be drawn from the testing of the penalty function algorithm against the full scale example problem, was that the algorithm was unable to have an influence on the selection of the value of problem variables which have no direct effect on the value of the objective function. A simple change in the value of the maximum water mass flow rate variable had no effect on the objective function. The algorithm rules are such that an improvement in objective function value is necessary for a new solution point to be accepted. Such problem variables that may have an indirect effect on the optimisation, by giving other problem variables scope to improve the objective function, are not sufficiently catered for by the algorithm.

CHAPTER 9.

THE DEVELOPMENT OF A UNIQUE ALGORITHM FOR THE SOLUTION OF HVAC SYSTEM DESIGN OPTIMISATION PROBLEMS.

Although the development and testing of direct search algorithms to solve HVAC system design optimisation problems described in this thesis has met with some success it can be concluded that the development is not sufficiently advanced to give full reliability when used against large scale optimisation problems. The characteristics of HVAC system design optimisation problems are complex and the approach for solving such problems, using the algorithms detailed in this thesis, is not considered to be sufficient for all HVAC system optimisation problems.

Three areas for improved development of the algorithm are considered. Firstly, specific improvement in the ordering of problem variables should be explored, secondly an investigation into the merits of decomposing the optimisation problem into a series of smaller more manageable sub-optimisation problems should take place, and thirdly, investigation into the introduction of occasional random selection of solution points to act as a check that the algorithm is proceeding to the correct optimum solution should be embarked upon. Such random selections of solution points would enable problem variables which have no direct influence on the objective function and sparse configuration constraints to be addressed more fully.

Investigation, development and incorporation of these elements into an optimisation algorithm would be a major undertaking and as such is outside the scope if this research. This chapter preliminarily addresses these areas for further investigation and development.

9.1 Problem Variable Sensitivity Analysis and Ordering of Variables.

From the development and testing of the penalty function algorithm using a pattern search, limitations associated with the definition of the optimisation problem were discovered. Improvements in the definition to be more in line with the specific optimisation problem are needed.

Separate routines were developed and incorporated into the definition of the problem. The routines investigated the extent of the influence of each problem variable on the objective function, and selection of the order in which the problem variables are presented to any given search algorithm.

From the initial starting position each problem variable is assessed in turn. The variable bounds are established and a step-length movement equal to one tenth of the problem variable range is made. The gradient of the objective function in relation to problem variables is calculated. The extent of the gradient is used as an assessment of the influence of the problem variable. By ordering the objective function gradients the problem variable order presented within the definition of the optimisation problem is established. The problem variable producing the greatest change in objective function value being presented as the first variable to the search algorithm, with the other problem variables being presented in descending

order of change.

The developed routines were incorporated into the penalty function algorithm employing the pattern search (Section 8.11). An active penalty weighting W_{11} relating to the upper bound of the face velocity constraint was included.

9.1.1. Results of the Ordered Problem Variable Penalty Function Algorithm with Pattern Search.

Two runs of the algorithm were made, the first was made using the algorithm in its previous form without ordered problem variables, the second was made after the problem variables had been ordered according to the change in objective function. Table 9.1 compares the results obtained from the two searches.

The first search run maintained the order of the problem variables as X_1 = number of coil rows, X_2 = coil width, X_3 = coil height, X_4 = number of coil circuits, and X_5 = maximum water mass flow rate. The second search run arranged the problem variables according to the influence on the objective function, as X_1 = number of coil rows, X_2 = coil height, X_3 = coil width, X_4 = number of coil circuits, and X_5 = maximum water mass flow rate. Hence the second search promoted the problem variable of coil height to a position before the problem variable of coil width. The results show a significant improvement in the final solution and the performance of the algorithm in terms of the overall speed of the search when the problem variables are ordered.

W11 ACTIVE	STARTING POSITION	SOLUTION FOUND	OBJECTIVE FUNCTION VALUE (£)	TOTAL FUNCTION CALLS
VARIABLE ORDER RUN 1				
X1	6	5	690	162
X2	1.60	1.85		
X3	1.80	1.05		
X4	30	30		
X5	9.5	9.5		
VARIABLE ORDER RUN 2	STARTING POSITION	SOLUTION FOUND	OBJECTIVE FUNCTION VALUE (£)	TOTAL FUNCTION CALLS
X1	6	5	662	108
X3	1.80	0.95		
X2	1.60	2.00		
X4	30	30		
X5	9.5	9.5		

Ordered Problem Variable Results. Table 9.1

A full investigation of the mechanism of ordering the problem variables in terms of the effect that this has on the pattern search should be considered. The initial results suggest that some benefits may be established from this undertaking.

Clearly other integral problems exist with the use of the pattern search with the penalty function algorithm. Problem variables which do not have a direct effect on the objective function are not dealt with satisfactorily by the pattern search which requires an improvement in the objective function for a new solution point to be accepted. The size of HVAC system design optimisation problems also cause the pattern search algorithm to perform poorly and ordering of the variables is probably only a partial solution.

9.2. Optimisation Problem Reduction and Simplification.

The weakness of the direct search algorithms developed during this research have been largely due to the robustness of the algorithm around the constraint function, in particular when discrete problem variables are present, and the inability of the algorithm to be successful when employed with large scale optimisation problems. A preliminary investigation into methods for reducing the size of the optimisation problem was therefore undertaken. Initially three methods have been considered:

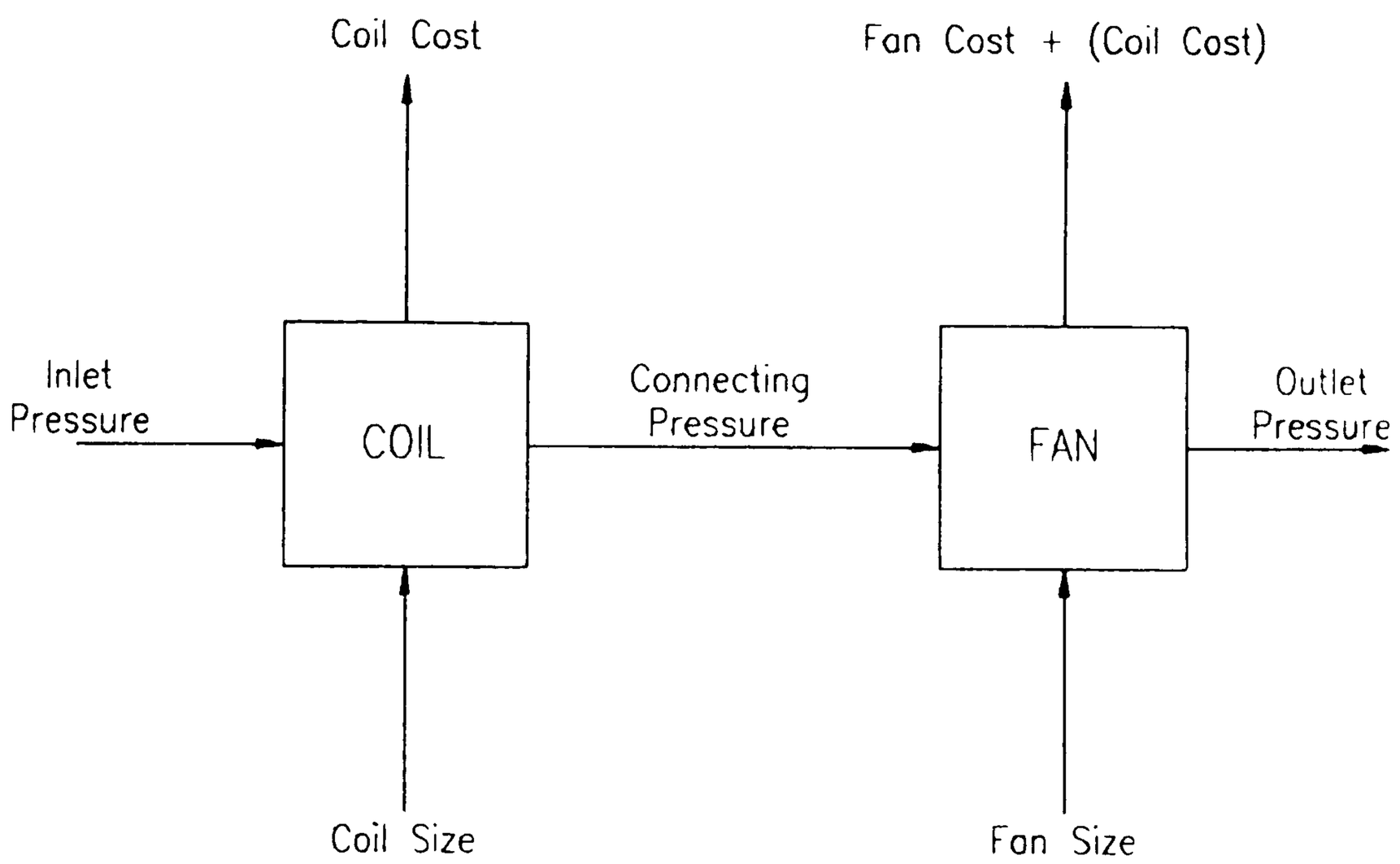
1) Decomposition of the problem into a sequence of sub-optimisation problems.

2) The use of a simplified system simulation model for intermediate search moves.

3) The elimination of problem variables and constraint functions from the optimisation problem.

9.2.1. Optimisation Problem Decomposition.

Generally large scale optimisation problems such as those involved with HVAC system design consist of a series of smaller sub-optimisations as there exists a link between the components within the system, (Siddel, 1982). It follows that the smaller the size of the optimisation problem the easier it is for an algorithm to solve, and as such there exists the potential to ease the problems experienced by the algorithm with large scale optimisation problems. For example, Figure 9.2 illustrates a constant mass flow fan-coil system. The inlet pressure to the coil and the outlet pressure from the fan are fixed. The link between components is represented by the connecting pressure of the air. The approach adopted by decomposing this system into sub-optimisations to minimise capital cost would be firstly to find the optimum size of the coil in terms of cost and to repeat this optimisation for a range of feasible connecting pressures. The fan would then be optimised with the connecting pressure values input as a problem variable, the associated coil cost for each given connecting pressure would be expressed as a function of that variable. The optimum capital cost found would be used to determine the component sizes.



Fan - Coil Optimisation Decomposition

Figure 5.2

This technique of decomposing an optimisation problem into a series of sub-optimisations is reliant on a positive linkage between the sub-optimisations, i.e. the sub-optimisations must follow each other sequentially with a solution or range of solutions from one forming an input variable to the next. HVAC systems often do not possess such straight forward linkage with feedback from one part of the system influencing a component previously dealt with. This can be visualised by consideration of a recirculating HVAC system where the recirculated air influences the condition of the supply air to the system, the sequential format necessary for decomposition is therefore broken.

A second drawback to this technique is that often a series of values are necessary from one sub-optimisation to form the range for a subsequent input variable to the next sub-optimisation as is the case in the fan-coil example. This means that the sub-optimisations have to be repeated to produce the range of values and hence detracts from the speed of the overall system optimisation. It is not clear as to weather this would be prohibitive to the objectives of the optimisation and should form the basis for further research.

9.2.2. Simplified System Simulation Models.

The time that an HVAC system design optimisation problem takes to find each new solution point is dependent on the solution time of the simulation procedure, therefore considerations should be given to the development of a simplified simulation. Research has recently been completed into methods for reducing the simulation of HVAC systems

that function without adversely effecting the stability of the system or component models, (Hanby, 1988). Until this area of research is suitably developed it maybe possible to produce approximate solution points by means of a simplified simulation.

The approach would be to predominantly produce solution points throughout an optimisation search using a simplified simulation of the system, a full simulation being used occasionally to check that the search is proceeding correctly towards the optimum solution, as it would be unlikely that the simplified simulation would correlate fully to the full simulation hence the need to check that the search was not being misled by the simplified simulation.

During the early development of simulation techniques for HVAC systems, simplified models were produced, (Stoecker, 1975), and development of this approach, using such models, linked to a robust optimisation algorithm may prove to be advantageous. However, as this is primarily concerned with simulation methodology it is outside of the scope of this research. A further development more associated with optimisation development using simplification of the component and system models may result in effectively reducing the size of the optimisation and hence yield the same advantageous reduction in solution time.

9.2.3. Elimination of Problem Variables and Constraint Functions.

One of the characteristics of the solution of HVAC system design optimisation problems is that it often lies on or near a constraint function. The successful implementation of optimisation algorithms developed throughout this research has been hampered by the difficulty of the algorithm to negotiate constraint functions towards the optimum solution. Elimination of constraint functions from the problem would therefore be advantageous. The reduction in the number of problem variables would also help the effectiveness of the optimisation algorithms, again this is well illustrated throughout this research. Fewer variables would make the optimisation algorithm easier, quicker and more robust in finding the optimum solution, (Wright and Lambert, 1991).

On going research (Wright, 1989), has investigated an approach to elimination of constraint functions and problem variables in HVAC system design optimisation. Most HVAC system constraints can be represented as either equality or inequality constraints. Where an inequality constraint is active at the solution, it can be considered as an equality constraint. The general form of an inequality constraint is:

$$C(X) > 0.0$$

If the solution lies on or near the constraint then the constraint value can be approximated into an equality form:

$$C(X) = 0.0$$

In this form problem reduction can take place, (Rao, 1979), by elimination of one problem variable and the constraint function itself. Taking the face velocity constraint from the cooling coil example problem as an example this problem reduction technique can be illustrated.

The upper limit of the face velocity constraint can be expressed in the following form:

$$2.5 - (\text{constant}/W \times H) > 0.0$$

relating to the maximum air face velocity of 2.5m/s. For optimisation of the coil capital cost the optimisation will reduce the dimensions W = coil width, and H = coil height until the constraint function upper limit is encountered, at which point the constraint function may be treated as an equality constraint:

$$2.5 - (\text{constant}/W \times H) = 0.0$$

rearranging this equality constraint yields:

$$W = (\text{constant}/2.5 \times H)$$

thus eliminating the constraint function and the problem variable W, coil width by expressing W in terms of the problem variable H, coil height.

One of the considerations with constraint function and problem variable elimination is the determination of the specific problem variables that are associated with the constraint functions. This is a relatively simple

operation for some constraint functions where the problem variables appear explicitly, the face velocity constraint for the cooling coil is one such example where the problem variables of coil width and coil height appear within the constraint. A means of evaluating the relationship between the problem variables and the constraint function in this instance has been established easily through evaluating the change in constraint function value with a given change in problem variable value, the gradient of each constraint against each variable determines this.

Development of constraint and variable elimination for HVAC system design optimisation problems appears to have merit in reducing the size of the optimisation problem and may also have benefit in reducing the optimisation to a series of sub-optimisations. It is considered worthy of further investigation and development. Particular consideration should be made to developing a similar approach to less manageable constraint functions in which the problem variables within the constraint bear no direct relationship to the value of the objective function. In this case constraint and variable elimination can take place but there would not be an in-built inference as to weather the upper or lower limit of the constraint should be eliminated and similarly the choice of variable elimination would not be readily determined.

The elimination technique would not be applicable to sparse configuration constraints, it would allow identification of the variables associated with the constraint but elimination due to the characteristics of the constraint would not be possible.

9.3. Random Solution Optimisation.

Throughout this research certain difficulties in the characteristics of the problem have been highlighted that have not been fully resolved. The existence of sparse constraint functions commonly associated with component configuration and problem variables that do not have a direct linkage to the objective function are two such difficulties. The direct search algorithms developed do not adequately deal with either of these characteristics, and although the sparse constraint is rare and usually bears no relationship to the system performance or capital cost it does have an influence in the final design of the system.

The use of a randomised selection of solution points occasionally occurring within a search algorithm would allow the algorithm to assess sparse constraints to a limited degree, but would not maintain control of the search path. Another approach which has been developed and implemented for optimisation of HVAC component sizing (Chu Kai Hung, 1991), employs the use of genetic engineering to address the optimisation. Development of genetic algorithms (GA's), have proved to be useful for optimisation of problems with complex characteristics such as those displayed in HVAC system design. They use a structured yet randomised exchange of information from selected feasible solution points to develop new stronger or 'fitter' solution points. Each exchange of information is called a generation. (Wright, 1992), has preliminarily developed such GA's applying them to a model HVAC system, it was found that quickly a near-optimal solution was developed after five generations of the algorithm, however, further generations yielded no further favourable solutions.

Certain advantages of the GA's approach were determined:

1) They are not concerned with the form of the problem.
2) Constrained optimisation problems can be tackled by simple implementation of a penalty function.
3) Random selection of the solution points ensure that false local optimums are not found by the algorithm and that sparse constraints are in some way catered for.
4) The speed of the search is not dependent on the number of problem variables within the optimisation as the GA treats all variables together as a single string of binary coding.

Further development of a optimisation algorithm should be considered which implements the use of GA's to quickly find the near optimum incorporated with a robust and more traditional optimisation method to develop the solution to the optimum.

9.4. Summary of Future Developments.

The most favourable next steps should be those which are most likely to successfully improve the performance of the optimisation. These are:

1) The development of problem variable and constraint elimination to reduce the size of the optimisation problem. In conjunction with this, better use of the constraint function should be made, as the solution is known to lie on or near a constraint. Ordering of problem variables

should be adopted to enhance the performance of the search.

2) The development of genetic algorithms to quickly find the near optimum, and to incorporate this with a robust direct search algorithm to find the optimum solution.

3) Simplification of the system simulation to improve the overall speed of the search, a full simulation being performed occasionally to maintain stability.

CHAPTER 10.

CONCLUSIONS AND FUTURE DEVELOPMENT.

The conclusion of this research is that it is possible to develop an optimisation algorithm to solve HVAC system design problems. The application of a single direct search method to solve specific problems is possible, but it may not be robust in solving all problems.

The characteristics of HVAC system design optimisation problems were established. The particular characteristics that proved to be of most significance to direct search optimisation algorithms, are that discrete problem variables are present, and that the solution lies on or near constraint functions. Two algorithms that address these characteristics have been evaluated, the complex algorithm and a pattern search with penalty function.

10.1. The Complex Algorithm Conclusions.

The complex algorithm relies on the rejection of infeasible solution points. Discrete problem variables caused the algorithm to lack robustness around constraint functions. Development and incorporation of a univariate search enhanced the complex algorithm robustness and allowed small scale HVAC system design problems to be solved. (Lambert and Wright, 1991).

The slow speed of the complex algorithm and the continued lack of robustness around constraint functions when the number of problem variables are increased, leads to the conclusion that the algorithm is unlikely to be successful with full scale HVAC system designs. The future development of this type of algorithm, which uses rejection of infeasible solutions, is not recommended because of the strong linkage between the number of problem variables and constraint functions, and the overall speed of the search.

10.2. The Penalty Function Algorithm Conclusions.

The penalty function algorithm approach is to attempt to prevent constraint violation. This improved the speed of the search since the system performance must be simulated for each constraint violation.

The Hooke and Jeeves pattern search used in parallel with the penalty function and was found to be effective in solving HVAC optimisation problems. This algorithm, however, was found to be sensitive to the value of the overall penalty multiplier r, and the extent to which the penalty multiplier was reduced for each search. It was concluded that there was no direct relationship between the successive values of r, and the stability of the pattern search, and that this was specific to the problem. The automation of the selection of the penalty multiplier r, was concluded to be impractical.

The implementation of active and inactive penalty weightings, proved to be a successful tool to assist the solving of HVAC optimisation problems.

The method was robust accurate and fast. The selection of the active penalty weighting was, however, reliant on knowledge of the design function of the constraint that it related too.

The selection of active penalty weightings is difficult for large scale HVAC system design optimisation problems with a large numbers of problem variables and constraint functions. The ordering of problem variables according to their influence on the objective function improves the performance of the pattern search within the penalty function algorithm. A similar method for considering which problem variables are effective in each constraint function allows selection of the active penalty weighting.

It is doubtful that the pattern search - penalty function method will solve problems that include variables which have no effect on the objective function. Such variables only influence certain constraint function values and therefore it is difficult to assess their impact on the direction of the search. A further complication is the handling of the highly sparse equality constraints. These are not common in HVAC systems but can arise from limitations on component configuration.

10.3. Future Development.

It is clear that as the number of problem variables and constraint functions increase, the optimisation of HVAC system design problems becomes more difficult. The direct search algorithms developed within this research have been successful for small scale problems, but as the problem size increases the robustness of the algorithms reduce.

The recommendation of this research is that development of a form of problem reduction be undertaken. The use of constraint and variable elimination to reduce the problem size has had initial success, (Wright, 1989). This approach has the ability to handle sparse constraints and incorporate variables unrelated to the objective function. It is known that one of the characteristics of the solution of HVAC system design optimisation is that it lies on or near a constraint, so development of the use, and handling of constraints would be advantageous.

The use of a genetic algorithm to find the near optimum solution is recommended. It is known that the problem definition is not effected by the use of a genetic algorithm and that the number of variables within the problem does not adversely effect the speed of the search.

The final recommendation for future development of this research is that a simplified system simulation be developed for use during the search, with a full simulation being undertaken occasionally to confirm that the search is stable.

The idealised form of the algorithm would incorporate the use of constraint and variable elimination to reduce the problem size, a genetic algorithm to quickly find the near optimum, using a simplified simulation, and the use of a robust direct search algorithm such as the pattern search - penalty function method to find the 'true' optimum.

REFERENCES.

Amstal, J.J. van, and Poirters, J.A.A.M., (1989), 'Design of Data Structures and Algorithms.' Prentice-Hall International, Hemel Hempstead, 1989.

APACHE User Manual, Oscar Faber Partnership, St. Albans, Hertfordshire, U.K.

Box, M.J. (1965) "A New Method of Constrained Optimization and a Comparison with Other Methods." The Computer Journal, 8, 42-52.

BS 5141, ' Specification for Air Heating and Cooling Coils', part 1, 1975, British Standards Institute.

Campey, I.G., and Nickols, D.G. (1961) "Simplex Minimization. Programme Specification", I.C.I. Ltd., August 1961.'

Carrol, C.W. (1961) "The Created Response Surface Technique for Optimising Non-linear Restrained Systems.", Operations Research, 9, 169-184.

Chu Kai Hung, E., (1991) ' Genetic Algorithms on Duct Sizing Optimisation.' Final Year Project, Dept. of Mechanical Engineering, University of Hong Kong.

Clark, D.R., Hurley, C.W., and Hill, C.R. (1985) 'Dynamic Models for HVAC System Components.' ASHRAE, Transactions, vol., 91, Part 1, pp. 737 - 750.

Davis, O.L. (Ed.) (1954) 'The Design and Analysis of Industrial Experiments.' Oliver and Boyd, Edinburgh.

Fiacco, A.V., and McCormick, G.P. (1964) 'The Sequential Unconstrained Minimization Technique for Non-linear Programming, a Primal-dual Method.' Management Science, 10, 360-366.

Gough, M.C.B., 'B:TAS: 'A Software Package for Component Based Systems Modelling.', 5th International Symposium on the Use of Computers for Environmental Engineering Related to buildings, July 1986, Bath, pp 264-273.

Hanby, V.I., (1988) ' Graph Directed Solution Procedure', Internal Research Document, Dept. of Civil Engineering, Loughborough University of Technology.

Hanby, V.I. and Wright, J.A.,(1989) ' HVAC Optimisation Studies : Component Modelling Methodology', Building Serv. Eng. Technol. 10(1) pp. 35-39

HEVACOMP. (1988) M & E Designers Package User Manual. HEVACOMP Sheffield, Yorkshire, U.K.

HEVASTAR.(1988) M & E Designers Package User Manual. HEVACOMP. Sheffield, Yorkshire, U.K.

Hill, C.R. 'Simulation Techniques for Building Systems.' Proceedings of the Workshop on HVAC Controls Modelling and Simulation, Georgia Institute of Technology, February 1984.

Hooke, R. and Jeeves, T.A., (1960), 'Direct Search Solution of Numerical and Statistical Problems.' Journal of the Association of Computer Machinery, Vol. 8, pp 212 - 229.

Lambert, G.C. and Wright, J.A., (1991). 'Direct Search Optimisation of HVAC Systems Design.' Building Environmental Performance '91, Canterbury, April 1991.

Leah, R.L., (1983) 'Quadratic Search Algorithms for Optimisation of Thermal Systems Models.' M.Sc. Thesis, University of Illinois.

Murray, M.A.P., Ph.D. Thesis 'Component Based Performance Simulation of HVAC Systems', University of Technology, Loughborough, December 1984.

Nelder, J.A., and Mead, R., (1965) A Simplex Method for Function Minimization. The Computer Journal, 7, 308-313.

Park, C., Bushby, S.T. and Kelly, G.E., (1989) 'Simulation of a Large Office Building System using the HVACSIM+ Program.' ASHRAE Transactions, Vol. 96, Part 1B, pp 642-651.

Powell, M.J.D., (1964) 'An Efficient Method of Finding the Minimum of a Function of Several Variables Without Calculating Derivatives.' The Computer Journal, 7, 155-162.

Rao, S.S., (1987), 'Optimisation Theory and Applications,'Wiley Eastern Limited.

Rosenbrock, H.H. (1960), 'An Automatic Method for Finding the Greatest or Least Value of a Function,' The Computer Journal, 3, 175-184.

Sedgewick, R. (1991) 'Algorithms', Addison-Wesley, Reading (Mass.), U.S.A.

Siddel, J.N., (1982), 'Optimal Engineering Design', Marcel Dekker Inc., New York.

Spendley, W. Hext, G.R., and Himsworth, F.R., (1962) 'Sequential Application of Simplex Designs in Optimisations and Evolutionary Operation', Technometrics, 4, 441-461.

Stoecker, W.F., (1975) 'Procedures for Simulating the Performance of Components and Systems for Energy Calculations.' ASHRAE, U.S.A.

Stoecker, W.F., (1989) 'Design of Thermal Systems', McGraw Hill.

Sugie, N., (1964) 'An Extension of Fibonaccian Searching to Multi-Dimensional Cases', I.E.E.E. Transactions on Automatic Control, AC-9, p105.

Wright, J.A., (1986) 'The Optimised Design of HVAC Systems,' Ph.D. Thesis, Loughborough University of Technology.

Wright, J.A. and Lambert, G.C., (1991) 'The Optimised Design of HVAC Systems.' Report to SERC. Grant: GR/D/91564. Dept. of Building Engineering and Architecture, The University of Liverpool.

Wright, J.A., (1989) (Written Communication) on Constraint and Variable Elimination. Dept. of Building Engineering and Architecture, The University of Liverpool.

Wright, J.A., (1992) (Written communication) on Genetic Algorithms. Dept. of Civil Engineering, Loughborough University of Technology.