COMPUTER AIDED DESIGN OF NONLINEAR CONTROL SYSTEMS VIA OPTIMIZATION

by

Masoud Sahba

A Thesis submited for the Degree
of Doctor of Philosophy

January 1984

Department of Electrical Engineering,
Imperial College of Science and Technology,
University of London.

ABSTRACT

In this thesis a design methodology is proposed for designing controllers for nonlinear systems. Since frequency response design techniques can be used only in special cases nonlinear systems are designed, in practice, by multiple simulation. The complexity of the design objectives can make design by simulation difficult and time consuming; thus, any reduction of simulation time will substantially reduce the the overall design time. The design methodology is based on expressing the stability and performance requirements as a collection of simple and (or) functional (i.e. infinite dimensional) constraints and then using optimization methods to find a controller which satisfies these constraints.

The most important constraint in control system design is stability, and to ensure stability it would appear necessary to compute the system trajectory for all initial states and all time. It is shown, in this thesis, how simulation time, required for ensuring stability, can be considerably reduced.

The algorithms which solve these problems replace the semi-infinite constraints by simple inequalities. There is, therefore, a need for efficient algorithms to solve these sub-problems; four algorithms have been developed for this purpose. A new feasible point algorithm which, under mild assumptions, finds a solution to a finite set of inequalities and converges quadratically, has been developed. This algorithm requires exact gradient calculations which are often difficult to make. Two new derivative free algorithms are,

therefore, proposed. Convergence of these algorithms has been established. The fourth algorithm presented, minimizes a cost function subject to equlity and inequality constraints. The algorithm is of the exact penalty type and includes a new method for computing the penalty parameter; in this algorithm, the penalty parameter is allowed to decrease. Global convergence is established. Examples show good performance.

Implementation of the design technique also requires an interactive environment to simulate nonlinear systems. SIMNON, a command driven simulation program for nonlinear system, has been implemented on a Perkin Elmer computer and enhanced to permit implementation of the design method (To improve the run-time capabilities of this package a compiler has been written to translate the code generated by SIMNON into machine code).

To demonstrate the usefulness of the optimization based design method several design studies were undertaken. A controller was designed to stablize a double inverted pendulum in an upright position. A different design methodology was used for the design of nonlinear controllers for torque controlled robot arms. A controller was designed to ensure robust performance and robust stability of the system despite variation in load mass. Finally, linear and nonlinear optimal controllers were designed for a seventh order nonlinear model of a single machine power system to improve the transient performance of the system and to satisfy stability and other soft constraints.

To my Family and Teachers

ACKNOWLEGEMENTS

I would like to express my most sincere thanks, appreciation and gratitude to my supervisor, Professor D.Q. Mayne, for his guidance, encouragement and patience during the course of the research that led to this thesis.

I would like to thank Mr. A.J. Heunis, most gratefully, for his invaluable help and advice.

The encouragement and help from my fellow research students and friends will be always remembered with gratitude.

The support of the U.K. Science and Engineering Research Council is gratefully acknowledged.

CONTENTS

CHAPTER 1

INTRODUCTION

1.1 - OPTIMIZATION BASED COMPUTER AIDED DESIGN

The introduction of computers and recent developments in computer science have resulted in changes, not only in society in general, but also in scientific methodology. Numerical solutions are not only acceptable but are now as important as analytical solutions. In other words, the class of methods which are practical has changed; this, in turn, has greatly influenced the development of theory advances. This can best be seen in solving engineering problems. Design, as opposed to synthesis, is iterative in nature because the often imprecise objective and constraints, and the possibility of trading off one desirable quality for another requires constant interaction with the designer. However, synthesis techniques which solve precisely specified problems are useful tools for solving sub-problems which may recur in the design process.

Many design problems can be formulated as constrained optimization problems, in which inequality constraints correspond to design specifications[1,2,3,4]. The solution of such precisely formulated problems is only one stage of the design process. The designer must be able to interact at each stage, changing the constraints, relaxing some and tightening others (e.g. to obtain a simpler controller he could relax perfor-

mance constraints). Since there is such a close relationship between design and constrained optimization, algorithms for solving constrained optimization problems, or more generally, for satisfying inequality constraints play an important role in computer aided design.

Many of the design specifications can be transcribed into standard inequality constraints, so that feasible point type algorithms may be employed. However, many other design specifications including control design requirements, involve infinite dimensional constraints of the form $\varphi(z, \omega) \leq 0$ for all $\omega \in \Omega$, where ω denotes frequency, time or parameter vectors. In design of controllers, for example, the parameters "z" of a controller are to be chosen such that the resultant closed loop system satisfies certain constraints, including stability and hard constraints on control and states. A typical constraint in this group of problems is: $y(z, t) \leq \alpha$ for $t \in [t1, t2]$, where $y(z, .)$ is the response of the closed loop system to a step input. In another class of problems, the parameter values of the actual system, structure or device differ from the nominal values used in the design. This difference may occur because of lack of precise knowledge of some of parameters in the system (e.g. unknown mass in the design of controller for robot arms). A satisfactory design may require satisfaction of certain constraints, not only by nominal design but also, by all possible realizations of certain parameters. A typical example is design of robust controllers, where the controller must be such that the design constraints are satisfied for all values of certain plant

parameters lying in a specified set. Taking into account conventional (finite dimensional) constraints, many design problems may, therefore, be expressed either as

a) determine a $z \in F$; or

b) minimize $\{f(z) \mid z \in F\}$,

where $F \triangleq \{z \in R^p \mid g(z) \leq 0,\ \varphi_\Omega(z) \leq 0\}$ (1.1.1)

and where $g : R^p \to R$, and $\varphi_\Omega : R^p \to R$ is defined by

$$\varphi_\Omega(z) \triangleq \max_{\omega \in \Omega} \max_{j \in \underline{m}} \psi^j(z, \omega) \tag{1.1.2}$$

where $\underline{m}$ denotes the set $\{1,2,\ldots,m\}$. These problems are obviously very complex and require global solution of a maximization problem. However, semi-infinite constraints can be replaced by an infinite sequence of inequalities. Let us ignore the conventional constraints and restrict the number of functional constraints to one so that F is defined by

$$F \triangleq \{z \in R^p \mid \varphi_\Omega(z) \leq 0\} \tag{1.1.3}$$

where, now $\varphi_\Omega : R^p \to R$ is defined by

$$\varphi_\Omega(z) \triangleq \max_{\omega \in \Omega} \psi(z, \omega) \tag{1.1.4}$$

Let Ω_0 denote any subset of Ω (e.g. $\Omega_0 = \{\omega_1, \omega_2, \ldots, \omega_s\}$); then the set F_{Ω_0} defined by

$$F_{\Omega_0} \triangleq \{z \mid \psi(z,\omega) \leq 0,\ \omega \in \Omega_0\} \qquad (1.1.5)$$

is clearly a superset of F. Hence, F_{Ω_0} is called an outer approximation to F. The outer approximation algorithms employ a sequence of outer approximations $F_{\Omega_1}, F_{\Omega_2}, \ldots,$ to F, each described by a finite number of inequalities. The following (conceptual) algorithm can be used to determine a feasible point [3].

Algorithm 1: to find a $z \in F$:

Data: $z_0 \in R^p$; Ω_0 a finite subset of Ω (e.g. $\Omega_0 = \{\omega_0\}$)

Step 0: Set i=0

Step 1: Compute any $z_i \in F_{\Omega_i}$.

Step 2: Compute ω_i to solve $\max\{\psi(z_i,\omega) \mid \omega \in \Omega\}$.

Step 3: Set $\Omega_{i+1} = \Omega_i \cup \{\omega_i\}$, set i=i+1 and go to Step1.

■

It can be shown [5,6] that any accumulation point z^* of an infinite sequence $\{z_i\}$ generated by the above algorithm is feasible ($z^* \in F$). The algorithm can be easily modified to deal with the constrained optimization problem $\min\{f(z) \mid z \in F\}$ as follows.

Algorithm 2: to solve $\min\{f(z) \mid z \in F\}$

Data: $z_0 \in R^p$; Ω_0 a finite subset of Ω.

Step 0: Set i=0

Step 1: Compute a z_i to solve

$$P_i: \min\{f(z) \mid z \in F_{\Omega_i}\}.$$

Step 2: Compute ω_i to solve $\max\{\psi(z_i,\omega) \mid \omega\in\Omega\}$.

Step 3: Set $\Omega_{i+1} = \Omega_i \cup \{\omega_i\}$, set i=i+1 and go to Step1.

■

These algorithms have several defects: exact solution of the problems in Steps 1 and 2 are required and the cardinality of Ω_i tends to infinity with i, making the problem in Step 1 progressively more difficult to solve. These defects have been removed in the implementable outer approximation algorithms [7,8] by modifying the alove algorithms so that problem P_i (Step 1 Algorithm 2) and the problem $\max\{\psi(z_i,\omega) \mid \omega\in\Omega\}$ (Step 2 Algorithms 1 and 2) need only be solved approximately. An outer approximation technique is proposed by Gonzaga and Polak [8] which offers considerable advantage in constraint dropping over earlier scheme such as those due to Eaves and Zangwill [5]. Becker, Heunis and Mayne [3] have proposed an implementable version of Algorithms 1 and 2 by making use of an additional finite approximation $\hat{\Omega}_i$ to Ω for Step 2. Thus, if Ω is frequency (or time) interval $[\omega_1, \omega_2]$, then $\hat{\Omega}_i \triangleq \{\omega_1, \omega_1+\Delta, \omega_1+2\Delta, \ldots, \omega_2\}$, where $\Delta=(\omega_2-\omega_1)/i$ is a suitable approximation.

Algorithm 3: to find a $z\in F$:

Data: $z_0\in R^p$, Ω_0, $\delta\in(0,1)$

Step 0: Set i=0

Step 1: Compute any $z_i\in F_{\Omega_i}$.

Step 2: Compute ω_i to solve $\max\{\psi(z_i,\omega) \mid \omega\in\hat{\Omega}_i\}$.

Store ω_i .

Step 3: Set $\Omega_{i+1} = \{\omega_i \,|\, \psi(z_i,\omega_j) \geq \delta^j-\delta^i, j=1,2,\ldots,i\}$,
set i=i+1 and go to Step 1. ▪

Notice that the test $\psi(z_i,\omega_j)\geq\delta^j-\delta^i$ (for inclusion of ω_i in Ω_{i+1}) is difficult for low values of j and becomes more difficult as $i\to\infty$ since $\delta^i\to 0$. If $\psi(.,.)$ is continuously differentiable, then, any accumulation point z^* of an infinite sequence $\{z_i\}$ generated by the algorithm is feasible [8]. Implementable version of Algorithm 2 is presented in [7]. Let $\theta_{\Omega_i}(.):R^p\to R$ be an optimality condition for P_i i.e. $\theta_{\Omega_i}(z) \leq 0$ for all z and $\theta_{\Omega_i}(z)=0$ if, and only if, $z\in F_{\Omega_i}(z)$ and satisfies certain necessary conditions of optimality. Then, one example of an implementable version of Algorithm 2 follows [3].

Algorithm 4: to solve $\min\{f(z) \,|\, z\in F\}$

Data: $z_0\in R^p$, Ω_0, $\delta\in(0,1)$, $\gamma\in(0,1)$

Step 0: Set i=0

Step 1: Compute a z_i such that
$\theta_{\Omega_1}(z_i) \geq -\gamma^i$

Step 2: Compute ω_i to solve $\max\{\psi(z_i,\omega) \,|\, \omega\in\hat{\Omega}_i\}$.
Store ω_i .

Step 3: set $\Omega_{i+1} = \{\omega_i \,|\, \psi(z_i,\omega_j) \geq \delta^j-\delta^i, j=1,2,\ldots,i\}$,
set i=i+1 and go to Step 1.

Sub-algorithms are required either for finding a feasible point or solving P_i. These are essentially conventional (finite dimensional) mathematical programming problems for which

many algorithms exist [9,10,11,12], however, because of the complexity of the total problem, large rewards are obtained through the use of efficient programs. It is the purpose of this thesis to obtain efficient algorithms for the above sub-problems and to investigate their use in design.

1.2 - STABILITY AND DESIGN OF NONLINEAR SYSTEMS

The modern stability theory of automatic control, no matter how presented, is based on the simple strong formulation due to A.M. Liapunov, of stability theory [13]. As a consequence of the many distinct behaviour of nonlinear systems, most methods of analysis are directed towards solving special problems such as the existence of the limit cycles. In several methods, the assumptions made are based on the expected form of solution, so that some knowledge of the possible forms of nonlinear behaviour is a requirement for the analyst[15]. Although frequency domain techniques are widely used for designing (stabilizing) linear systems, the extension of these methods to nonlinear systems has only been utilized in a certain class of nonlinear systems, namely those which have a forward path which is a linear time invariant dynamics system and a feedback element which is nonlinear or nonstationary or both [18]. The resulting stability criteria (e.g. Popov's criterion and the circle criterion) are frequency domain constraints. However, many design problems do not fall into this restricted class and existing tools do not appear adequate. On the other hand, control

design specifications often relate to system time response behaviour and there are few analytical techniques for nonlinear systems which can be used with confidence to guarantee these design requirements (for a survey of the existing methods see [15,16]). The limitations of the various analytical methods for nonlinear systems mean that simulation is frequently used in practice as a design tool but the complexity of the design objectives (e.g. the satisfaction of the step response of constraints at every time in a specified interval), the large number of control parameters, can make design by simulation difficult and time consuming. However, it is possible to utilize optimization methods to aid this methodology. In this thesis it is shown how optimization can be used in a new methodology for the design of feedback controllers for nonlinear systems to satisfy various stability and performance constraints.

1.3 - OUTLINE OF THE THESIS AND LIST OF CONTRIBUTIONS

In Chapter 2 a procedure for designing of feedback controllers for nonlinear systems is proposed, analysed and incorporated in SIMNON, an interactive simulation package which is implemented on a Perkin Elmer 8/32 computer. Illustrative examples are given.

In Chapter 3 a new feasible point algorithm is presented which, under mild assumptions, finds a solution in a finite number of iterations. A complete theoretical analysis is

given. Quadratic rate of convergence (in the absence of a stopping condition) is established. Numerical examples are computed and a comparison is made with existing results.

In Chapter 4 two new derivative free feasible point algorithms are presented. Convergence properties are established. Numerical results are given. The algorithms are incorporated in SIMNON.

In Chapter 5 a new globally convergent optimization algorithm for constrained problems is presented. In this algorithm a new method for computing the penalty parameter is proposed. The penalty parameter computed in this way can be decreased. It is proposed to solve a linear program which ensures existence of a search direction vector. A complete theoretical analysis, under mild assumptions, is given. Numerical examples are computed and comparison is made with the existing results.

In the next few chapters the new design method and the new algorithms are applied to practical examples.

In Chapter 6 a double inverted pendulum is stablized at the upright position using the design technique proposed in Chapter 1. Minimum order and functional observers are designed and the resultant system is simulated under severe disturbances.

In Chapter 7 a design methodology is proposed for design of

nonlinear controllers for torque controlled robot arms. Robust performance and robust stability of such a system for unknown mass is established by numerical simulation of the resultant system.

In Chapter 8 linear and nonlinear optimal controllers are designed for a seventh order nonlinear model of a single machine power system to improve the transient performance of the system and to satisfy stability and other soft constraints. Simulation results are given.

In Chapter 9 our conclusions are presented.

Since SIMNON is heavily machine dependent, it was extensively modified to implement our design procedure. Several new commands are included. A compiler is written to translate the Reverse Polish Notation of SIMNON into machine language; this is presented in Appendix I. This improved the efficiency of the package by a factor of five.

1.4 - REFERENCES

[1] - Zakian V., Al-Naib U., "Design of dynamical and control systems by method of inequalities", Proc. IEE, Vol 120, pp 1421-1427, 1973.

[2] - Mayne D.Q., Polak E., and Sangiovanni-Vincentelli A., "Computer aided design via optimization", Automatica, Vol 18, No.2, pp 147-154, 1982.

[3] - Becker R.G., Heunis A.J., Mayne D.Q., "Computer aided design of control systems via optimization", Proc. IEE, Vol 126, No. 6, pp. 573-578, 1979.

[4] - Zakian V.,"New formulation for the method of inequalities", Proc. IEE, Vol. 126, pp. 579-584, 1979.

[5] - Eaves B.C., Zangwill W.I.,"Generalized cutting plane algorithms", SIAM J. Control & Optimization, Vol 9, pp. 529-542, 1971.

[6] - Blankenship J.W., Falk J.E., "Infinitely constrained optimization problems", The George Washington Univ., Inst. for Management Science and Eng., Serial T-301, 1974.

[7] - Mayne D.Q., Polak E., Trahan R.,"An outer approximation algorithm for computer aided design problems", JOTA Vol 28, pp. 331, 1979.

[8] - Gonzaga C., Polak E.,"On constrained dropping schemes and optimality functions for a class of outer approximation algorithms", SIAM J. Control and Optimiz., Vol 17 pp. 477, 1979.

[9] - Mayne D.Q., Polak E. and Heunis A.J.,"Solving nonlinear inequalities in a finite number of iterations", JOTA,

Vol. 33, pp. 207-222, 1981.

[10] - Garcia-Palomares U.M., Restuccia A.,"A global quadratic algorithm for solving a system of mixed equalities and inequalities", Math. Prog., Vol 21, pp. 290-300, 1981.

[11] - Mayne D.Q., Polak E.,"A superlinearly convergent algorithm for constrained optimization problems", Math. Prog., Vol 16, pp. 45-61, 1982.

[12] - Powell M.J.D.,"A fast algorithm for nonlinearly constrained optimization calculations", in: G.A. Watson, ed., Numerical Analysis, Lecture Notes in Mathematics, Vol 630 (Springer-Verlag, Berlin) pp. 144-157, 1978.

[13] - Rouche N., Habets P., Laloy M.,"Stability theory by Liapunov's direct method", Springer-Verlag, 1977.

[14] - Cook P.A.,"System stability" in Modern Approaches to Control System Design, Peter Peregrinus Ltd, London, 1979.

[15] - Atherton D.P.,"Analysis and design of nonlinear feedback systems", IEE Proc., Vol. 128, Pt. D, No. 5, pp.173-180, 1981.

[16] - Pyatnitsky E.S.,"New research on the absolute stability of automatic control systems", J. of Autom. and Remote Control Vol. 6, pp. 855-881, 1968.

[17] - Mayne D.Q.,"Optimization based design of nonlinear systems", Internal Report Electrical Engineering Department, Imperial College, UK., 1983.

[18] - Aizerman M.A., Gantmacher F.R.,"Absolute stability of regulator systems", Translated by E. Polak, Holden-Day Inc., 1964.

CHAPTER 2

2. DESIGN OF FEEDBACK CONTROLLERS FOR NONLINEAR SYSTEMS

2.1 - INTRODUCTION

The substantial progress in the design of linear multivariable systems makes it opportune to re-examine the difficult problem of designing controllers for nonlinear multivariable systems. Of course, it has always been recognized that linear models merely approximate the nonlinear processes mainly encountered in practice. Hence, several design procedures have been proposed; one of the most widely used of these is the describing function method. In common with most of the available methods, this method assumes that the feedback loop consists of two subsystems, one linear, the other nonlinear, connected in tandem. The method approximates the nonlinearity by an amplitude dependent transfer function, and assesses stability from the Nyquist plots of the transfer function of the linear system and the amplitude parameterized family of approximate transfer functions of the nonlinear system. It is known (see e.g. [1]) that satisfaction of the describing function criterion does not, in general, ensure stability, even when the nonlinearity is memoryless and time invariant. For the latter case, rigorous stability criteria (the circle criterion and Popov's criterion) have been developed and are amenable to simple graphical interpretations. All these methods, however, are restricted to systems consisting of linear and nonlinear (possibly memoryless) subsystems. For

more general systems, it appears that the Liapunov's stability criterion is the only resort. However, it is notoriously difficult and virtually impossible in practice to obtain Liapunov functions for systems of order higher than three. Since obtaining such a function represents only part of the design process, this approach is not generally applicable. Efficient methods for determining optimal open loop controls do exist and are employed in aero-space and some process control applications; these methods cannot be used for determining satisfactory feedback controllers.

In the face of such poor assistance from system theory it appears that many engineers employ simulation as their major design tool. Controllers, whose structure is motivated by past experience, are added to a simulation of the nonlinear system and parameters are adjusted until closed-loop simulations indicate (but do not guarantee) stability and adequate performance. For reasons given below, it is believed that any generally applicable method for the design of nonlinear systems will, of necessity, involve repeated simulations. Attention should, therefore, be devoted to the problem of reducing the substantial computation involved. To assess stability it appears necessary, at first sight, to evaluate the state trajectory at all times (in the finite interval $[0,\infty)$) and for every initial state (in some compact subset X of the state space R^n). It appears impossible to relax the latter requirement; however, recent works by Polak and Mayne [2] and Mayne and Sahba [3] show that it is possible, in the assessment of stability, to reduce the interval of simulation from the in-

finite interval $[0,\infty)$ to a finite interval $[0,T]$, where T may be quite small, resulting in a considerable reduction in computation. The requirement that all initial states in X must be considered may be handled by formulating the stability requirement as an inifinite dimensional constraint. Algorithms for such constraints are given in Chapter 1. It will be shown how these procedures [2,6] can be used, with the aid of numeriacl examples, for designing of nonlinear systems. It will be also shown how the new algorithms may be employed to obtain satisfactory performance (e.g. low tracking error).

2.2 - STABILITY OF NONLINEAR SYSTEMS

Suppose that the system to be considered is described by

$$\dot{x}(t) = \tilde{f}(x(t), u(t)) \qquad (2.2.1)$$

where $\tilde{f} : R^n \times R^m \rightarrow R^n$ is continously differentiable. We also suppose that the set of initial states of interest is some compact subset X of R^n, and that the origin is the equlibrium state ($\tilde{f}(0,0) = 0$) and lies in the interior of X. Suppose that the controller structure has been chosen so that

$$u(t) = h(x(t), z) \qquad (2.2.2)$$

where $h : R^n \times R^r \rightarrow R^m$ defines the controller and it is assumed to be continuously differentiable and to satisfy $h(0,z)=0$ for all $z \in R^r$; $z \in R^r$ specifies the controller parameters to be chosen. Hence the closed loop system satisfies

$$\dot{x}(t) = f(x(t), z) \qquad (2.2.3)$$

where $f : R^n \times R^r \to R^n$ is defined by $f(x, z) \triangleq \tilde{f}(x, h(x,z))$ and is continuously differentiable and satisfies $f(0,z)=0$ for all $z \in R^r$.

For any initial state x_0 at $t=0$ let $x(t; x_0, z)$ denote the solution of (2.2.3) at time t. The formal definitions of stability [1] involve the solution $x(t; x_0, z)$ of (2.2.3) at all $t \in [0, \infty)$ and all $x_0 \in X$ (the origin is globally asymptotically stable if $x(t; x_0, z) \to 0$ as $t \to \infty$ for all x_0 and if for all $\varepsilon > 0$ there exists a $\delta > 0$ such that $\|x_0\| < \delta$ implies that $\|x(t; x_0, z)\| < \varepsilon$ for all $t \geq 0$). A frequency domain characterization is not possible since the system is nonlinear so that this escape from an infinite number of time responses is not available. If a candidate Liapunov function $(x,z) \to W(x,z)$ were available, to test it would still require the evaluation of $\dot{W}(x) \triangleq W_x(x)f(x,z)$ at all $x \in X$; while infinite dimensiona lity in the time domain is now absent, it must be remembered that determination of W involves even more computation.

Let $V : R^n \to R$ be continuous function with the following properties

a) $V(x) > 0$ for all $x \in R^n$

b) $V(\alpha x) = \alpha V(x)$ for all $\alpha > 0$, all $x \in R^n$

c) $V(x) = 0$ if and only if $x = 0$.

An example of such a function is $V(x) = (x^T P x)^{1/2}$ where P is a positive definite matrix. For all x let B(x), a subset of R^n, be defined as follows

$$B(x) \triangleq \{x' \in R^n \mid V(x') < V(x)\} \qquad (2.2.4)$$

It is easily established that $B(\alpha x) = \alpha B(x)$ for all $\alpha \in [0, \infty)$, where $\alpha B(x)$ is defined in the usual way[2]

$$\alpha B(x) \triangleq \{\alpha x' \in R^n \mid x' \in B(x)\} \qquad (2.2.5)$$

We immediately obtain

$$V(x') < V(x) \text{ implies that } B(x') \subset B(x) \qquad (2.2.6)$$

clearly $B(\gamma x)$ is a subset of B(x) for all $\gamma \in (0,1)$ and $B(0)=\{0\}$.

If a controller could be chosen so that the resultant trajectories satisfy $\dot{W}(x) < 0$ for all $x \neq 0$ where $W(x) \triangleq V(x)^2$, then the system would be globally asymptotically stable with a Liapunov fuction W(x). Since this approach is not fruitful, we relax the demand on V (or W). Instead of requiring that V decreases monotonically along trajectories, we merely require that a decrease in V is achieved at times T, 2T, 3T, ... etc, for some finite T; V is allowed to increase in the intervals [kT, (k+1)T), k = 1,2,... provided that certain boundedness conditions are met. The following stability theorem [2] make the comment more precise.

THEOREM 2.2.1

Suppose $\bar{x} \in R^n$ is such that X is a subset of $\bar{B} \triangleq B(\bar{x})$. If there exists $\beta \in (0,1)$, $\gamma \in (1,\infty)$, $z \in R^r$ and $T \in (0,\infty)$ such that

i) $x(T, x_0, z) \in \beta B(x_0)$ for all $x_0 \in \bar{B}$

ii) $x(t, x_0, z) \in \gamma B(x_0)$ for all $x_0 \in \bar{B}$ and all $t \in [0,T]$.

Then

a) $x(t, x_0, z) \in \gamma B(x_0) \subset \gamma \bar{B}$ for all $x_0 \in \bar{B}$, and all $t \in [0,\infty)$,

b) $x(t, x_0, z) \to 0$ as $t \to \infty$, for all $x_0 \in \bar{B}$.

A proof is given in [2]; we present here a graphical motivation (see Fig. 2.1). The set of initial states of interest is X; X is a subset of $\bar{B}$ which is chosen to be (the smallest) set of form $B(\bar{x})$ (for some $\bar{x} \in R^n$ such that X is a subset of $\bar{B}$. Theorem 2.2.1 states that if z and T can be chosen so that a trajectory starting at any x_0 in $\bar{B}$ lies within $\gamma B(x_0)$ for all $t \in [0,T]$ and terminates (at T) in $\beta B(x_0)$, then the origin is asymptotically stable with a region of attraction $\bar{B}$, which includes X. It is obvious that in the interval [T,2T] this trajectory will be within $\beta\gamma B(x_0)$ and terminate (at 2T) in $\beta^2 B(x_0)$, etc. Hence, asymptotic stability for all initial states in $\bar{B}$ is ensured if hypotheses (i) and (ii) are satisfied; these conditions can be organized as infinite dimensional constraints (to be satisfied for all $x_0 \in \bar{B}$, all $t \in [0,T]$).

2.3 - RESPONSE FUNCTION V

The procedure obtained above depends on the choice of function V and associated set value function B; a wise choice will permit a small value for T. The following choice should be suitable for many applications:

(a) Choose a control structure (possiblly dynamic) which permits linear control (e.g. by setting certain components of z to zero); if dynamic, incorporate the extra states in x.

(b) Linearise the system (eqn. 2.2.1) about the equilibrium point yielding

$$\dot{w}(t) = Aw(t) + Bu(t) \qquad (2.3.1)$$

where $A \triangleq f_x(0,0)$ and $B \triangleq f_u(0,0)$ if the equilibrium point is the origin. The linearized output equation is

$$y(t) = cw(t) \qquad (2.3.2)$$

where $c \triangleq g_x(0,0)$.

(c) Design a linear controller

$$u = -Kcw(t) + \omega(t) \qquad (2.3.3)$$

so that the closed loop system

$$\dot{w}(t) = (A - BKc)w(t) + B\omega(t) \qquad (2.3.4)$$

is stable.

(d) For some symmetric matrix $Q>0$, compute the sysmmetric positive definite solution P of the Liapunov equation for eqn. (2.3.4) (with the same linear controller) in the neighbourhood of the origin. In this neighbourhood, provided that β and γ are appropriately chosen, $T=0_+$ will suffice.

It is, therefore, plausible that this choice of V will permit a relatively small value of T to be chosen for the nonlinear design problem. Since $V(x) = \|x\|_P$, it follows that

$$\|x(kT, x_0, z)\|_P \leq \beta^k \|x_0\|_P \qquad (2.3.5)$$

so that appropriate choices are $\beta = e^{\delta T}$, $\gamma = 1.5\beta$, in which case $(1/\delta)$ is effectively the "time constant" of the control system and should be appropriately chosen.

A disadvantage of the above procedure is that the controlled nonlinear system is forced to behave like a linear system, at least at multiples of T. This may require excessive control action when x is large. One way of avoiding this is to permit a variable "time constant" $(1/\delta)$ by allowing β or T (or both) to vary with x. To save computation, it may be preferable to replace T by T(x), where T(x) is small if $\|x\|$ is small and large if $\|x\|$ is large. The time constant is now $(1/\delta(t)) = T(x)/\ln\beta$ and is only small when $\|x\|$ is small. The stability theorem is easily extended.

THEOREM 2.3.1

Suppose $\bar{x}\in R^n$ is such that X is a subset of $\bar{B} \triangleq B(\bar{x})$. If there exist $\beta\in(0,1)$, $\gamma\in(1,\infty)$, $T:\bar{B} \to [T_1, T_2]$ and $z\in R^r$ such that

i) $x(T(x_0),x_0,z)\in\beta B(x_0)$ for all $x_0\in\bar{B}$

ii) $x(t,x_0,z)\in\gamma B(x_0)$ for all $x_0\in\bar{B}$, $t\in[0,T(x)]$

then

a) $x(t,x_0,z)\in\gamma B(x_0) \subset \gamma\bar{B}$ for all $x_0\in\bar{B}$ and all $t\in[0,\infty)$

b) $x(t,x_0,z) \to 0$ as $t \to \infty$ for all $x_0\in\bar{B}$.

PROOF: If $x'\in\beta^k B(\bar{x})=B(\beta^k\bar{x})$, from (i)

$$x(T(x'),x',z)\in\beta B(\beta^k\bar{x}) = \beta^{k+1}B(\bar{x}).$$

Since f is autonomous it follows that

$$x(t_k,x_0,z)\in\beta^k B(\bar{x}) \text{ for all } x_0\in B(\bar{x})$$

where

$$t_k = \sum_{i=1}^{k-1} T(x_i).$$

It follows from (ii) that

$x(t,x',z) \in \gamma B(x') \subset \gamma\beta^k B(\bar{x}) \subset \gamma B(\bar{x})$ for all $t \in [t_k, t_{k+1}]$

then as $t \to \infty$, $k \to \infty$ and

(a) $x(t,x_0,z) \in \gamma B(x_0) \subset \gamma\bar{B}$ for all $x_0 \in \bar{B}$, all $t \in [0,\infty)$

(b) $x(t,x_0,z) \to 0$ as $t \to \infty$ for all $x_0 \in \bar{B}$. ∎

Since values of $T(x)$ are restricted to lie in the range $[T_1, T_2]$ the effective "time constant" lies in the range $[T_1/\ln\beta, T_2/\ln\beta]$.

It is possible under certain circumstances, to discard the second infinite dimensional constraint (hypothesis (ii)) in each of the above theorems. Suppose that $(x,u) \to \tilde{f}(x,u)$ and $(x,p) \to h(x,p)$ are continuously differentiable, so that $h(0,z) = 0$, for all $z \in Z$ where Z, the set of feasible solution of z, is compact. From the mean value theorem we have that

$$f(x,z) = f(0,z) + (\partial f(\zeta,z)/\partial x)x \quad \zeta \in \bar{B} \tag{2.3.6}$$

or

$$\|f(x,z)\|_p \leq \|f(0,z)\|_p + \|\partial f(\zeta,z)/\partial x\|_p \|x\|_p \tag{2.3.7}$$

Since $\partial f(x,z)/\partial x$ is continuous and $\bar{B}$ is compact

$\|\partial f(x,z)/\partial x\|_p \leq M$ for all $x \in \bar{B}$ and all $z \in Z$,

where M is finite. Hence,

$$\|f(x,z)\|_P \leq M\|x\|_P \text{ for all } x\in\bar{B}, \text{ all } z\in Z \tag{2.3.8}$$

But

$$\|x(t,x_0,z)\|_P \leq \|x_0\|_P + \int_0^t \|f(x(s),z)\|_P ds$$

$$\leq \|x_0\|_P + \int_0^t M\|x(s)\|_P ds \tag{2.3.9}$$

and from the Bellman-Gronwall Lemma

$$\|x(t,x_0,z)\|_P \leq \|x_0\|_P \, e^{Mt} \tag{2.3.10}$$

for all $x_0\in\bar{B}$ and all $z\in Z$. Hence, if T (T_2) is chosen to be sufficiently small (such that $e^{MT} < \gamma$) it follows that

$$x(t,x_0,z)\in\gamma B(x_0) \tag{2.3.11}$$

for all $x_0\in\bar{B}$ and all $t\in[0,T]$ ($[0, T(x)]$).

Hence, for such a T (T_2) hypothesis (ii), in Theorems 2.2.1 and 2.3.1 is automatically satisfied. Under such conditions asymptotic stability is obtained with $z\in Z$ satisfying the inequality

$$\|x(t,x_0,z)\|_P - \beta\|x_0\|_P \leq 0 \text{ for all } x_0\in\bar{B} \tag{2.3.12}$$

2.4 - PERFORMANCE

Satisfaction of the stability constraint automatically ensures satisfaction of one performance criterion; the closed loop system must have a specified "time constant", in the sense that $x(kT,x_0,z) \in \beta^k B(x_0)$ if a constant T is employed, and $x(T_k(x_0),x_0,z) \in \beta^k B(x_0)$ otherwise, where $T_k(x_0) \in [kT_1,kT_2]$. Hence, by choice of T and β, a satisfactory rate of recovery from an initial state is ensured. Other performance constraints (e.g. tracking error less than specified limit, zero steady-state error to step, ramp and parabolic inputs) can also be satisfied (these constraints are, usually, infinite dimensional constraints). For example, in the sigle-input, single-output case, the output y(t) due to test input r(t)=aH(t) (and zero initial state) may be required to satisfy

$$y(t) < y_h(t,a) \qquad (2.4.1)$$

and

$$y(t) > y_l(t,a) \qquad (2.4.2)$$

for all $t \in [0,T]$ and all a in the set A of amplitudes of test signals. This performance criterion is illustrated in Fig. 2.2. Note that, typically, a longer response time is permitted for a larger change in demand output. Tighter tracking under transient conditions can be achieved as follows. Suppose that the reference input r belongs to the class R de-

fined by

$$R \triangleq \{r \mid r(t) = a_0 + a_1 t + \ldots + a_{k-1} t^{k-1}, a \in A\} \qquad (2.4.3)$$

where $a \triangleq (a_0, \ldots, a_{k-1})^T$ and A is a compact subset of R^k. An element r of R has the finite parametric representation

$$r(t) = \langle a, \hat{p}(t) \rangle \qquad (2.4.4)$$

where $\hat{p}(t) \triangleq (1, t, t^2, \ldots, t^{k-1})$.

The closed loop system with a controller specified by z and an input $r \in R$ can be described by

$$\dot{x}(t) = \bar{f}(x(t), z, a) \qquad (2.4.5)$$

$$y(t) = g(x(t)) \qquad (2.4.6)$$

The instantaneous tracking error is

$$e(t, x_0, z, a) \triangleq [g(x(t, x_0, z, a))]^2 - [\langle a, \hat{p}(t) \rangle]^2 \qquad (2.4.7)$$

and a scalar valued tracking criterion is

$$c(z) = \max\{e(t, x_0, z, a) \mid x_0 \in X, a \in A, t \in T\} \qquad (2.4.8)$$

where T = [0, t'] is an interval of interest. Typically there are also constraints on the magnitude of u(t) and in some cases on the magnitude of $\dot{u}(t)$. Thus constraints of the form

$$|u(t)| \leq U \qquad (2.4.9)$$

must also be satisfied for all $t\in[0,T]$ and all $a\in[0,A]$ where $u(t)$ is the control signal in response to a step input $r(t) = aH(t)$.

Summarising, the performance constraints may be expressed in the form

$$\varphi_j(z,t,a) \leq 0, \quad j = 1,\ldots,J \qquad (2.4.10)$$

for all $t\in[0,T]$ and all $a\in A$. The parameter z emphasises the fact that the responses $y(t)$ and $u(t)$ depend on the controller parameters. For example, the constraint $|u(t)| \leq U$ may be expressed as

$$\varphi_1(z,t,a) = u(t) - U \leq 0 \text{ and}$$
$$\varphi_2(z,t,a) = -u(t) - U \leq 0.$$

Another design objective is that of robust performance, i.e. that the performance constraints must be satisfied even if the plant differs (with limits) from the model employed for design [7]. To quantify this suppose that the closed loop system is described by

$$\dot{x}(t) = f(x(t), r(t), z, p) \qquad (2.4.11)$$

where p is a vector $(p_1,\ldots,p_q)$ of plant parameters which may be known or may, indeed, vary. Suppose p_0 specifies the nomi-

nal plant and that p always lies in a known set Ω. Then the response y(t) and control u(t) will depend on p as well as z and a (i.e. r). In order to ensure robustness performance the following constraints must be satisfied

$$\varphi_j(z,t,a,p) \leq 0, \quad j = 1,\ldots,J \qquad (2.4.12)$$

for all $t\in[0,T]$, all $a\in A$ and all $p\in\Omega$. The constraint in (2.4.12) represents a constraint on the output y(t) or the control u(t) in response to test input (specified by a) when the plant parameter is p and the controller parameter is z. Satisfaction of (2.4.12) ensures that the performance constraints are satisfied for all plants such that p lies in Ω.

2.5 - CONTROLLER STRUCTURE

There has, unfortunately, been relatively little interaction between the areas of system theory and controller design [7]. However, recent results on characterising admissible controllers in linear multivariable systems have suggested useful controller structure; recent results on transforming nonlinear to equivalent linear systems [8] may permit similar techniques to be used.

In many cases, controller structure will be chosen either on the basis of prior experience or in some standard form (e.g multivariable proportional plus integral controller, or a controller with transfer function $[1/d(s)]N(s)$ or $D(s)^{-1}N(s)$ where N(s) is a matrix polynomial, D(s) a diagonal matrix of

polynomials and d(s) a polynomial). The parameters of any such controller may be specified by the vector z and the dynamic equations of the plant and the controller combined to yield a closed loop system described by $\dot{x}(t) = f(x,z)$ (or $\dot{x}(t) = f(x,z,r)$) and $y = g(x)$.

2.6 - ALGORITHMS FOR SOLVING SEMI-INFINITE CONSTRAINTS

The design constraints can be expressed as a finite number of conventional ($\varphi(z) \leq 0$) and semi-infinite ($\varphi(z,a) \leq 0$ for all $a \in A$) constraints. Hence, the design problem can be expressed as

P1: Determine a z such that $\varphi(z,a) \leq 0$ for all $a \in A$, or

P2: Minimize $\{c(z) \mid \varphi(z,a) \leq 0,\ a \in A\}$

where A is an infinite dimensional set (e.g. $A = \bar{B}$ for the stability constraint in hypothesis (i) of Theorems 2.2.1 and 2.3.1, $a = \bar{B} \times [0,T]$ for the stability constraints of hypothesis (ii)). Algorithms for these problems are described in Chapter 1.

2.7 - ILLUSTRATIVE EXAMPLES

EXAMPLE 1. Harmonic oscillator with two control variables.

Let us examine the stability of a simple dynamic system with two control variables

$$\dot{x}_1 = -\alpha x_1 + \omega x_2 + r_1$$
$$\dot{x}_2 = -\omega x_1 - \alpha x_2 + r_2$$

Let the feedback signal be given by

$$\begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = \begin{bmatrix} z_1 & 0 \\ 0 & z_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

where (z_1, z_2) are the control parameters. The closed loop system is then given by

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = [F] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

where

$$F \triangleq \begin{bmatrix} -(\alpha+z_1) & \omega \\ -\omega & -(\alpha+z_2) \end{bmatrix}.$$

The Liapunov function is defined by

$$V(x) \triangleq (x^T P x)^{1/2}$$

where P is a symmetric positive definite matrix defined by

$$Q = -(PF + F^T P)$$

Let $Q = I$ and $\alpha = 0$, $\omega = 1$, $z_1 = 1$ and $z_2 = 2$, then, P, the solution of the above equation, is given by

$$P = \begin{bmatrix} 8 & -1 \\ -1 & 5 \end{bmatrix}.$$

Now, let us choose $\bar{x} = (0, 1)$; this implies that $V(\bar{x}) = 2.36$, and

$$B(\bar{x}) = \{x \mid (x^T P x)^{1/2} < 2.36\}.$$

We have employed a feasible point algorithm to find the design parameters (z_1, z_2) satisfying the stability constraints; this means that the stability constraints must be satisfied for all initial points in the set $B(\bar{x})$. We have employed the Monte Carlo method for randomly choosing 500 initial states in $B(\bar{x})$. Figure 2.3. shows sample trajectories, computed over the interval [0,5sec], for iterations 1 to 3.

EXAMPLE 2. Two coupled alternators

The open loop system equations are described by

$$\dot{x}_1 = x_1 + 0.5 - \sin(x_3 + \pi/6) + u_1$$
$$\dot{x}_2 = 2x_2 - 0.5 + \sin(x_3 + \pi/6) + u_2$$
$$\dot{x}_3 = x_1 - x_2.$$

The controller is described by

$$u_1 = -z_1 x_1$$
$$u_2 = -z_2 x_2$$

The system equations and their adjoints (required for the computation of the gradient $\varphi_z(z,a)$) are integrated using an

efficient program due to Sargent and Sullivan [9]. The algorithm yielded a stable controller (assessed by 500 initial states in X, randomly chosen) within three iterations of the master algorithm. The state trajectories and V for x_0 = (1.6,-1,-1) are plotted in Figure 2.4. Note that V initially increases before decreasing, unlike conventional Liapunov function.

2.8 - REFERENCES

[1] - Cook P.A.,"System stability" in Modern Approaches to Control System Design, Peter Peregrinus Ltd, London, 1979.

[2] - Polak E., Mayne D.Q.,"Design of nonlinear feedback controllers", Memorandum No. UCB/ERLM80/12. Electronics Research Lab., University of California, Berkeley.

[3] - Mayne D.Q., Sahba M.,"Design of feedback controllers for nonliear systems", International Conference on Control and its Applications, University of Warwick, U.K., pp. 276-280, 1981.

[4] - Zakian V., Al-Naib U., "Design of dynamical and control systems by method of inequalities", Proc. IEE, Vol 120, pp 1421-1427, 1973.

[5] - Mayne D.Q., Polak E.,Trahan R.,"An outer approximation algorithm for computer aided design problems", JOTA Vol 28, pp. 331-352, 1979.

[6] - Gonzaga C., Polak E.,"On constrained dropping schemes and optimality functions for a class of outer approximation algorithms", SIAM J. Control and Optimiz., Vol 17 pp. 477, 1979.

[7] - Mayne D.Q.,"Optimization based design of nonlinear systems", Internal Report, Electrical Engineering Department, Imperial College, U.K., 1983.

[8] - Su R.," On the linear equivalent of nonlinear systems", System and Control Letters, Vol 2, pp. 48-52, 1982.

[9] - Sargent R.W.H., Sullivan G.R.,"Optimization techniques", Proc. Eighth IFIP Conference, Wurzburg, 1977.

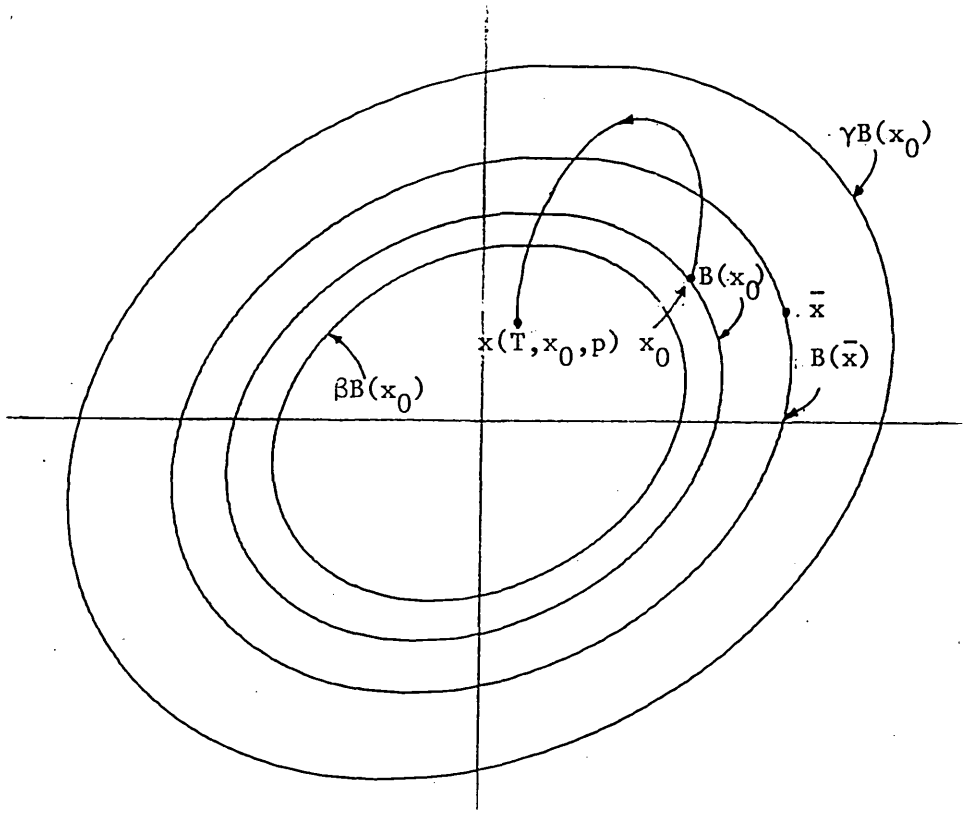

Figure 2.1 Stability Theorem

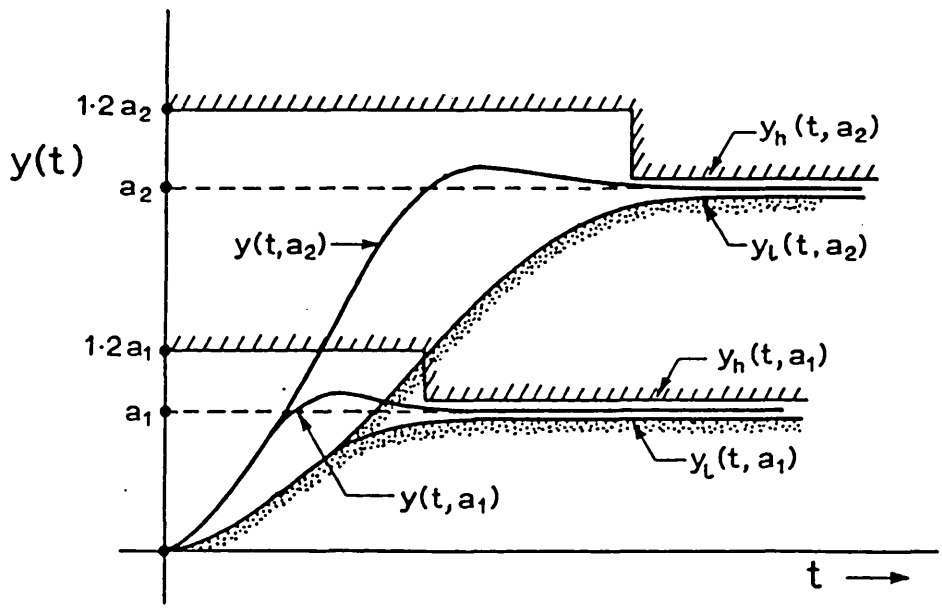

Fig. 2.2 Performance criterion

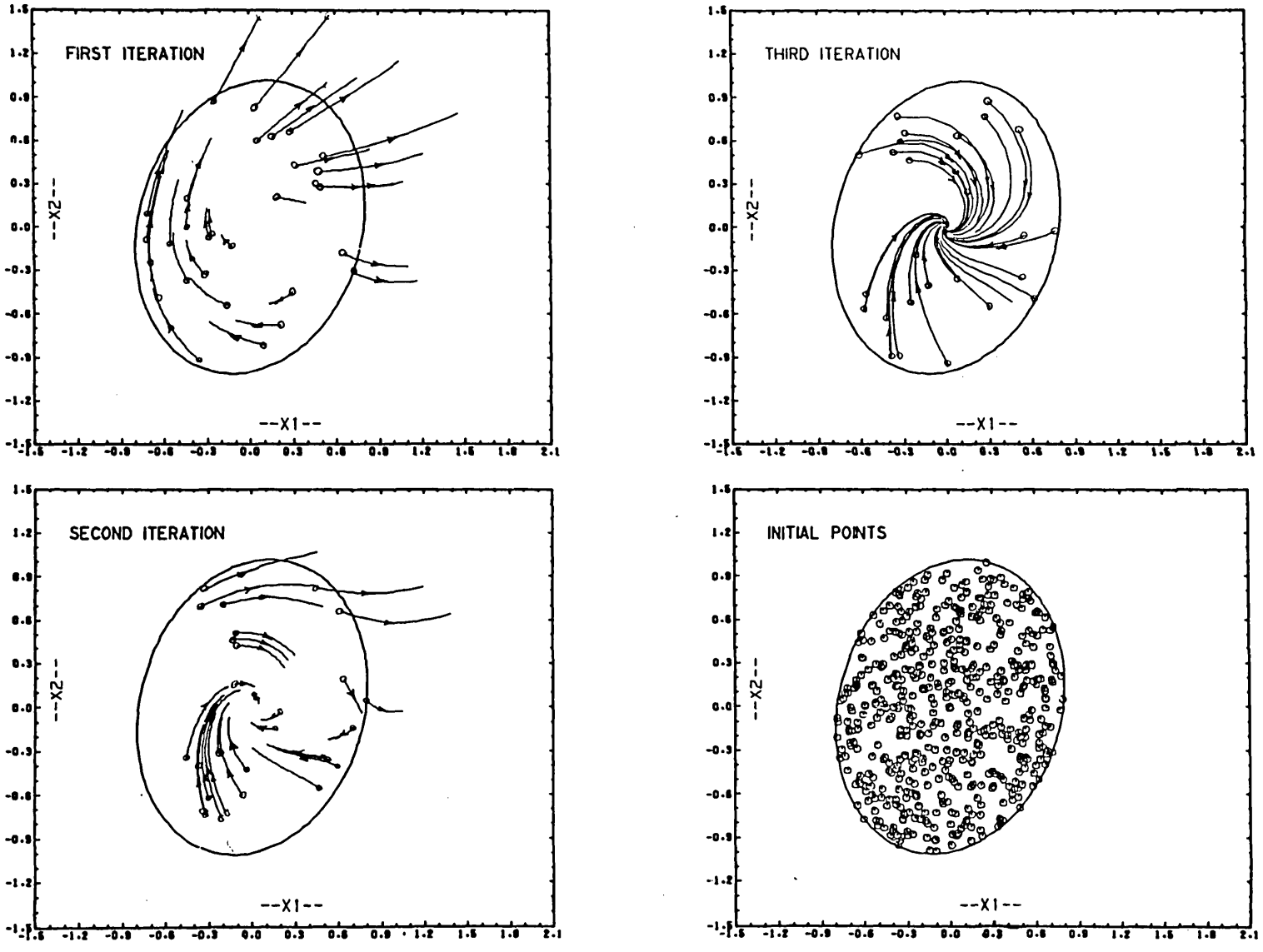

Fig. 2.3 Second-order System

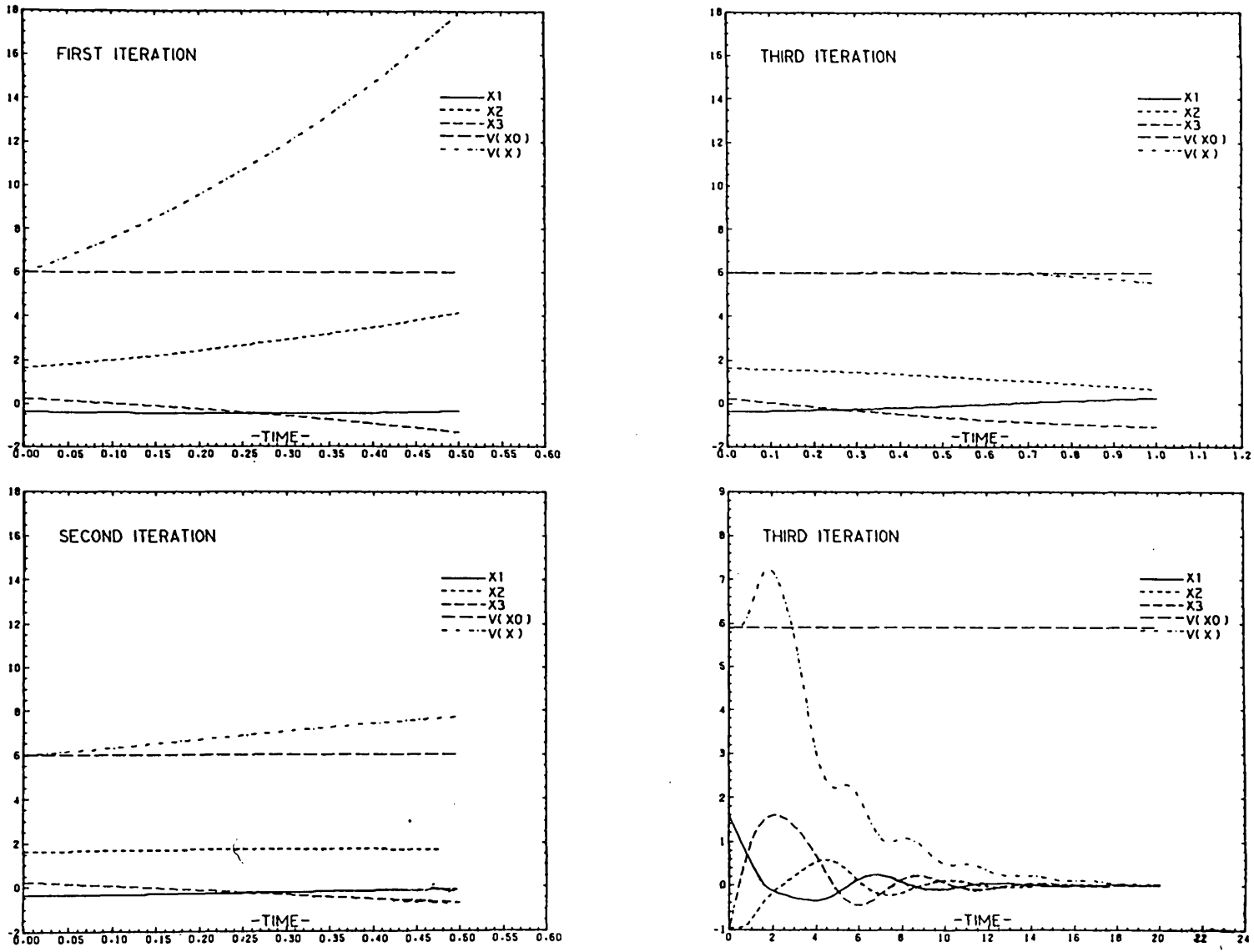

Fig 2.4 Third-order system

CHAPTER 3

AN EFFICIENT ALGORITHM FOR SOLVING INEQUALITIES

3.1 - INTRODUCTION

The problem considered is that of finding a point z in R^n satisfying the finite set of inequalities

$$g^j(z) \leq 0, \quad j = 1, \ldots, m. \tag{3.1.1}$$

Several algorithms have been presented for this problem. It is well known, for example, that a standard feasible direction algorithm may be employed; however, convergence can be slow [1]. In Reference 2 a modified Newton step is employed as the search direction, provided that it exists and satisfies certain properties; the modification consists of adding to the Newton step a perturbation directed to the interior of (the first order approximation to) the feasible set. The modification ensures finite convergence. If the Newton step does not exist or does not satisfy certain requirements, a first order descent direction for $\psi(z)$, where $\psi:R^n \to R$ is defined by

$$\psi(z) \triangleq \max\{g^j(z) \mid j = 1,\ldots,m\}, \tag{3.1.2}$$

is employed. In Reference 3 an alternative approach is employed. Again a modified Newton step is employed. In this case the modified step is that p in R^n of minimum norm which

satisfies

$$g^j(z) + g^j_z(z)p \leq -\varepsilon, \quad j = 1,\ldots,m. \tag{3.1.3}$$

Again, if such a p does not exist or does not satisfy certain properties, a descent direction (for a surrogate cost function) is employed. A scheme for adaptively reducing ε consistently with finite convergence completes the algorithm.

Although these algorithms, which have been extensively used in many design studies, generally work well, it has been observed that in some cases the reversion to a first order search direction, when the modified Newton step does not exist or does not satisfy certain conditions, can cause slow convergence. An alternative Newton type proposed in Reference 4 employs an active set strategy. Quadratic rate of convergence is established under stronger assumptions than those employed in our work. This algorithm does not necessarily find a solution in a finite number of iterations. The algorithm presented in this chapter attempts to avoid these deficiencies by always employing a Newton step directed to the interior of the linearized feasible set as the search direction. The algorithm employs as its search direction that vector p, of minimum norm, which solves

$$g^j(z) + g^j_z(z)p \leq -\varepsilon, \quad j = 1, \ldots, m, \tag{3.1.4}$$

where ε is chosen, subject to certain constraints, to be as large as possible. The value of ε is ascertain by solving a

linear program. Thus, whenever possible, the search direction p is computed such that z + p lies in the interior of the first order approximation to the feasible set. This ensures finite convergence.

3.2 - THE ALGORITHM

For all z and p in R^n let $\hat{\psi}(z,p)$ denote the following first order approximation to $\psi(z+p)$

$$\hat{\psi}(z,p) \triangleq \max\{g^j(z) + g_z^j(z)p \mid j\in\underline{m}\} \qquad (3.2.1)$$

where $\underline{m}$ denotes the set $\{1,2,\ldots,m\}$. The Newton step at z, if it exists, is that p in R^n which solves:

$$\min\{\ \|p\| \mid \hat{\psi}(z,p) < 0\}. \qquad (3.2.2)$$

Since the set $\{p \mid \hat{\psi}(z,p) < 0\}$ may be empty (implying nonexistence of the Newton step) another approach is required. Let the functions $\hat{\psi}^0$ and $\hat{\psi}^0_\varepsilon : R^n \to R$ be defined, for all $\varepsilon>0$, by

$$\hat{\psi}^0(z) \triangleq \min\{\ \hat{\psi}(z,p) \mid p\in P\} \qquad (3.2.3)$$

and

$$\hat{\psi}^0_\varepsilon(z) \triangleq \max\{\hat{\psi}^0(z),\ -\varepsilon\}, \qquad (3.2.4)$$

where

$$P \triangleq \{ p \in R^n \mid \|p\|_\infty \leq L \} \tag{3.2.5}$$

and L is some suitably chosen large number (without the constraint $p \in P$ the solution of (3.2.3) may be unbounded). If $\hat{\psi}^0(z) < 0$ then a solution to (3.2.2) exists, since the set $\{p \mid \hat{\psi}(z,p) < 0\}$ is not empty. Note that (3.2.3) is equivalent to a linear program which we denote LP(z).

We can now define our search direction at z. It is that p_ε which solves the following quadratic program ($QP_\varepsilon(z)$)

$$\min\{ \|p\| \mid \hat{\psi}(z,p) \leq \hat{\psi}^0_\varepsilon(z) \}. \tag{3.2.6}$$

Since the level sets of $p \to \|p\|$ are strictly convex and $\{p \mid \hat{\psi}(z,p) \leq \hat{\psi}^0_\varepsilon(z)\}$ is convex $p_\varepsilon(z)$, the p which solves $QP_\varepsilon(z)$, is unique.

To complete the algorithm we have to specify the step length $\lambda_\varepsilon(z)$. Let $\theta: R^n \times R^n \to R$ be defined by

$$\theta(z,p) \triangleq \hat{\psi}(z,p) - \psi(z). \tag{3.2.7}$$

Clearly $\theta(z,p)$ is a first order approximation of $\psi(z+p) - \psi(z)$. Similarly let θ' and $\theta_\varepsilon: R^n \to R$ be defined by

$$\theta'(z) \triangleq \hat{\psi}^0(z) - \psi(z), \tag{3.2.8}$$

and

$$\theta_\varepsilon(z) \triangleq \hat{\psi}^0_\varepsilon(z) - \psi(z) = \max\{\theta'(z),\ -(\psi(z)+\varepsilon)\}. \qquad (3.2.9)$$

Clearly

$$\theta_\varepsilon(z) = \theta(z,\ p_\varepsilon(z)) \qquad (3.2.10)$$

is an estimate of $\psi(z + p_\varepsilon(z)) - \psi(z)$, i.e. an estimate of the change of cost obtained by employing the search direction $p_\varepsilon(z)$. The step length $\lambda_\varepsilon(z)$ is chosen to be the largest step, in a finite set, such that the actual reduction in ψ is at least half of the estimated reduction. More precisely, $\lambda_\varepsilon(z)$ is the largest number in the set $S \triangleq \{1,\ \beta,\ \beta^2, \ldots\}$, $\beta \in (0,1)$, satisfying

$$\psi(z + \lambda p_\varepsilon(z)) - \psi(z) \leq \lambda\theta_\varepsilon(z)/2$$

$$= \lambda[\hat{\psi}(z, p_\varepsilon(z)) - \psi(z)]/2 \qquad (3.2.11)$$

We now have all the ingredients for defining the first version of the algorithm.

ALGORITHM 1

Data: $z_0 \in R^n$, $\varepsilon' \in (0,\ 1)$, $L >> 1$, $\beta \in (0,\ 1)$.

Step 0: Set $i = 0$; set $\varepsilon = \varepsilon'\psi(z_0)$,

Step 1: Compute $\hat{\psi}^0(z_i) = \min\{\hat{\psi}(z_i,\ p) \mid p \in P\}$.

Step 2: Compute $p_i = p_\varepsilon(z_i) = \text{argmin}\{\|p\| \,\big|\, \hat{\psi}(z_i,p) < \hat{\psi}^0_\varepsilon(z_i)\}$.

Step 3: Compute $\lambda_i = \lambda_\varepsilon(z_i)$, the largest λ in S such that

$$\psi(z_i + \lambda p_i) - \psi(z_i) < \lambda\theta_\varepsilon(z_i)/2.$$

Step 4: Set $z_{i+1} = z_i + \lambda_i p_i$, set $i = i + 1$.
Go to step 1. ∎

3.3 - CONVERGENCE

Let the set F be defined by

$$F \triangleq \{z \,\big|\, g^j(z) < 0,\ j\in\underline{m}\} = \{z \,\big|\, \psi(z) < 0\} \qquad (3.3.1)$$

and let F^C denote the complement of F, i.e.

$$F^C = \{z \,\big|\, \psi(z) \geq 0\}. \qquad (3.3.2)$$

We make the following assumptions:

H1: The functions $g^j : R^n \to R$, $j\in\underline{m}$, are continuously differentiable.

H2: For all z in F^C,

$$\theta'(z) < 0. \qquad (3.3.3)$$

Assumption H2 ensures that $\psi(z)$ can be decreased at all z in F^C (since $\theta'(z) = \min\{\hat{\psi}(z,p) \,\big|\, p\in P\} - \psi(z)$) is a first order

estimate of $\psi(z+p(z)) - \psi(z)$. A sufficient condition for H2 is the positive linear independence, for all z in F_ϵ, of the set $\{\nabla g^j(z) \,|\, j\in I(z)\}$, $I(z) \triangleq \{j\in\underline{m} \,|\, g^j(z) = \psi(z)\}$ (i.e. the set of gradients of the most active constraints).

Our first task is to establish that $\hat{\psi}(z,p)$ is indeed a first order estimate of $\psi(z+p)$. This is easily done. For all z, all $\delta>0$ let $B_\delta(z)$ denote $\{z' \,|\, \|z'-z\| < \delta\}$.

PROPOSITION 3.3.1

For all $\eta>0$, all z in R^n there exists a $\delta>0$ such that

$$|\psi(z'+p) - \hat{\psi}(z',p)| < \eta\|p\|$$

for all $z'\in B_\delta(z)$, all $p\in B_\delta(0)$.

PROOF: As shown in Reference 2,

$$|\psi(z'+p) - \hat{\psi}(z'+p)| < \max\{|g^j(z'+p) - \hat{g}^j(z',p)| \,|\, j\in\underline{m}\}$$

where

$$\hat{g}^j(z,p) \triangleq g^j(z) + g^j_z(z)p.$$

Now

$$g^j(z'+p) - \hat{g}^j(z',p) = \left[\int_0^1 [g^j_z(z'+tp) - g^j_z(z')]dt\right]p,$$

so that

$$|g^j(z'+p) - \hat{g}^j(z',p)| \leq \left[\int_0^1 \|g_z^j(z'+tp) - g_z^j(z')\|dt\right]\|p\|.$$

Since g_z^j is uniformly continuous in any compact set and since $z' \in B_\delta(z)$ and $p \in B_\delta(0)$ imply that $z'+tp \in B_{2\delta}(z)$ for all $t \in [0,1]$, it follows that, for all $\eta > 0$, δ can be chosen so that

$$|\psi(z'+p) - \hat{\psi}(z',p)| \leq \eta\|p\|$$

for all $z' \in B_\delta(z)$ and all p in $B_\delta(0)$. ∎

Our next task is to establish that the algorithm map $z \to A_\varepsilon(z) \triangleq z + \lambda_\varepsilon(z)p_\varepsilon(z)$ (defined by Steps 1 - 4 of the algorithm) has certain continuity properties. Our first result is:

PROPOSITION 3.3.2

$\hat{\psi}^o : R^n \to R$ is continuous.

PROOF: Let z be any arbitrary point in R^n and let $\delta > 0$. Let z' be an arbitrary point in $B(z)$ and let $p_\varepsilon(z)$ ($p_\varepsilon(z') \in P$) satisfy

$$\hat{\psi}^o(z) = \hat{\psi}(z,\ p_\varepsilon(z))$$

$$\hat{\psi}^{o}(z') = \hat{\psi}(z', p_{\varepsilon}(z')).$$

Hence

$$\hat{\psi}^{o}(z') - \hat{\psi}^{o}(z) = \hat{\psi}(z', p_{\varepsilon}(z')) - \hat{\psi}(z, p_{\varepsilon}(z))$$

$$< \hat{\psi}(z', p_{\varepsilon}(z')) - \hat{\psi}(z, p_{\varepsilon}(z'))$$

and

$$\hat{\psi}^{o}(z) - \hat{\psi}^{o}(z') = \hat{\psi}(z, p_{\varepsilon}(z)) - \hat{\psi}(z', p_{\varepsilon}(z'))$$

$$< \hat{\psi}(z, p_{\varepsilon}(z')) - \hat{\psi}(z', p_{\varepsilon}(z')).$$

since $\hat{\psi}$ is uniformly continuous in $B_{\delta}(z) \times P$ it follows that

$$\left| \hat{\psi}^{o}(z') - \hat{\psi}^{o}(z) \right| \to 0 \quad \text{as} \quad z' \to z. \qquad \blacksquare$$

COROLLARY

For all $\varepsilon > 0$, $\hat{\psi}^{o}_{\varepsilon} : R^{n} \to R$ is continuous. $\blacksquare$

PROPOSITION 3.3.3

(i) θ' and θ_{ε}, for all $\varepsilon > 0$, are continuous.
(ii) For all $z \in F^{c}$ and all $\varepsilon > 0$, $\theta_{\varepsilon}(z) < 0$.

PROOF: (i) The continuity of θ' and θ_{ε} follows from the continuity of $\hat{\psi}^{o}$, $\hat{\psi}^{o}_{\varepsilon}$ and ψ.

(ii) By H2, $\theta'(z) < 0$ for all z in F^C. From (3.2.9), $\theta_\varepsilon(z) = \max\{\theta'(z), -(\psi(z) + \varepsilon)\}$. Since $\psi(z) > 0$, then, $\theta_\varepsilon(z) < 0$. ∎

PROPOSITION 3.3.4

For all $z \in F^C$ there exist a $\delta > 0$ and a (positive) $\lambda_1 \in S$ such that

$$\psi(z' + \lambda_1 p_\varepsilon(z')) - \psi(z') \leq \lambda_1 \theta_\varepsilon(z')/2$$

for all $z' \in B_\delta(z)$ and all $\varepsilon \geq 0$.

PROOF: By Proposition 3.3.3 there exists a $\delta_1 > 0$ such that

$$\theta_\varepsilon(z') \in [(3/2)\theta_\varepsilon(z), (1/2)\theta_\varepsilon(z)]$$

where

$$\theta_\varepsilon(z) < 0$$

for all $z' \in B_{\delta_1}(z)$. From Proposition 3.3.1 there exists a $\delta \in (0, \delta_1]$ such that

$$\left|\psi(z'+\lambda p) - \hat{\psi}(z', \lambda p)\right| \leq (-\theta_\varepsilon(z)/4)\lambda$$

for all $z' \in B_\delta(z)$, all $p \in P$, all $\lambda \in [0, \delta/L]$ (so that $\lambda p \in B_\delta(0)$ for all $p \in P$). Hence, for all $z' \in B_\delta(z)$, all $\lambda \in [0, \delta/L]$:

$$\psi(z'+\lambda p_\varepsilon(z')) - \psi(z') \leq \hat{\psi}(z',\lambda p_\varepsilon(z')) - \psi(z')$$

$$+ \left|\psi(z'+\lambda p_\varepsilon(z')) - \hat{\psi}(z',\lambda p_\varepsilon(z'))\right|$$

Since $\lambda \to \hat{\psi}(z',\lambda p_\varepsilon(z')) - \psi(z')$ is convex,

$$\psi(z'+ \lambda p_\varepsilon(z')) - \psi(z') \leq [\hat{\psi}(z',\lambda p_\varepsilon(z')) - \psi(z')] - \lambda\theta_\varepsilon(z)/4$$

$$\leq \lambda[\hat{\psi}(z',p_\varepsilon(z')) - \psi(z')] - \lambda\theta_\varepsilon(z)/4$$

$$= \lambda[\theta_\varepsilon(z') - \theta_\varepsilon(z)/4]$$

for all $z' \in B_\delta(z)$, all $\lambda \in [0,\delta/L]$. Since $\theta_\varepsilon(z') \leq \theta_\varepsilon(z)/2$ implies that $-\theta_\varepsilon(z) \leq -2\theta_\varepsilon(z')$ it follows that

$$\psi(z' + \lambda p_\varepsilon(z')) - \psi(z') \leq \lambda\theta_\varepsilon(z')/2$$

for all $z' \in B_\delta(z)$, all $\lambda \in [0,\delta/L]$ and all $\varepsilon \geq 0$. The desired result follows with λ_1 the largest number in S which is not greater than δ/L. ∎

COROLLARY

For all $z \in F^C$ there exist a $\delta > 0$ and a $\lambda_1 > 0$ such that the step length $\lambda_\varepsilon(z')$ generated by the algorithm satisfies

$$\lambda_\varepsilon(z') \geq \lambda_1$$

for all $z' \in B_\delta(z)$ and all $\varepsilon > 0$.

We can now easily establish that the algorithm generates convergent subsequences.

THEOREM 3.3.1

Any accumulation point z^* of an infinite sequence $\{z_i\}$ generated by Algorithm 1 satisfies $\theta_\varepsilon(z^*) = 0$ and, therefore, lies in F.

PROOF: Let z be any point such that $\theta_\varepsilon(z) < 0$ (and, hence, not lying in F). Then, by Proposition 3.3.4, and its Corollary, there exist a $\delta > 0$, and a $\lambda_1 > 0$ such that

$$\psi(A_\varepsilon(z')) - \psi(z') = \psi(z' + \lambda_\varepsilon(z')p_\varepsilon(z')) - \psi(z')$$

$$\leq \lambda_\varepsilon(z')\theta_\varepsilon(z')/2$$

$$\leq \lambda_1\theta_\varepsilon(z')/2$$

for all $z' \in B_\delta(z)$. Since θ_ε is continuous, δ can be chosen so that

$$\theta_\varepsilon(z') \leq \theta_\varepsilon(z)/2$$

for all $z' \in B_\delta(z)$. Hence the algorithm map $z \to A_\varepsilon(z) \triangleq z + \lambda_\varepsilon(z)p_\varepsilon(z)$ has the uniform continuity property

$$\psi(A_\varepsilon(z')) - \psi(z') < \lambda_1\theta_\varepsilon(z)/4$$

for all $z' \in B_\delta(z)$. Hence, by Theorem 1.3.3. in Reference 1, any accumulation point z^* of an inifinite sequence $\{z_i\}$ generated by the algorithm must satisfy $\theta_\varepsilon(z) = 0$ and, hence, lie in F. ∎

Since every accumulation point z^* of an infinite sequence $\{z_i\}$ generated by the algorithm satisfies $\psi(z^*) < 0$, it follows that there exists a finite i such that $\psi(z_i) < 0$, i.e. the inequalities are solved in a finite number of iterations. (The practical version of the algorithm employs a stopping condition: if $\psi(z_i) < 0$, stop).

3.4 - RATE OF CONVERGENCE

Theorem 3.3.1 establishes finite convergence. We now examine rate of convergence assuming that the algorithm does not incorporate a stopping condition. Although rate of convergence does not appear relevant to an algorithm stopping in a finite number of iterations, a superlinear rate of convergence requires an asymptotic step length of unity which contributes to the efficiency of the algorithm.

Suppose the algorithm generates an infinite sequence $\{z_i\}$ converging to z^* where $\hat{\psi}^0(z^*) < -\varepsilon$. It follows that $\hat{\psi}^0(z_i) < -\varepsilon$ so that $\hat{\psi}^0_\varepsilon(z_i) = -\varepsilon$ for all i sufficiently large. For all such i, therefore, the search direction $p_\varepsilon(z_i)$ solves

$$\min\{\|p\| \,\big|\, \hat{\psi}(z_i,p) < -\varepsilon\}$$

which is equivalent to

$$\min\{\|p\| \,\big|\, g^j(z_i) + g^j_z(z_i)p < -\varepsilon,\ j\in\underline{m}\}$$

and can, therefore, be recognized as the Newton step for the problem of determining a $z\in R^n$ such that

$$g^j_\varepsilon(z) \triangleq g^j(z) + \varepsilon < 0, \quad j\in\underline{m}.$$

Hence, under standard assumptions, quadratic convergence (to a point z^* satisfying $\psi(z^*) < -\varepsilon$) is easily established. However, since ε cannot be chosen a priori sufficiently small to ensure that any accumulation point z^* satisfies $\psi(z^*) < -\varepsilon$, it is necessary to modify the algorithm slightly as follows.

ALGORITHM 2

Data: $z_0\in R^n$, $\varepsilon'\in(0, 1)$, $L >> 1$, $\beta\in(0, 1)$.

Step 0: Set $i = 0$; set $\varepsilon = \varepsilon'\psi(z_0)$,

Step 1: Compute $\hat{\psi}^0(z_i) = \min\{\hat{\psi}(z_i, p) \,\big|\, p\in P\}$.

Step 2: Compute

$$p_i = p_\varepsilon(z_i) = \operatorname{argmin}\{\|p\| \,\big|\, \hat{\psi}(z_i,p) < \hat{\psi}^0_{\varepsilon_i}(z_i)\}.$$

Step 3: Compute $\lambda_i = \lambda_{\varepsilon_i}(z_i)$, the largest λ in S such that

$$\psi(z_i + \lambda p_i) - \psi(z_i) < \lambda\theta(z_i, p_i)/2.$$

Step 4: Set $z_{i+1} = z_i + \lambda_i p_i$. If $\hat{\psi}^o(z_i) \geq -\varepsilon_i$, set ε_{i+1} equal to the largest ε in the set $\{\varepsilon_i, \varepsilon_i/2, \ldots\}$ such that $\hat{\psi}^o(z_i) < -\varepsilon$; else set $\varepsilon_{i+1} = \varepsilon_i$. Set $i = i + 1$. and go to Step 1. ∎

We note that the only change is the introduction in Step 4 of a mechanism to reduce ε_i. In order to recover the convergence results established for Algorithm 1, we need to establish that ε_i is reduced only finitely often, so that it eventually becomes constant. For this we need an additional assumption

H3: The set $\{z \mid \psi(z) < \psi(z_0)\}$ is compact. (Alternatively, any infinite sequence $\{z_i\}$ generated by the algorithm is compact).

PROPOSITION 3.4.1

Given H1 - H3, ε_i is reduced in Step 4 only finitely often so that $\varepsilon_i = \varepsilon^* > 0$ for all i sufficiently large.

PROOF: It is easily established, as in the proof of Theorem 3.3.1, that Algorithm 2 is well defined so that ψ is reduced at each iteration. Hence, $\psi(z_i) < \psi(z_0)$ for all $i \geq 0$ so that

$\{z_i\}$, generated by the algorithm, is compact and therefore possesses accumulation points. Let z^* be an accumulation point of $\{z_i\}$. Suppose contrary to what is to be proven, that ε_i is reduced infinitely often in Step 2 when $i \in K$. Hence, $\hat{\psi}^o(z_i) \geq -\varepsilon_i$ for all $i \in K$. Since $\hat{\psi}^o(z_i) \overset{K}{\to} \hat{\psi}^o(z^*)$ and $\varepsilon_i \to 0$ as $i \to \infty$, it follows that $\hat{\psi}^o(z^*) \geq 0$ and, hence, $\psi(z^*) \geq 0$. It follows from H2 that $\theta'(z^*) = \hat{\psi}^o(z^*) - \psi(z^*) < 0$ so that $\psi(z^*) > 0$.

Steps 1 to 3 (with $z_i = z$, $\varepsilon_i = \varepsilon$) define an algorithm map $z \to A_\varepsilon(z) = z + \lambda_\varepsilon(z) p_\varepsilon(z)$ (so that $z_{i+1} = A_{\varepsilon_i}(z_i)$). It is clear that $\hat{\psi}^o_\varepsilon(z) \leq \hat{\psi}^o_0(z)$ for all z and all $\varepsilon \geq 0$ so that

$$\theta(z, p_\varepsilon(z)) \leq \theta_0(z) \triangleq \hat{\psi}^o_0(z) - \psi(z)$$

for all $\varepsilon \geq 0$ and all $z \in R^n$. Clearly θ_0 is continuous, $\theta_0(z) < 0$ for all z such that $\psi(z) > 0$ and $\theta_0(z) = 0$ if $\psi(z) = 0$ (in contrast with θ_ε which satisfies $\theta_\varepsilon(z) < 0$ for all z such that $\psi(z) \geq 0$). The convergence analysis given in the proof of Theorem 3.3.1 with θ_ε replaced by θ_0 reveals that any accumulation point z^* of an infinite sequence $\{z_i\}$ generated by Algorithm 2 satisfies $\theta_0(z^*) = 0$, so that $\psi(z^*) \leq 0$. But this contradicts the fact that $\psi(z^*) > 0$. Hence, ε_i is reduced only finitely often in Step 4. ∎

COROLLARY

Suppose that $\theta'(z) < 0$ for all z such that $\psi(z) \geq -\varepsilon_0$. Then any accumulation point z^* of an infinite sequence $\{z_i\}$ generated by Algorithm 2 satisfies $\theta_{\varepsilon^*}(z^*) = 0$ and $\psi(z^*) \leq -\varepsilon^*$, for some $\varepsilon^* \in (0, \varepsilon_0]$. The inequality $\psi(z) \leq 0$ is satisfied in a finite number of iterations.

PROOF: That $\theta_{\varepsilon^*}(z^*) = 0$, $\varepsilon^* \in (0, \varepsilon_0]$ and $\psi(z^*) < 0$ follow from Proposition 3.4.1 and Theorem 3.3.1. Since $\theta_{\varepsilon^*}(z^*) = \max\{\theta'(z^*), -(\varepsilon^* + \psi(z^*))\}$ it follows that $\theta_{\varepsilon^*}(z^*) < 0$ if $\psi(z^*) > -\varepsilon^*$. Hence $\psi(z^*) \leq -\varepsilon^*$ ∎

It follows from Step 4 of Algorithm 2 that for all $i \geq i^*$, say, the inequality $\hat{\psi}^0(z_i) < -\varepsilon_i = -\varepsilon^*$ holds. Hence, for all $i \geq i^*$, $\hat{\psi}^0_{\varepsilon_i}(z_i) = -\varepsilon^*$ so that $p_{\varepsilon_i}(z_i)$ solves

$$\min\{\|p\| \,\big|\, g^j(z) + g^j_z(z)p \leq -\varepsilon^*,\ j \in \underline{m}\}$$

and is, therefore, a Newton step for the problem of solving the inequalities

$$g^j(z) + \varepsilon^* \leq 0, \quad j \in \underline{m}.$$

We replace H2 by the strengthened hypothesis

H2A: For all z such that $\psi(z) \geq -\varepsilon_0$, the set

$\{\nabla g^j(z),\ j\in I(z)\}, \quad I(z) \triangleq \{j\in \underline{m} \,|\, g^j(z) = \psi(z)\}$

is linear independent (so that $\theta'(z) < 0$).

We assume, in the sequel, that H1, H2a and H3 hold.

PROPOSITION 3.4.2

Let $\varepsilon^*\in(0, \varepsilon_0]$ and let z^* be any point in F satisfying $\psi(z^*) = -\varepsilon^*$. Then there exist a $\delta>0$, k and $k_1\in(o,\infty)$ such that

(i) $\|p_{\varepsilon^*}(z)\| \leq k_1[\psi(z) + \varepsilon^*]$

(ii) $\|p_{\varepsilon^*}(z + p_{\varepsilon^*}(z))\| \leq k\|p_{\varepsilon^*}(z)\|^2$

for all z in $B_\delta(z^*)$.

PROOF: Let $\tilde{p}(z)$ denote the minimum norm solution (when a solution exists) of

$$(g^j(z) + \varepsilon^*)_+ + g^j_z(z)p = 0, \quad j\in I(z^*) \qquad (3.4.1)$$

where, for any $\alpha\in R$, $(\alpha)_+$ denotes $\max\{\alpha, 0\}$. Equation (3.4.1) may be written as

$$A(z)p + (b(z))_+ = 0 \qquad (3.4.2)$$

where the matrix A(z) and the vector b(z) are constructed,

respectively, from the sets $\{g_z^j(z) \mid j\in I(z^*)\}$ and $\{g^j(z)+\varepsilon^* \mid j\in I(z^*)\}$ and $(b(z))_+$ denotes the vector whose components satisfy $b^j(z) = (b^j(z))_+$, $j\in I(z^*)$.

Since $A(z^*)$ has full rank and A and $(b)_+$ are continuous there exists $\delta_1>0$ such that $\tilde{p}$ is continuous and satisfies

$$\tilde{p}(z) = [A(z)^T A(z)]^{-1} A(z)^T (b(z))_+ \qquad (3.4.3)$$

for all $z\in B_{\delta_1}(z^*)$. Since $\max\{g^j(z^*) \mid j\in I(z^*)\} = \psi(z^*)$ and $\max\{g^j(z^*) \mid j\notin I(z^*)\} < \psi(z^*)$, there exists a $\delta_2\in(0,\delta_1]$ such that

$$\psi(z) = \max\{g^j(z) \mid j\in I(z^*)\} \qquad (3.4.4)$$

$$\|(b(z))_+\|_\infty = \psi(z) + \varepsilon^* \qquad (3.4.5)$$

for all $z\in B_{\delta_2}(z^*)$. It follows from (3.4.3) that there exists a $k_1\in(0,\infty)$ such that

$$\|\tilde{p}(z)\| \leq k_1 \|(b(z))_+\|_\infty = k_1[\psi(z) + \varepsilon^*] \qquad (3.4.6)$$

for all $z\in B_{\delta_2}(z^*)$. Next there exists a $\delta_3\in(0,\delta_2]$ such that

(a) $g^j(z) + \varepsilon^* + g_z^j(z)\tilde{p}(z) < 0, \quad j\in I(z^*)$

(b) $g^j(z) + \varepsilon^* + g_z^j(z)\tilde{p}(z) < 0, \quad j\in I(z^*)$

for all $z \in B_{\delta_3}(z^*)$ ((a) follows from the definition of $\tilde{p}$ and (b) follows from (3.4.6) and the fact that $g^j(z^*) + \varepsilon^* < \psi(z^*) + \varepsilon^* = 0$ for all $j \notin I(z^*)$). Hence, $p_{\varepsilon^*}(z)$, the minimum norm solution of (a) and (b), satisfies

$$\|p_{\varepsilon^*}(z)\| \leq \|\tilde{p}(z)\| \tag{3.4.7}$$

for all $z \in B_{\delta_3}(z^*)$, thus providing (i).

Choose $\delta \in (0, \delta_3]$ such that $\|p_{\varepsilon^*}(z)\| \leq \delta_3$ for all $z \in B_\delta(z^*)$. There exists a constant $k_2 \in (0, \infty)$ such that

$$\|g_z(z) - g_z(y)\| \leq k_2 \|z - y\| \tag{3.4.8}$$

for all z, y in $B_{2\delta_3}(z^*)$. Now:

$$b(z + p_{\varepsilon^*}(z)) = b(z) + A(z)p_{\varepsilon^*}(z) + e(z) \tag{3.4.9}$$

where by virtue of (3.4.8),

$$\|e(z)\| \leq k_2 \|p_{\varepsilon^*}(z)\|^2 \tag{3.4.10}$$

for all z in $B_\delta(z^*)$. Since, from the definition of p_{ε^*},

$$b(z) + A(z)p_{\varepsilon^*}(z) \leq 0 \tag{3.4.11}$$

it follows that

$$\|[b(z + p_{\varepsilon^*}(z)]_+\| \leq \|e(z)\| \leq k_2\|p_\varepsilon(z)\|^2 \qquad (3.4.12)$$

for all z in $B_\delta(z^*)$. The desired result follows from (3.4.6) and (3.4.7) with $k = k_1k_2$. ∎

PROPOSITION 3.4.3

Let z^* and ε^* be as in Proposition 3.4.2. Then there exists a $\delta>0$ such that the step length $\lambda_{\varepsilon^*}(z)$ (defined in Step 3 of Algorithm 2 with z_i replaced by z, ε_i replaced by ε^*) is unity for all z in $B_{\varepsilon^*}(z^*)$.

PROOF: It follows from the proof of Proposition 3.4.2 that there exists a $\delta_1\in(0,\infty)$ such that $p_{\varepsilon^*}(z)$ exists (and satisfies $g^j(z) + g^j_z(z)p_{\varepsilon^*}(z) \leq -\varepsilon^*$) for all $z\in B_{\delta_1}(z^*)$. Hence, $\theta_{\varepsilon^*}(z) = \hat{\psi}^0_{\varepsilon^*}(z) - \psi(z) = -[\psi(z) + \varepsilon^*]$ for all $z\in B_{\delta_1}(z^*)$. From Proposition 3.4.2, there exists a $\delta\in(0,\delta_1]$ and a $k_1\in(0,\infty)$ such that $\|p_{\varepsilon^*}(z)\| \leq k_1[\psi(z) + \varepsilon^*]$ for all $z\in B_\delta(z^*)$. It is evident from Proposition 3.3.1 that δ can be chosen sufficiently small so that

$$\left|\psi(z + p_{\varepsilon^*}(z)) - \hat{\psi}(z, p_{\varepsilon^*}(z))\right| \leq \|p_{\varepsilon^*}(z)\|/4k_1$$

$$\leq [\psi(z) + \varepsilon^*]/4$$

$$\leq -\theta_{\varepsilon^*}(z)/4$$

for all $z \in B_\delta(z^*)$. Hence

$$\psi(z + p_{\varepsilon^*}(z)) - \psi(z) \leq [\hat{\psi}(z, p_{\varepsilon^*}(z)) - \psi(z)]$$

$$+ \left|\psi(z + p_{\varepsilon^*}(z)) - \hat{\psi}(z, p_{\varepsilon^*}(z))\right|$$

$$\leq \theta_{\varepsilon^*}(z) - \theta_{\varepsilon^*}(z)/4$$

$$\leq \theta_{\varepsilon^*}(z)/2$$

so that $\lambda_{\varepsilon^*}(z) = 1$ for all $z \in B_\delta(z^*)$. ∎

THEOREM 3.4.2

Let $\{z_i\}$ and $\{\varepsilon_i\}$ be infinite sequences generated by the algorithm. Then $\varepsilon_i = \varepsilon^* > 0$ for all i sufficiently large and z_i converges quadratically to a $\bar{z}$ satisfying $\psi(\bar{z}) \leq -\varepsilon^*$.

PROOF: From Proposition 3.4.1 there exists an integer i^* such that $\varepsilon_i = \varepsilon^* \in (0, \varepsilon_0]$ for all $i \geq i^*$. Let z^* be an accumulation

point of $\{z_i\}$. From Corollary to Proposition 3.4.1, $\psi(z^*) \leq -\varepsilon^*$. Suppose $\psi(z^*) < -\varepsilon^*$. Since $p_{\varepsilon^*}(z) = 0$ for all z such that $\psi(z) \leq -\varepsilon^*$, it follows that $p_i = p_{\varepsilon_i}(z_i) = p_{\varepsilon^*}(z_i) = 0$ (so that $z_i = z^*$) for all i sufficiently large. Then, suppose that $\psi(z^*) = -\varepsilon^*$. Choose $\delta > 0$ to satisfy the hypotheses of Proposition 3.4.2 and 3.4.3. By Proposition 3.4.2 there exist an $i_1 \geq i^*$ and a $k \in (0,\infty)$ such that

(a) $z_{i_1} \in B_{\delta/2}(z^*)$

(b) $\|p_{i_1}\| = \|p_{\varepsilon^*}(z_{i_1})\| \leq 1/(2/k)$

(c) $\|p_{i_1}\| \leq \delta/4$

From Proposition 3.4.3, $\lambda_{i_1} = 1$ so that

$$\|z_{i_1+1} - z^*\| \leq \|z_{i_1} - z^*\| + \|p_{i_1}\|$$

$$\leq \delta/2 + \delta/4 = (3/4)\delta$$

Hence $\lambda_{i_1+1} = 1$ and

$$\|p_{i_1+1}\| \leq k\|p_{i_1}\|^2 \leq (k/2k)\|p_{i_1}\| \leq \delta/8$$

so that

$$\|z_{i_1+2} - z^*\| < (7/8)\delta$$

Proceeding in this fashion we find that $\lambda_i = 1$ and $z_i \in B_\delta(z^*)$ for all $i \geq i_1$. Also, since $\|p_{i+1}\| < k\,\|p_i\|^2$ for all $i \geq i_1$ it follows that the sequence $\{z_i\}$ is a Cauchy sequence and hence converges to a $\bar{z}$ in $B_\delta(z^*)$. Clearly (Proposition 3.4.1 and its Corollary), $\theta_{\varepsilon^*}(\bar{z}) = 0$ and $\psi_{\varepsilon^*}(\bar{z}) < 0$. Finally,

$$\lim_{i\to\infty} \frac{\|p_{i+1}\|}{\|p_i\|^2} < k.$$

This is a sufficient condition for the sequence $\{z_i\}$ to converge quadratically to $\bar{z}$ as $i \to \infty$. ∎

3.5 - NUMERICAL EXAMPLES

In the following two examples we compare the performance of our algorithm with that in Reference 2.

EXAMPLE 1: The feasible set consists of a pair of squares of sides π, centered at $(-\pi/2, 0)$ and $(3\pi/2, 0)$, and is defined by

$$\begin{aligned} \sin z_1 &< 0 \\ -\cos z_2 &< 0 \\ z_1 - 2\pi &< 0 \\ z_2 - \pi/2 &< 0 \end{aligned}$$

$$-z_1 - \pi \leq 0$$

$$-z_2 - \pi/2 \leq 0.$$

Starting from $z_0 = (0, 75)$, a feasible point $z = (-1.5, 0)$ was located in 2 iterations, compared with four iterations taken by algorithm in Reference 2.

EXAMPLE 2: The feasible set is specified as follows

$$z_1 \geq 0,$$

$$z_3 \geq 0,$$

$$(z_4 - z_6)^2 - (z_5 - z_7)^2 - 4 \geq 0,$$

$$(z_i - 1 \geq 0, \quad i = 5, 7$$

$$(z_3 z_i - z_2 z_{i+1})/(z_2^2 + z_3^2)^{1/2} - 1 \geq 0' \quad i = 4, 6$$

$$[(z_2 - z_1)z_{i+1} + (z_1 - z_i)z_3]/[z_3^2 + (z_2 - z_1)^2]^{1/2} - 1 \geq 0, \quad i = 4, 6$$

$$24 - z_1 z_3 \geq 0.$$

Starting from an initial point $z_0 = (3, 0, 2, -3, 1.5, 5, 0)$, $\varepsilon_0 = 0.0025$ a feasible point z = (6.734, 2.192, 3.561, 1.807, 1.01, 3.809, 1.01) was located in six iterations, compared with thirteen iterations taken by algorithm of Reference 2. Figure 3.1 shows the performance of the algorithm when the stop condition is removed; the rate of convergence is clearly quadratic.

3.6 - DISCUSSION

The algorithm has been programmed in Fortran and extensively tested. It has also been incorporated in SIMNON and employed for the design of nonlinear dynamic systems to satisfy nonlinear constraints. The algorithm works well in high order problems where the linearized feasible set is often empty. In Step 1 of the algorithm $\hat{\psi}^{O}(z) = \psi(z)$ indicates that the linearized feasible set is empty (this indicates that the feasible set is, probably, empty).

3.7 - REFERENCES

[1] - Polak E., "Computational methods in optimization: A unified approach", Academic Press, N.Y., 1971.

[2] - Polak E., Mayne D.Q., "On the finite solution of nonlinear inequalities",IEEE Trans. Automatic Control, AC-24, pp. 443-444, 1979.

[3] - Mayne D.Q., Polak E. and Heunis A.J.,"Solving nonlinear inequalities in a finite number of iterations", JOTA, Vol. 33, pp. 207-222, 1981.

[4] - Garcia-Palomares U.M., Restuccia A.,"A global quadratic algorithm for solving a system of mixed equalities and inequalities", Math. Prog., Vol 21, pp. 290-300, 1981.

[5] - Garcia-Palomares U.M.,"Superlinearly convergent algorithms for linearly constrained optimization", in Nonlinear Programming 2, O.L. Mangasarian, ed., Academic Press, New York, pp. 101-120, 1975.

[6] - Mayne D.Q., Sahba M., "An efficient algorithm for solving inequalities", Accepted for publication by JOTA, 1983.

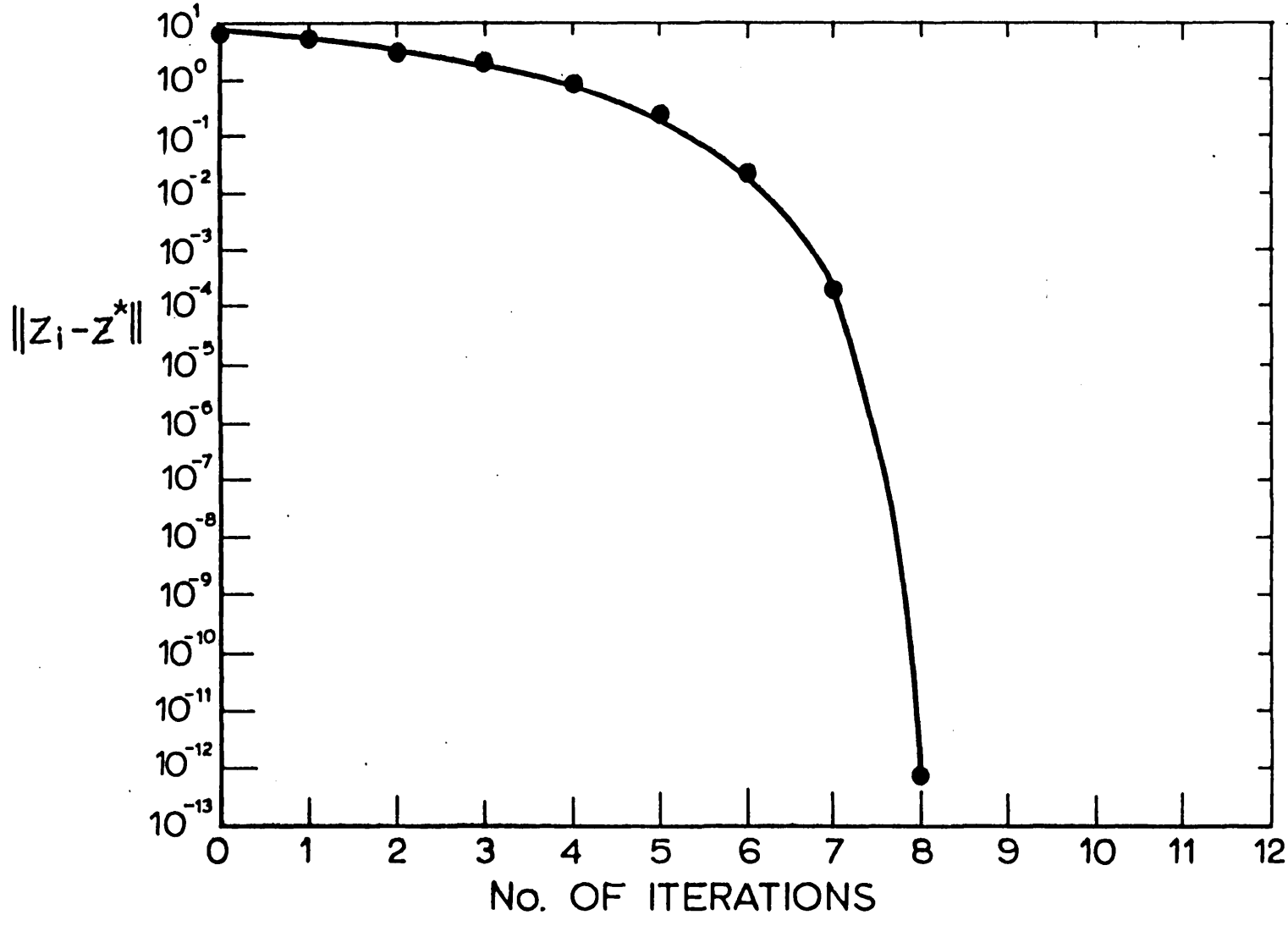

Fig. 3.1 Rate of Convergence in Example 2

CHAPTER 4

DERIVATIVE FREE ALGORITHMS FOR SOLVING NONLINEAR INEQUALITIES IN A FINITE NUMBER OF ITERATIONS

4.1 - INTRODUCTION

It is well known that an important group of engineering problems can be transcribed into the problem of finding any point satisfying:

$$g^j(z) < 0 \qquad j \in \{1,\ldots,m\} \tag{4.1.1}$$

where $g^j: R^n \to R$ is continuously differentiable. There are several algorithms for solving a set of inequalities [1,2,3,4], but they all require explicit computation of the gradient $\nabla g^j(z)$, $j=1,\ldots,m$. In some engineering problems, it is not possible to evaluate the gradient analytically (e.g. when the constraint values are computed by solving a set of differential equations [5]). In these cases, in order to evaluate $\nabla g(z)$, one has either to employ a finite difference approximation, or solve a system of adjoint equations and integrate a differential equation. The latter is not always available, and in any case the operation may be very expensive. Hence, there is a great incentive to construct an algorithm, which uses function values only, for solving inequalities. Since Newton type algorithms are generally known to be very efficient we need to find an approximation to the gradient $\nabla g^j(z)$, $j = 1,\ldots,m$. The most direct way of avoiding

the computation of $\nabla g^j(z)$ is to use a finite difference approximation of the partial derivatives. A method for approximating $\nabla g(z_i)$ has been proposed by Broyden [6]. A desirable feature of the algorithm is that it should find a feasible point in a finite number of iterations and, if allowed to proceed, should have a reasonable rate of convergence. There are a few methods [1,2,3] which find a solution in a finite number of iterations. In this chapter we construct two algorithms which only require function evaluations and nevertheless find solutions in a finite number of iterations. The first algorithm only utilizes finite difference approximation of the gradient, while in the second algorithm a mixture of finite difference approximation and Broyden type approximation of the gradient is employed.

4.2 - DEFINITIONS AND ASSUMPTIONS

We define $\psi: R^n \to R$ by

$$\psi(z) \triangleq \max\{g^j(z) \mid j \in \underline{m}\} \tag{4.2.1}$$

where $\underline{m} \triangleq \{1,2,\ldots,m\}$. Let $g(z)$ denote the (column) vector $(g^1(z),\ldots,g^m(z))$.

Assumption 1: The function $g(.):R^n \to R^m$ is continuously differentiable.

For any $z \in R^n$, $\tau > 0$, $j \in \underline{m}$, let $\nabla_\tau g^j(z)$ be an approximation to the gradient $\nabla g^j(z)$, where τ indicates the precision of the

approximation.

Assumption 2: Given any compact set C, a subset of R^n, and any $\mu>0$, there exists a $\bar{\tau}>0$ such that for all $z\in C$, all $\tau\in[0,\bar{\tau}]$ and all $j\in\underline{m}$

$$\|\nabla_\tau g^j(z) - \nabla g^j(z)\| < \mu.$$

Assumption 3: For all $\tau>0$ and $j\in\underline{m}$, $\nabla_\tau g^j(.)$ is continuous.

An example which satisfies the above assumptions is the simple difference formula given by [5]:

$$\nabla_\tau g^j(z) = \begin{bmatrix} (g^j(z+\tau_1 e_1) - g^j(z))/\tau_1 \\ \vdots \\ (g^j(z+\tau_n e_n) - g^j(z))/\tau_n \end{bmatrix} \qquad (4.2.2)$$

where e_j is the jth column of identity matrix and $\tau_j\in(0,\bar{\tau}]$, $j = 1,\ldots,n$, $\bar{\tau}>0$ are positive constants. The matrix $D_\tau g(z)$ is defined to be the approximation to the gradient matrix whose rows are $D_\tau g^j(z) \triangleq [\nabla_\tau g^j(z)]^T$, $j = 1,\ldots,m$. For $\tau=0$, we define

$$D_0 g(z) \triangleq g_z(z), \quad D_0 g^j(z) \triangleq g_z^j(z). \qquad (4.2.3)$$

Let the set F be defined by:

$$F \triangleq \{z \mid g^j(z) < 0,\ j\in\underline{m}\}. \qquad (4.2.4)$$

For all z, p in R^n let $\hat{\psi}(z,p)$ denote the following first order approximation to $\psi(z+p)$

$$\hat{\psi}(z,p) \triangleq \max\{g^j(z) + D_0 g^j(z)p \,\big|\, j\in\underline{m}\} \tag{4.2.5}$$

For all $\tau>0$, let $\hat{\psi}_\tau(z,p)$ be defined by

$$\hat{\psi}_\tau(z,p) \triangleq \max\{g^j(z) + D_\tau g^j(z)p \,\big|\, j\in\underline{m}\}. \tag{4.2.6}$$

The approximate Newton step at z, if it exists, is that p in R^n which solves

$$\min\{\|p\| \,\big|\, \hat{\psi}_\tau(z,p) \leq 0\}. \tag{4.2.7}$$

Since the set $\{p \,\big|\, \hat{\psi}_\tau(z,p) \leq 0\}$ may be empty (implying the non existence of the Newton step) another approach is required. Let functions $\hat{\psi}^0$, $\hat{\psi}^0_\tau$, $\hat{\psi}^0_{\tau,\varepsilon}$, and $\hat{\psi}^0_{0,\varepsilon}: R^n \to R$ be defined, for all $\varepsilon>0$ and all $\tau>0$, by

$$\hat{\psi}^0(z) \triangleq \min\{\hat{\psi}(z,p) \,\big|\, p\in P\}, \tag{4.2.8}$$

$$\hat{\psi}^0_\tau(z) \triangleq \min\{\hat{\psi}_\tau(z,p) \,\big|\, p\in P\} \tag{4.2.9}$$

and

$$\hat{\psi}^0_{\tau,\varepsilon}(z) \triangleq \max\{\hat{\psi}^0_\tau(z), -\varepsilon\}, \tag{4.2.10}$$

$$\hat{\psi}^0_{0,\varepsilon}(z) \triangleq \max\{\hat{\psi}^0(z), -\varepsilon\}, \tag{4.2.11}$$

where

$$P \triangleq \{p \in R^n \mid \|p\|_\infty \leq L\} \tag{4.2.12}$$

and where L is some suitably chosen large number (without the constraint $p \in P$ the solutions of (4.2.8) and (4.2.9) may be unbounded). If $\hat{\psi}^o_\tau(z) < 0$, the solution of (4.2.7) exists (the set $\{p \mid \hat{\psi}_\tau(z,p) < 0\}$ is not empty).

We now define our search direction $p_{\tau,\varepsilon}(z)$ given z, τ, and ε; it is that p which solves the following quadratic program

$$\min\{\|p\| \mid \hat{\psi}_\tau(z,p) \leq \hat{\psi}^o_{\tau,\varepsilon}(z)\}. \tag{4.2.13}$$

Since the level sets of $p \to \|p\|$ are strictly convex and since $\{p \mid \hat{\psi}_\tau(z,p) \leq \hat{\psi}^o_{\tau,\varepsilon}(z)\}$ is convex, $p_{\tau,\varepsilon}(z)$ is unique. Also for all τ, ε, and z, $p_{\tau,\varepsilon}(z) \in P$.

For all $\tau > 0$, let θ and $\hat{\theta}_\tau : R^n \times R^n \to R$ be defined by

$$\theta(z,p) \triangleq \hat{\psi}(z,p) - \psi(z), \tag{4.2.14}$$

and

$$\hat{\theta}_\tau(z,p) \triangleq \hat{\psi}_\tau(z,p) - \psi(z). \tag{4.2.15}$$

Clearly $\theta(z,p)$ is a first order approximation of $\psi(z+p) - \psi(z)$. Similarly, for all $\tau > 0$ and all $\varepsilon > 0$, θ^o, θ^o_ε and $\hat{\theta}^o_{\tau,\varepsilon} : R^n \to R$ are defined by

$$\theta^o(z) \triangleq \hat{\psi}^o(z) - \psi(z), \tag{4.2.16}$$

$$\theta^{o}_{\varepsilon}(z) \triangleq \hat{\psi}^{o}_{0,\varepsilon}(z) - \psi(z) \qquad (4.2.17)$$

and

$$\hat{\theta}^{o}_{\tau,\varepsilon}(z) \triangleq \hat{\psi}^{o}_{\tau,\varepsilon}(z) - \psi(z). \qquad (4.2.18)$$

Clearly $\theta^{o}_{\varepsilon}(z) = \max\{\theta^{o}(z), -(\psi(z)+\varepsilon)\}$. (4.2.19)

Also

$$\hat{\theta}^{o}_{\tau,\varepsilon}(z) = \hat{\theta}_{\tau}(z, p_{\tau,\varepsilon}(z)) \qquad (4.2.20)$$

is an estimate of $\psi(z+p_{\tau,\varepsilon}(z)) - \psi(z)$, i.e. an estimate of change of cost obtained by employing the search direction $p_{\tau,\varepsilon}(z)$. Let F^{c} denote the complement of F, i.e.

$$F^{c} \triangleq \{z \,|\, \psi(z) > 0\}. \qquad (4.2.21)$$

Assumption 4: For all z in F^{c}, $\theta^{o}(z) < 0$.

The above assumption ensures that $\psi(z)$ can be decreased at all z in F^{c}. A sufficient condition for A4 is the positive linear independence, for all z in F, of the set $\{\nabla g^{j}(z) \,|\, j \in I(z)\}$, where $I(z) \triangleq \{j \in \underline{m} \,|\, g^{j}(z) = \psi(z)\}$ (i.e. the set of gradient of the most active constraints).

The step length is chosen to be the greatest number in the set $S \triangleq \{1, \beta, \beta^{2}, \ldots\}$, $\beta \in (0,1)$ such that the actual change in ψ is at least half of the estimated change.

4.3 - ALGORITHM

We now have all the ingredients to state our first algorithm:

Algorithm 1.

Data: $z_0 \in R^n$, $\varepsilon' \in (0,1)$, $L >> 1$, $\beta \in (0,1)$, $\tau_0 > 0$, $\lambda_{min} > 0$.

Step 0: Set $i = 1$, set $\varepsilon = \varepsilon' \psi(z_0)$.

Step 1: If $\psi(z_i) < 0$ stop.

Step 2: Find $\tau_i = \tau(z_i)$ the largest $\tau \in \{\tau_{i-1}, \tau_{i-1}/2, \ldots\}$ such that $\hat{\theta}^0_{\tau_i,\varepsilon}(z_i) < -\tau_i$.

Step 3: Compute

$$p_{\tau_i,\varepsilon}(z_i) = \operatorname{argmin}\{\|p\| \,\big|\, \hat{\psi}_{\tau_i}(z_i,p) < \hat{\psi}^0_{\tau_i,\varepsilon}(z_i)\}.$$

Step 4: Find, if possible, $\lambda_i = \lambda_{\tau_i,\varepsilon}(z_i)$ the largest number in S not less that $\lambda_{min}\tau_i$ such that

$$\psi(z_i+\lambda_i p_{\tau_i,\varepsilon}(z_i)) - \psi(z_i) < \lambda_i \hat{\theta}^0_{\tau_i,\varepsilon}(z_i)/2$$

else, set $\tau_{i-1} = \tau_{i-1}/2$ and go to Step 2.

Step 5: Set $z_{i+1} = z_i + \lambda_i p_{\tau_i,\varepsilon}(z_i)$, set $i = i+1$. Go to Step 1. ∎

4.4 - CONVERGENCE

For all z, all $\delta>0$ let $B_\delta(z) \triangleq \{z' \mid \|z'-z\| < \delta\}$.

We shall first establish that the algorithm has certain continuity properties.

PROPOSITION 4.4.1

For all $\tau\geq 0$, $z \to \hat{\psi}^o_\tau(z)$ is continuous.

PROOF: Let z be any arbitrary point in R^n and let $\delta>0$. Let z' be any arbitrary point in $B_\delta(z)$. Let $p_\tau(z)$, $p_\tau(z')$ be any points in P satisfying

$$\hat{\psi}^o_\tau(z) = \hat{\psi}_\tau(z,p_\tau(z)),$$

$$\hat{\psi}^o_\tau(z') = \hat{\psi}_\tau(z',p_\tau(z')).$$

Hence,

$$\hat{\psi}^o_\tau(z') - \hat{\psi}^o_\tau(z) = \hat{\psi}_\tau(z',p_\tau(z')) - \hat{\psi}_\tau(z,p_\tau(z))$$

$$\leq \hat{\psi}_\tau(z',p_\tau(z)) - \hat{\psi}_\tau(z,p_\tau(z))$$

and

$$\hat{\psi}^{o}_{\tau}(z) - \hat{\psi}^{o}_{\tau}(z') = \hat{\psi}_{\tau}(z,p_{\tau}(z)) - \hat{\psi}_{\tau}(z',p_{\tau}(z'))$$

$$\leq \hat{\psi}_{\tau}(z,p_{\tau}(z')) - \hat{\psi}_{\tau}(z',p_{\tau}(z')).$$

Since $\hat{\psi}_{\tau}$ is uniformly continous in $B_{\delta}(z)\times P$, it follows that

$$\left|\hat{\psi}^{o}_{\tau}(z') - \hat{\psi}^{o}_{\tau}(z)\right| \to 0 \quad \text{as } z' \to z. \qquad \blacksquare$$

COROLLARY: For all $\varepsilon>0$ and all $\tau\geq 0$, $\hat{\psi}^{o}_{\tau,\varepsilon}:R^{n} \to R$ is continuous.

PROPOSITION 4.4.2

For all $\varepsilon\geq 0$,

i) $\theta^{o},\theta^{o}_{\varepsilon}:R^{n} \to R$ are continuous,

ii) for all $z\in F^{c}$, $\theta^{o}_{\varepsilon}(z) < 0$,

iii) the function $(z,\tau) \to \hat{\theta}^{o}_{\tau,\varepsilon}(z)$ is continuous in $R^{n}\times R^{+}$,

iv) for all $z_{0}\in F^{c}$, there exists a $\tau_{1}>0$, and a $\varrho>0$, such that $\hat{\theta}^{o}_{\tau_{1},\varepsilon}(z) \leq -\tau_{1}$, for all $z\in B_{\varrho}(z_{0})$.

PROOF: i) The continuity of θ^{o} and θ^{o}_{ε} follows from the continuity of $\hat{\psi}^{o}$, $\hat{\psi}^{o}_{0,\varepsilon}$ and ψ.

ii) By Assumption 4, $\theta^{o}(z) < 0$, for all $z\in F^{c}$. Thus

$$\theta^{o}_{\varepsilon}(z) = \max\{\theta^{o}(z), -(\psi(z)+\varepsilon)\} < 0$$

for all $z\in F^{c}$.

iii) Consider the function $\eta^{j}_{i}:R^{n}\times R^{+} \to R$, defined by

$$\eta_i^j(z,\tau) \triangleq \left[[g^j(z+\tau e_i)-g^j(z)]/\tau\right] = \int_0^1 \left[[g^j_{z_i}(z+s\tau e_i)\tau]/\tau\right] ds$$

$$= \int_0^1 g^j_{z_i}(z+s\tau e_i) ds$$

for all $i = 1,\ldots,n$, all $j \in \underline{m}$. Since $g^j_{z_i}$ is uniformly continuous on any compact set, then, for all $\delta > 0$, there exists a $\rho > 0$ such that $\|g^j_{z_i}(z'') - g^j_{z_i}(z')\| < \delta$ for all z' and z'' in $B_\rho(z)$. Thus

$$\left|\eta_i^j(z',\tau') - \eta_i^j(z,\tau)\right| < \left|\int_0^1 [g^j_{z_i}(z'+s\tau' e_i)-g^j_{z_i}(z+s\tau e_i)] ds\right|$$

$$< \int_0^1 \left|[g^j_{z_i}(z'+s\tau' e_i)-g^j_{z_i}(z+s\tau e_i)]\right| ds.$$

Hence for all $\delta > 0$, there exists a $\rho > 0$ such that

$$\left|\eta_i^j(z',\tau') - \eta_i^j(z,\tau)\right| < \int_0^1 \delta ds = \delta$$

for all $z' \in B_{\rho/2}(z)$, all $\tau' \in B_{\rho/2}(\tau)$ (so that $(z'+s\tau' e_i) \in B_\rho(z+s\tau e_i)$ if $s \in [0,1]$), all $i = 1,2,\ldots,n$. and all $j \in \underline{m}$. This implies that $(z,\tau,p) \to \hat{\psi}_\tau(z,p)$ is continuous. The continuity of $(z,\tau) \to \hat{\psi}^o_\tau(z) = \min\{\hat{\psi}_\tau(z,p) \,|\, p \in P\}$ (and, hence, of $(z,\tau) \to \hat{\psi}^o_{\tau,\varepsilon}(z)$) is then established by Berge's maximum theorem [9]. Continuity of $(z,\tau) \to \hat{\theta}^o_{\tau,\varepsilon}(z)$ follows from continuity for $\varepsilon > 0$ of $(z,\tau) \to \hat{\psi}^o_{\tau,\varepsilon}(z)$, and ψ.

iv) Let $\varepsilon > 0$ and $z_o \in F^c$. From (iii), $(z,\tau) \to \hat{\theta}^o_{\tau,\varepsilon}(z)$ is con-

tinuous and from (ii) $\theta^{o}_{\varepsilon}(z_o)<0$, hence, there exists a $\tau_1 >0$ such that $\hat{\theta}^{o}_{\tau_1,\varepsilon}(z) \leq -\tau_1 < 0$, for all $z\in B_{\varrho}(z_o)$. ▪

By virtue of Proposition 4.4.2(iv) Step2 of the algorithm is well defined, i.e. if $z\in F^c$, a τ_1 satisfying $\hat{\theta}^{o}_{\tau_1,\varepsilon}(z) \leq -\tau_1 < 0$ is achieved in a finite number of iterations.

PROPOSITION 4.4.3

For all $\eta>0$, all z in R^n, there exists a $\varrho>0$ such that

$$\left|\psi(z'+p) - \hat{\psi}(z',p)\right| \leq \eta\|p\|$$

for all $z'\in B_{\varrho}(z)$, and all $p\in B_{\varrho}(0)$.

For a proof of proposition 4.4.3 see the proof of Proposition 3.3.1 of Chapter 3.

PROPOSITION 4.4.4

Let $\varepsilon>0$. For all z in F^c, there exists a $\varrho>0$ and a positive $\bar{\tau}$ in S such that

$$\psi(z'+\tau p_{\tau,\varepsilon}(z')) - \psi(z') \leq \tau\hat{\theta}^{o}_{\tau,\varepsilon}(z')/2 \leq \theta^{o}_{\varepsilon}(z)/4 < 0$$

for all z' in $B_{\varrho}(z)$, and all $\tau\in[0,\bar{\tau}]$.

PROOF: By Proposition 4.4.2 for all $\varepsilon>0$ and $z\in F^C$ there exists a $\varrho_1>0$ and a $\tau_1>0$ (sufficiently small) such that

$$\hat{\theta}^{O}_{\tau,\varepsilon}(z') \in [(3/2)\theta^{O}_{\varepsilon}(z),\ (1/2)\theta^{O}_{\varepsilon}(z)], \qquad \theta^{O}_{\varepsilon}(z) < 0$$

for all $\tau\in[0,\tau_1]$ and all $z'\in B_{\varrho_1}(z)$. From Proposition 4.4.3 (since $p\in P$ implies $\|\lambda p\| \leq \lambda L$ for some $L < \infty$) there exists a $\varrho\in(0,\varrho_1]$ such that

$$\left|\psi(z'+\lambda p) - \hat{\psi}(z',\lambda p)\right| \leq (-\theta^{O}_{\varepsilon}(z)/8)\lambda$$

for all $z'\in B_{\varrho}(z)$, all $p\in P$, all $\lambda\in(0,\varrho/L]$. Hence, for all $\tau\in[0,\tau_1]$, all $z'\in B_{\varrho}(z)$, and all $\lambda\in(0,\varrho/L]$

$$\psi(z'+\lambda p_{\tau,\varepsilon}(z')) - \psi(z') \leq \hat{\psi}(z',\lambda p_{\tau,\varepsilon}(z')) - \psi(z') +$$

$$\left|\psi(z'+\lambda p_{\tau,\varepsilon}(z')) - \hat{\psi}(z',\lambda p_{\tau,\varepsilon}(z'))\right|.$$

$$\leq \hat{\psi}(z',\lambda p_{\tau,\varepsilon}(z')) - \psi(z') - \lambda\theta^{O}_{\varepsilon}(z)/8.$$

(We have made use of the fact that $p_{\tau,\varepsilon}(z')\in P$.)
Since $\lambda \to \hat{\psi}(z',\lambda p_{\tau,\varepsilon}(z')) - \psi(z')$ is convex,

$$\psi(z'+\lambda p_{\tau,\varepsilon}(z')) - \psi(z') \leq \lambda[\hat{\psi}(z'+p_{\tau,\varepsilon}(z')) - \psi(z')] - \lambda\theta^{O}_{\varepsilon}(z)/8$$

$$\leq \lambda[\hat{\psi}_{\tau}(z',p_{\tau,\varepsilon}(z')) - \psi(z')] +$$

$$\lambda\left|\hat{\psi}(z'+p_{\tau,\varepsilon}(z')) - \hat{\psi}_{\tau}(z',p_{\tau,\varepsilon}(z'))\right| - \lambda\theta^{O}_{\varepsilon}(z)/8$$

$$= \lambda\hat{\theta}^{o}_{\tau,\varepsilon}(z') + \lambda\left|\hat{\psi}(z'+p_{\tau,\varepsilon}(z')) - \hat{\psi}_{\tau}(z', p_{\tau,\varepsilon}(z'))\right|$$

$$- \lambda\theta^{o}_{\varepsilon}(z)/8.$$

Since $\hat{\theta}^{o}_{\tau,\varepsilon}(z') \leq \theta^{o}_{\varepsilon}(z)/2$ implies $-\theta^{o}_{\varepsilon}(z) \leq -2\hat{\theta}^{o}_{\tau,\varepsilon}(z')$, and also

$$\left|\hat{\psi}(z', p_{\tau,\varepsilon}(z')) - \hat{\psi}_{\tau}(z', p_{\tau,\varepsilon}(z'))\right| = \left|\max_{j}\{g^{j}(z') + g^{j}_{z}(z')p_{\tau,\varepsilon}(z')\}\right.$$

$$\left. - \max_{j}\{g^{j}(z') + D_{\tau}g^{j}(z')p_{\tau,\varepsilon}(z')\}\right|, \quad j \in \underline{m}$$

$$\leq \max_{j}\left|\{g^{j}(z') + g^{j}_{z}(z')p_{\tau,\varepsilon}(z')\right.$$

$$\left. - g^{j}(z') - D_{\tau}g^{j}(z')p_{\tau,\varepsilon}(z')\}\right|, \quad j \in \underline{m}$$

$$\leq \max_{j}\left|\{[g^{j}_{z}(z') - D_{\tau}g^{j}(z')]p_{\tau,\varepsilon}(z')\}\right| \quad j \in \underline{m}.$$

Hence

$$\left|\hat{\psi}(z', p_{\tau,\varepsilon}(z')) - \hat{\psi}_{\tau}(z', p_{\tau,\varepsilon}(z'))\right| \leq \|g_{z}(z') - D_{\tau}(z')\| . \|p_{\tau,\varepsilon}(z')\|.$$

Given Assumption 2, there exists a $\bar{\tau} \in (0, \tau_1] \cap S$ such that for all $\tau \in [0, \bar{\tau}]$, and all $z' \in B_{\varrho}(z)$

$$\left|\hat{\psi}(z', p_{\tau,\varepsilon}(z')) - \hat{\psi}_{\tau}(z', p_{\tau,\varepsilon}(z'))\right| \leq -\theta^{o}_{\varepsilon}(z)/8.$$

Thus

$$\psi(z'+\lambda p_{\tau,\varepsilon}(z')) - \psi(z') \leq \lambda\hat{\theta}^{O}_{\tau,\varepsilon}(z')/2 \leq \lambda\theta_{\varepsilon}(z)/4$$

for all $z' \in B_{\varrho}(z)$, all $\lambda \in [0, \varrho/L]$, and all $\tau \in [0, \bar{\tau}]$. The desired result follows with τ, the largest number in S which is not greater than $\max[\varrho/L, \bar{\tau}]$. ∎

COROLLARY: For all $\varepsilon > 0$, and all $z \in F^{C}$, there exists a $\varrho > 0$ and $\bar{\tau} > 0$ such that $\lambda_{\tau,\varepsilon}(z') \geq \tau$ for all $z' \in B_{\varrho}(z)$, and all $\tau \in [0, \bar{\tau}]$. ∎

Let $A_{\varepsilon}: R^{+} \times R^{n} \to R^{n}$ be defined by $A_{\varepsilon}(\tau, z) = z + \tau p_{\tau,\varepsilon}(z)$. Then, as a result of Proposition 4.4.4, for all $z \in F^{C}$, and $\varepsilon > 0$ there exists a $\mu(z)$, $\varrho(z)$, and $\tau(z) > 0$ such that

$$\psi(z'') - \psi(z') \leq -\mu(z) \qquad (4.4.1)$$

for all $z' \in B_{\varrho(z)}(z)$, all $z'' = A_{\varepsilon}(\tau, z')$, and all $\tau \in (0, \tau(z)]$.

Given the above propositions, the algorithm is well defined. We can now state our main convergence results.

PROPOSITION 4.4.5

If the algorithm constructs an infinite sequence $\{z_i\}_0^{\infty}$, either $\tau_i \to 0$ or $\{z_i\}_0^{\infty}$ has no accumulation points in F^{C}.

PROOF: Let z^{*} be any accumulation point of an infinite sequence $\{z_i\}$, i.e. $z_i \overset{K}{\to} z^{*}$ for some set K in $\{0, 1, 2, \ldots\}$.

Since $\{\tau_i\}$ is a monotonically decreasing sequence, bounded from below, it must converge. Suppose, contrary to what is to be proven, that $\tau_i \to \tau^* > 0$. Then, by construction, for some i^* large enough and for all $i \geq i^*$, $\tau_i = \tau^* > 0$. For all $z_i \in F^C$ and $i \geq i^*$ from Step 2,

$$\hat{\theta}^O_{\tau^*,\varepsilon}(z_i) < -\tau^*$$

and from Step 4,

$$\psi(z_{i+1}) - \psi(z_i) < \lambda_{\tau^*,\varepsilon}(z_i)\hat{\theta}^O_{\tau^*,\varepsilon}(z_i)/2$$

$$< \lambda_{\min}\tau^*\hat{\theta}^O_{\tau^*,\varepsilon}(z_i)/2.$$

If $z^* \in F^C$, then

$$\lim_{\substack{i\to\infty \\ i,j\in K}} [\psi(z_j) - \psi(z_i)] < \lambda_{\min}\tau^*\hat{\theta}^O_{\tau^*,\varepsilon}(z^*)/2 < 0 \qquad (4.4.2)$$

where i and j are successive numbers in K. Since $\{\psi(z_i)\}$ is a monotonically decreasing sequence, bounded from below, it must converge, i.e.

$$\lim_{\substack{i\to\infty \\ i,j\in K}} [\psi(z_j) - \psi(z_i)] = 0$$

which contradicts (4.4.2); we conclude that $\tau_i \to 0$. ∎

THEOREM 4.4.1

Let $\{z_i\}$ be an infinite sequence constructed by the algorithm. If $\{z_i\}$ is finite then its last element is desirable; if $\{z_i\}$ is infinite every accumulation point is desirable.

PROOF: Suppose the sequence $\{z_i\}$ is finite and z_{i^*} is the last element. The algorithm then constructs an infinite sequence $\{y_j\}_0^\infty$, where $y_j \in A_\varepsilon\left[\tau_{i^*-1}/2^j, z_{i^*}\right]$, $j = 0,1,2,\ldots$. Suppose, contrary to what is to be proven that $z_{i^*} \notin F$, then

$$\psi(y_j) - \psi(z_{i^*}) > \lambda_{\min}[\tau_{i^*-1}/2^j][\hat{\theta}^o_{\tau_{i^*},\varepsilon}(z_{i^*})/2], \quad j = 0,1,2,\ldots \qquad (4.4.3)$$

As a result of Proposition 4.4.4, there exists a $\mu(z_{i^*}) > 0$, $\varrho(z_{i^*}) > 0$, and a $\tau(z_{i^*}) > 0$ such that

$$\psi(z'') - \psi(z') \leq -\mu(z_{i^*})$$

for all $z' \in B_{\varrho(z_{i^*})}(z_{i^*})$, all $z'' = A_\varepsilon(\tau, z')$, and all $\tau \in (0, \tau(z_{i^*})]$. Let $j^* \in \{0,1,\ldots\}$ be such that

$$\tau_{i^*-1}/2^{j^*} \leq \min\left[-\mu(z_{i^*})/[\lambda_{\min}\hat{\theta}^o_{\tau,\varepsilon}(z_{i^*})/2],\ \tau(z_{i^*})\right].$$

Hence

$$\psi(y_j) - \psi(z_{i^*}) \leq -\mu(z_{i^*}) \leq \lambda_{\min}[\tau_{i^*-1}/2^j][\hat{\theta}^o_{\tau_{i^*}}, \varepsilon(z_{i^*})/2]$$

for all $j \geq j^*$. This contradicts (4.4.3) and we conclude that $z_{i^*} \in F$.

Now suppose $\{z_i\}$ is infinite and has an accumulation point z^*. Let K, a subset of $\{0,1,2,\ldots\}$, define a sequence such that $z_i \xrightarrow{K} z^*$. Suppose that $z^* \notin F$. By Proposition 4.4.5, $\tau_i \to 0$ and hence, there exists a $k_1 \in K$ such that $\tau_i \leq \tau(z^*)$ for all $i \geq k_1$. Let $k^* \geq k_1$ be such that $z_i \in B_{\varrho(z^*)}(z^*)$, for all $i \geq k^*$, and $i \in K$. As a result of Proposition 4.4.4, for $z \in F^c$ and for all $i \in \{0,1,\ldots\}$ we obtain

$$\psi(z_{i+1}) - \psi(z_i) \leq -\mu(z^*) \qquad (4.4.4)$$

for all $i \geq k^*$, and $i \in K$. Hence,

$$\lim_{\substack{i\to\infty \\ i,j\in K}} [\psi(z_j) - \psi(z_i)] \leq -\mu(z^*).$$

where i and j are successive numbers in K. But $\{\psi(z_i)\}_{i=0}^{\infty}$ is a monotonically decreasing sequence, bounded from below, it must converge, thus

$$\lim_{\substack{i\to\infty \\ i,j\in K}} [\psi(z_j) - \psi(z_i)] = 0$$

which contradicts (4.4.4). ∎

Since every accumulation point z^* of an infinite sequence $\{z_i\}$ generated by the algorithm satisfies $\psi(z^*) < 0$, then, there exists an i^*, large enough, such that for all $i \geq i^*$, $\psi(z_i) \leq 0$, i.e. the inequalities are solved in a finite number of iterations.

4.5 - SECANT METHOD

Gradient evaluations by finite difference method requires m(n+1) function evaluations. It may be desired to reduce the computation required for gradient approximation. Broyden [6] has derived a method for solving a system of equations. We shall use his method of gradient approximations.

Given Assumption 1 and an open convex set C such that for given z_i in C and $p \neq 0$, the vector $z_{i+1} = z_i + p_i$ belongs to C, let $G(z_{i+1})$ denote the Broyden approximation to $g_z(z_{i+1})$ given by

$$G(z_{i+1}) = G(z_i) + \frac{(y - G(z_i)s)s^T}{\langle s,s \rangle} \qquad (4.5.1)$$

where $s \triangleq z_{i+1} - z_i$ and $y \triangleq g(z_{i+1}) - g(z_i)$. For a full description of this method see [6,7]. Broyden's approximation can be carried out with m scalar function evaluations. The price paid is a reduction from quadratic to, probably, super-linear rate of convergence. The degree of approximation in this method increases as $\|z_{i+1}-z_i\|$ decreases. It is obvious

that if the initial point is far from the solution, Broyden's method may only give a good approximation for a few iterations. The gradient approximation must then be refined using finite difference method.

Algorithm 2

Data: $z_0 \in R^n$, $\varepsilon' \in (0,1)$, $L >> 1$, $\beta \in (0,1)$, $\tau_0 > 0$, $\lambda_{min} > 0$, isec > 1.

Step 0: Set $i = 1$, set $\varepsilon = \varepsilon' \psi(z_0)$.

Step 1: If $\psi(z_i) < 0$ stop.

Step 2: Find $\tau_i = \tau(z_i)$ the largest $\tau \in \{\tau_{i-1}, \tau_{i-1}/2, \ldots\}$ such that $\hat{\theta}^0_{\tau_i,\varepsilon}(z_i) < -\tau_i$.

Step 3: Compute

$$p_{\tau_i,\varepsilon}(z_i) = \operatorname{argmin}\{\|p\| \,\big|\, \hat{\psi}_{\tau_i}(z_i,p) < \hat{\psi}^0_{\tau_i,\varepsilon}(z_i)\}.$$

Step 4: Find, if possible, $\lambda_i = \lambda_{\tau_i,\varepsilon}(z_i)$ the largest number in S not less that $\lambda_{min}\tau_i$ such that

$$\psi(z_i+\lambda_i p_{\tau_i,\varepsilon}(z_i)) - \psi(z_i) < \lambda_i \hat{\theta}^0_{\tau_i,\varepsilon}(z_i)/2.$$

Else, if Broyden's approximation is used at this iteration, use the finite difference approximation and go to Step 2; if the finite difference approximation is used at this iteration, set $\tau_{i-1} = \tau_{i-1}/2$ and go to Step 2.

Step 5: Save z_i and $g(z_i)$, set $z_{i+1} = z_i + \lambda_i p_{\tau_i, \varepsilon}(z_i)$, set $i = i+1$. Use Broyden's approximation for `isec' consecutive iterations, then use the finite difference approximation for one iteration and go to Step 1. ▪

Since the finite difference approximation is used at least, at every (isec+1)th iteration, the algorithm's convergence property in a finite number of iterations is established by Theorem 4.4.1.

4.6 - SCALING

In the numerical approximation of the derivatives by finite difference it may be necessary to take special precautions to ensure sufficient accuracy. A method for choosing appropriate step length is proposed by Curtis and Ried [8]. In their method estimates of truncation and round-off errors are given by:

$$\Delta_t = \left| [g(z+h)-g(z-h)]/2h - [g(z+h)-g(z)]/h \right| \qquad (4.6.1)$$

$$\Delta_r = h \max\{ \left| [g(z+h)-g(z)]/h \right| , \left| [g(z+h)-g(z)]/h \right| \} \qquad (4.6.2)$$

The balance between Δ_t and Δ_r is maintained if $u \triangleq \Delta_t/\Delta_r$ is kept in the range of $[u_{min}, u_{max}]$. If $u \notin [u_{min}, u_{max}]$, then $h_{new} = h_{old}[\hat{u}/\max(u,1)]^{1/2}$, where $\hat{u}$ is chosen in the range

$[u_{min}, u_{max}]$. Finally, h is restricted to a range $[h_{min}, h_{max}]$.

4.7 - NUMERICAL EXAMPLES

The algorithms have been successfully applied to several test problems and incorporated in an interactive optimization based design package. To illustrate performace of the algorithms, we present three examples, in which, the following parameters are used.

$\beta = 0.1$, isec = 3, L = 1.e6, $\tau_0 = 1.e-4$.

EXAMPLE 1 [3]: The feasible set consists of a pair of squares of sides π centered at $(-\pi,0)$, and $(3\pi/2,0)$ and is defined by

$$\begin{aligned} \sin z_1 &\leq 0 \\ -\cos z_2 &\leq 0 \\ z_1 - 3\pi &\leq 0 \\ z_2 - \pi/2 &\leq 0 \\ -z_1 - \pi &\leq 0 \\ -z_2 - \pi/2 &\leq 0 \end{aligned}$$

Starting from z_0=(0, 75), Algorithm 1 located a feasible point in two iterations (z=(-1.56, 0.)), while Algorithm 2 found a solution in four iterations (z=(-0.417, -1.153)). In the first iteration the linearized feasible set was expanded

to ensure a solution. The algorithm given in [3], employing exact gradients, found a solution in four iterations.

The following two examples are minimization problems cited in [10]. We have employed the cost functions as additional constraints by perturbing the optimal cost to ensure that the feasible sets satisfy the assumptions of our algorithms.

EXAMPLE 2 (Colville's first problem): The minimum cost value is -33.87. We added an extra constraint i.e "cost + 33.80 $\leq$ 0" to the existing fifteen constraints with five variables. Algorithm 1 found a solution in eleven iterations while Algorithm 2 took only nine iterations to find a solution.

EXAMPLE 3 (Colville's third problem): The optimal value is -30665.538. We added the constraint "cost + 30665.5 $\leq$ 0" to the existing sixteen constraints with five variables. Both algorithms took three iterations to find a feasible solution.

4.8 - DISCUSSION

We have presented two new derivative free algorithms for solving a set of inequalities in a finite number of iterations. One of the algorithms presented uses a Broyden type approximation to the gradient matrix to improve the efficiency of the algorithm. If the algorithm is used interactively, designer has the additional flexibility of controlling whether the algorithm uses Broyden or the finite difference approximation of the derivatives.

4.9 - REFERENCES

[1] - Polak E., "Computational methods in optimization: A unified approach", Academic Press, N.Y., 1971.

[2] - Polak E., Mayne D.Q., "On finite solution of nonlinear inequalities",IEEE Trans. Automatic Control, AC-24, pp. 443-444, 1979.

[3] - Mayne D.Q., Polak E., Heunis A.J., "Solving nonlinear inequalities in a finite number of iterations", JOTA, Vol. 33, pp. 207-222, 1981.

[4] - Mayne D.Q., Sahba M., "An efficient algorithm for solving inequalities", Accepted for publication by JOTA, 1983.

[5] - Trahan R., Polak E., "A derivative free algorithm for a class of infinitely constrained problems", University of California, Berkeley, Electrocics Research Laboratory Memorandom UCB/ERL M78/75, 1977.

[6] - Broyden C.G., "A class of methods for solving nonlinear simultanious equations", Math. Comp., Vol 19, pp. 577-593, 1965.

[7] - Dennis Jr. J.E., More J.J., "Quasi-Newton methods, motivation and theory", SIAM Review, Vol 19, No. 1, pp. 46-87, 1977.

[8] - Curtis A.R., Reid J.K., "The choice of step lengths when using differences to approximate Jacobian matrices", Journal of Institute of Math. and Applic., Vol. 13, pp. 121-126, 1974.

[9] - Berge C., "Topological spaces", The McMillan Co., N.Y.,1962.

[10] - Himmelblau D.M., "Applied nonlinear programming", McGraw-Hill, N.Y. 1972.

[11] - Sahba M.,"A derivative free algorithm for solving inequalities in a finite number of iterations", Submitted for publication to Journal of Mathematical Programming Study.

CHAPTER 5

A GLOBALLY CONVERGENT ALGORITHM FOR NONLINEARLY CONSTRAINED OPTIMIZATION CONVERGENCE

5.1 - INTRODUCTION

In this chapter a new algorithm for the problem of minimizing a cost function subject to nonlinear equality and inequality constraints is presented and analysed. The problem considered is:

$$P: \quad \min \{ f(z) \mid g(z) \leq 0, \quad h(z) = 0 \} \qquad (5.1.1)$$

where $f:R^n \to R$, $g:R^n \to R^m$, $h:R^n \to R^r$. Several algorithms possessing a superlinear rate of convergence but which are only locally convergent are described in the literature. The exact penalty function technique was employed by several authors [1-5] for globally stabilizing these algorithms. In this class of algorithm the constrained optimization problem P is replaced by an unconstrained non-differentiable optimization problem. The search direction is determined by solving a first or second order approximation to the original problem and step length is then determined by approximately minimizing an exact penalty function. The main difficulties encountered in these methods [1],[5] are the choice of penalty parameter, the choice of step length, and the "Maratos effect"[3] (the exact penalty function step length procedure can truncate the step length near a solution, thus destroying

superlinear convergence). Mayne and Polak [5] have proposed an algorithm to overcome these problems. They utilize a quadratic approximations for determining the search direction but resort to a first order approximation if the Newton step is unsatisfactory; an exact penalty function for choosing the step length. When sufficiently close to a solution, an arc search vector is employed to avoid the "Maratos effect" and to achieve superlinear convergence. Their algorithm is relatively complex. Chamberlain et al [6] have proposed an algorithm based on Han's algorithm [1]. They employ the watchdog technique to avoid truncation of the step length near a solution; this technique allows, at some iterations, step lengths that are much longer than those that would be normally allowed by the line search objective function. This method is said to be effective, if employed in a controlled way. Powell's earlier algorithm [8] has, despite good numerical performance on some examples, several drawbacks: the procedure for updating the penalty parameter does not guarantee global convergence.(see [9] for a counter example) and unity step length in the neighbourhood of the solution (essential for superlinear convergence) is not ensured. Schittkowski [15] replaces the exact nondifferentiable penalty function of Powell[8] with a differentiable augmented Lagrange function. Under certain assumptions he proves convergence of this algorithm.

In this chapter we present an algorithm which avoids some of the complications involved in other algorithms; the algorithm employs a method for computing the penalty parameter which is

not based on the Lagrange multipliers. A procedure is then proposed for automatically increasing or decreasing the penalty parameter to ensure global convergence. Since the linearized feasible set is often empty, we propose to solve a linear program which ensures existence of a search vector. Global convergence of the algorithm using only linear search direction is established. Computational results suggest good numerical performance.

5.2 - DEFINITIONS

Let $L:R^n \times R^m \times R^r \rightarrow R$ denote the Lagrangian defined by:

$$L(z,\lambda,\mu) \triangleq f(z)+\langle\lambda,g(z)\rangle+\langle\mu,h(z)\rangle. \qquad (5.2.1)$$

Let $H \in R^{n\times n}$ denote a positive definite estimate of the Hessian $L_{zz}(z,\lambda,\mu)$. The exact penalty function employed $\gamma:R^n \times R \rightarrow R$ is defined by

$$\gamma(z,c) \triangleq f(z)+c\ \psi(z) \qquad (5.2.2)$$

where $\psi:R^n \rightarrow R$ is defined by

$$\psi(z) \triangleq \max\left\{g^j(z), j\in\underline{m}\ ;\left|h^j(z)\right|, j\in\underline{r};\ 0\right\} \qquad (5.2.3)$$

and where $\underline{m} \triangleq \{1,2,\ldots m\}$, $\underline{r} \triangleq \{1,2,\ldots r\}$.

The first order estimates $\hat{\gamma}(z,p,c)$ of $\gamma(z+p,c)$ and $\hat{\psi}(z,p)$ of

$\psi(z+p)$ are defined by

$$\hat{\gamma}(z,p,c)=f(z)+f_z(z)p+c\ \max\{\hat{\psi}(z,p);0\} \tag{5.2.4}$$

$$\hat{\psi}(z,p) \triangleq \max\left\{g^j(z)+g^j_z(z)p, j\in\underline{m}; \left|h^j(z)+h^j_z(z)p\right|, j\in\underline{r}\right\} \tag{5.2.5}$$

Let the function $\hat{\psi}^o:R^n \to R$ be defined by

$$\hat{\psi}^o(z) \triangleq \max\{\min[\hat{\psi}(z,p)\,|\,p\in\bar{P}]\ ;0\} \tag{5.2.6}$$

where

$$\bar{P} \triangleq \left\{p\in R^n \,\middle|\, \|p\|_\infty \leq J\right\} \tag{5.2.7}$$

where J is some suitably chosen large number. Notice that $0\leq \hat{\psi}^o(z) \leq \psi(z)$; if $\hat{\psi}^o(z)=\psi(z)$ and $\psi(z)>0$ then the feasible set of $\min\{\hat{\psi}(z,p)\,|\,p\in\bar{P}\}$ is empty. The search direction $p(z,H)$ is obtained by solving the following quadratic program:

$$QP(z,H):\quad \min\left\{f_z(z)p+0.5p^THp \,\middle|\, \hat{\psi}(z,p) \leq \hat{\psi}^o(z), p\in\bar{P}\right\} \tag{5.2.8}$$

where H is a positive definite matrix. The set of desirable points D is defined by

$$D \triangleq \left\{z\in R^n \,\middle|\, (z,\lambda,\mu)\text{is a Kuhn Tucker triple for P}\right\} \tag{5.2.9}$$

Let $T:R^n\times R^m\times R^r \to R^+$ denote a function satisfying $T(z,\lambda,\mu)=0$

if (z,λ,μ) is a Kuhn Tucker triple for P. A simple and suitable function is:

$$T(z,\lambda,\mu) \triangleq \psi(z)+\|\nabla f(z)+g_z^T(z)\lambda+h_z^T(z)\mu\|^2. \qquad (5.2.10)$$

The penalty parameter c defined in this chapter must satisfy

$$f_z(z)p+c\,[\hat{\psi}^o(z)-\psi(z)] \leq -[\hat{\psi}^o(z)-\psi(z)]^2$$

which is equivalent, when $\hat{\psi}^o(z)\neq\psi(z)$, to

$$c \geq \psi(z)-\hat{\psi}^o(z)+[f_z(z)p]/[\psi(z)-\hat{\psi}^o(z)]. \qquad (5.2.11)$$

Let $\theta:R^n\times R^n\times R \to R$ be defined by

$$\theta(z,p,c) \triangleq \hat{\gamma}(z,p,c)-\gamma(z,c)$$

$$=f_z(z)p+c\,[\max(\hat{\psi}(z,p)\ ;0)-\psi(z)] \qquad (5.2.12)$$

so that $\theta(z,p,c)$ is a first order estimate of $\gamma(z+p,c)-\gamma(z,c)$; p is a descent direction for $\gamma(z,c)$ if $\theta(z,p,c) < 0$.

Let $\theta_c^o(.,H):R^n \to R$ be defined by

$$\theta_c^o(z,H) \triangleq \theta(z,p(z,H),c)$$

$$= f_z(z)p(z,H) + c\,[\hat{\psi}^o(z)-\psi(z)] \qquad (5.2.13)$$

If $[\hat{\psi}^o(z)-\psi(z)]$ is negative, a positive c always exists such that $\theta_c^o(z,H) \leq -[\hat{\psi}^o(z)-\psi(z)]^2 < 0$, i.e. such that p(z,H) is a descent direction for $\gamma(z,c)$. However, if $z \in F$, then $\theta_c^o(z,H)=f_z(z)p(z,H) < 0$ (see Proposition 5.6.2) i.e. p(z,H) is a descent direction for $\gamma(z,c)$ for all $c>0$. If $\theta_c^o(z,H)=0$, for any c satisfying (5.2.11), then $z \in D$ (see Proposition 5.6.1).

Finally, let F, the set of feasible points, and F^c, the complement of F, be defined by:

$$F \triangleq \{x \mid \psi(x) = 0\} \tag{5.2.14}$$

$$F^c \triangleq \{x \mid \psi(x) > 0\} \tag{5.2.15}$$

5.3 - THE PENALTY PARAMETER

The constrained optimization problem P can be replaced by the equivalent unconstrained optimization problem P_c defined by:

$$P_c: \quad \min\left[\gamma(z,c) \mid z \in R^n\right] \tag{5.3.1}$$

for c large enough. Let function $\hat{c}: F^c \to R$ be defined by

$$\hat{c}(z) \triangleq \psi(z)-\hat{\psi}^o(z) + \frac{f_z(z)p(z,H)}{\psi(z)-\hat{\psi}^o(z)} \tag{5.3.2}$$

Since $\psi(z)-\hat{\psi}^o(z)>0$ for all $z \in F^c$ (this is implied by assump-

tion H2), $\hat{c}:F^C \to R$ is well defined. Our test function $\bar{c}:R^n \to R$ is defined by:

$$\bar{c}(z) \triangleq \begin{cases} \max\{\hat{c}(z), 0\} & \text{if } \psi(z) > 0 \\ \tilde{c} & \text{otherwise} \end{cases} \qquad (5.3.3)$$

where $\tilde{c}>0$. Powell [8] notes that a choice of c which is too large, may be inefficient, because too much weight is given to satisfying the constraints. Existing procedures (with guaranteed convergence) require that $c_i \to \infty$ if c_i is changed infinitely often[5] and usually require that c_i must be non-decreasing. In this chapter we propose a method for choosing c_i in which $c_i \to \infty$ only if c_i is changed infinitely often, but c_i may remain constant or even decrease if certain tests are satisfied. A procedure for choosing c_i is given below

PROCEDURE FOR CHOOSING c_i:

Step 1: Compute $\bar{c}(z_i)$.

Step 2: (test a) If $\bar{c}(z_i) < c_{i-1}$ and $\psi(z_i) < \psi_{old} - \varepsilon$, set $c_i = \max\{ \bar{c}(z_i) , \tilde{c} \}$ and set $\psi_{old} = \psi(z_i)$. Return to main algorithm.

Step 3: (test b) If $\bar{c}(z_i) > c_{i-1}$ (and $\psi(z_i) > 0$), set $c_i = \max\{\bar{c}(z_i), (c_{i-1}+\delta)\}$. Return to main algorithm.

Step 4: (test c) If $\bar{c}(z_i) < c_{i-1}$ or $\psi(z_i) = 0$, set $c_i=c_{i-1}$. Return to main algorithm.

5.4 - THE STEP LENGTH

The choice of step length is important, since it forces convergence from any starting point and an asymptotic step length of unity is necessary for superlinear convergence of the algorithm. Since γ is not differentiable, the standard Armijo test, can not be employed. We shall use an Armijo like test, similar to that of [5], which compares the actual change in $\gamma(z,c)$ with its first order estimate $\hat{\gamma}(z,p,c)$. Instead of $\alpha\gamma_z(z,c)p$ in the Armijo test we use the estimate $\theta(z,\alpha p,c)$ of $\gamma(z+\alpha p,c)-\gamma(z,c)$. Since $\theta(z,\alpha p,c)\leq\alpha\theta(z,p,c)$ for all $\alpha\in[0,1]$, the modified Armijo test is given by

$$\gamma(z+\alpha p,c) - \gamma(z,c) \leq \tau\alpha\theta(z,p,c) \qquad (5.4.1)$$

The step length is chosen to be the largest α in the set $S \triangleq \{\beta,\beta^2,\ldots\}$ satisfying (5.4.1), where $\beta\in(0,1)$ and $\tau\in(0,1)$; a sensible choice for τ is $\tau=0.1$ or $\tau=0.05$.

5.5 - ALGORITHM

We can now state our algorithm.

Main Algorithm.

Data: $z_1\in R^n$, $H_1=I_n$, $J>>1$ (e.g. J=1.E+6), $\beta\in(0,1)$ (e.g. $\beta=0.1$), $\tau\in(0.,0.25)$ (e.g. $\tau=0.05$), $\tilde{c} > 0$ (e.g. $\tilde{c}=1.$), $\delta > 0$ (e.g. $\delta=0.01$), $\varepsilon > 0$ (e.g. $\varepsilon=0.01\psi(z_0)$).

Step0: Set i=1, $c_0 = \tilde{c}$ and $\psi_{old} = \psi(z_0)$.

Step1: Compute $\hat{\psi}^o(z_i)$.

Step2: Compute $p_i=p(z_i,H_i)$ the minimum norm solution of $QP(z_i,H_i)$.

Step3: Compute c_i using the given procedure.

Step4: Compute $\alpha_i=\alpha_c(z_i)$ the largest α in S such that

$$\gamma(z_i+\alpha_i p_i,c_i) - \gamma(z_i,c_i) \leq \tau\alpha_i\theta_c^o(z_i,H_i).$$

Step5: Set $z_{i+1}=z_i+\alpha_i p_i$. Update H_i to $H_{i+1}>0$. Set i = i+1, and go to Step1. ▪

5.6 - GLOBAL CONVERGENCE

For all $z\in R^n$, let

$$I(z) \triangleq \{j\in\underline{m} \mid g^j(z) = \psi(z)\} \qquad (5.6.1)$$

and

$$E(z) \triangleq \{j\in\underline{r} \mid h^j(z) = \psi(z)\} \qquad (5.6.2)$$

We make the following assumptions.

H1: the functions f, g, and h are continuously differentiable.

H2: For all z the vectors $\{\nabla g^j(z), j\in I(z), \nabla h^j(z), j\in E(z)\}$ are positive linearly independent.

H3: The sequences $\{z_i\}$ and $\{H_i\}$ generated by the algorithm are bounded.

These assumptions are reasonable; in particular, H2 can not be further relaxed without loosing the ability of the algorithm to find a feasible point.

To proceed we need to define the set D_c

$$D_c \triangleq \{z \in R^n \mid \theta_c^o(z,H) = 0\}. \qquad (5.6.3)$$

Note that D_c is not necessarily the set of locally optimal points for P_c.

PROPOSITION 5.6.1.

For all $c \geq \bar{c}(z)$ and all $H>0$, $z \in D$ if and only if $z \in D_c$.

PROOF: Let $c \geq \bar{c}(z)$ and $z \in D_c$. Suppose that $\psi(z)>0$, then $c \geq \bar{c}(z) \geq \hat{c}(z)$ implies that

$$\theta_c^o(z,H) = f_z(z)p(z,H)+c\,[\hat{\psi}^o(z)-\psi(z)] < 0$$

which contradicts $z \in D_c$. Hence $\psi(z) = 0$ which implies $g(z) \leq \underline{0}$ and $h(z)=\underline{0}$. Since $\theta_c^o(z,H)=0$, and $\psi(z)=\hat{\psi}^o(z)=0$ then, $f_z(z)p(z,H) = 0$. From QP(z,H) we obtain

$$f_z(z)p(z,H)+0.5p^T(z,H)Hp(z,H)=0,$$

since H is positive definite, $p(z,H)=0$. From the dual problem of $QP(z,H)$ there exist $\lambda \in R^n$, and $\mu \in R^r$ such that

$$\nabla f(z)+g_z^T(z)\lambda+h_z^T(z)\mu=0$$

$$g^T(z)\lambda=0$$

$$\lambda \geqslant 0$$

which imply $z \in D$.

Now, let $z \in D$ which implies that there exist $\lambda \in R^m$ and $\mu \in R^r$ such that

$$g(z) \leqslant \underline{0}, \quad h(z)=\underline{0}, \quad \lambda \geqslant \underline{0}, \quad \lambda^T g(z)=0, \text{and}$$

$$\nabla f(z) + g_z^T(z)\lambda + h_z^T(z)\mu = 0 \qquad (5.6.4)$$

Let p be the unique solution of $QP(z,H)$. From (5.6.4) and $z \in F$ we obtain

$$f_z(z)p = p^T \nabla f(z)$$

$$= - p^T g_z^T(z)\lambda - p^T h_z^T(z)\mu$$

But $\lambda^T(g(z)+g_z(z)p)=0$ and $h(z)+h_z(z)p=\underline{0}$, hence

$$f_z(z)p = \lambda^T g(z) + \mu^T h(z) = 0.$$

Since $\psi(z)=0$ and $f_z(z)p(z,H)=0$ imply that $\theta_c^o(z,H) = 0$, we conclude that $z\in D_c$. ∎

PROPOSITION 5.6.2.

i) $\hat{\psi}^o:R^n \to R$ is continuous.

ii) For all $z\in F$, $z\notin D$, and $c\in R^n$, $\theta_c^o(z,H)<0$

iii) For all $H > 0$, the function $p(.,H):R^n \to R^n$ is continuous

iv) $\bar{c}:F^c \to R$ is continuous.

v) For all $c\geq\bar{c}(z)$ and $H > 0$, $\theta_c^o(.,H):R^n \to R$ is continuous.

PROOF: i) the proof is similar to that of Proposition 3.3.2.

ii) Let $z\in F$, $z\notin D$, and $c\in R^n$. Let (p,λ,μ) be a Kuhn-Tucker triple for $QP(z,H)$, satisfying the following conditions:

$$\nabla f(z)+Hp+g_z^T(z)\lambda+h_z^T(z)\mu = 0 \qquad (5.6.5)$$

$$h(z)+h_z(z)p = 0 \qquad (5.6.6)$$

$$g(z)+g_z(z)p \leq 0 \qquad (5.6.7)$$

$$\lambda \geq 0$$

$$\lambda^T(g(z)+g_z(z)p) = 0. \qquad (5.6.8)$$

We also have

$$\theta_c^o(z,H) = f_z(z)p \qquad (5.6.9)$$

or from eqn (5.6.5)

$$f_z(z)p = p^T \nabla f(z)$$

$$= -p^T H p - p^T g_z^T(z)\lambda - p^T h_z^T(z)\mu$$

$$= -p^T H p + \lambda^T g(z) + \mu^T h(z). \qquad (5.6.10)$$

But $h(z) = \underline{0}$, $g(z) \leq \underline{0}$ and $\lambda \geq 0$, hence, $\theta_c^o(z,H) \leq 0$.

iii) Let $H > 0$. We note (see [1]) that (5.2.8) is equivalent to:

$$q(z,H)=\min\left\{ f_z(z)p+0.5p^T Hp+d[\hat{\psi}(z,p)-\hat{\psi}^o(z)]_+ \;\middle|\; p\in\bar{P} \right\} \qquad (5.6.11)$$

for d sufficiently large (since $[\hat{\psi}(z,p)-\hat{\psi}^o(z)]_+$ is convex and $f_z(z)p+0.5p^T Hp$ is strictly convex, $d>0$ exists such that (5.2.8) and (5.6.11) are equivalent). From [11,pp 115, 116] $q(z,H)$ is continuous and solution set is upper semi-continuous. Since (5.6.11) has a unique solution, the solution set contains a single element $p(z,H)$. It, then, follows [11,pp. 117] that $p(.,H)$ is continuous.

iv) Follows from continuity of ψ, $\hat{\psi}^o$, $p(.,H)$, and ∇f.

v) Follows from continuity of ψ, $\hat{\psi}^o$, $p(.,H)$, $\bar{c}$, and ∇f. ▪

Steps 1 to 4 of the algorithm define a map $A_c(.,H):R^n \to R^n$;

$$A_c(z,H) \triangleq z+\alpha_c(z)p(z,H) \qquad (5.6.12)$$

Since $p(z,H)$ always exists and is unique, A_c is well defined.

PROPOSITION 5.6.3.

i) For all $\eta>0$, all $z\in R^n$, and $c\in R^n$, there exists a $\delta>0$ such that

$$\left|\gamma(z'+p',c)-\hat{\gamma}(z',p',c)\right| \leq \eta\|p'\| \qquad (5.6.13)$$

for all $z'\in B_\delta(z)$, all $p'\in B_\delta(0)$.

ii) For all $z\notin D_c$, all $H>0$, and all $c\geq\bar{c}(z)$, there exist a $\varrho_1>0$ and a $\delta_1>0$ such that

$$\gamma(A_c(z',H),c)-\gamma(z',c) \leq -\varrho_1 \qquad (5.6.14)$$

for all $z'\in B_{\delta_1}(z)$.

iii) For all $c\geq\bar{c}(z)$, $z\in D_c$ is a necessary condition of optimality for P_c.

PROOF: i) Let $c\in R^n$, then

$$\left|\gamma(z'+p',c)-\hat{\gamma}(z',p',c)\right| \leq \left|f(z'+p')-\hat{f}(z',p')\right| +$$

$$c \max\{\left|g^j(z'+p')-\hat{g}^j(z',p')\right|, j\in\underline{m};$$

$$\left|h^j(z'+p')-\hat{h}^j(z',p')\right|, j\in\underline{r}\} \qquad (5.6.15)$$

But

$$g^j(z'+p')-\hat{g}^j(z',p') = \left[\int_0^1 (g_z^j(z'+tp')-g_z^j(z'))dt\right]p' \qquad (5.6.16)$$

so that

$$\left|g^j(z'+p')-\hat{g}^j(z',p')\right| \leq \left[\int_0^1 \|g_z^j(z'+tp')-g_z^j(z')\|dt\right]\|p'\| \qquad (5.6.17)$$

Similarly,

$$\left|f(z'+p')-\hat{f}(z',p')\right| \leq \left[\int_0^1 \left|f_z(z'+tp')-f_z(z')\right|dt\right]\|p'\| \qquad (5.6.18)$$

and

$$\left|h^j(z'+p')-\hat{h}^j(z',p')\right| \leq \left[\int_0^1 \|h_z^j(z'+tp')-h_z^j(z')\|dt\right]\|p'\| \qquad (5.6.19)$$

Since f_z, g_z^j, and h_z^j are uniformly continuous in any compact set and since $z'\in B_\delta(z)$ and $p'\in B_\delta(0)$ imply that $z'+tp'\in B_{2\delta}(z)$, for all $t\in[0,1]$ it follows that, for all $\eta>0$, δ can be chosen so that

$$\left|\gamma(z'+p',c)-\hat{\gamma}(z',p',c)\right| \leq \eta\|p'\| \qquad (5.6.20)$$

for all $z'\in B_\delta(z)$ and all $p'\in B_\delta(0)$.

ii) let $z\notin D_c$, $H>0$, $c\geq\bar{c}(z)$. From (5.2.12) and since $\hat{\gamma}(z,\alpha_1 p,c)$ is convex in α_1 (for all z, p, and c), we obtain

$$\hat{\gamma}(z',\alpha_1 p',c)-\gamma(z',c) \leq \alpha_1\theta_c^o(z',H) \qquad (5.6.21)$$

for all $\alpha_1\in[0,1]$ and $p' \triangleq p(z',H)$. Since $\theta_c^o(.,H)$ is continuous, there exists $\delta>0$ such that $\theta_c^o(z',H)\in[(3/2)\theta_c^o(z,H),(1/2)\theta_c^o(z,H)]$, for all $z'\in B_\delta(z)$. From (i), there exists a $\delta_1\in(0,\delta]$ such that

$$\left|\gamma(z'+\alpha_1 p',c)-\hat{\gamma}(z',\alpha_1 p',c)\right| \leq (-\theta_c^o(z,H)/4)\alpha_1 \qquad (5.6.22)$$

for all $z'\in B_{\delta_1}(z)$, all $p\in\bar{P}$, all $\alpha_1\in[0,\delta_1/J]$ (so that $\alpha_1 p\in B_{\delta_1}(0)$ for all $p\in\bar{P}$). Hence, for all $z'\in B_{\delta_1}(z)$, and all $\alpha_1\in[0,\delta_1/J]$,

$$\gamma(z'+\alpha_1 p',c)-\gamma(z',c) \leq \alpha_1[\theta_c^o(z',H)-\theta_c^o(z,H)/4]$$

$$\leq \alpha_1\theta_c^o(z,H)/4. \qquad (5.6.23)$$

Now, for all $\alpha_c(z) \geq \alpha_1\in[0,\delta_1/J]$, we obtain

$$\gamma(A_c(z',H),c)-\gamma(z',c) \leq -\varrho_1 \qquad (5.6.24)$$

for all $z' \in B_{\delta_1}(z)$ and $\varrho_1 \triangleq -\alpha_c(z)\theta_c^o(z,H)/4$.

iii) Follows from (ii). ▪

PROPOSITION 5.6.4.

Any accumulation point z^* of an infinite sequence $\{z_i\}$ generated by the main algorithm, where $z_{i+1}=A_c(z_i,H_i)$, $H_i>0$ and $c>c_i$, for all i satisfies $z^* \in D$.

PROOF: Let $c>c_i$ for all i, where c_i is obtained from the procedure for choosing c_i. From Proposition 5.6.3(ii) and Theorem (1.3.3) of [12] we obtain that $z^* \in D_c$. If z^* is in F, then from the procedure for choosing c_i, $c_i>\max\{\bar{c}(z_i),\tilde{c}\}$ for all i (this is obvious in Step 2 of the procedure for choosing c_i and can be easily deduced from Steps 3 and 4), hence $c>\bar{c}(z^*)=\tilde{c}$. It then follows from Proposition 5.6.1 that $z^* \in D$.

Suppose z^* is in F^c, since $z_i \overset{X}{\to} z^*$, $X \subset \{1,2,\ldots\}$ and since $\bar{c}:F^c \to R$ is continuous, it follows that $\bar{c}(z_i) \overset{X}{\to} \bar{c}(z^*)$. Hence $c > \bar{c}(z_i)$ for all i implies that $c \geq \bar{c}(z^*)$. This implies that $\theta_c^o(z^*)<0$ which is a contradiction (since $z^* \in D_c$ implies that $\theta_c^o(z^*) = 0$) and we conclude that z^* is in F and, hence, in F and, hence, in D. ▪

THEOREM 5.1.

Any accumulation point z^* of an infinite sequence $\{z_i\}$ generated by the algorithm satisfies $z^* \in D$.

PROOF: We consider the following possible cases:

Case a (satisfaction, infinitely often, of "test a" in the procedure for choosing c_i) Suppose that there exists an infinite subset of $X_+ \triangleq \{0,1,..\}$, say K, such that test (a) is satisfied. The fact that $\psi(z_{i+1}) \leq \psi(z_i) - \varepsilon$ for all i such that "test a" is satisfied; contradicts the fact that $\psi(z_i) \geq 0$ for all i. Hence, "test a" cannot be satisfied infinitely often. Hence, there exist an i_0 (i_0 is the last element in K) such that for all $i \geq i_0$, c_i is either increased (case b) or kept constant (case c).

Case b (satisfaction, infinitely often, of "test b") Suppose that there exists an infinite subset of X_+, say M, such that $\{c_i\}_{i \in M}$ is a monotonically increasing sequence and $z_i \overset{M}{\to} z^*$. Notice that for all $i \in M$, z_i is in F^c. However, there exists an i_0 large enough such that for all $i \geq i_0$ and $i \in M$, z_i is in N, where N is a small compact neighbourhood of z^*. Since $\bar{c}: F^c \to R$ is continuous, $\bar{c}(z_i) \overset{M}{\to} \bar{c}(z^*)$. However, c_i, when increased, is increased by an amount not less than δ, so that eventually, c_i remains constant. This is a contradiction i.e. "test b" cannot be satisfied infinitely often. The proof,

then, follows from (case c).

Case c (satisfaction, infinitely often, of "test c") This implies that there exists an i_0 such that for all $i \geq i_0$, c_i remains constant. Then, from Proposition 5.6.4, any accumulation point z^* of an infinite sequence $\{z_i\}_{i=i_0}^{\infty}$ satisfies $z^* \in D$.■

5.7 - NUMERICAL RESULTS

The algorithm has been successfully applied to several test problems. We have employed the BFGS procedure with Powell's modification to update the estimate of the Hessian. To illustrate the procedure for choosing c, let us consider the following simple example:

$$\text{minimize} \quad z_1^2+z_2^2$$

$$\begin{aligned} \text{s.t.} \quad \sin(z_1) &\leq 0, \\ -\cos(z_1) &\leq 0, \\ z_1^2+z_2^2-\pi/2 &\leq 0, \\ -z_1-\pi &\leq 0, \\ -z_2-\pi/2 &\leq 0. \end{aligned}$$

i	z_1	z_2	$\hat{\psi}^o(z_i)$	cold	$\bar{c}(z_i)$	c_i	α_i	$T(z_i)$
1	-1.	75.	1.34E1	1.E6	5.61E3	5.61E3	1	5.43E6
2	2.54E1	3.79E1	0.	5.61E3	2.08E3	2.08E3	1	1.24E4
3	2.51E1	-1.57	0.	2.08E3	6.33E2	6.33E2	1	2.73E3
4	-3.14	-0.74	0.	6.33E2	9.47	9.48	1	5.5E-3
5	9.8E-4	3.3E-2	0.	9.48	-3.38	1.	1	5.8E-6
6	-3.E-10	-1.1E-3	0.	1.	F	1.	1	3.E-14
7	-8.8E-8	2.5E-9	0.	1.	F	1.	1	7.E-29

Table 1.

The optimal point is $z^* = (0, 0)$, and the starting point $z_0 = (-1, 75)$ is chosen. Table 1 shows the numerical results computed by the algorithm. Notice that in the first iteration the linearized feasible set is expanded to ensure the existence of the Newton step. It can be seen that c is decreased in the first four iterations and then remained constant at its lower bound ($\tilde{c}$=1.).

The second problem considered is that of Chamberlain[9]:

$$\text{minimize} \quad z_2$$

$$\text{s.t.} \quad -2z_1^2+z_1^3+z_2 \geq 0,$$
$$-2(1-z_1)^2+(1-z_1)^3+z_2 \geq 0.$$

Chamberlain has shown that when starting from $z_0 = (0, 0)$, Powell's algorithm [8] cycles between z_0 and $z_1 = (1, 0)$. This occurs because of the updating scheme for penalty parameter in [8]. Table 3 illustrates the computation results.

i	z_1	z_2	c_i	α_i	$T(z_i)$
1	0.	0.	2.01	0.4	2.246
2	0.4	0	2.248	1.	0.159
3	0.501	0.379	1.733	1.	2.38E-6
4	0.5	0.375	2.003	1.	6.91E-14

Table 3.

Other interesting problems cited in the literature are Colville's problems[16]. Starting from the initial points suggested by the author fast convergence is obtained and displayed in Table 4.

Problem	number of constraints	number of variables	number of iteration	$T(z^*)$
Colville 1	15	5	8	1.457e-6
Colville 2	20	15	16	2.13e-6
Colville 3	16	5	3	7.742e-11

Table 4.

5.8 - DISCUSSION

The new algorithm presented is, under mild assumptions, globally convergent. In an attempt to solve some of the outstanding problems in optimization algorithm, several features are employed (in the algorithm) which are not present in previous algorithms. All the algorithms (known by the author), at best, assume that the columns of the gradient matrix of the most active constraints are linearly independent. We have relaxed this assumption to positive linear independence, and by employing a linear program (in Step 1 of the algorithm), a feasible solution to the Newton step is ensured, A fixed positive definite matrix (e.g. $H_i=I$, for all i) can replace the estimate of the Hessian, in which case assumption H3 can be further relaxed. An interesting feature of the algorithm is the new rule for choosing the penalty parameter c, which allows c to decrease. This permits high initial values and low final values. However, to establish superlinear rate of convergence of the algorithm, we need to show that step length of unity can be achieved in a neighbourhood of the solution. This has not been the aim of this project; interested readers are refered to an interesting work by Pantoja [18].

5.9 - REFERENCES

[1] - Han S.P., "A globally convergent method for nonlinear programming", JOTA 22 pp. 297-309, 1977.

[2] - Mayne D.Q., Maratos N., "A first order exact penalty function algorithm for equality constrained optimization problems", Math. Prog. Study,16 pp. 303-309,1979.

[3] - Maratos N., "Exact penalty function algorithm for finite dimensional and optimization problems", Ph.D. Thesis, Imperial College, University of London, 1978.

[4] - Powell M.J.D., "A fast algorithm for nonlinearly constrained optimizations" , in: G.A. Watson, ed., Numerical analysis, Lecture notes in Mathematics, vol 630 (Springer-Verlay, Berlin) pp. 144-157, 1978.

[5] - Mayne D.Q., Polak E., "A Superlinearly convergent algorithm for constrained optimization problems", Math. Prog. Study, 16 pp. 45-61, 1982.

[6] - Chamberlain R.M., Powell M.J.D., Lemarechal C., Pederson H.C., "The wacthdog technique for forcing convergence in algorithm for constrained minimization", Math. Prog. Study, 16 pp. 1-17, 1982.

[7] - Pietrzykowski T., "An exact potential method for constrained maxima", SIAM Journal of Numerical Analysis 6, pp. 229-303, 1969.

[8] - Powell M.J.D., "A fast algorithm for nonlinearly constrained optimization calculations", Dundee Conference on Numerical Analysis, 1977.

[9] - Chamberlain R.M., "Some examples of cycling in variable metric methods for constrained minimization", Math. Prog. Study,16 pp. 378-384, 1979.

[10] - Mayne D.Q., Sahba M., "An efficient algorithm for solving inequalities", Accepted for publication by JOTA, 1983.

[11] - Berge C., "Topological spaces", The McMillan Co., N.Y.,1962.

[12] - Polak E., "Computational methods in optimization: A unified approach", Academic Press, N.Y., 1971.

[13] - Robinson S.M., "Perturbed Kuhn-Tucker points and rates of convergence for a class of nonlinear programming algorithms", Math. Prog. Study, 7 pp. 1-16 ,1974.

[14] - Garcia-Palomares U.M., Mangasarian O.L., "Superlinearly convergent quasi-Newton algorithms for nonlinearly constrained optimization problems", Math. Prog. Study, 11, pp. 1-13, 1976.

[15] - Schittkowski K., "The nonlinear programming method of Wilson, Han, and Powell with an augmented Lagrangian type line search function, part 1", Numer. Math. 38, pp. 38-83, 1981.

[16] - Himmelblau D.M., "Applied nonlinear programming", McGraw-Hill, N.Y. 1972.

[17] - Fiacco A.V., McCormic G.P.,"Nonlinear programming: Sequential unconstrained minimization Techniques", John Wiley and Sons, U.S.A., 1968.

[18] - Pantoja J.F.D.,"Algorithms for constrained optimization problems", PhD Thesis, Imperial College, 1983.

CHAPTER 6

DESIGN OF OPTIMAL CONTROLLERS FOR A DOUBLE INVERTED PENDULUM

6.1 - INTRODUCTION

In this chapter the design methodology of Chapter 2, embedded in an interactive CAD package SIMNON, is employed to design a stabilizing controller for a double inverted pendulum [1,2]. While the double inverted pendulum has been well studied, it, nevertheless, presents an interesting and challenging control problem. Strugen and Loscutoff [3] have treated this problem using state space approach. They considered a linear model and designed a feedback controller to stabilize the pendulum at upright position using a full order observer. K. Furuta, T. Okulani and H sone [9] considered the same linear system and designed a state feedback optimal controller by minimizing a quadratic criterion function. They used a first order functional observer. These controllers do not work satisfactory when

(a) the system state is outside a small neighbourhood of the equilibrium state
(b) the pendulum system is not identified accurately
(c) a disturbance arising, for example, from inclination of the rail exists.

A controller effective for the latter two cases, must include an integrator to accommodate constant disturbances, and

is called a servo-controller. K. Furuta, H Kajiwara and K. Kosuge [6] used a servo-controller to stabilize a double inverted pendulum on an inclined rail at upright position and at a given reference point. We shall show how our multipurpose CAD package SIMNON can be used to design an optimal controller for the nonlinear model of a double inverted pendulum. SIMNON is an interactive simulation package written in Fortran [10], which includes the optimization algorithm [8] of Chapter 3 as a subsystem. It provides graphics facilities to monitor progression of the design and the user can interrupt the task, if he wishes, to change the design parameters and continue without losing any computation prior to the interruption of the task.

The following notation is used:

e = input voltage to the amplifier, V

e_0 = constant parameter, 0.3 V

r = position of the cart, m

r_0 = constant, 0.5 m

θ_1 = angular position of lower pendulum, rad

θ_{10} = constant, $\pi/3$ rad

θ_2 = angular position of upper pendulum, rad

θ_{20} = constant, $\pi/3$ rad

M = equivalent mass of the cart-drive system, 0.574 kg

m_1 = mass of lower pendulum, 0.103 kg

m_2 = mass of upper pendulum, 0.07 kg

F = equivalent friction constant of the cart-drive system, 2.81 kg/s

a = gain of overall cart driving system and the amplifier, 46.7 N/V

c_1 = friction constant of lower pendulum, 1.92×10^{-3} kgm^2/s

c_2 = friction constant of upper pendulum, 8.93×10^{-4} kgm^2/s

α = angle of the inclined rail, rad

l_1 = distance from pin of cart and pendulum to center of gravity of lower pendulum, 0.225 m

l_2 = distance from pin of upper and lower pendulum to center of gravity of upper pendulum, 0.177 m

L = length of lower pendulum, 0.379 m

J_1 = mass moment of inertia of lower pendulum, 2.386×10^{-3} kgm^2

J_1 = mass moment of inertia of upper pendulum, 1.521×10^{-3} kgm^2

g = acceleration due to gravity, 9.8 m/s^2.

6.2 - PHYSICAL SYSTEM AND MATHEMATICAL MODEL

A double inverted pendulum consists of two rods (Fig. 6.1), the lower pendulum is hinged to the cart whose motion is restricted to the vertical plane containing the line of the rail. The upper pendulum is connected to the lower pendulum in the same way. Since we do not have experimental apparatus we use the model and specifications of a double pendulum system given by K. Furuta et al [6]. The cart moves on a rigid length of 1.0m. The cart driving system consists of an 80 W DC motor, a pulley and a belt transmission system using a timing belt and a DC power amplifier. The mathematical model

of the system is constructed under the following assumptions [9]:

(a) Each pendulum is a rigid body.
(b) Length of the belt does not change.
(c) Driving force to the cart is directly applied to the cart without delay, and is proportional to the input to the amplifier.
(d) Friction forces against the motion of the cart, and those generated at connecting hinges of the pendulum, are proportional to the velocities of the upper and lower pendulum, respectively.

According to the above assumptions, a mathematical model can be derived based on the Lagrange equation (see [6]) and is given by

$$K_1 \ddot{x}_0 = K_2 \dot{x}_0 + K_3 + K_4 u + K_5 \sin\alpha \qquad (6.2.1)$$

where $x_0 \triangleq [r, \theta_1, \theta_2]^T$, $u \triangleq e/e_0$

and

$$K_1 = \begin{bmatrix} m_1+m_2+M & (m_1 l_1+m_2 L)\cos(\alpha+\theta_1) & m_2 l_2 \cos(\alpha+\theta_2) \\ (m_1 l_1+m_2 L)\cos(\alpha+\theta_1) & J_1+m_1 l_1^2+m_2 L^2 & m_2 l_2 L\cos(\theta_1-\theta_2) \\ m_2 l_2 \cos(\alpha+\theta 2) & m_2 l_2 L\cos(\theta_1-\theta_2) & J_2+m_2 l_2^2 \end{bmatrix} \qquad (6.2.2)$$

$$K_2=\begin{bmatrix} -F & (m_1l_1+m_2L)\sin(\alpha+\theta_1)\dot{\theta}_1 & m_2l_2\sin(\alpha+\theta_2)\dot{\theta}_2 \\ 0 & -c_1-c_2 & m_2l_2L\sin(\theta_1-\theta_2)\dot{\theta}_2+c_2 \\ 0 & m_2l_2L\sin(\theta_1-\theta_2)\dot{\theta}_1+c_2 & -c_2 \end{bmatrix} \qquad (6.2.3)$$

$$K_3 = \begin{bmatrix} 0 \\ (m_1l_1 + m_2L)g \sin \theta_1 \\ m_2l_2\sin \theta_2 \end{bmatrix} \qquad (6.2.4)$$

$$K_4 = \begin{bmatrix} e_0a \\ 0 \\ 0 \end{bmatrix} \qquad (6.2.5)$$

$$K_5 = \begin{bmatrix} -(m_1 + m_2 + M)g \\ 0 \\ 0 \end{bmatrix} \qquad (6.2.6)$$

where e is the input voltage to the amplifier satisfying

$$|e| < e_0 \qquad (6.2.7)$$

Let

$$x \triangleq \begin{bmatrix} x_0 \\ \dot{x}_0 \end{bmatrix} \qquad (6.2.8)$$

The mathematical model of the double inverted pendulum is then given by

$$\dot{x} = \begin{bmatrix} \dot{x}_0 \\ K_1^{-1}(K_2\dot{x}_0 + K_3 + K_4u + K_5\sin \alpha \end{bmatrix} \qquad (6.2.9)$$

In the neighbourhood of x = 0, the following linear model can be derived:

$$\dot{x} = \bar{A}x + \bar{b}u + \bar{h}d \qquad (6.2.10)$$

where d = sin α

$$\bar{A} = \begin{bmatrix} 0_{3\times3} & I_3 \\ L_1^{-1}[L_{21} : L_{22}] \end{bmatrix} \qquad \bar{b} = \begin{bmatrix} 0_{3\times1} \\ L_1^{-1}K_4 \end{bmatrix} \qquad \bar{h} = \begin{bmatrix} 0_{3\times1} \\ L_1^{-1}K_5 \end{bmatrix}$$

$$L_1 = K_1 \Big| \ \theta_1 = \theta_2 = 0$$

$$L_{21} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & (m_1 l_1 + m_2 L)g & 0 \\ 0 & 0 & m_2 l_2 g \end{bmatrix}$$

$$L_{22} = \begin{bmatrix} -F & 0 & 0 \\ 0 & -c_1 - c_2 & c_2 \\ 0 & c_2 & -c_2 \end{bmatrix}$$

Parameters of the model are either measured or experimentally identified and are given in Section 6.1 (see Reference 6). These parameters are substituted in the linear model (eqn. (6.2.10)), and since eigenvalues of matrix $\bar{A}$ are [7.2193, 4.1242, 0, -3.223, -6.087, -10.7545], there exist two unstable modes. The system, however, is observable, because

$$\begin{bmatrix} C \\ C\bar{A} \end{bmatrix} = I_6 \qquad (6.2.11)$$

and the observability index is 2. The controllability matrix V, i.e.

$$V = [\bar{b}, \bar{A}\bar{b}, \ldots, \bar{A}^5\bar{b}]$$

has full rank. This can be verified by examining the eigenvalues of the Grammian matrix V^TV, which are [6.33E+13, 4.03E+9, 2.77E+7, 1.28E4, 1.45E+3, 1.13E+1]. The square root of the ratio of the maximum and minimum root of V^TV is found to be 2.36E+6, which means that the system is very difficult to control [3,6,9]. Since the linear system is controllable, the poles of the closed loop system are asignable symmetrically and arbitrary, and the system can be stabilized. But a nonlinear unstable system, like a double inverted pendulum may not be stabilized by using the controller designed for its linearized model, since the system state sometimes deviates considerably from the equilibrium state, whereas the linearized model is valid [9]. There are also other physical constraints on the system, such as saturation of the power amplifier, limitations on cart movement etc. Since satisfaction of these constraints are not explicit in terms of the location of poles or the criterion function, application of the classical control is a formidable task. Our method of designing a controller for such a system as a double inverted pendulum is based on minimization of a criterion function such that all the performance and stability constraints are satisfied. In the following sections we shall show how this can be achieved.

6.3 - DESIGN OF OPTIMAL SERVO CONTROLLER FOR THE DOUBLE INVERTED PENDULUM

In the presence of constant disturbance, such as unknown angle of inclination of the rail, the linear model of the system can be stabilized using a servo control system [6]. In order to stabilize the system, at a given reference position on the rail when constant or random disturbances exist, we employ a servo controller of the form

$$u = -z_1^T x + z_2 \int_0^t \tilde{r} dt \qquad (6.3.1)$$

where $\tilde{r} \triangleq r_d - x_1$ and $z_1 \in R^6$, $z_2 \in R^1$ are the vectors of design parameters. The following equations for the closed loop system can be derived from eqns. (6.2.10) and (6.3.1) for the constant disturbance d and constant command signal r_d:

$$\frac{d}{dt}\begin{bmatrix}\dot{x}\\ \dot{\tilde{r}}\end{bmatrix} = \left\{\begin{bmatrix}\bar{A} & 0\\ [-1.0_{1\times 5}] & 0\end{bmatrix} - \begin{bmatrix}\bar{b}\\ 0\end{bmatrix}[z_1^T \; z_2]\right\}\begin{bmatrix}\dot{x}\\ \dot{\tilde{r}}\end{bmatrix} \qquad (6.3.2)$$

The above linear system has been stabilized using the CAD package SIMNON [10], yielding

$$z_1^T = [2.196, -17.14, 39.44, 5.61, 2.196, 6.03]$$

$$z_2 = 0.315.$$

Eqn. (6.3.2) can be rewritten as

$$\frac{d}{dt}\begin{bmatrix}\dot{x}\\ \dot{\tilde{r}}\end{bmatrix} = \tilde{A}\begin{bmatrix}\dot{x}\\ \dot{\tilde{r}}\end{bmatrix} \qquad (6.3.3)$$

where $\tilde{A}$ is given by

$$\tilde{A} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -50.7 & 393.4 & -910.7 & -134.1 & -49.97 & -139.6 & 7.265 \\ 147.3 & -1102. & 2634. & 389.6 & 144.9 & 405.9 & -21.11 \\ -17.13 & 80.46 & -258.5 & -45.29 & -16.22 & -47.63 & 2.454 \\ -1.0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad (6.3.4)$$

The eigenvalues of $\tilde{A}$ are given by

$$\lambda_1, \lambda_2 = -11.507 \pm 25.837J$$

$$\lambda_3, \lambda_4 = -3.0478 \pm 4.221J$$

$$\lambda_5, \lambda_6 = -0.21 \pm 0.1379J$$

$$\lambda_7 = -7.299$$

We can now compute the symmetric positive definite matrix P of the Liapunov equation

$$\tilde{A}^T P + P\tilde{A} = -I_7$$

where matrix P is given by

$$P = \begin{bmatrix} 28.7 & 5.36 & 42.5 & 13.56 & 5.39 & 6.88 & -9.97 \\ 5.36 & 38.7 & -65.0 & -4.76 & -2.41 & -9,71 & -2.6 \\ 42.5 & -65.0 & 250.3 & 45.76 & 19.62 & 38.66 & -12.16 \\ 13.56 & -4.76 & 45.76 & 11.21 & 4.66 & 7.24 & -4.27 \\ 5.39 & -2.41 & 1962 & 4.66 & 1.97 & 3.1 & -1.67 \\ 6.88 & -9.71 & 38.66 & 7.24 & 3.1 & 6.05 & -1.98 \\ -9.97 & -2.6 & -12.16 & -4.27 & -1.67 & -1.98 & 4.99 \end{bmatrix}$$

(6.3.4)

Let us define function $V : R^n \rightarrow [0,\infty)$

$$V(x) \triangleq (x^T P x)^{1/2} \tag{6.3.5}$$

where positive definite matrix P is given by eqn. (6.3.4).

Let us choose

$$\bar{x}^T = [0.1,\ 0.05,\ 0.05,\ 0,\ 0,\ 0,\ 0]. \tag{6.3.6}$$

The set of initial points (domain of attraction) is given by

$$B(\bar{x}) = \{x \,|\, V(x) - V(\bar{x}) < 0\}. \tag{6.3.7}$$

We are now in a position to set up a system of inequalities for the design requirements. In setting up the system of inequalities $\varphi_j(z) < 0$, the function $\varphi(.)$ must be chosen to reflect the design requirements. Let response of the system (position of the cart on the rail) to a step input ($r_d = \hat{r}H(t)$) be $x(t,x_0,z)$. The requirements are: a rise time of less than or equal to t_f seconds; an overshoot of less that or equal to 10%; and the absolute value of the control signal must not exceed 1.0. Other constraints are stability constraints to ensure asymptotic stability of the system in $B(\bar{x})$. The above requirements can be specified by

(a) $x_1(t,x_0,z) > 0.65\hat{r}$ for all $t_f < t < T$ and $x_0 \in B(\bar{x})$

(b) $x_1(t,x_0,z) \leq 1.1\hat{r}$ for all $0 \leq t \leq T$ and all $x_0 \in B(\bar{x})$

(c) $|u(t,x_0,z)| \leq 1.0$ for all $0 \leq t \leq T$

(d) $V(x(T,x_0,z)) \leq \beta V(\bar{x})$ for all $x_0 \in B(\bar{x})$, $\beta \in (0,1)$.

Notice that the above inequalities are actually infinite dimensional, each inequality being index by a time point in the relevant time interval; or by an initial state in $B(\bar{x})$. In order to reduce the number of inequalities and, indeed, to improve the efficiency of the optimization program, we have reformulated the above set of constraints as

$$\text{(i)}\quad \varphi_1(x_0,z) = \int_{t_f}^{T} \left| x_1(t,x_0,z) - \hat{r} \right| dt \leq \alpha_1$$

$$\text{(ii)}\quad \varphi_2(x_0,z) = \int_{0}^{T} \left| x_1(t,x_0,z)^2 - 1.21\tilde{r}^2 \right| dt \leq 0 \quad \text{if } \left| x_1(t,x_0,z) \right| > 1.1\hat{r}$$

$$\text{(iii)}\quad \varphi_3(x_0,z) = \int_{0}^{T} \left| u(t,x_0,z) - 1.0 \right| dt \leq 0 \quad \text{if } \left| u(t,x_0,z) \right| > 1.$$

(iv) $\varphi_4(x_0,z) = V(x(T,x_0,z)) - \beta V(\bar{x}) \leq 0$ for all $x_0 \in B(\bar{x})$, $\beta \in (0,1)$

where α_1 is a scalar which is initially set to some positive value and then can be interactively reduced to obtain an optimal design.

The above optimization problem is programmed in SIMNON language, and the algorithm yielded a stable controller as-

sessed by 500 initial states in $B(\bar{x})$, randomely chosen (uniformely distributed in $B(\bar{x})$). The vectors of the design parameters are computed as

$$z_1^T = [5.62, -16.024, 41.943, 4.9031, 0.9736, 5.926]$$

$$z_2 = 2.8358$$

and the characteristic roots of the linearized stable system are found to be

$$\lambda_1, \lambda_2 = -6.8179 \pm 6.9274J$$

$$\lambda_3, \lambda_4 = -1.506 \pm 6.483J$$

$$\lambda_5 = -88.2258$$

$$\lambda_6 = -3.9214$$

$$\lambda_7 = -0.61237 \times 10^{-1}.$$

Figure 6.3 shows the simulation results for ten initial states in $B(\bar{x})$. Since the system is only considered to behave linearly when the angle of first pendulum θ_1 is less than 3^o ($|x_2| < 0.05$) and $\theta_2 - \theta_1$ is less than 0.3^o ($|x_2 - x_3| < 0.005$) [6], results of Fig. 6.3 clearly show that the controller designed by our algorithm can stabilize the pendulum for much larger values of θ_1 and $\theta_1 - \theta_2$. Fig-

ures 6.4 and 6.5 show the response of the system, where the reference value ($\hat{r}_d$) and disturbance (incline of the rail d) are changed stepwise ($\hat{r}_d$ = 0.1m, $\alpha = 15^o$).

6.4 - DESIGN OF OBSERVER

In the design of feedback controller for the double inverted pendulum we assumed that all the six states are observable, but in practice, only three of the states, namely r, θ_1, and $\theta_1 - \theta_2$, are directly measurable, the rest must be constructed using an observer. It is well known that a minimal order observer (third order) can be designed to construct the unobservable states [4]. We shall employ the Gopinath method [7] for the design of minimal order observer. Since the position of the cart, the angle θ_1 and the difference of angles ($\theta_2 - \theta_1$) are measured by potentiometers, their values are taken as output y:

$$y = \bar{C}x = [H_0, 0]x \qquad (6.4.1)$$

where

$$H_0 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix}$$

For the design of the observer, instead of the system represented by eqn. (6.2.10), the following equivalent system is used:

$$\dot{w} = Aw + bu + hd$$

$$y = Cw = [I_3 \; 0]w \tag{6.4.2}$$

where $w \epsilon R^6$, $y \epsilon R^3$ and $u \epsilon R$ are the new states, output and input to the system,

$$w = Hx$$

$$A = H\bar{A}H^{-1}$$

$$b = H\bar{b}$$

$$C = \bar{C}H^{-1}$$

$$h = H\bar{h} \tag{6.4.3}$$

$$H = \begin{bmatrix} H_0 & 0 \\ 0 & H_0 \end{bmatrix}$$

$$A = \begin{bmatrix} 0_3 & I_3 \\ A_{21} & A_{22} \end{bmatrix} \quad b = \begin{bmatrix} 0_{1\times 3} \\ b_2 \end{bmatrix} \quad h = \begin{bmatrix} 0_{1\times 3} \\ h_2 \end{bmatrix}.$$

The minimal order observer designed for eqn. (6.4.2) is given by

$$\dot{\bar{x}} = \hat{A}\bar{x} + \hat{B}y + \hat{J}u \tag{6.4.4}$$

$$\hat{x} = \hat{C}\bar{x} + \hat{D}y$$

where $\hat{x}$ is the estimate of w,

$$\hat{A} = A_{22} - L$$

$$\hat{B} = \hat{A}L + A_{21}$$

$$\hat{C} = \begin{bmatrix} O \\ I \end{bmatrix} \qquad (6.4.5)$$

$$\hat{D} = \begin{bmatrix} I \\ L \end{bmatrix}$$

$$\hat{J} = b_2$$

L is an arbitrary matrix to make $\hat{A}$ stable. In this study L is chosen as

$$L = 10I_3 \qquad (6.4.6)$$

The observer matrices are then given by

$$\hat{A} = \begin{bmatrix} -14.6313 & 0.00472 & -.00925 \\ 13.459 & -9.312 & 1.0647 \\ -15.0236 & -3.7326 & -14.7784 \end{bmatrix} \qquad (6.4.7)$$

$$\hat{B} = \begin{bmatrix} -146.313 & -2.2185 & -0.0249 \\ 134.59 & -58.051 & -2.625 \\ -150.236 & -76.475 & -85.344 \end{bmatrix} \qquad (6.4.8)$$

The eigenvalues of $\hat{A}$ are:

$$\lambda_1 = -14.727, \quad \lambda_2 = -13.851, \quad \lambda_3 = -10.1418$$

To make the studies more realistic, we have to include the

measurement and observation noises. A full study of different noises in the system is outside the scope of this work. We shall, however, assume that a single white noise is affecting the controller system. A standard deviation of 0.05 (high noise) is assumed. Figures 6.6 and 6.7 show the response of the double pendulum system with a minimal order observer to a step reference ($\hat{r}$ = 0.1 m) input, and to step-reference input with step disturbance ($\hat{r}$ = 0.1 m, $\alpha = 15^{o}$), respectively.

6.5 - DESIGN OF FUNCTIONAL OBSERVER

In the previous section we designed a minimal (third) order observer, but to reconstruct a linear combination of the states ($z_1^T w$), a lower order observer (functional observer) can be used. This has the advantage of reducing the computation time, which allows a shorter sampling period. We employ an algorithm by Inoue [5] (also see Ref. 6) to design a functional observer (Fig. 6.2). The observer is given by

$$\dot{\bar{x}} = \hat{A}\bar{x} + \hat{b}y + \hat{j}u$$

$$v = \hat{c}\bar{x} + \hat{d}y \qquad (6.5.1)$$

where v is an estimate of $z_1^T w$ and

$$\hat{A} = (-10.0) \quad \hat{b} = (-301.85,\ -517.78,\ -103.65)^T$$

$$\hat{j} = 94.0732 \quad \hat{c} = 1$$

$\hat{d} = (35.805,\ 77.583,\ 80.188)^T$

A similar noise to the one used with the minimal order observer is added to the control signal. Figures 6.8 and 6.9 show response of the double pendulum system with the functional observer to a step reference input ($\hat{r}_d$ = 0.1 m) and to a step reference input with step disturbance ($\hat{r}_d$ = 0.1 m, $\alpha = 15^o$), respectively. A sampling time of 4 mili-seconds was assumed (compared with a sampling time of 10 mili-seconds for the minimal order observer).

6.6 - DISCUSSION

An alternative approach to feedback design of nonlinear system (presented in Chapter 2) is successfully applied to a double inverted pendulum system. A feedback controller is designed using an optimization algorithm which is based on multiple simulation of the nonlinear system in a finite interval. The results of this study clearly show the advantages of this method compared with other methods where a controller is designed for a linearized model of the system. A minimal order observer and a functional order observer are designed and the response of the system to step reference input and step constant disturbance is simulated. The functional order observer appears to be favourable in practice, although the minimal order observer produces better results when severe constant disturbance exists.

6.7 - REFERENCES

[1] - Mayne D.Q., Sahba M.,"Design of feedback controllers for nonliear systems", International Conference on Control and its Applications, University of Warwick, U.K., pp. 276-280, 1981.

[2] - Sahba M.,"Computer aided design of nonlinear systems with application to control of a double inverted pendulum", Proc. D, IEE, Vol 130, pp. 350-358, 1983.

[3] - Sturgen W.R., Loscutoff M.V.,"Application of model control and dynamic observers to control a double pendulum", Proc. of JACC pp. 857-865, 1972. [4] - Luenberger D.G.,"Observer for multivariable systems", IEE Trans. Aut. Contr. AC - 11 No. 2, pp. 190-197, 1966.

[5] - Inoue A., SICE Trans., Vol 10 pp.487, 1979.

[6] - Furuta K., Kajiwara H., Kosuge K.,"Digital control of a double inverted pendulum on an inclined rail", IJC, Vol 32, No. 5, pp. 907-924, 1980.

[7] - Gopinath B.,"On the control of linear multivariable input-output systems", The Bell Syst. Tech. J., Vol 50, pp. 1063-1081, 1971.

[8] - Mayne D.Q., Sahba M., "An efficient algorithm for solving inequalities", Accepted for publication by JOTA.

[9] - Furuta K.,Okutani T., Sone H.,"Computer control of a double inverted pendulum", Computer and Elec. Eng.,Vol 5, pp. 67, 1978.

[10] - Elmquist H.,"SIMNON: An interactive simulation program for nonlinear systems", Proc. Simu. 77, Montreaux, 1978.

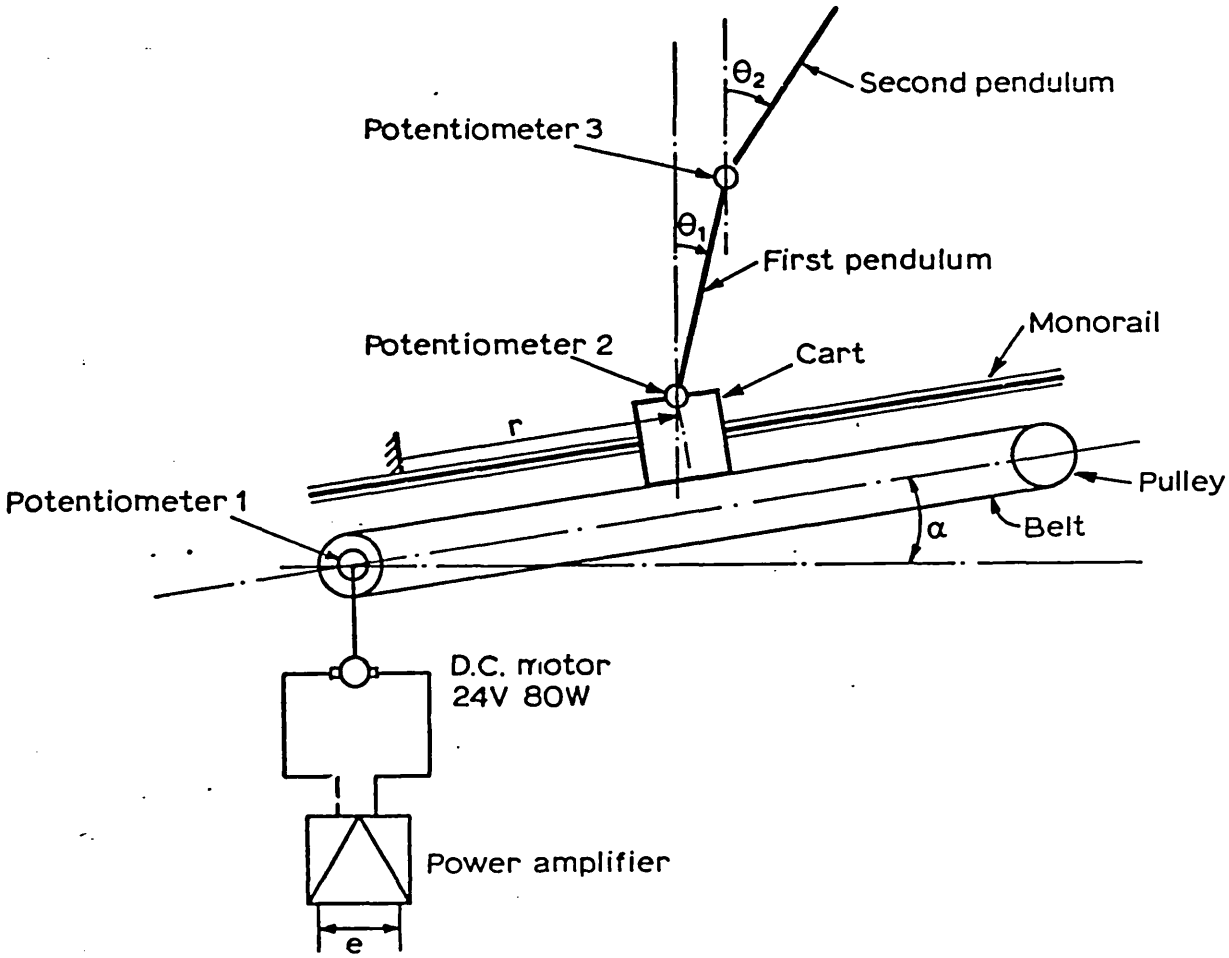

Fig. 6.1 Double Inverted Pendulum System

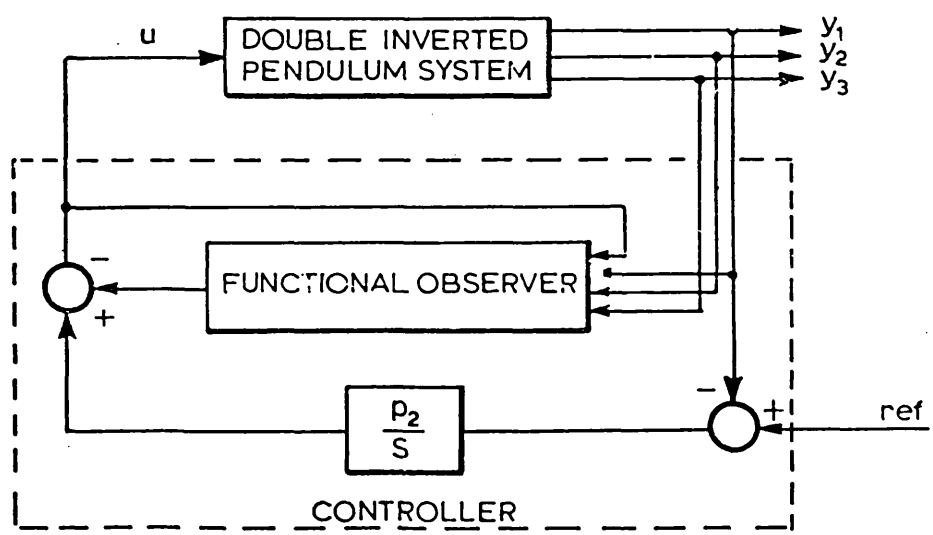

Fig. 6.2 Control System for Double Inverted Pendulum with Functional Observer

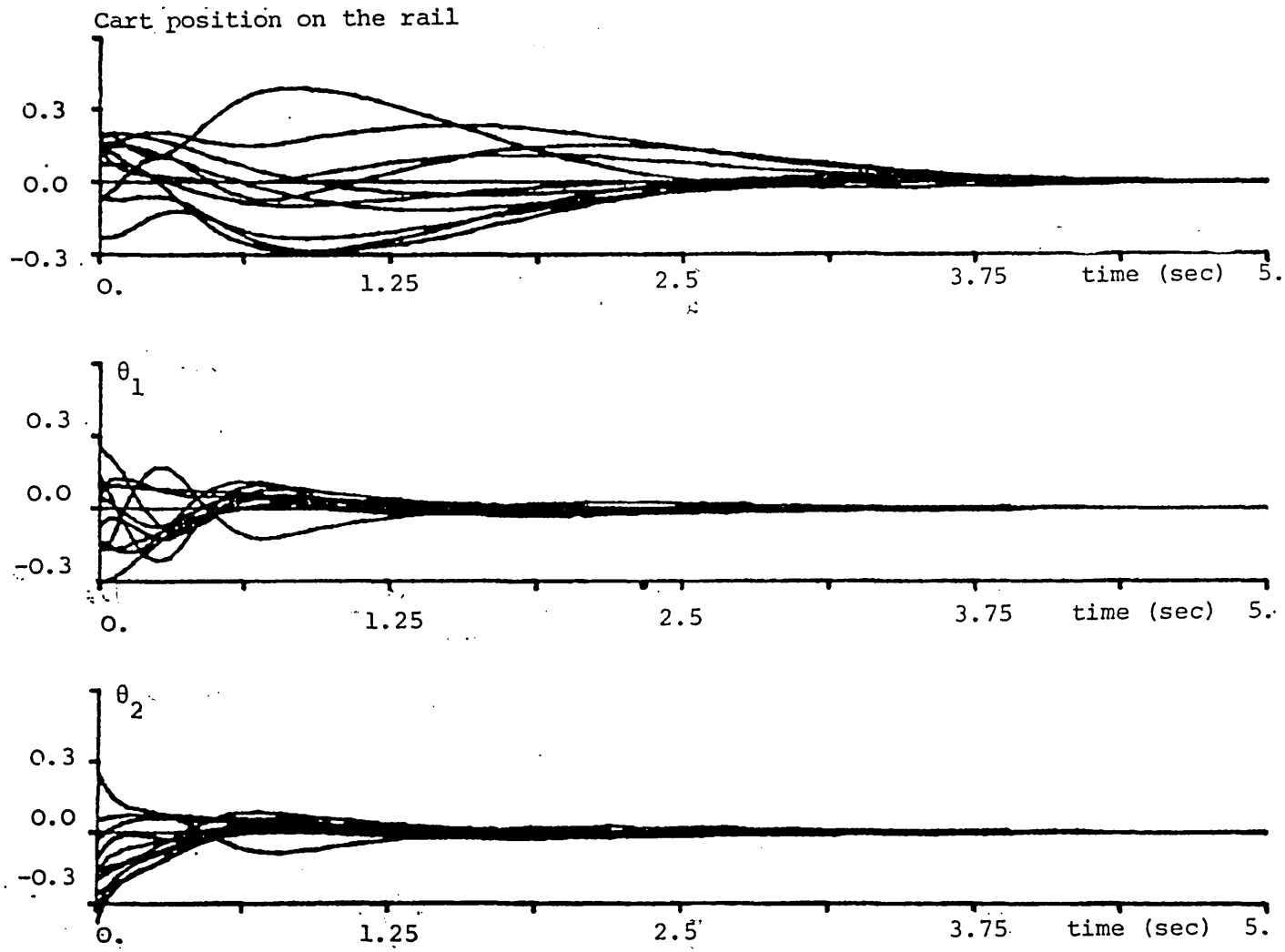

Fig.6.3 Simulation results for 10 starting point in the set of initial conditions.

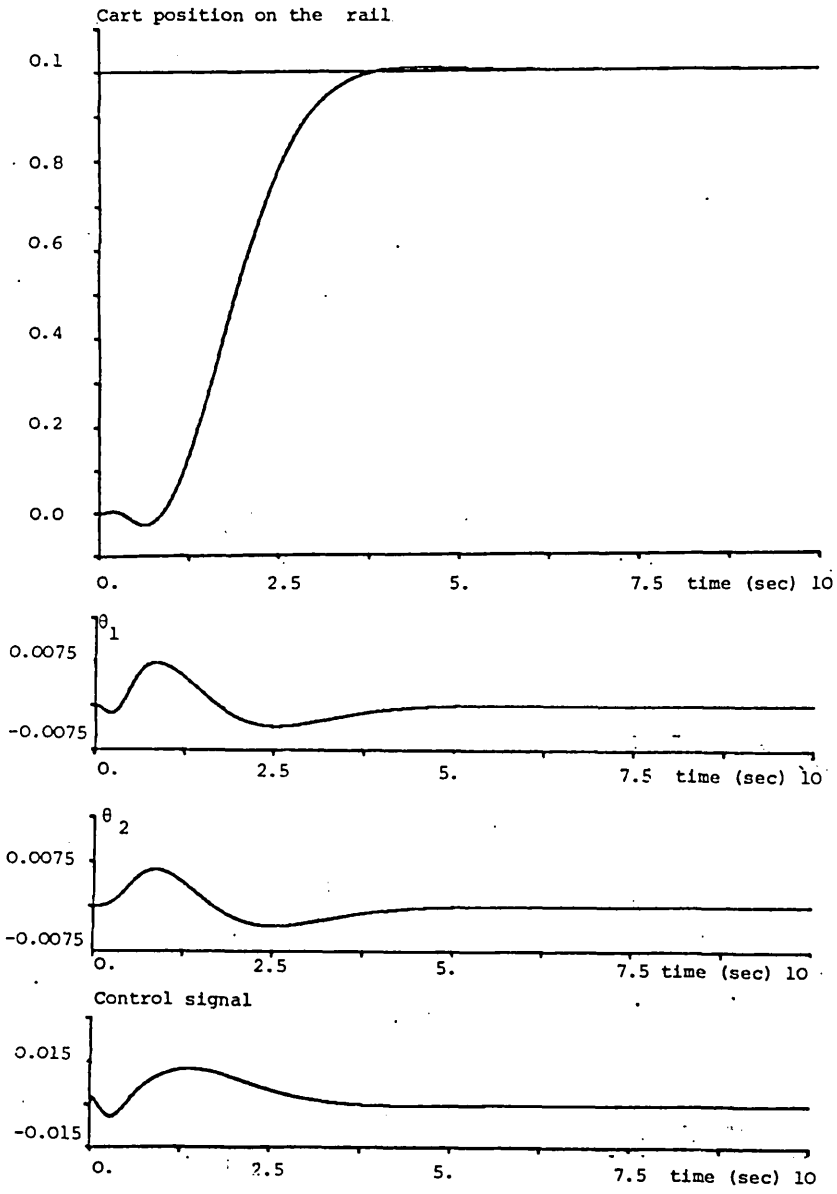

Fig.6.4 Response of the system to step input.

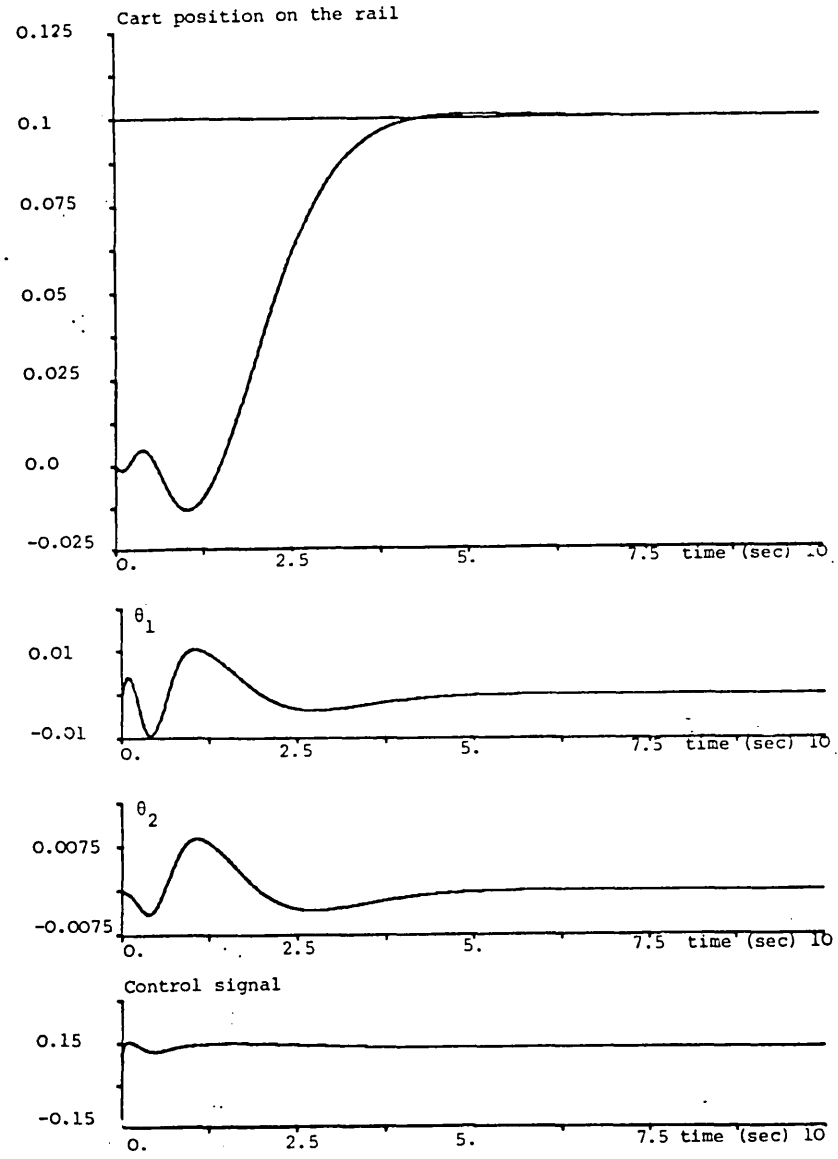

Fig.6.5 Response of the system to step input and step disturbance.

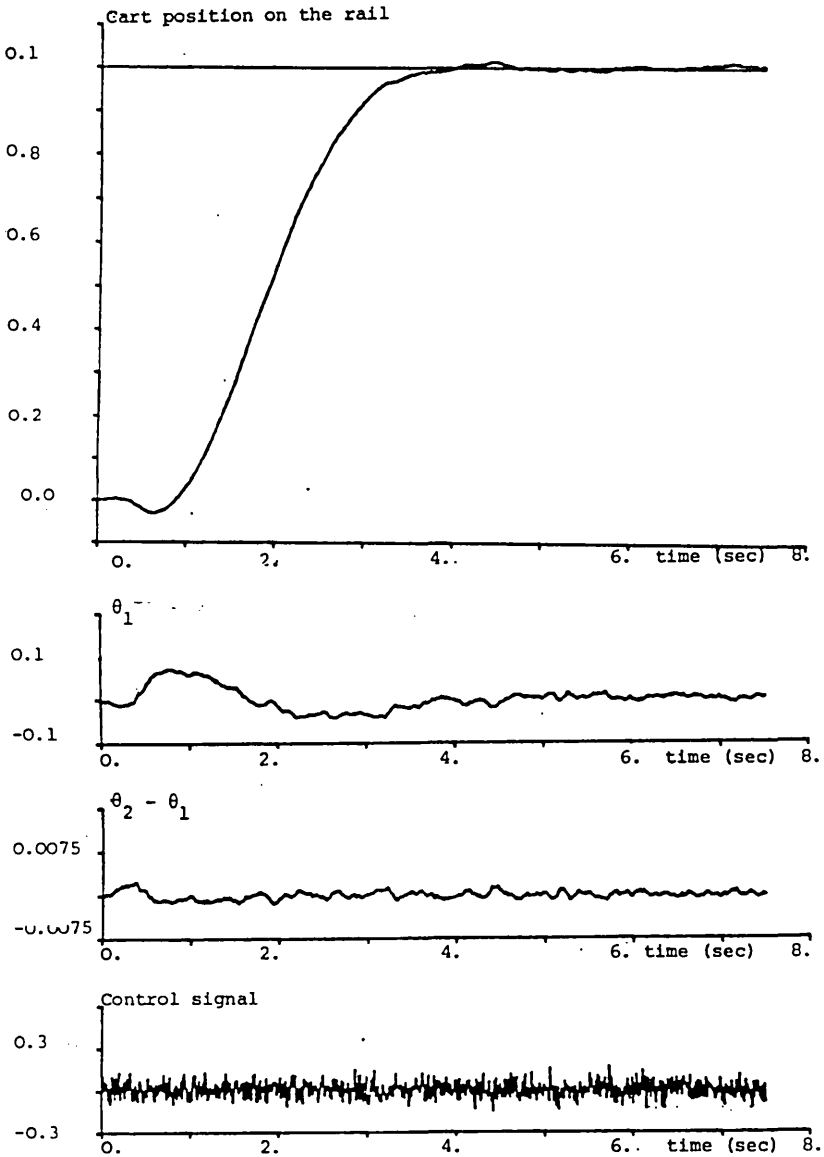

Fig. 6.6 Response of the system with minimal order observer to step input.

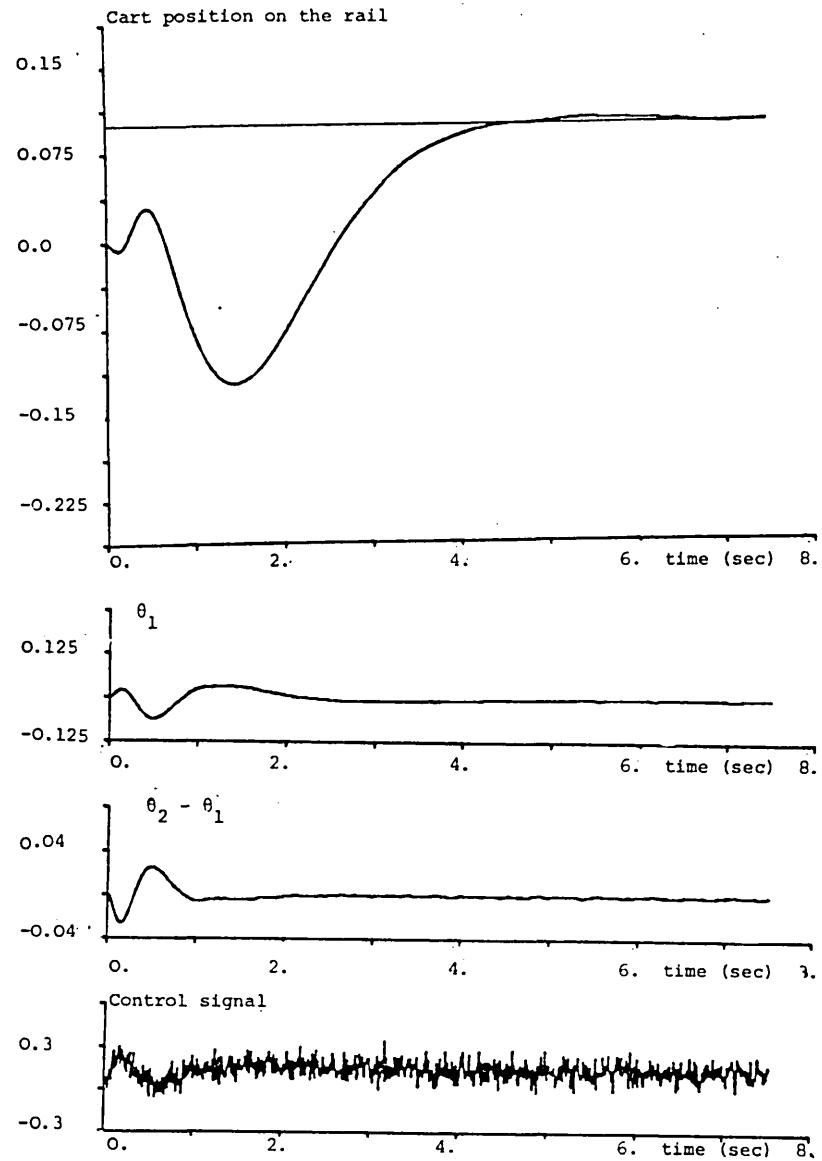

Fig.6.7 Response of the system with minimal order observer to step input and step disturbance.

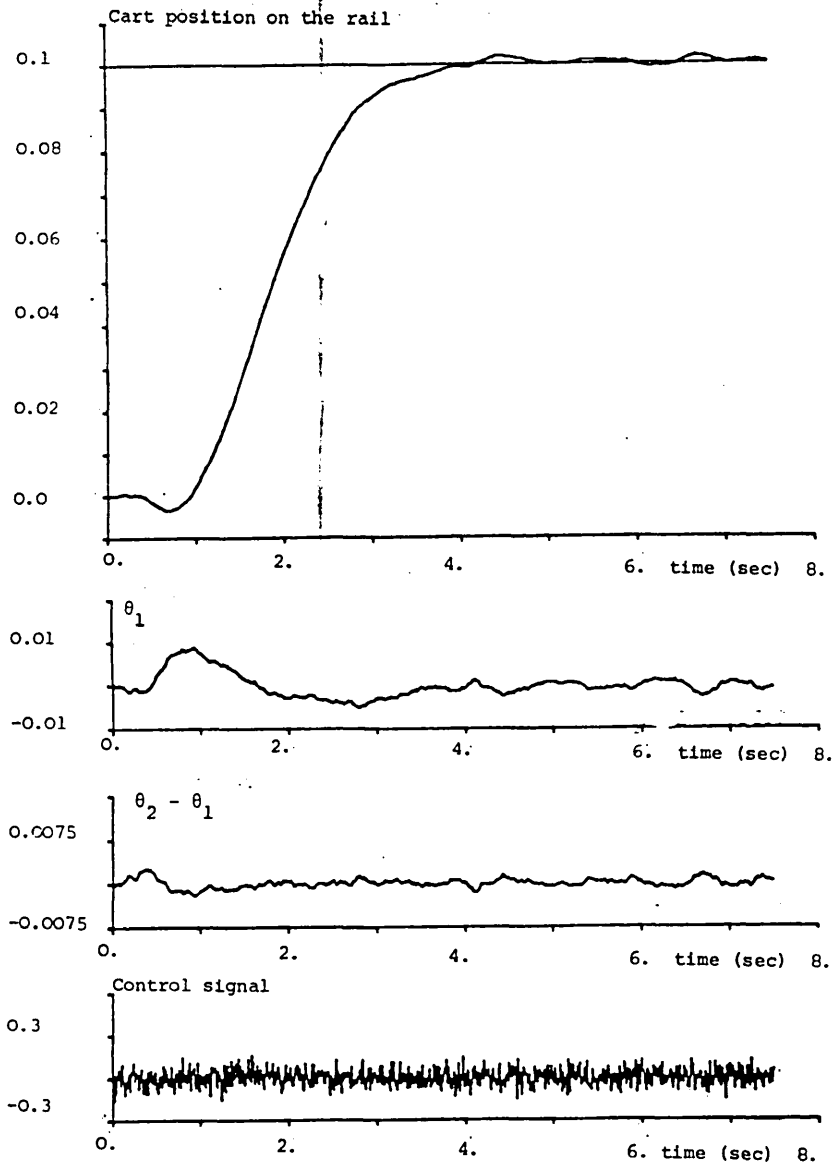

Fig. 6.9 Response of the system with functional observer to step input.

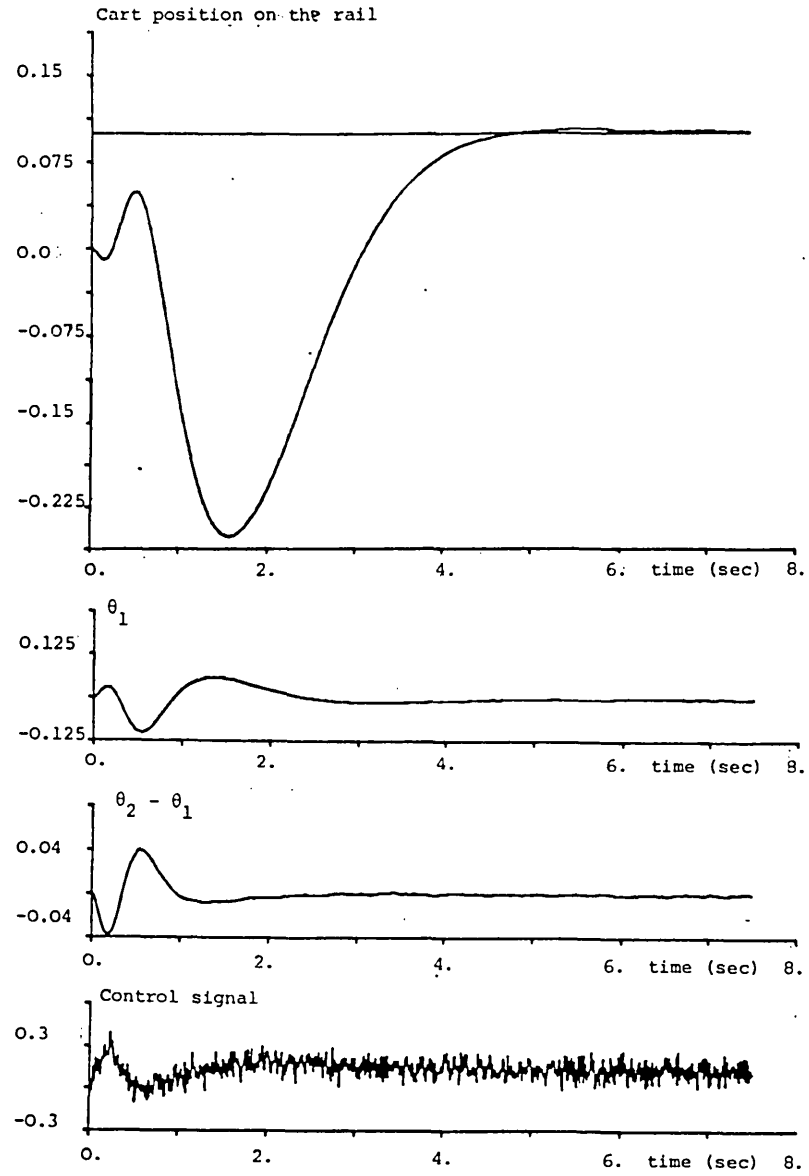

Fig. 6.10 Response of the system with functional observer to step input and step disturbance

CHAPTER 7

COMPUTER AIDED DESIGN OF NONLINEAR CONTROLLERS FOR TORQUE CONTROLLED ROBOT ARMS

7.1 - INTRODUCTION

It has been shown that many design objectives, including stability, of nonlinear systems may be expressed as infinite dimensional inequalities [1-3]. A range of algorithms, with established convergence, have also been developed to solve the associated problems of computing the parameters of a controller which satisfies these complex constraints or minimizes an objective function subject to these constraints. However, applying these procedures automatically without employing the insight yielded by control theory to specify the structure of the controller is naive and likely to be computationally expensive. On the other hand, control synthesis techniques (such as linear optimal controllers, pole allocation, etc.) are inadequate as a practical design methodology because they are unable (without much manipulation, for example, of quadratic cost coefficient) to meet design constraints. An ideal design methodology would combine control theoretical results to choose the structure of the controller with sophisticated mathematical programming techniques to adjust parameters to satisfy the complex design constraints.

Such a methodology has already been proposed [4] for the design of linear multivariable systems. The design parameters

specify a family of achievable (exponentially) stable closed-loop transfer functions $H(s, z)$ where $z \triangleq (z_1, \ldots, z_k)$ denote the vector of design parameters. (Vectors such as $z = (z_1, \ldots, z_k)$, $x = (\theta, \dot{\theta})$, etc are column vectors in all the equations). The same approach cannot, in general, be utilized for nonlinear system design. However, there exists a class of nonlinear systems which can be transformed into (equivalent) linear systems using nonlinear transformations [5]. If $\dot{x} = f(x, u)$ is a member of this class of systems then there exists a nonlinear transformation $(x, u) \rightarrow (w, v) = T(x, u)$ such that the transformed system satisfies $\dot{w} = Aw + Bv$, i.e. is linear having a transfer function $P(s)$. Using the procedure described above, the desired closed loop transfer function $H(s, z)$ uniquely determines a (linear) controller $C(s, z)$ for the transformed linear system. This can be transformed back into a nonlinear controller for the original system. The parameter z can then be chosen to satisfy design constraints.

The rich literature on the control of robot arms shows that the controllers for these systems can be designed in this way. Indeed torque controlled robot arms can be transformed into equivalent linear systems by a particularly simple transformation of the control u (and not the state x).

Dynamic models of robot arms have been obtained in References 6, 7 and 8. It is shown there that the models of a wide variety of robot arms have the form:

$$H(\theta)\ddot{\theta} = j(\theta,\dot{\theta}) + Du \qquad (7.1.1)$$

where θ is a vector of angles, the dimension n/2 of θ being the number of degrees of freedom, $H(\theta)$ is a n/2xn/2 matrix valued function of θ (representing inertia defined by the masses and dimensions of each arm) and $j(\theta,\dot{\theta})$ is a n/2 vector including torques due to gravitational, coriolis and centripetal forces as well as friction. Finally, D is a n/2xn/2 real matrix which can be set equal to the identity matrix by suitable redefinition of u, the n/2-vector of applied torques. Several control techniques have been developed. Yaun [6] linearizes the system and applies linear decoupling theory to obtain a system of n/2 equivalent single joint systems. Freund [9] has shown how nonlinear systems of the form $\dot{x} = A(x, t)x + B(x, t)u$, $y = C(x, t) + D(x, t)u$ may be decoupled and this has been applied, for example in [9,10], to obtain an equivalent set of decoupled linear systems (one for each degree of freedom) to which standard control techniques can be applied. Ali and Taylor [11] have studied a similar model. They propose the nonlinear control transformation $(u, x) \rightarrow v$ defined by $u = B(x, t)^{*}(A_m x - A(x, t) + \lambda B_m v)$ (where B^{*} denotes the pseudo inverse of B). If $A_m x - A(x, t) + \lambda B_m v$ lies in the range of $B(x, t)$ for all x, t, v (this is not generally the case but is true for these robot arm models) then $\dot{x} = A(x, t)x + B(x, t)u$ is transformed to $\dot{x} = A_m x + \lambda B_m v$ by this transformation; with suitable choice of A_m and B_m these equations correspond to a set of decoupled linear systems for which suitable controllers can be designed, These transformations for torque controlled

robot arms all reduce to a simple transformation (see [12, 13]) which can be obtained by inspection from (7.1.1).

In Section 7.2 we present the control strategy. In Section 7.3 we discuss controller parametrization. In Section 7.4 the design constraints are discussed. Finally, in Section 7.5 we illustrate the procedure by designing a controller for a robot arm with two degrees of freedom.

7.2 - CONTROL STRATEGY

Rewriting (7.1.1) with D = I (redefining u) gives

$$H(\theta)\ddot{\theta} = j(\theta, \dot{\theta}) + u \qquad (7.2.1)$$

If we define (implicitly) a nonlinear trasformation $(x, u) \rightarrow v$ by:

$$u = H(\theta)[v - j(\theta, \dot{\theta})] \qquad (7.2.2)$$

then (7.2.1) is transformed into the equivalent system:

$$H(\theta)\ddot{\theta} = H(\theta)v \qquad (7.2.3)$$

which, if $H(\theta)$ is invertible for all (relevant) θ, is equivalent to the decoupled linear system:

$$\ddot{\theta} = v \qquad (7.2.4)$$

This transformation, which is implicit in [12], is used in this chapter to obtain a controller which satisfies design constraints (such as hard constraints on u, which from (7.2.2), is a nonlinear function of state $x \triangleq (\theta, \dot{\theta})$ for the nominal system (i.e. assuming H and j are exactly known)). However, as Hewit [13] has shown, model errors (e.g. in H and j due to unknown system parameters) may cause instability. Hence, practical design constraints include robust stability and performance. To cope with unknown friction (and, hence, unknown j) Hewit [13] proposes the use of accelerometers so that $\ddot{\theta}$ is known. From eqn. (7.2.1), $j(\theta,\dot{\theta})$ can then be deduced by calculating $H(\theta)\ddot{\theta} - u$ (provided that $H(\theta)$ is exactly known). In this work knowledge of $\ddot{\theta}_1$ is used to obtain the unknown load mass and optimization techniques are employed to ensure stability and performance for all $j(\theta, \dot{\theta})$ in a given region of uncertainty.

So far we have considered the problem of controlling θ, the arm coordinate vector. However, our main concern is control of the world coordinate vector q which is a nonlinear function of θ:

$$q = \varphi(\theta) \qquad (7.2.5)$$

Since the dimension of q will be generally less than the dimension of θ, q may be augmented by the addition of additional elements. This enables control of q and, furthermore, satisfaction of additional constraints (such as maintaining the angle of one link). It is undesirable to solve (7.2.5), i.e.

to compute θ for given q. This can be avoided, as shown by Hewit [13], by substituting (7.2.5) (and its derivatives) in (7.2.1) to yield:

$$H_1(\theta)\ddot{q} = j_1(\theta, \dot{\theta}) + u \qquad (7.2.6)$$

where $H_1 \triangleq HJ^{-1}$ and $j_1 \triangleq j + HJ^{-1}L$ where, in turn, $J \triangleq \varphi_\theta$ and $L \triangleq \varphi_{\theta\theta}$. With the nonlinear transformation (x, u) → v defined (implicitly) by

$$u = H_1(\theta)v - j_1(\theta, \dot{\theta}) \qquad (7.2.7)$$

(7.2.6) is transformed into:

$$\ddot{q} = v \qquad (7.2.8)$$

provided that H_1 is everywhere invertible. Hence, the problem of controlling q is essentially the same as that of controlling θ (although more computation is involved).

7.3 - CONTROL PARAMETRIZATION

In any design procedure a control structure must be proposed, and then the parameters of the controller chosen to satisfy design constraints. Here the controller structure is nonlinear being defined by the nonlinear transformation presented above.

The robot arm is described by

$$H(\theta)\ddot{\theta} = j(\theta, \dot{\theta}) + u \tag{7.3.1}$$

The nonlinear transformation (x, u) → v defined by:

$$u = H(\theta)v - j(\theta, \dot{\theta}) \tag{7.3.2}$$

(see Fig. 7.1) yields the equivalent system

$$\ddot{\theta} = v \tag{7.3.3}$$

if H(θ) is invertible. If v is defined by:

$$v_j = -k_j^{(1)}\theta_j - k_j^{(2)}\dot{\theta}_j + k_j^{(1)}r_j' \tag{7.3.4}$$

for j = 1, ..., n/2, where r_j' is the desired angle of the jth link, then the closed loop transfer function from r_j' to θ_j is:

$$z_{jj}'(s) = k_j^{(1)}/[s^2 + k_j^{(2)}s + k_j^{(1)}] \tag{7.3.5}$$

If we choose $k_j^{(1)}$ and $k_j^{(2)}$ to satisfy:

$$k_j^{(1)} = \omega_j^2 \tag{7.3.6}$$

$$k_j^{(2)} = 2\zeta_j\omega_j \tag{7.3.7}$$

then

$$z_{jj}'(s) = \omega_j^2/[s^2 + 2\zeta_j\omega_j + \omega_j^2] \tag{7.3.8}$$

which is a damped linear system with damping factor ζ_j and natural frequency ω_j. If the system is subjected to sudden changes in demanded position, it might be advantageous to introduce further compensators $1/[1 + sT_j]$, $j = 1, \ldots, n/2$, outside the feedback loop (see Fig. 7.2) in order to reduce peak torque. The controller parameters $(T_1, k_1^{(1)}, k_1^{(2)}, \ldots, T_{n/2}, k_{n/2}^{(1)}, k_{n/2}^{(2)})$ are completely specified, via (7.3.6) and (7.3.7) by the design parameter z defined by:

$$z \triangleq (T_1, \zeta_1, \omega_1, \ldots, T_{n/2}, \zeta_{n/2}, \omega_{n/2})^T \qquad (7.3.9)$$

These variables are chosen to satisfy the design constraints which are discussed next. Note that the (nonlinear) controller for the original system is defined by (7.3.2), (7.3.4), (7.3.6) and (7.3.7).

7.4 - DESIGN CONSTRAINTS

7.4.1 STABILITY OF THE NOMINAL SYSTEM

The most important constraint is that of stability. Since the total system is constrained to behave like a set of decoupled linear systems, with transfer functions $z_{jj}(s) = \omega_j^2/[1 + sT_j][s^2 + 2\zeta_j\omega_j + \omega_j^2]$, $j = 1, \ldots, n/2$, stability (of the nominal system) is achieved by the simple constraints $T_j > 0$, $\zeta_j > 0$, $\omega_j > 0$, $j = 1, \ldots, n/2$ on the design variables. Relative stability can be ensured by specifying ζ_j (e.g. $\zeta_j = 0.7$), $j = 1, \ldots, n/2$ and choosing (T_j, ω_j), $j = 1, \ldots, n/2$ to satisfy other constraints (sub-

ject, of course, to the constraints $T_j > 0$, $\omega_j > 0$, $j = 1, \ldots, n/2$).

7.4.2 - CONTROL CONSTRAINTS

Let (the column vector) $w \triangleq (\theta, \dot{\theta}, r')$ denote the state of the complete system (robot arm plus controller, where $r' \triangleq (r'_1, \ldots, r'_{n/2})$. The dimension of w is 3n/2. Assembling (7.3.2)-(7.3.4) yields:

$$\dot{w} = F(z)w + G(z)r \tag{7.4.1}$$

$$y = Mw \tag{7.4.2}$$

$$u = \varphi(w, z) \tag{7.4.3}$$

where

$$F(z) \triangleq \begin{bmatrix} 0 & I & 0 \\ -K_1(z) & -K_2(z) & K_1(z) \\ 0 & 0 & -T(z)^{-1} \end{bmatrix} \tag{7.4.4}$$

$$G(z) \triangleq \begin{bmatrix} 0 \\ 0 \\ T(z)^{-1} \end{bmatrix} \tag{7.4.5}$$

$$K_1(z) \triangleq \mathrm{diag}\{k_1^{(1)}, \ldots, k_{n/2}^{(1)}\} \tag{7.4.6}$$

$$K_2(z) \triangleq \mathrm{diag}\{k_1^{(2)}, \ldots, k_{n/2}^{(2)}\} \tag{7.4.7}$$

$$T(z) \triangleq \text{diag}\{T_1, \ldots, T_{n/2}\} \tag{7.4.8}$$

and $\varphi(w, z)$ is defined by:

$$\varphi_k(w, z) \triangleq h_k(\theta)[-K_1(z) \; -K_2(z) \; K_1(z)]w - j_k(\theta, \dot{\theta}) \tag{7.4.9}$$

where φ_k and j_k are the kth elements of φ and j respectively and $h_k(\theta)$ is the kth row of $H(\theta)$. As before $z \triangleq (T_1, \zeta_1, \omega_1, \ldots, T_{n/2}, \zeta_{n/2}, \omega_{n/2})$ is the vector of design parameters. Suppose the hard control constraint is

$$u(t) \in \Omega \tag{7.4.10}$$

for all t where Ω is typically the set $\{u \in R^{n/2} \mid |u_j| < a_j, \ldots\}$. Let $\hat{R}$ denote the set of test inputs r and $\hat{W}$ the set of initial states w for which the system must satisfy the control constraints. As a simple example $\hat{R}$ may consist of separate step functions applied to each input and $\hat{W}$ may be the set $\{0\}$. Let $w(t; z, r, w_0)$ and $u(t; z, r, w_0)$ denote the values of the state and control at time t if r is applied and w_0 is the initial ($t = 0$) state. Then the control constraint (to be satisfied by z) is:

$$u(t; z, r, w_0) \in \Omega \tag{7.4.11}$$

for all $t \in [0, t_1]$, all $r \in \hat{R}$ and all $w_0 \in \hat{W}$. This is an infinite dimensional constraint.

7.4.3 - PERFORMANCE CONSTRAINTS

Typical performance constraints are concerned with tracking and disturbance errors. The former can be specified as

$$y(t; z, r, w_0) \in Y(t; r, w_0) \qquad (7.4.12)$$

for all $t \in T^p$, all $r \in \hat{R}^p$ and all $w_0 \in \hat{W}^p$, where T^p, $\hat{R}^p$ and $\hat{W}^p$ denote, respectively, a set of time intervals, a set of test inputs and a set of initial states for which good tracking is required. The set $Y(t; r, w_0)$ denotes an appropriate "envelope" surrounding the input r to be tracked. This is also an infinite dimensional constraints. However, in our case adequate tracking performance can be achieved by constraints on ζ_j (e.g. setting $\zeta_j = 0.7$ for $j = 1, \ldots, n/2$) to ensure damping and constraints on ω_j and T_j, $j = 1, \ldots, n/2$ to ensure speed of response.

7.4.4 - ROBUST STABILITY

The above constrains, if satisfied, ensure that the nominal system satisfies our design objectives. However, the dynamics of the actual system differ from those of the nominal due, inter alia, to lack of knowledge of the true system parameters. Let vector p denote those actual parameters which may differ (appreciably) from the nominal parameter p^0. For example, p may include the load mass and the coefficients of friction of the joints of the robot. Then the actual robot dynamics are described by:

$$H(\theta, p)\ddot{\theta} = j(\theta, \dot{\theta}, p) + u \qquad (7.4.13)$$

whereas the nonlinear transformation:

$$u = H(\theta, p^0)v - j(\theta, \dot{\theta}, p^0) \qquad (7.4.14)$$

uses the minimal parameter. Hence, the equivalent system is now described by:

$$\ddot{\theta} = H(\theta, p)^{-1}[H(\theta, p^0)v + j(\theta, \dot{\theta}, ,p) - j(\theta, \dot{\theta}, p^0)] \qquad (7.4.14)$$

i.e.

$$\ddot{\theta} = v + \eta(x, v, p) \qquad (7.4.15)$$

where η is defined by

$$\eta(x, v, p) \triangleq [H(\theta, p)^{-1}H(\theta, p^0) - I]v$$

$$+H(\theta, p)^{-1}[j(\theta, \dot{\theta}, p) - j(\theta, \dot{\theta}, p^0)] \qquad (7.4.16)$$

where x denotes the vector $(\theta, \dot{\theta})$. It is easily seen that $\eta(x, v, p^0) = 0$ in which case (7.4.15) reduces to $\ddot{\theta} = v$. Suppose we know that p always lies inside the set P. Then robust stability is ensured provided that the nonlinear system described by (7.4.15) with

$$v = -K_1(z)\theta - K_2(z)\dot{\theta} + K_1(z)r' \qquad (7.4.17)$$

is stable for all $p \in P$. Assembling (7.3.2), (7.3.3) and (7.4.15) yields:

$$\dot{w} = F(z)w + G(z)r + \eta^1(z, w, p) \qquad (7.4.18)$$

$$y = Mw \qquad (7.4.19)$$

$$u = \varphi(w, z) \qquad (7.4.20)$$

where $\eta^1(w, z, p)$ is defined by

$$\eta^1(w, z, p) \triangleq [0, (x, -K_1(z)\theta - K_2(z)\dot{\theta} + K_1(z)r', p), 0] \qquad (7.4.21)$$

Hence, robust stability is ensured if (7.4.18) is stable for all p in P. To turn this statement into a standard constraint consider the case when $r'(0) = 0$ and $r(t) = 0$ for all $t > 0$. Then stability of (7.4.18) is equivalent to stability of

$$\dot{x} = F_1(x)x + \eta^2(x, z, p) \qquad (7.4.22)$$

where $\eta^2(x, z, p)$ consists of the first n elements of $\eta^1(w, z, p)$ defined in (7.4.21) and

$$F_1(z) \triangleq \begin{bmatrix} 0 & I \\ -K_1(z) & -K_2(z) \end{bmatrix} \qquad (7.4.23)$$

Unless $\eta^2(0, z, p) = 0$ for all z, p (this will be the case if p represents an error in terms depending on $\dot{\theta}$ (e.g. fric-

tion) but will not be the case if p represents errors in load mass) then the equilibrium point of (7.4.22) is no longer the origin and will depend on z and p. Let $\tilde{x}(t; z, p)$ denote the deviation of x from the equilibrium point. Then $\tilde{x}$ satisfies:

$$\dot{\tilde{x}} = F_1(z)\tilde{x} + \eta^3(\tilde{x}, z, p) \tag{7.4.24}$$

where $\eta^3(0, z, p) = 0$ for all z and p.

Let $V(x, z) \triangleq (1/2)x^T S x$ be a Liapunov function for $\dot{x} = F_1(z)x$ so that :

$$Q(z) \triangleq -[F_1(z)^T S + S F_1(z)] \tag{7.4.25}$$

is positive definite. Along trajectories of (7.4.24),

$$\dot{V}(\tilde{x}, z, p) = -\tilde{x}^T Q(z)\tilde{x} + \tilde{x}^T S\eta^3(\tilde{x}, z, p) \tag{7.4.26}$$

Hence, (7.4.24) will be stable if, for some $\delta > 0$,

$$\tilde{x}^T Q(Z)\tilde{x} \geq \tilde{x}^T S\eta^3(\tilde{x}, z, p) + \delta\|\tilde{x}\|^2 \tag{7.4.27}$$

for all $\tilde{x}$ and all p in P. This again is an infinite dimensional inequality. Note that (7.4.27) is sufficient but not necessary for stability; it may be necessary to relax the condition as in [3].

7.4.5 - ROBUST PERFORMANCE

Let $y(t; z, r, w_0, p)$ denote the output of the system at time t in response to input r and initial state w_0 when the uncertain parameters have the value p. The performance constraint can be specified as

$$y(t; z, r, w_0, p) \in Y(t, r, w_0) \tag{7.4.28}$$

for all $t \in T^p$, all $r \in \hat{R}^p$, all $w_0 \in \hat{W}^p$ and all $p \in P$.

7.5 - DESIGN STUDY

We consider a two degree of freedom robot arm (Fig. 7.3) with $n = 4$, $m_1 = 25kg$, $m_2 = 20kg$, $m_A = 5kg$, $l_1 = 1.2m$, $l_2 = 1.0m$, $|u_1| \leq 1500N.m$ and $|u_2| \leq 400N.m$ where m_i and l_i are the mass and length of link, $i = 1, 2$, m_A is the mass of the first join and $m_B \in [0, 20]kg$ is the load mass. The resultant expression for H and j are

$$H(\theta) = \begin{bmatrix} C_1 + C_2\cos\theta_2 + C_3 & C_2\cos\theta_2 + C_3 \\ C_2\cos\theta_2 + C_3 & C_3 \end{bmatrix} \tag{7.5.1}$$

$$j(\theta, \dot{\theta}) = \begin{bmatrix} C_2\dot{\theta}_2(2\dot{\theta}_1 + \dot{\theta}_2)\sin\theta_2 - C_4\cos(\theta_1 + \theta_2) - C_5\cos\theta_1 \\ -C_4\cos(\theta_1 + \theta_2) - C_2\dot{\theta}_1^2\sin\theta_2 \end{bmatrix} \tag{7.5.2}$$

where

$$C_1 = J_1/3 + J_A + (J_2 + J_B)\alpha^2$$

$$C_2 = (J_2/2 + J_B)/\alpha$$

$$C_3 = J_1/3 + J_B$$

$$C_4 = gl_2(m_2/2 + m_B)$$

$$C_5 = gl_1(m_1/2 + m_A + m_B)$$

and where

$$J_1 \triangleq m_1 l_1^2, \qquad J_2 \triangleq m_2 l_2^2$$

$$J_A \triangleq m_A l_1^2, \qquad J_B \triangleq m_B l_2^2$$

$$\alpha \triangleq l_1/l_2.$$

The designs were carried out using SIMNON (an interactive simulator for nonlinear systems [14]) with an added outer approximation algorithm to satisfy infinite dimensional constraints. Derivatives of the various constraint functions were estimated by finite differences.

7.5.1 - NOMINAL DESIGN

The state x and load m_B are assumed known. We set $\zeta_1 = \zeta_2 = 0.7$, $\omega_1 = \omega_2 = \omega_n$ and $T_1 = T_2 = T$ so that $z = (T, \omega_n)$. Stability of the nominal system is ensured by

the constraints $\omega_n > 0$ and $T > 0$. The interactive simulator (with optimization) was employed to determine a $T > 0$ and a $\omega_n > 0$ satisfying the control constraint for all $r \in \hat{R} \triangleq \{r \mid r < 0.1\text{rad}\}$ and $w(0) = 0$ for each of the following loads: $m_B = 0, 2, 4, \ldots, 20\text{kg}$. It was found that $T = 0.2\text{secs}$ and $\omega_n = 47.54\exp(-0.091 m_B)$ satisfied the design constraints. Note that bandwidth (speed of response) decreases as m_B increases. Simulation results for $w(0) = 0$, $r_1(t) = r_2(t) = 0.1H(t)$ are shown in Fig. 7.4.

7.5.2 - UNKNOWN LOAD

In the system equations both H and j are functions of m_B. Hence in principle the load can be determined by finding the value of m_B that satisfies the equation

$$H(\theta, p)\ddot{\theta} = j(\theta, \dot{\theta}, p) + u. \qquad (7.5.3)$$

Solving the first component of this equation yields the estimate

$$\hat{m}_B' = \{u_1 - (m_1 l_1^2/3 + m_A l_1^2 + m_2 l_1^2 + m_2 l_2 l_1 \cos\beta + m_2 l_2^2/3)\ddot{\theta}$$

$$- [(m_2 l_2 l_1/2)\cos\beta + m_2 l_2^2/3]\ddot{\beta} + (m_2 l_2 l_1/2)\dot{\beta}(2\dot{\theta} + \dot{\beta})\sin\beta$$

$$- (g l_2 m_2/2)\cos(\theta + \beta) - g l_1 (m_1/2 + m_A + m_2)\cos\theta\}$$

$$/\{l_2^2[\ddot{\theta}(l_1^2/l_2^2 + 2(l_1/l_2)\cos\beta + 1) + \ddot{\beta}[(l_1/l_2)\cos\beta + 1]$$

$$- (l_1/l_2)\dot{\beta}(2\dot{\theta} + \dot{\beta})\sin\beta] + gl_2\cos(\theta + \beta) + g_1 l_1\cos\theta\}. \tag{7.5.4}$$

Since this estimate is noisy it is further processed to obtain

$$\hat{m}_B = 20 + F[\text{sat}(\hat{m}'_B) - 20] \tag{7.5.5}$$

where the operation F is a linear filter whose transfer function is $1/[1 + s/100]$ and

$$\text{sat}(a) \triangleq \begin{cases} a & \text{if } a \in [0, 20] \\ 0 & \text{if } a < 0 \\ 20 & \text{if } a > 20 \end{cases} \tag{7.5.6}$$

This ensures that $\hat{m}_B \in [0, 20]$. To implement this estimator knowledge of $\ddot{\theta}_1$ is required.

Since $\hat{m}_B$ is not necessarily equal to m'_B the resultant system is nonlinear and is described by ((7.4.18). However, the estimate of the load is good so that the system behaviour is close to that of system in which the load mass is known. Simulation responses of the system when $x_0 = 0$ and $m_B = 10$ for three sets of inputs ($r_1(t) = r_2(t) = 0.1H(t)$; $r_1(t) = 0$, $r_2(t) = 0.1H(t)$, and $r_1(t) = 0.1H(t)$, $r_2(t) = 0$) are shown in Fig. 7.5. It can be seen that $\hat{m}_B$ converges to m_B rapidly and that thus ensures low interaction between the two arms (less that 0.25% for the first and less that 2.5% for the second).

7.5.3 - LOAD AND STATE UNKNOWN

In this case we assume that the angular position (θ_1, θ_2) but not the angular velocities are measured. To design the observer it was assumed that m_B was known so that θ_1, for example, satisfies the linear equations

$$(d/dt)\theta_1 = \dot{\theta}_1 \tag{7.5.7}$$

$$(d/dt)\dot{\theta}_1 = -\omega_n^2\theta_1 - 1.4\omega_n\dot{\theta}_1 + \omega_n^2 r_1'. \tag{7.5.8}$$

Given θ_1 it is possible to design a nominal observer of the form:

$$\dot{\eta}_1 = a\eta_1 + b\theta_1 \tag{7.5.9}$$

$$\dot{\theta}_1' = c\eta_1 + d\theta_1. \tag{7.5.10}$$

The same observer was designed for each arm, and a, b, c, and d were chosen so that the estimate errors

$$\theta_1^e \triangleq \theta_1 - \theta_1', \quad \theta_2^e \triangleq \theta 2 - \theta_2'$$

satisfied

$$\dot{\theta}_1^e = -(1.4\omega_n + 10)\theta_1^e \tag{7.5.11}$$

$$\dot{\theta}_2^e = -(1.4\omega_n + 10)\theta_2^e \tag{7.5.12}$$

so that the observer time constant (for each arm) is less than 0.05 sec which is small compared with the system response.

To estimate the mass, (7.5.4) was again used with $\dot{\theta}_1$ and $\dot{\theta}_2$ replaced by the estimate $\dot{\theta}'_1$ and $\dot{\theta}'_2$ obtained from the observers, $\ddot{\theta}_2$ assumed known (provided by an accelerometer), and $\ddot{\theta}_1$ replaced by v_1 (from Fig. 7.5, it is clear that v_1 is an estimate of $\ddot{\theta}_1$ provided that $\hat{m}_B$ approximates m_B), the acceleration $\ddot{\theta}_1$ can also be computed from (7.4.1), yielding

$$\ddot{\theta}_1 = [H(x, \hat{m}_B)^{-1}]_{1*}[u + j(x, \hat{m}_B)] \qquad (7.5.13)$$

where $[.]_{1*}$ denotes the first row of the matrix [.] and the notation has been altered to show that both H and j are functions of load mass.

In simulating the complete system (robot arm, nonlinear controller, linear feedback controller, external compensator and the mass estimator) process noise and measurement noise having standard deviation of 0.05 and 0.001, respectively, were added. Simulated responses for the case $x_0 = 0$, $r_1(t) = r_2(t) = 0.1H(t)$, $m_B = 10kg$, $\ddot{\theta}_2$ assumed known and $\ddot{\theta}_1$ replaced by v_1, are shown in Figure 7.6. In this simulation the cross couplings were less than 3%.

7.5.4 - ROBUSTNESS

It is possible to include the robustness constraints (for stability and performance) given above in the design stage. We have not done so because of the limitations in our current computer aided design system. However, we have checked the robustness of the (nominal) design with the load mass estimator by checking performance and stability over the following set of (plant) variations:

(a) FRICTION: to account for dynamic friction, $j(\theta, \dot{\theta})$ in (7.3.1) we replaced (as in (7.4.13)) by:

$$j(\theta, \dot{\theta}, p) = j(\theta, \dot{\theta}) + \begin{bmatrix} -f_1\dot{\theta}_1 + f_2(\dot{\theta}_2 - \dot{\theta}_1) \\ -f_2(\dot{\theta}_2 - \dot{\theta}_1) \end{bmatrix} \qquad (7.5.14)$$

where f_1 and f_2 could take any value in the range $[0, 0.1]$(N.m.s/rad).

(b) LOAD MASS: to allow for error in estimating mass we replace m_B in (7.3.1) with $\hat{m}_B = m_B + \tilde{m}$, where $\tilde{m}$ may have any value in the range $[-2, 2]$kg.

Thus, in this case $p = (f_1, f_2, \tilde{m})$ and $P = [0, .1]\times[0, .1]\times[-2, 2]$. To check performance and stability the trajectory of the actual system was computed for a 100 initial states, randomely chosen in the set $\{\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2, m \mid \theta_1, \theta_2 \in [0, 2\pi], \dot{\theta}_1, \dot{\theta}_2 \in [0, 0.5], m \in [-10, 10]\}$, with $r(t) = 0$ and $p = (f_1, f_2, \tilde{m})$ chosen randomely in the set

P. Process noise (standard deviation 0.05) and measurement noise (0.001) were added. The filter constant (for the load mass esimator) was increased to 0.1 sec to increase transient errors. The simulation results (for 20 of the starting points) are shown in Figure 7.7.

7.6 - DISCUSSION

It appears that the most successful approaches to computer aided design will employ a combination of structural results from control theory and numerical techniques for obtimizing and satisfying design constraints, both employed in a flexible interactive system. For linear systems and a selected class of nonlinear systems it is possible to choose the structure of the controllers so that the controller is parametrized by parameters of a pre-specified closed loop system. These parameters (in our case damping factor and natural frequency of a second order system) can then be chosen to ensure stability and satisfaction of design constraints (on control magnitudes and performance). In the design of a torque controlled robot, the consequence of this methodology is that the design problem reduces to the design of a set of linear, decoupled second order systems with nonlinear state constraints (corresponding to the control magnitude constraints). Standard outer approximation algorithms have been employed to satisfy the design constraints. It is also shown how to modify the controller when the state is not completely accessible and when the load mass is unknown. Simulation results show that the resultant system performs well in the

range of test constraints. Although robustness constraints were not included at the design stage, repeated simulation shows that the nominal design copes well with a (wide) range of errors in load mass and friction coefficients.

7.7 - REFERENCES

[1] - Zakian V., Al-Naib U., "Design of dynamical and control systems by method of inequalities", Proc. IEE, Vol 120, pp 1421-1427, 1973.

[2] - Mayne D.Q.,"Control system design via mathematical programming", in Design of Modern Control Systems, P. Peregrinus for IEE D.J. Bell, P.A. Cook, N. Munro (eds), IEE control Eng. Series 20, pp. 144-157, 1982.

[3] - Mayne D.Q., Sahba M.,"Design of feedback controllers for nonlinear systems", International Conference on Control and its Applications, University of Warwick, U.K., pp. 276-280, 1981.

[4] - Gustafson C.L., Desoer C.A.,"Controller design for linear multivariable feedback systems with stable plants, using optimization with inequalitiy constraints", Memo No. UCB/ERL M81/51, July 1981, Electronic Research Lab., University of California, Berkeley.

[5] - Su R.,"On the linear equivalents of nonlinear systems", Systems and Control Letters, Vol 2, pp. 48-52, 1982.

[6] - Yaun J.S.,"Dynamic decoupling of remote manipulator system", IEEE Trans. AC 23, pp. 713-717, 1978.

[7] - Horn K.P., Hirokawa K., Vazirani V.V.,"Dynamics of three degree of freedom kinematic chain", A.I. Memo 478, Massachussets Inst. of Tech. Artificial Intelligence Laboratory, 1977.

[8] - Gould D.J.,"Robot arm control using inverse system method", MSc Thesis, Imperial College, 1979.

[9] - Freund E.,"The structure of decoupled nonlinear systems", International Journal of Control, Vol 21, pp. 443-450, 1975.

[10] - Freund E., Syrbe M.,"Control of industrial robots by means of microprocessors", Colloque IRIA, Versailles, Rocquencourt, 1976.

[11] - Ali A., Taylor P.M.,"Nonlinear control of an industrial robot", Proc. IEEE Conf. on Application of Adaptive and Multivariable Control, University of Hull, pp.19-21, 1982.

[12] - Hewit J.R. Padova J.,"Decoupled feedback control of robot and manipulator arms", Proc. of Third International CISM-IFTOMM Symposium on Theory and Practice of Robots and Manipulators, Udine, pp. 251-266, 1978.

[13] - Hewit J.R., Burdens J.R., "Fast Dynamic decoupled control for robotics using adaptive force control", Report, Department of Mechanical Eng., University of Newcastle upon Tyne, 1980.

[14] - Elmquist H.,"SIMNON:- An interactive simulation program for nonlinear systems", Proc. Simulation 77, Montreaux 1978.

[15] - Hewit J.R.,"Decoupled control of robot movement", Electronics Letters, Vol 15, pp. 670-671, 1979.

[16] - Sahba M.,"Computer aided design of nonlinear controllers for torque controlled robot arms", To appear in IEE -D.

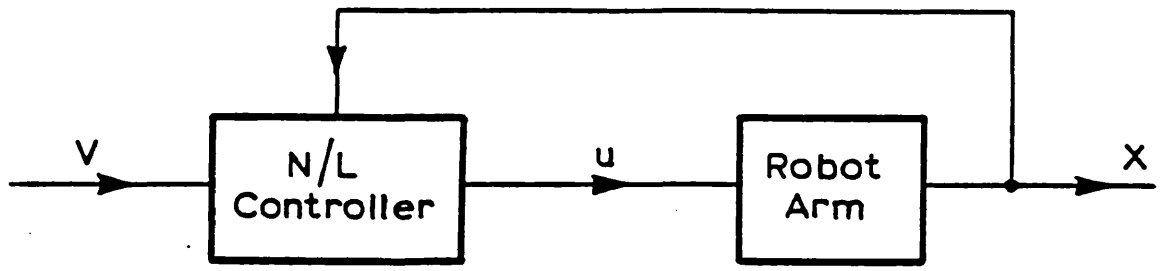

Fig. 7.1 Controller Structure

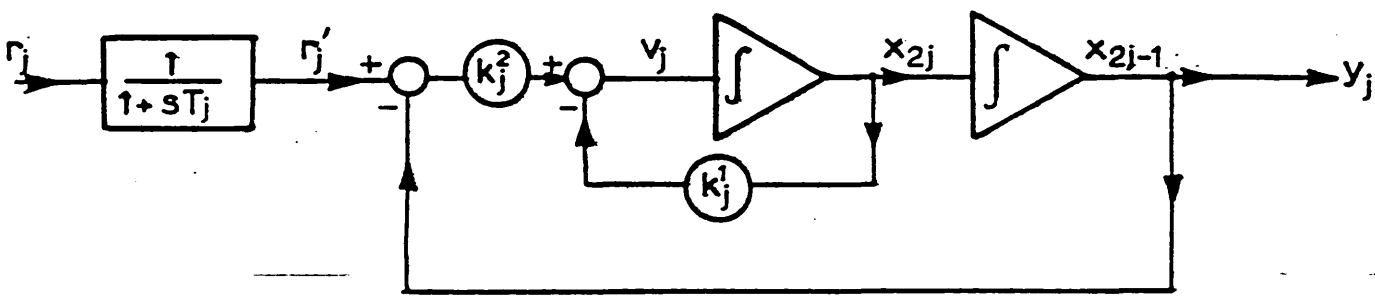

Fig. 7.2 Equivalent Closed Loop System

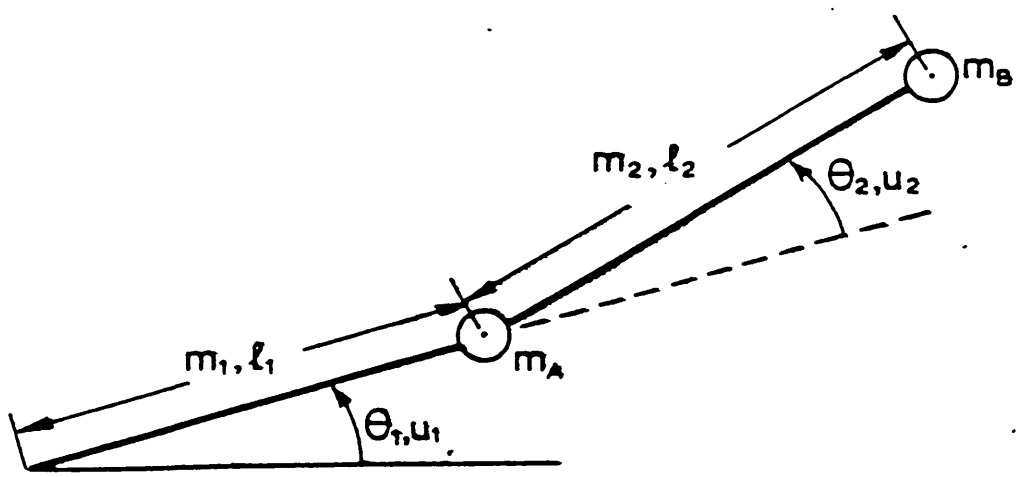

Fig. 7.3 Robot Arm with Two Degrees of Freedom

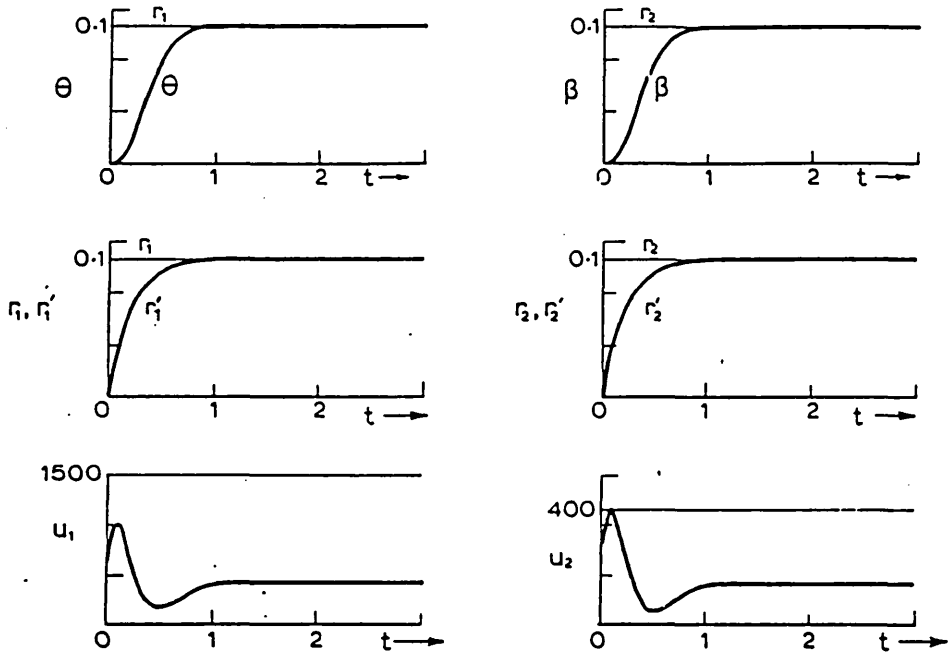

Fig. 7.4 Simulation Results for m_B = 20 kg

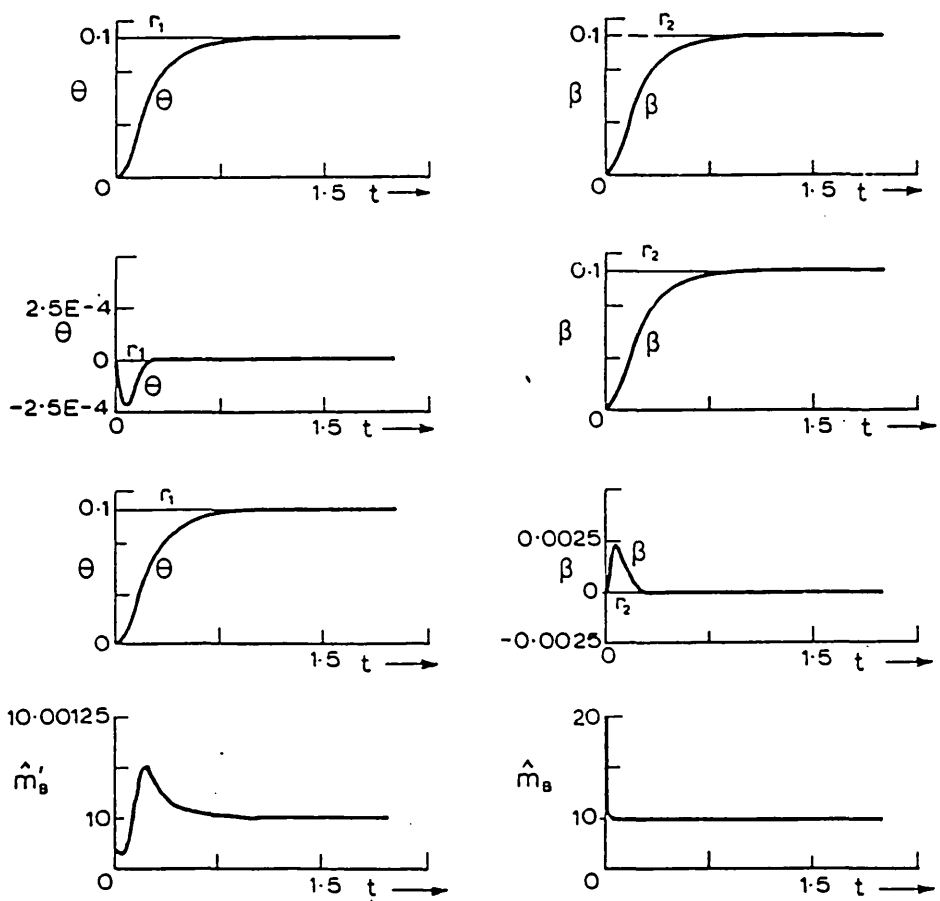

Fig. 7.5 Simulation results for m_B = 20 kg with mass estimator

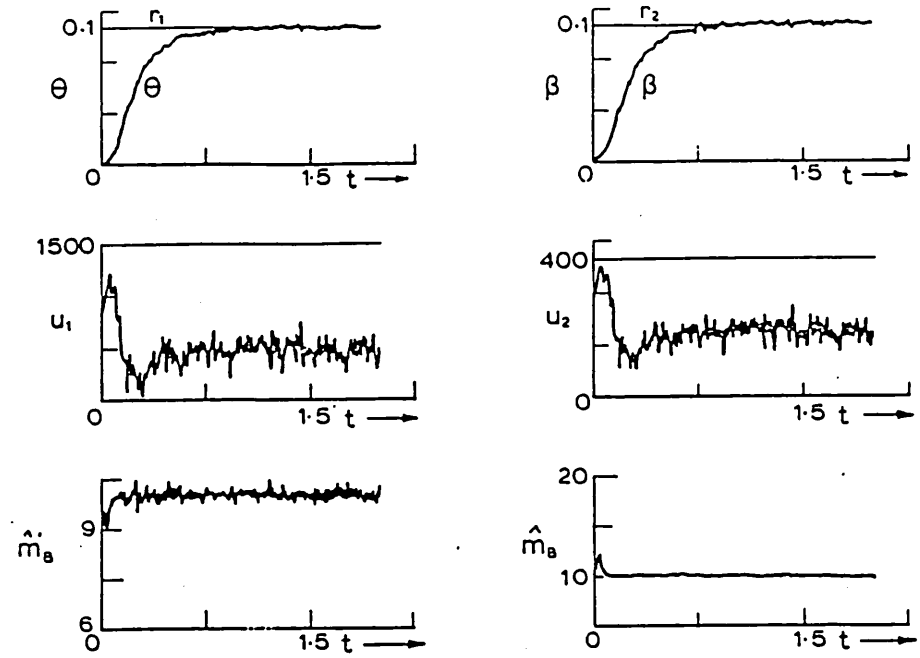

Fig. 7.6 Simulation Results for m_B = 10 kg with mass estimator and observer

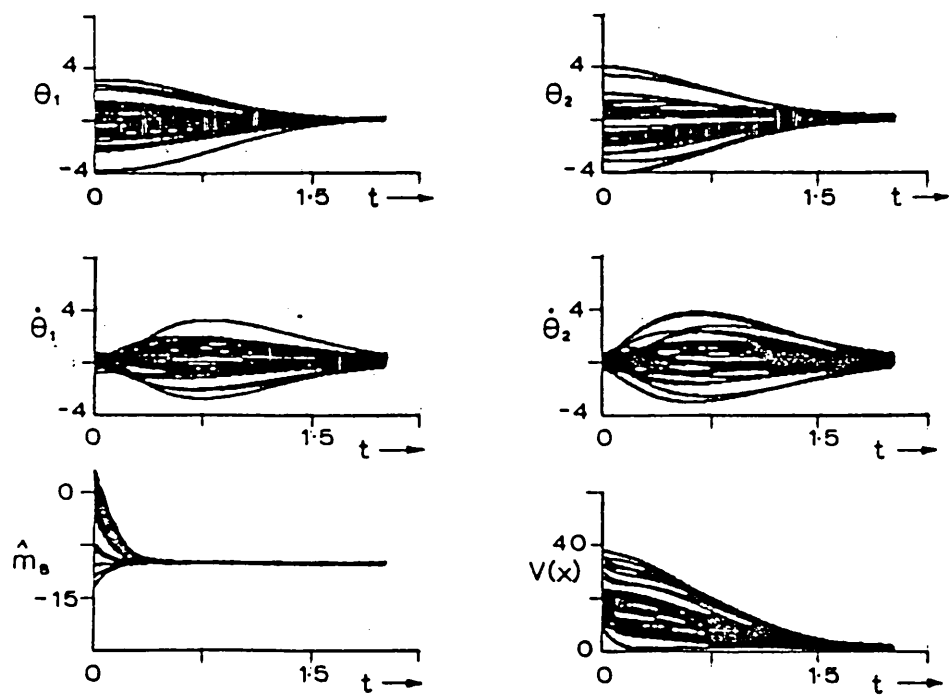

Fig. 7.7 Simulation Results for m_B = 10kg with mass estimator for 20 starting points

CHAPTER 8

OPTIMAL CONTROL OF POWER SYSTEM GENERATORS INCORPORATING NONLINEAR STATE FEEDBACK

8.1 - INTRODUCTION

Two types of stability of power systems are defined: steady state stability and transient stability. The steady state stability of the system is the capability of the system to withstand small disturbances (normal fluctuations), whereas the transient stability is the ability of the generators to regain and maintain synchronism after a large and sudden disturbance (faults, switchings). In a power system the mechanical power delivered from the turbine to the generator is converted to electrical power and transformed to the load. After a disturbance the balance between the mechanical and electrical power is changed, causing the generator speed to vary. There are three ways of controlling such a generator to maintain synchronism with the rest of the system and to provide good damping [1]:

(a) A signal may be sent to the governor system to change the mechanical input power.
(b) The voltage regulator setting may be varied to change the terminal voltage and consequently electrical power output of the generator.
(c) The shape of the network (load) may be changed.

The improvement obtained by voltage regulators (AVR) on system steady state and transient stability has been well established [2 - 8]. Various feedback signals in addition to terminal voltage, have been proposed and used [4 - 8]. The use of stabilizing signals in both the voltage regulator and the turbine governor has been studied [9, 10]. Frequency response methods (classical or modern multivariable techniques), modern linear multivariable state space techniques, and optimal control based techniques have been employed for the design of additional controllers for the excitation and governor loops. Several papers have been devoted to the first two techniques [4, 9, 11]. Most of the work on optimal performance of a power system has mainly employed linearized models of the system. An optimal controller for such a system consists of a linear feedback for all the state variables and is obtained through solving the matrix Riccati equation. Some work has also been done to obtain sub-optimal control employing feedbacks of output variables by an indirect method using the Liapunov equation. A few authors have considered nonlinear feedback controllers [1, 12]. The optimization techniques used, so far, are generally unable to satisfy design constraints. More elaborate techniques have been recently developed [15 - 19] (also chapters 1 to 5); these can be used as a practical tool to optimize a cost function (by adjusting a set of parameters) to satisfy all the constraints. In this Chapter feedback controllers (linear and nonlinear) are designed for a seventh order nonlinear model under a severe disturbance to satisfy stability and performance constraints; sensitivity analysis is then used to eliminate unnecessary

feedback signals. The closed loop system is then subjected to disturbances and its behaviour is assessed.

8.2 - SYSTEM REPRESENTATION AND STATE EQUATIONS

This study has been performed on a single machine model of a power system using the parameters of a typical generator described in Reference 1. The generator is coupled through a transformer and a double circuit transmission line to a large system as shown in Fig. 8.1. The state equations for this system are given below:

$$\dot{\delta} = \omega$$

$$\dot{\omega} = -(k/J)\omega - (1/J)M_t + M_e/J$$

$$\dot{I}_f = D_4 I_f + D_1 V_f + V_{mb}(D_2\omega \sin\delta + D_3\omega\cos\delta)$$

$$\dot{V}_e = -(1/t_a)V_e - G_a V_t/t_a + G_a/t_a(V_r + u_1) \qquad (8.2.1)$$

$$\dot{V}_f = -(G_e/t_e)V_e - (1/t_e)V_f$$

$$\dot{A} = (G_g/t_v)\omega - (1/t_v)A + (1/t_v)(Y_0 + u_2)$$

$$\dot{M}_t = (1/t_s)A - (1/t_s)M_t$$

where

$$M_e = C_1 V_{mb}^2 \sin^2\delta + C_2 V_{mb}^2 \cos^2\delta + C_3 I_f^2 + C_4 V_{mb}^2 \sin\delta \cos\delta$$

$$+ C_5 V_{mb} I_f \sin\delta + C_6 V_{mb} I_f \cos\delta \qquad (8.2.2)$$

$$D_1 = \omega_0/a$$

$$D_2 = -(x_{qe}x_{md})/Z_f^3$$

$$D_3 = -(x_{md}r_{ae})/Z_f^3$$

$$D_4 = -\omega_0 r_f/a$$

$$Z_f^3 = Z^2(x_{md} + x_f) - x_{md}^2 x_{qe}$$

$$Z^2 = r_{ae}^2 + x_{qe}x_{de}$$

$$a = Z_f^3/Z^2$$

$$C_1 = 0.5(x_d - x_q)r_{ae}x_{de}/Z^4$$

$$C_2 = -0.5(x_d - x_q)r_{ae}x_{qe}/Z^4$$

$$C_3 = 0.5x_{md}^2 r_{ae}(x_q^2 + r_{ae}^2)/Z^4$$

$$C_4 = 0.5(x_d - x_q)(r_{ae}^2 - x_{de}x_{qe})/Z^4$$

$$C_5 = (1/2Z^4)(x_d - x_q)(r_{ae}^2 - x_{de}x_{qe})x_{md} + (1/2Z^2)x_{md}x_{de}$$

$$C_6 = r_{ae}x_{md}[-(x_d - x_q)x_q/Z^2 + 1/2]/Z^2$$

and

$$V_t = [(v_d^2 + v_q^2)/2]^{1/2} \qquad (8.2.3)$$

where

$$v_d = l_1 \sin\delta + l_2 \cos\delta + l_3 i_f$$

$$v_q = f_1 \sin\delta + f_2 \cos\delta + f_3 i_f$$

$$l_1 = V_{mb}(x_q x_{de} + r_a r_{ae})/Z^2$$

$$l_2 = V_{mb}(x_q r_{ae} - r_a x_{qe})/Z^2$$

$$l_3 = (x_q x_{md} r_{ae} - r_a x_{qe} x_{md})/Z^2$$

$$f_1 = V_{mb}(-x_d r_{ae} + r_a r_{ae})/Z^2$$

$$f_2 = V_{mb}(x_d x_{qe} + r_a r_{ae})/Z^2$$

$$f_3 = (x_d x_{qe} x_{md} + r_a x_{md} r_{ae})/Z^2 - x_{md}$$

$$r_{ae} = r_a + r_t + r_e$$

$$x_{ae} = x_a + x_t + x_e$$

$$x_{de} = x_d + x_t + x_e$$

$$x_{qe} = x_q + x_t + x_e.$$

The system parameters together with the base values are given in Table 1. The parameters are those of a 588 MVA CEGB generator.

G_a	Voltage regulator amplifier gain	0.001
G_e	Exciter gain	5.56
G_g	Governor speed gain	0.0709
t_a	AVR amplifier time constant (sec)	0.05
t_e	Exciter time constant (sec)	0.05
t_s	Entrained steam time constant (sec)	0.3
t_v	Turbine valve time constant	0.05
r_a	Armature resistance	0.00115
r_e	Transmission line resistance	0.0209
r_f	Field resistance	0.00114
r_t	Transformer resistance	0.0044
x_d	Direct axis synchronous reactance	2.98
x_e	Transmission time reactance	0.3333
x_q	quadratic axis synchronous reactance	2.83
x_t	Transformer reactance	0.157
x_{md}	Magnetising reactance	2.82
x_{mq}	Magnetising reactance	2.67

Table 1: System Parameters

In the steady state all the derivatives of the state variable are zero and a set of algebraic equations is solved to give the initial steady state conditions.

δ Rotor angle ($\bar{\delta} = -1.442$)

ω	Rotor speed ($\bar{\omega} = 0.0$)
I_f	Field current ($\bar{I}_f = -1.4855$)
V_e	Exciter voltage ($\bar{V}_e = 3.04 \times 10^{-4}$)
V_f	Field voltage ($\bar{V}_f = -1.69 \times 10^{-3}$)
A	Valve position ($\bar{A} = 0.86087$)
M_t	Mechanical torque ($\bar{M}_t = 0.86087$)
V_t	Terminal voltage ($\bar{V}_t = 1.0$)
$\bar{P}$	Active power at the generator ($\bar{P} = -0.847$)
$\bar{Q}$	Reactive power at the generator ($\bar{Q} = -0.276$)

Table 2: Steady state values with δ in rad, $\dot{\delta}$ in rad/sec, and other variables in p.u.

8.3 - OPTIMIZATION OF SYSTEM PERFORMANCE WITH LINEAR CONTROLLER

The controller $u = (u_1, u_2)^T$ is introduced as shown in Fig. 8.1. A linear controller is employed i.e.

$$u = Zx \tag{8.3.1}$$

where Z is a matrix of design parameters and

$$x \triangleq (\delta - \bar{\delta},\ \omega - \bar{\omega},\ I_f - \bar{I}_f,\ V_e - \bar{V}_e,\ V_f - \bar{V}_f,\ A - \bar{A},\ M_t - \bar{M}_t) \tag{8.3.2}$$

The general quadratic performance index I, defined as

$$I = \int_0^T (x^T Q x + u^T H u) dt \qquad (8.3.3)$$

where $x = x(t)$ is the solution of eqn. (8.2.1) with initial state $\bar{x}$ given in Table 2 and control $u \triangleq u(t) = Zx(t)$. This reduces to

$$I = \int_0^T x^T (Q + Z^T H Z) x dt \qquad (8.3.4)$$

The major objectives in design of controllers for power systems are :

(a) the reduction of the first rotor excursion in response to a large disturbance (improvement of transient stability).
(b) the rapid recovery of the terminal voltage.

To satisfy the above objectives, the choice of weighting matrices Q and H is important (some guidelines which help the choice of these weighting matrices are given in Reference 1). In this study H and Q are chosen to be diagonal and are given by

$$Q = \text{diag}\,[10.0,\ 1.0,\ 1.0,\ 1.0,\ 1.0,\ 1.0,\ 1.0]$$

$$H = \text{diag}\,[0.001,\ 0.01] \qquad (8.3.5)$$

In general, the synchronous machine field voltage V_f and excitation voltage V_e are limited. The ceiling values are chosen as ± 3 times the rated load value [1, 14]. On the other

hand the steam valve position cannot be greater than 1 (fully open) or less than 0 (fully closed). The performance of the system when subjected to a large disturbance in the form of a three-phase short circuit of 100 ms at high voltage busbar is to be optimized so that all the constraints are satisfied. The system performance is, often, optimized for a particular set of operating (steady state) conditions. It is, however, important that the final system with optimal controllers should not be sensitive to changes in initial system operating conditions. We shall, therefore, optimize the system performance over a range of initial systems operating conditions.

8.4 - ALGORITHM

The design problem can be expressed as either

(a) Determine a z such that $\varphi(z, a) \leqslant 0$ for all $a \in B$, or

(b) Minimize $\{c(z) \mid \varphi(z, a) \leqslant 0, a \in B\}$

where B is an infinite dimensional set (in this study z is vector of elements of the control matrix Z, $\varphi(z, a) \leqslant 0$ summarizes the design constraints and B is the set of all initial system operating conditions). Algorithms for these problems, with proven convergence, have been fully described in Chapters 1, 3, 4 and 5.(also [15 - 19]). Here we have used the algorithm presented in Chapter 4 to solve problem (i). The problem can be reformulated as: to find a pair (z, α) which minimizes α subject to

$$I(z) = \int_0^T x^T(Q + ZHZ)xdt \leq \alpha$$

$$|V_f(t, z)| \leq 0.006 \quad \text{for } 0 \leq t \leq T$$

$$|V_e(t, z)| \leq 0.001 \quad \text{for } 0 \leq t \leq T$$

$$0 \leq A(t; z) \leq 1 \quad \text{for } 0 \leq t \leq T$$

SIMNON, a command driven interactive simulation program for nonlinear systems [20], which is implemented on a Perkin Elmer 8/32 computer and includes the algorithm of Chapter 4, has been used for the above optimization problem. We approximate this by choosing α, forcing z to satisfy the above set of constraints and then reducing α interactively. Fig. 8.2 shows the response of the system with a conventional controller [1] and Fig. 8.3 shows the response of the system, starting from the steady state values, with the optimal linear controller. It may be possible to choose the parameters of the conventional controller to give a more acceptable response. The effect on the optimal feedback gains of small changes (less than 10%) in the operating conditions (initial values) is negligible. Controller matrix Z is given by

$$Z = \begin{bmatrix} 0.008, & -0.571, & -0.371, & -0.0014, & 0.0023, & -0.179, & -0.073 \\ -0.222, & -0.333, & 0.296, & -0.0042, & 0.00179, & 0.3199, & 0.2319 \end{bmatrix}$$

8.5 - NONLINEAR CONTROLLER DESIGN

It is desired to design a controller which provides the system with good damping for small disturbances and a high loop gain during large disturbances. A nonlinear controller is suggested which has these advantages; the linear part provides high damping for small disturbances and the nonlinear component dominates the performance during large disturbances. It is obvious that such a controller (F) must be a function of state variables and the optimal controller (u) can be expanded into a Taylor series about a nominal point (u_0) to give [12]:

$$u - u_0 = \left[\frac{\partial F}{\partial x}\right]_{u=u_0} \partial x + \left[\frac{\partial^2 F}{\partial x^2}\right]_{u=u_0} \partial x^2 + \ldots \qquad (8.5.1)$$

The linear optimal controller designed in the previous section accounts only for the first order term in (8.5.1), the second order terms will now be added to the control signals u_1 and u_2 to give

$$u_1 = z_1^T x + x^T R_1 x$$

$$u_2 = z_2^T x + x^T R_2 x \qquad (8.5.2)$$

where R_1 and R_2 are upper triangular matrices of the design parameters and z_1^T, z_2^T are the first and second rows of Z. Fig. 8.4 shows the response of the system with an optimal nonlinear controller. This figure clearly shows the improve-

ment in both rotor angle and terminal voltage. The performance index, however, may not be sensitive to all 70 terms in the controllers (7 linear and 28 nonlinear terms in each controller). Hence, we now eliminate some of the feedback signals (i.e. replace elements of z_i, R_i, i = 1,2 by zero), subject to the condition that the change in the performance index is less than 5%. The system with simplified nonlinear controller (total of 22 linear and nonlinear terms) has been reoptimized and the results are illustrated in Fig. 8.5. In this Figure variations of rotor angle, terminal voltage, mechanical torque, AVR setting and governor setting (control signals) are shown. The controllers are given by

$$u_1 = -.049x_2 - 0.57x_3 - 0.76x_1^2 + 0.39x_1x_2$$

$$+0.563x_1x_7 + 0.07x_2^2 - 1.567x_3^2 - 1.69x_3x_6 - 2.0x_3x_7 + 1.7x_6^2$$

$$u_2 = -0.154x_2 - 0.524x_3 + 0.524x_6 - 1.47x_1^2 - 0.634x_1x_2$$

$$+ 2.63x_1x_3 + 1.675x_1x_6 + 0.864x_1x_7 - 0.1x_2^2 + 0.184x_2x_7$$

$$- 138.68x_4x_6 + 0.822x_6x_7.$$

In a further study, any term in the controller which did not change the performance index more than 10% was eliminated. The total number of linear and nonlinear terms in the controller is now reduced to 9. The system has been reoptimized and the results are shown in Fig. 8.6. The controllers are given by

$$u_1 = 0.0508x_2 - 1.4523x_3 - 0.604x_3^2 - 0.1005x_3x_7$$

$$u_2 = -0.6536x_6 - 0.149x_1x_6 - 0.8065x_1^2 + x_1x_3 + 0.215x_1x_7$$

Figures 8.7 and 8.8 compare the variations of rotor angle and terminal voltage of the system controlled by conventional controllers with system controlled by optimal linear, optimal nonlinear, optimal reduced 22-term nonlinear controllers and optimal reduced 9-term nonlinear controllers. These figures show that the (full and reduced) nonlinear controller has decreased the first swing while the system damping is as good as (or even better than) that of the linear optimal controller.

8.6 - SYSTEM PERFORMANCE UNDER SMALL DISTURBANCES

In this section, system performance with reduced nonlinear controllers under small disturbances is studied. The disturbance chosen is a 10% variation of the system voltage (infinite busbar) for a 100 ms. Fig. 8.9 shows the performance of the system under such a disturbance. This figure compares the performance of the system with the 22-term controller with the 9-term controller. It is clear that the 22-term controller performs much better under small disturbances.

8.7 - DISCUSSION

The use of linear and nonlinear state feedbacks in the AVR and governor settings of the nonlinear model of a power sys-

tem generator has been investigated. The optimization algorithm of Chapter 4 has been employed to minimize a performance index subjected to a set of constraints. The design is based on satisfaction of performance constraints by analysing the nonlinear system equations. Most other designs are based on linearized models of the system. An attempt to improve the damping in rotor oscillations and to achieve fast recovery of the terminal voltage following a large system disturbance with a combination of linear and nonlinear state feedbacks was successful. It is shown that a good performance can be obtained using reduced nonlinear feedbacks. The optimal reduced 22-term controller performs very well under small disturbances. In the nonlinear controller (F) only first and second order terms are considered; higher order terms may permit better reponse. A more comprehensive study must consider the performance of the system under a wide range of operating conditions. The effect of fault detection time on the system performance must also be considered. The designs were carried out interactively. It is, therefore, not possible to give a good estimate of the computing time required for each design.

8.7 - REFERENCES

[1] - Vaahedi E.,"Optimal control of power generators using self-tuning state estimators", PhD Thesis, Imperial College, 1979.

[2] - Jacovides L.J., Adkins B.,"Effect of excitation regulation on system machine stability", Proc. IEE Vol 113, pp. 1021-1034, 1966.

[3] - Dinely J.L., Kennedy M.W.,"Concept of synchronous generator stability", ibid.,Vol 111, pp. 95-97, 1964.

[4] - Demello F.P., Concordia C.,"Concepts of synchronous machine stability as defined by excitation control", IEEE Trans., PAS 88, pp. 317-329, 1969.

[5] - Shier R.M., Blythe A.L."Field tests of dynamic stability using a stabilizing signal and computer program verification", ibid., PAS 87, pp. 315-322, 1968.

[6] - Schief F.R., Hunkins H.D., Martin G.E., Hattan E.E.,"Excitation control to improve powerline stability", ibid, PAS 87, pp. 1426-1434, 1968.

[7] - Dandeno P.L., Karas A.N., McClymont K.R.,Watson W.,"Effect of high speed rectifier exitation systems on generator stability limits", ibid., PAS 87, pp. 190-201, 1968.

[8] - Dinely J.L., Morris A.J. Preece C.,"Optimized transient stability from excitation control of synchronous generator", ibid., PAS 87, pp. 1696-1705, 1968.

[9] - Coles H.E.,"Effect of prime-mover governing and voltage regulation on turbo alternator performance", Proc. IEE, Vol 112, pp. 1395-1405, 1965.

[10] - Hughes F.M.,"Improvement of turbo alternator transient performance", ibid., Vol 120, pp. 233-240, 1973.

[11] - Marshal W.K., Smolinski W.J.,"Dynamic stability determination by synchronizing and damping torque analysis", IEEE Trans., PAS 92, pp. 266-279, 1968.

[12] - Chana G.S., Daniels A.R.,"Turbogenerator excitation control incorporating nonlinear state feedback", Proc. IEE, Vol 125, pp. 1245-1246, 1978.

[13] - Moya O.E.O,"Optimal transient stability control of sychronous generators by on line digital computer", PhD Thesis, Imperial College 1976.

[14] Iyer S.N., Cory B.J."Optimization of turbo generator transient performance by differential dynamic programming", IEEE Trans., PAS 90, pp. 2149-2157, 1971.

[15] - Polak E., Mayne D.Q.,"An algorithm for optimization problems with functional inequality constraints", ibid.,AC 21, pp. 184-193, 1976.

[16] - Mayne D.Q., Polak E., Trahan R.,"An outer approximation algorithm for computer aided design problems", JOTA Vol 28, pp. 331-352, 1979.

[17] - Gonzaga C., Polak E.,"On constrained dropping schemes and optimality functions for a class of outer approximation algorithms", SIAM J. Control and Optimiz., Vol 17 pp. 477, 1979.

[18] - Mayne D.Q., Polak E., Heunis A.J.,"Solving nonlinear inequalities in a finite number of iterations", JOTA, Vol. 33, pp. 207-222, 1981.

[19] - Sahba M.,"A derivative free algorithm for solving inequalities", submitted for publication to Journal of

Mathematical Prog. Study.

[20] - Elmquist H.,"SIMNON: an interactive simulation program for nonlinear systems", Proc. Simulation 77, Montreaux, 1978.

[21] - Sahba M.,"Optimal control of power system generators incorporating nonlinear state feedback", IEE-D, Vol 130, pp. 359-362, 1983.

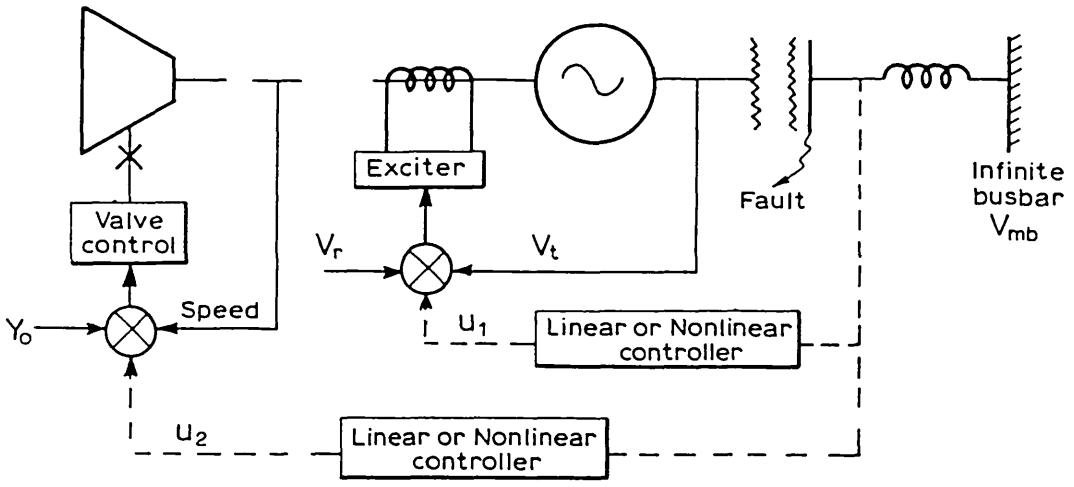

Fig. 8.1 Basic System with Proposed Controllers.

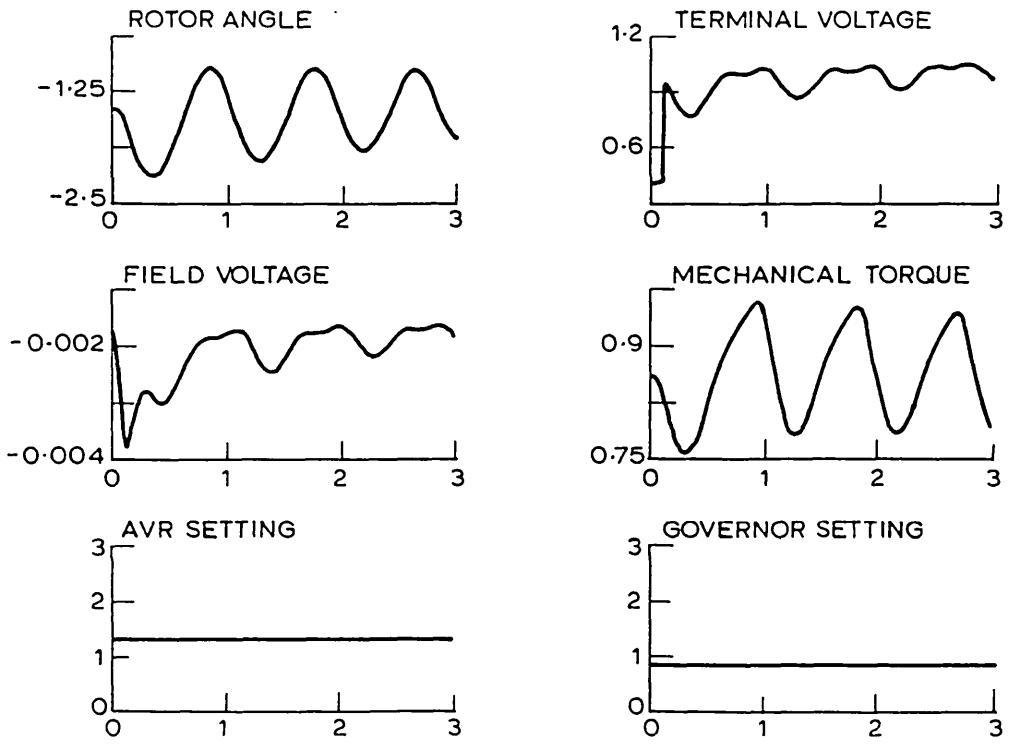

Fig. 8.2 System performance with conventional controllers following a large disturbance.

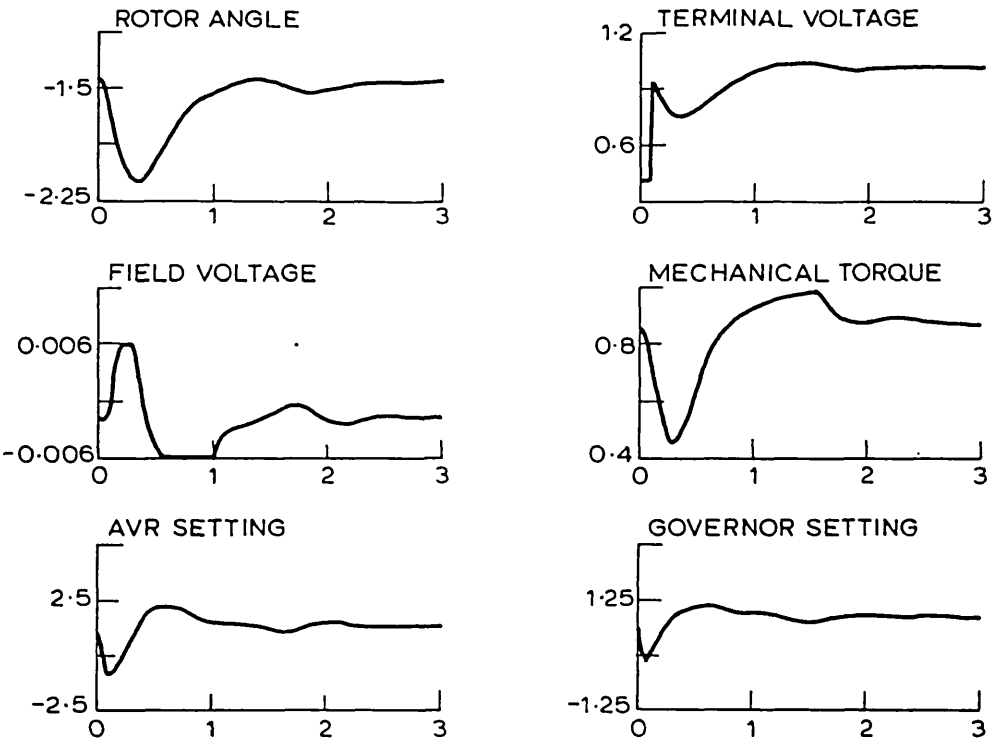

Fig. 8.3 System performance with optimal linear controllers following large disturbance.

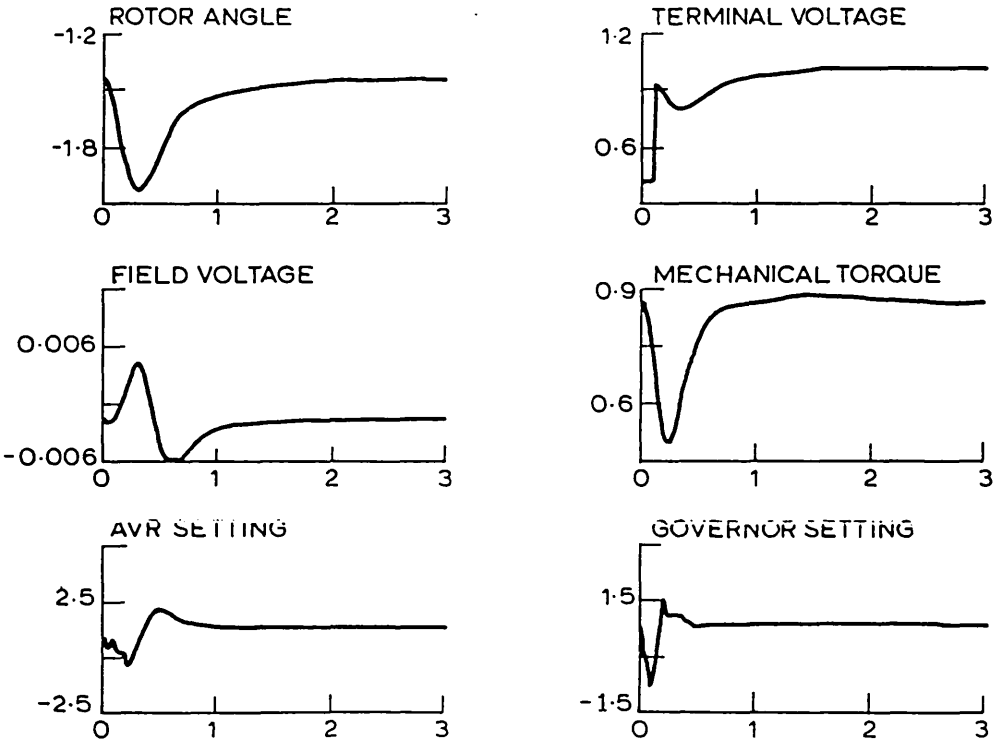

Fig. 8.4 System performance with optimal nonlinear controllers following a large disturbance.

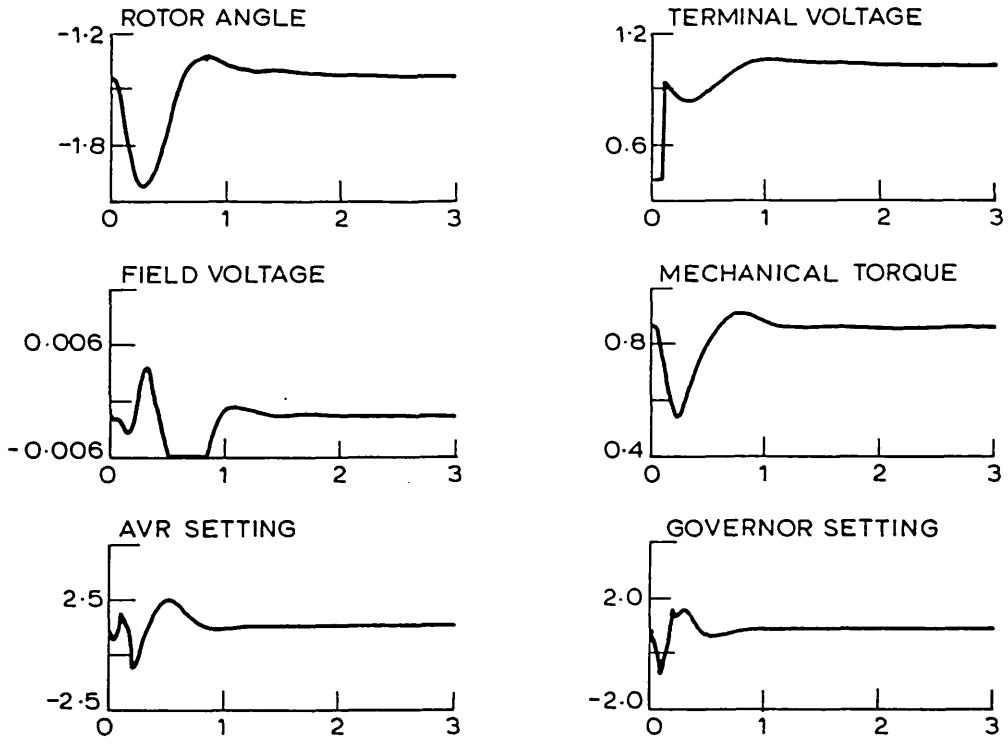

Fig. 8.5 System performance with optimal reduced (total of 22 terms) nonlinear controllers following a large disturbance.

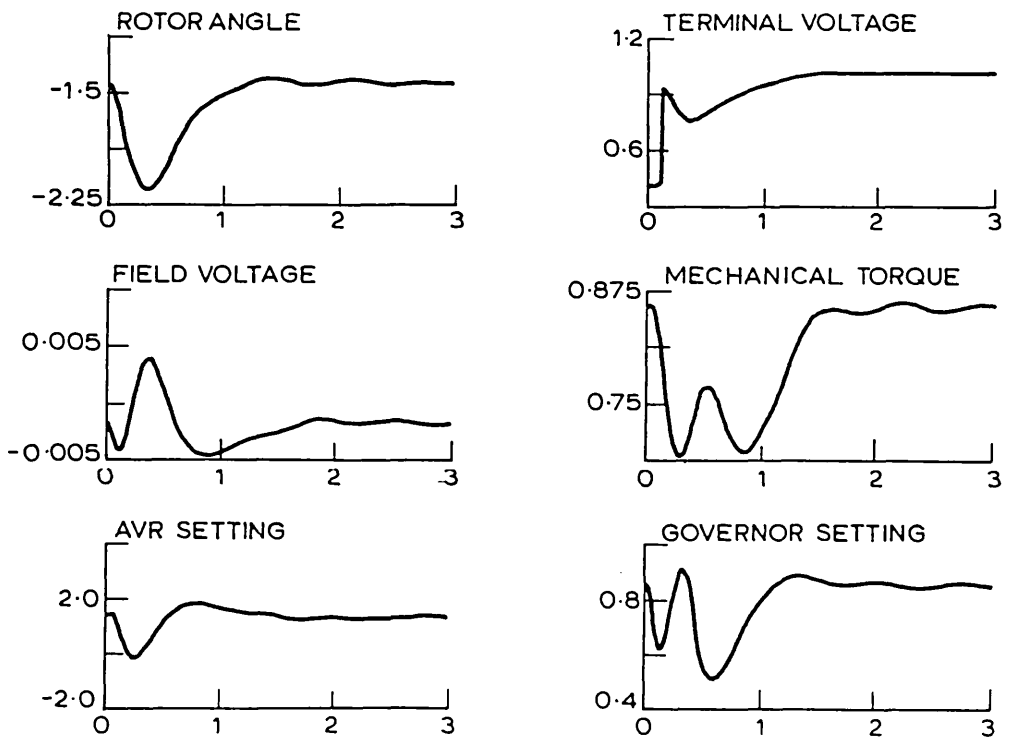

Fig 8.6 System performance with optimal reduced (total of 9 terms) nonlinear controllers following a large disturbance.

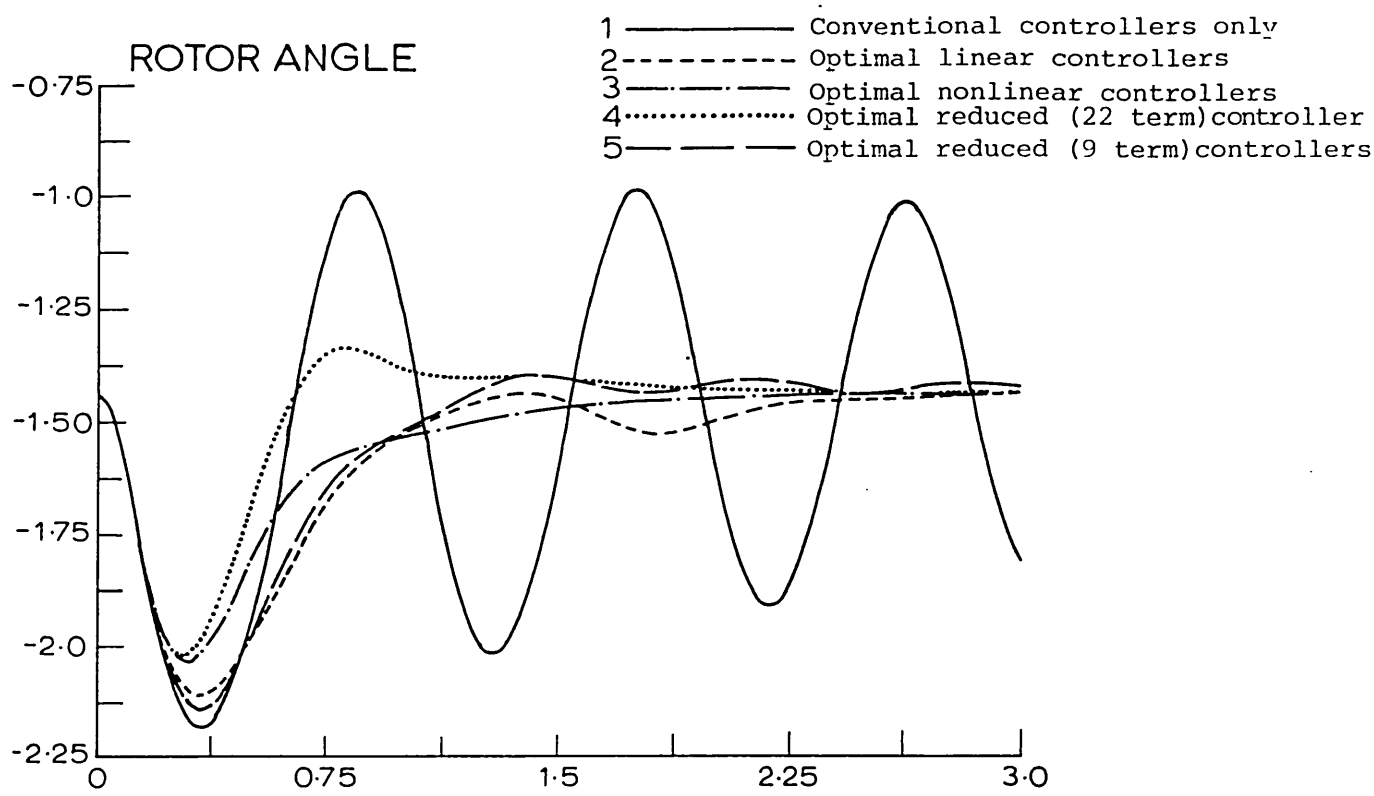

Fig. 8.7 Load angle swing following a large disturbance.

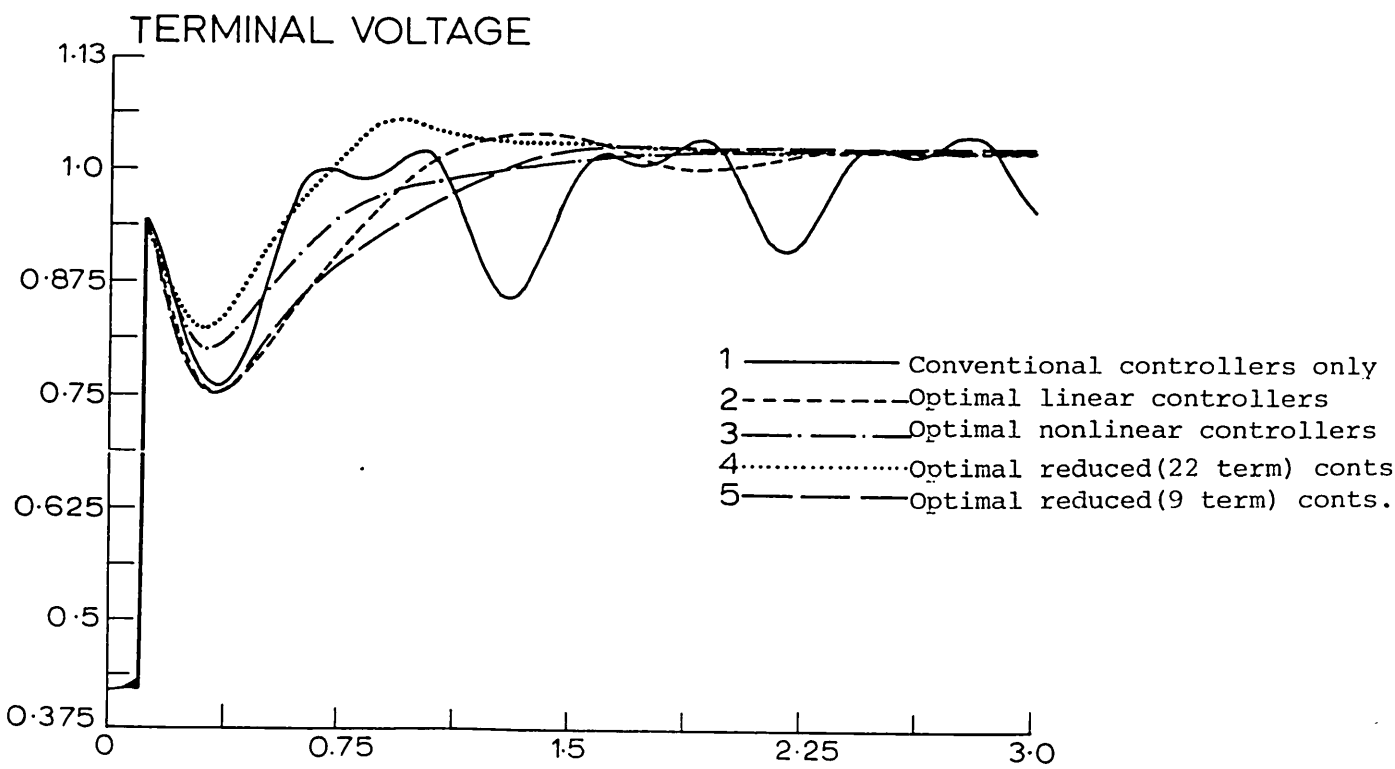

Fig. 8.8 Terminal voltage recovery following a large disturbance.

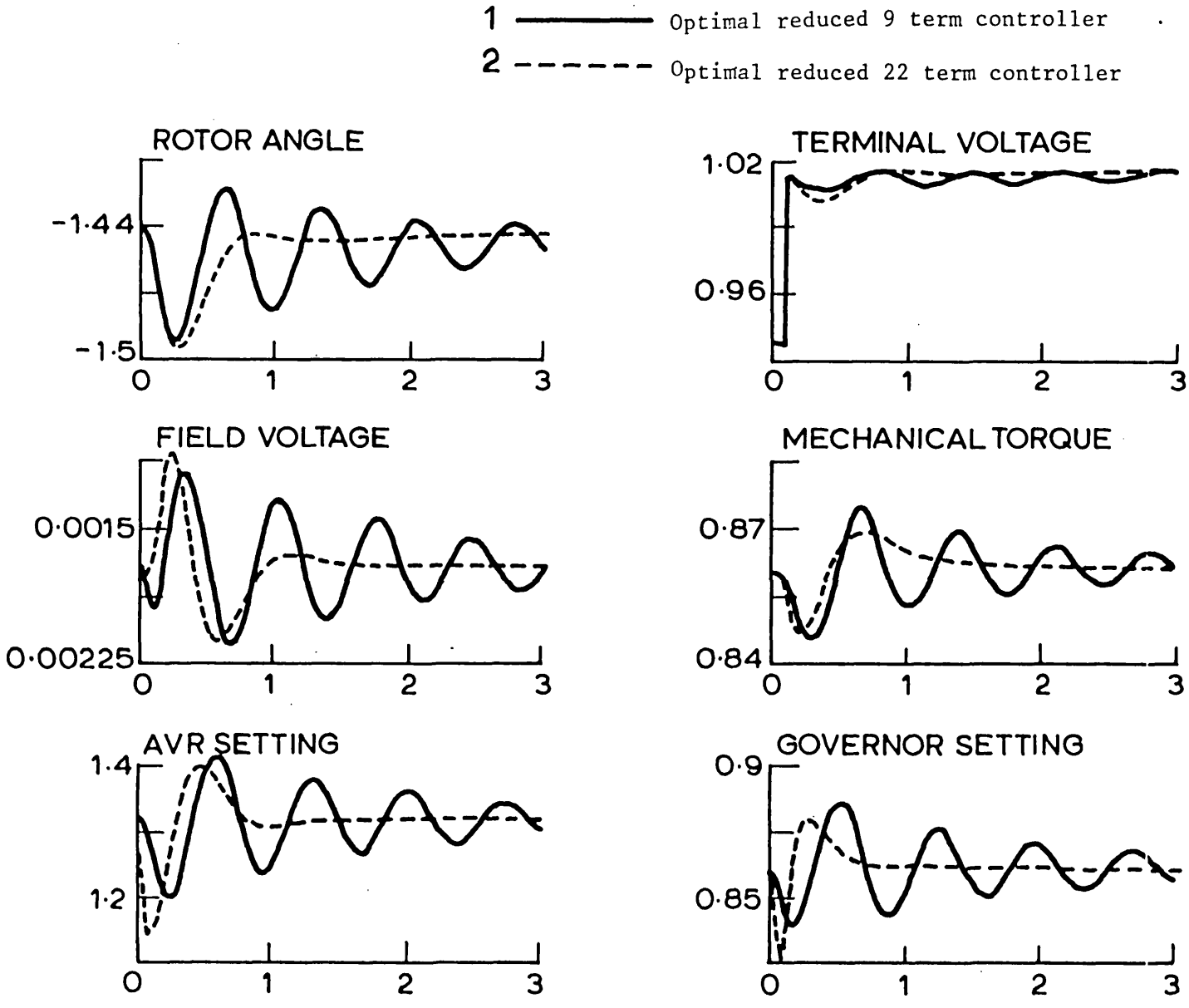

Fig. 8.9 System performance with reduced order controllers under small disturbance

CHAPTER 9

CONCLUSIONS

This thesis has shown how optimization techniques can be used along with system theory to design robust controllers which satisfy time domain performance and stability objectives. It has been argued that an ideal design methodology should combine control theoretic results (to choose the structure of the controller) with sophisticated mathematical techniques (to adjust controller parameters to satisfy design constraints). Since frequency response design methods can be used only in special cases and since most design constraints are formulated in the time domain, a design methodology is proposed which is based on multiple simulations. To test stability by simulation it would appear necessary to compute the system trajectory for all initial states and for all $t \in [0, \infty)$. However, it has been shown that the duration of the simulation may be reduced to a finite interval $[0, T]$, where T may be very small. It has been shown that the stability constraints may be expressed as functional (infinite dimensional) constraints, and it has also been shown how existing outer approximation algorithms can be employed to replace the functional constraints with a collection of simple inequality constraints.

In Chapter 3, an alternative Newton type algorithm for solving inequalities has been proposed and analysed. The algorithm always employs a Newton step directed to the interior

of the linearized feasible set to ensure rapid convergence in a finite number of iterations. The linearized feasible set is expanded, when necessary, to ensure the existence of a search direction. The algorithm is thus particularly efficient in high order problems where the linearized set is often empty. When the stopping condition is removed the algorithm possesses a quadratic rate of convergence.

In Chapter 4, two derivative free versions of the above algorithm have been presented. One algorithm uses only finite difference approximations for the gradient, while in the second algorithm a mixture of finite difference and Broyden type approximations of the gradient matrix are employed. The algorithms have been analysed and proven to find a solution in a finite number of iterations.

In Chapter 5, an algorithm for constrained minimization is presented. This algorithm, under mild assumptions, is globally convergent. In this algorithm several features are employed which are not present in the previous algorithms. All the algorithms known (by the author) assume, at best, that the columns of the gradient matrix of the most active constraints are linearly independent. This assumption has been further relaxed to positive linear independence of the most active constraints and by employing a linear program, a feasible solution to the Newton step has been ensured. A new feature of the algorithm is the rule for choosing the penalty parameter c, which allows c to decrease. Numerical results show good performance.

In Chapter 6, the design methodology developed in previous chapters has been successfully applied to a double inverted pendulum system. The results of this chapter clearly show the advantages of our alternative approach to feedback design of nonlinear systems. A minimal order observer and a functional order observer have been designed and the response of the system to a reference step input and a step input disturbance has been simulated.

In Chapter 7, the design of a nonlinear controller for torque controlled robot arms has been investigated. It has been shown that the system belongs to the class of systems that can be transformed into (equivalent) linear systems by means of a bijective transformation of state and control. The design methodology presented in this chapter utilizes this transformation to make the input-output map linear and then employs linear feedback (for the transformed system). The controller has been parametrized and designed to satisfy design constraints. The resultant (linear) controller is then transformed yielding a nonlinear controller for the original system. With this approach, stability of nominal closed loop system is ensured by the controller structure and its parametrization. It has also been shown how to modify the controller when the state is not completely accessible and when the load mass is unknown. Simulation results show that the resultant system performs well under a range of test conditions. Although robustness constraints were not included at the design stage, repeated simulation showed that the nominal design copes well with a wide range of errors in load mass

and friction coefficients.

In Chapter 8, The use of linear and nonlinear state feedbacks in the AVR and governor settings of the nonlinear model of a power system generator has been investigated. While most other designs are based on the linearized model of the system, our design method is based on satisfaction of performance constraints by analysing the nonlinear system equations. The system with a controller, consisting of linear and nonlinear state feedbacks, has achieved rapid recovery of the terminal voltage and has improved the damping in rotor oscillations following a large system disturbance. It has been shown that a good performance can be obtained using a reduced nonlinear feedback controller.

A important requirement in the above design procedure has been an interactive design system. The Lund nonlinear simulator package SIMNON was chosen as an interactive simulation base. SIMNON was implemented on a PERKIN ELMER 8/32 computer and further developed to include new commands and algorithms. A compiler was written to translate the code generated by SIMNON into machine code. This improved the speed of the package by a factor of five.

APPENDIX I

A COMPILER FOR SIMNON

I.1 - INTRODUCTION

SIMNON is an interactive simulation program written in FORTRAN. It contains a processor for the simulation language that produces a pseudocode which is interpreted and executed by a FORTRAN subroutine (CALCUL) at simulation time. This makes the execution of the code slow compared with a compiler which produces machine code. It is possible to add a new stage to SIMNON to generate machine code from the pseudocode [3]. The code is then executed directly by the machine rather than interpreted by a program. The cost of generating machine code from the pseudocode is negligible compared with reduction in the running cost.

Section I.2 describes SIMNON compiler output which is in Reverse Polish Notation (RPN). The information is mostly taken from Reference 2. Section I.3 describes the code generation in general [3], and finally Section I.4 describes the code generation for PERKIN ELMER Model 8/32.

I.2 - SIMNON COMPILER

SIMNON has a compiler which reads in the system description from the input files (every subsystem is kept in a file) and generates pseudocode. This section describes the pseudocode

format(or output from the compiler).

The pseudocode is stored in an integer array in common block/PSCODE/. It is organized as linked lists. Every node in the pseudocode may contain one or more equations or a call to a section in an external FORTRAN subsystem(via subroutine SYSTS). The pseudocode area contains five different lists, each list for a specified range of computations in the simulation part such as: initial computations, derivative computations or computation of discrete states. It is irrelevant to the code generator which list is to be used, but starting point of each list must be known to the system. Common block /ENTRYS/ is used to keep the pointers to these lists in the pseudocode area. Table 1 shows the description of a node head.

FP	Forward pointer (points to the next node)
BP	Backward pointer
LEN	Length of data in the node
IASYST	Index to subsystem at generation time
NODE	Compiler node at generation time
NEQ	Number of equations in this node

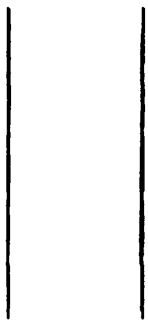

Table 1 Node organization

Each list in the pseudocode consists of a list head and zero or more nodes. A list head is an empty node (LEN=0).

The pseudocode consists of operators (integer 1 - 22) followed by zero or more operands. See Table 2.

OPERATION	MNEMONIC	CODE
Logical or	OR	1
Logical and	AND	2
Logical not	NOT	3
Test less than	TLT	4
Test greater than	TGT	5
Add	ADD	6
Subtract	SUB	7
Multiply	MUL	8
Divide	DIV	9
Negate	NEG	10
Raise	RAI	11
Jump if false	JMPF	12
Jump	JMP	13

Stack	FETCH	14 pointer
Unstack	DEPOS	15 pointer
Apply function	FUNC	16 function No
Call FORTRAN system	CALL	20 isyst ipart
Skip if not sampling	SCOND	21 system No
No operation	NOP	22

Table 2: Operation set

EXAMPLE

Let us consider the pseudocode generated by SIMNON for a node containing only one equation:

A = B + C

PSEUDOCODE	DESCRIPTION
5	FP - Forward Pointer
5	BP - Backward Pointer
7	LEN - Length of data in the node
1	IASYST - Index to subsystem at generation time

```
5          NODE - Compiler Node at generation time
1          LEQ - Number of equations in the node
14         FETCH
2          Pointer to the variable table (i.e. B )
14         FETCH
3          Pointer to the variable table (i.e. C )
6          ADD
15         DEPOS
1          Pointer to the variable table (i.e. A )
```

The SIMNON compiler produces RPN (Reverse Polish Notation) code operating on a stack. A detailed description of the operations is given below.

P(n) is the top stack element

n is the stack pointer.

k is the index in the pseudocode (=pc, program counter).

Logical variable types are given values 0.0 (false) and 1.0 (true). A value is thus true if it is greater than or equal to 0.5 .

The pointer used in FETCH and DEPOS has the following meaning: if pnt> 10000 it points to a literal stored in common block/VALUES/V(pnt-10000) otherwise it points to a variable whose address is stored in common block /VARTB2/IPNTS(pnt).

OP.	DESCRIPTION
OR	P(n-1):=(P(n-1)>=0.5) or (P(n)>=0.5); n:=n-1; k:=k+1
AND	P(n-1):=(p(n-1)>=0.5) and (P(n)>=0.5); n:=n-1; k:=k+1
NOT	P(n):= not (P(n)>=0.5); k:=k+1
TLT	P(n-1):=P(n-1)<P(n); n:=n-1;k:=k+1
TGT	P(n-1):=P(n-1)>P(n); n:=n-1;k:=k+1
ADD	P(n-1):=P(n-1)+P(n); n:=n-1;k:=k+1
SUB	P(n-1):=P(n-1)-P(n); n:=n-1;k:=k+1
MUL	P(n-1):=P(n-1)*P(n); n:=n-1;k:=k+1
DIV	P(n-1):=P(n-1)/P(n); n:=n-1;k:=k+1
NEG	P(n):=-P(n); k:=k+1
RAI	P(n-1):=P(n-1)**P(n); n:=n-1; k:=k+1
JMP nr	k:=k+nr+1 (nr>0)
FETCH pnt	n:=n+1; P(n)=var_{pnt}; k:=k+2
DEPOS pnt	var_{pnt}:=P(n); n:=n-1; k:=k+2
FUNC nr	for one-argument functions:
	P(n):=$func_{nr}$(P(n)); k:=k+2
	for two-argument functions:
	P(n-1):=$func_{nr}$(P(n-1), P(n)); n:=n-1;k:=k+2
CALL	ISYST:=isyst; IPART:=ipart; CALL SYSTS;
	if ISTOP is true then exit from CALCUL;
	k:=k+3
SCOND nr	if LCOND(nr) is true then k:=next code;
	else k:=k+2
NOP	k:=k+1

The following functions are used in FUNC:

NO	NAME	DESCRIPTION
1.	SQRT(x)	square root of x, x ≥ 0.
2.	EXP(X)	expotential function of x
3.	LN(X)	natural logarithm of x, x > 0.
4.	LOG(X)	logarithm base 10 of x, x > 0.
5.	SIN(X)	sine of x (x in radins)
6.	COS(X)	cosine of x (x in radians)
7.	TAN(X)	tangent of x (x in radians)
8.	ATAN(X)	arctangent of x
		result in radians [-π/2, π/2]
9.	ABS(X)	absolute value of x
10.	SIGN(X)	sign of x
		+1. if x > 0.
		0.0 if x = 0.
		-1. if x < 0.
11.	INT(X)	integer part of x
12.	ATAN2(X, Y)	arctangent of x/y
		result in radians [-π, π]
13.	MOD(X, Y)	x module y (x-int(x/y)*y)
14.	MIN(X, Y)	minimum of x and y
15.	MAX(X, Y)	maximum of x and y
16.	ARCSIN(X)	arcsine of x [-1, 1]
17.	ARCCOS(X)	arccosine of x [-1, 1]
18.	SINH(X)	hyperbolic sine of x
19.	COSH(X)	hyperbolic cosine of x
20.	TANH(X)	hyperbolic tangent of x

A few examples of the SIMNON pseudocode are as follows:

```
Y1 = IF A<B THEN C ELSE 2

FETCH     A
FETCH     B
TLT
JMPF        5
FETCH     C
JMP         3
FETCH     2.0
DEPOS     Y1

Y2 = IF A AND B>C THEN C ELSE IF NOT C THEN A ELSE B

FETCH     A
FETCH     B
FETCH     C
TGT
AND
JMPF        5
FETCH     C
JMP        12
FETCH     C
NOT
JMPF        5
FETCH     A
JMP         3
FETCH     B
DEPOS     Y2
```

```
Y3 = (-B)/(A+B-2)*(-C)

FETCH     B
NEG
FETCH     A
FETCH     B
ADD
FETCH     2.0
SUB
DIV
FETCH     C
NEG
MUL
DEPOS     Y3

Y4 = IF A>B AND C THEN TAN(A) ELSE B^(0.2*A)+LN(B)

FETCH     A
FETCH     B
TGT
FETCH     C
AND
JMPF        7
FETCH     A
FUNC      TAN
JMP        14
FETCH     B
FETCH     0.2
```

```
FETCH    A
MUL
RAI
FETCH    B
FUNC     LN
ADD
DEPOS    Y4
```

I.3 - CODE GENERATOR

The code generator scans through the pseudocode and generates machine instructions for each pseudocode instruction. The pseudocode contains forward jumps only, and it is, therefore, necessary to go back to the generated code and insert the jump addresses when the target of jump is processed. The RPN pseudocode is stack oriented which corresponds to the operand address stack used by the code generator but there is no explicit stack or stack instructions in the generated code. The code generator uses eight floating point registers for arithmetic computations (PERKIN ELMER computer has eight floating point registers and sixteen general purpose registers). General purpose registers are also used, whenever possible, to make maximum use of the available registers and reduce the execution time (code instructions using general purpose registers are usually faster). The intermediate results of the computation are stored in the registers which are allocated from a stack of free registers. Since the number of available registers are generally smaller than the

worst case expression in SIMNON the code generator automatically allocates temporary memory cells when the register stack is exhausted.

THE OPERAND STACK AND REGISTER ALLOCATION

Let us consider the following equation:

```
Y = (A + 2) * B
```

SIMNON compiler generates the following pseudocode:

```
1   FETCH    A
3   FETCH    2
5   ADD
6   FETCH    B
8   MUL
9   DEPOS    Y
```

The code generator performs the following actions:

(1) Push addr(A) onto the operand stack
(2) Push addr(2) onto the operand stack
(3) Pop stack twice. If the operands are registers, return the registers to register stack. If both operands are registers; then issue an add register instruction, the result will be stored in the first register

```
AER  R1, R2       (adds A to 2, result in R1)
```

Else if one of the operands is a register issue an add instruction, the result will be stored in the register.

```
AE  R1, 2         (adds A to 2, result in R1)
```

Else get a register from register stack; issue a load instruction to load A into a register.

```
LE  R1, A
```

Issue an add instruction, the result will be stored in the register.

```
AE  R1, 2         (adds A to 2, result in R1)
```

Push the register onto the operand stack.

(4) Push addr(B) onto operand stack

(5) Pop the stack twice. If the operands are registers, return the registers to the register stack. If both operands are registers, then, issue a multiply register instruction, the result will be stored in the first register.

```
MER  R1, R2       (multiplies(A+2) by B, result in R1)
```

Else if one of the operands is a register, issue a mul-

tiply instruction, the result will be stored in the register.

```
ME  R1, B        (multiplies (A+2) by B, result in R1)
```

Else get a register from register stack. Issue a load instruction to load the first operand into a register.

```
LE  R1, OP1
```

Issue a multipy instruction, the result will be stored in the register.

```
ME  R1, B        (multiplies (A+2) by B, result in R1)
```

Push the register onto the operand stack.

(6) Pop the stack, release the register. Issue a store instruction to store the register in Y.

```
STE  R1, Y
```

As a result of the above operations, the address stack will contain a mixture of(addresses of) the variables and registers. The registers represent results from already issued instructions. Note the difference between this stack and the stack used by the interpreter for the pseudocode that contains only values of the variables or intermediate computations.

THE REGISTER STACK

Addresses of free registers are stored in a stack called register stack. The only variable associated with this stack is the stackpointer with increment or decrement of 2. The reason for this is that the registers are considered to be an ordered set and allocation of registers are always made in order. This makes it possible to compute the addresses of the registers on top of the stack directly from the stack pointer. This allocation principle is important in the code generation for conditional branches.

JUMPS

The SIMNON compiler produces three branch or jump instructions, all of which are forward jumps. JMPF and JMP are resulted from IF-THEN-ELSE expressions and the given displacement is a relative pseudocode address. SCOND is used for execution of equations only when specified conditions are met (i.e. time sampling). The displacement is always to the end of the current node. The corresponding pseudocode address is easily found, using the information in the node head. When a jump instruction is encountered in the pseudocode the code generator issues a branch instruction with zero displacement and stores the following information in a special jump address table:

a) the target pseudocode code address

b) the current absolute code address

c) the type of jump instruction

For each pseudocode instruction processed, a check is made if the current pseudocode address is in column (a) of the jump table. If the jump destination is found the absolute code address is inserted in the absolute code at the address indicated by (b) and the entry is removed from the table.

CONDITIONAL BRANCHES

The equation :

Y = IF cond THEN expr1 ELSE expr2

generates the following pseudocode sequence:

```
    cond-code
    JMPF  L1
    ---------
    expr1-code  BLOCK A
    ---------
    JMP   L2
    ---------
L1:expr2-code  BLOCK B
    ---------
L2:DEPOS Y
```

The interpreter evaluates the condition and then one of the blocks A or B. The code generator proceeds through both A and

B and generates code for both cases. Since both blocks in an IF-THEN-ELSE construction consist of expressions the result at the end of the block will be on top of the operand stack either in a register or in a variable (only when the expression is a simple variable). Furthermore, the operand stack has increased by one element from the beginning of the block to the end. A necessary condition for correct code after the two branches is that at the end of both blocks the result is at the same address, i.e. the top element on the operand stack must be exactly the same. The code generator solves this in the following way:

At the end of each block it checks if the top stack element is a variable, in which case a register is popped from the register stack and an instruction to load the variable into the register is issued. The register operand is pushed on the operand stack. It is also necessary to restore both the operand stack and the register stack to the initial status of the block at the end of block A to ensure that the status is the same at the end of both blocks. This is easily done by popping the operand stack and releasing the register. The end of block A is always followed by a JMP instruction and the end of block B is followed by the target of the JMP instruction.

TEMPORARY VARIABLES:

The temporary variables are allocated linearly from an array and are dellocated only at the end of an equation (DEPOS).

Temporary variables are needed when the register stack is exhausted. In this case the operand stack is scanned from the top and when a register is found it is released and a load instruction to a temporary variable is issued. The address of the variable will then replace the register address in the operand stack. The released register is always the last register in the register stack.

CALCUL- The Interface between SIMNON and ABS. CODE:

To execute code in one of the five lists, SIMNON takes one of the values from common block /ENTRYS/ and stores it in common block /ENTRY/, then calls subroutine CALCUL (without any arguments). The code generator replaces the pseudocode entrypoints in /ENTRYS/ with absolute code start addresses and, therefore, /ENTRY/ carries the asbsolute starting address in the generated code. CALCUL is written in assembler, it makes a subroutine jump to the address in /ENTRY/.

DEBUGGING

Information concerning the code generation is required both for debugging and for checking of the generated code. The code generator prints information in four levels:

a) the source equations
b) the input pseudocode
c) the generated code in symbolic assembler format
d) the generated machine code in hexadecimal format

The printout is governed by flags in common block /VXCLOG/ and the values are taken from global variables which can be changed by command LET in SIMNON.

a) is turned on if LOGSRC.PE is non-zero
b) is turned on if LOGPSE.PE is non-zero
c) is turned on if LOGINS.PE is non-zero
d) is turned on if LOGHEX.PE is non-zero

The logical unit number for the output can be given using command LET as follows:

```
LET LUN.PE = 3
```

I.4 - CODE GENERATION FOR PERKIN ELMER MODEL 8/32

Assembly language programming is very close to actual machine operation, therefore, it is essential for the assembly language programmer have a good understanding of the system architecture. All assemblers have one basic purpose, which is to simplify the direct control of the processor by providing the programmer with a way of representing actual machine operations in symbolic form that is easy to read and understand. In performing this function, the assembler translates symbolic representation of machine instructions into a binary form that can be executed by the processor. The code generator is a small compiler which generates binary form for the processor. The architecture of the PERKIN ELMER includes four types of machine instructions. These are Regis-

ter to Register (RR format), Register to indexed storage (RX format), Register and Immediate (RI format), and Short Form(SF format). Instructions used in the code generator require two operands except branching instructions which need only one operand. The first operand instruction is contained in a register. The result usually replaces the content of the first operand register. RR and SF are sixteen bit, RI and RX2 are thirty two bit, and RX3 is fourty eight bit instructions. There are sixteen general registers numbered R0 R1 ... R15, and eight floating registers numbered R0 R2 ... R14. The floating registers are used for arithmetic calculation. Constants 0.0, 0.5, and 1.0, which are frequently used for logical calculations are stored in floating registers R0, R8, and R10. Detailed description of the instructions and addressing modes are given in the PERKIN ELMER Reference Manual No. S29-640R02.

The code generated for each pseudocode operation will be briefly described. The PERKIN ELMER code is represented in assembler mnemonics. Constants 0.0, 0.5, and 1.0 are used as literals in registers R0, R8, and R10. Rx means a register from register stack (R2, R4, or R6). A list of the assembler code used in the code generator is given in Table 3.

INSTRUCTION	MNEMONIC	OP-CODE
Add floating point	AE	6A
Add floating point register	AER	2A
Branch unconditional	B	430

Branch and Link	BAL	41
Branch Equal forward Short	BES	233
Branch on Low	BL	428
Branch on Low forward Short	BLS	218
Branch Not Plus forward Short	BNPS	232
Branch Not Equal Short	BNES	213
Branch on Plus	BP	422
Branch on Plus forward Short	BPS	212
Branch unconditional via Register	BR	300
Branch unconditional forward Short	BS	230
Compare floating point	CE	69
Compare floating Register	CER	29
Compare Logical Immediate	LI	F5
Divide floating point	DE	6D
Divide floating Register	DER	2D
Fix Register	FXR	2E
Float Register	FLR	2F
Load	L	58
Load floating point	LE	68
Load floating Register	LER	28
Load immediate Short	LIS	24
Multiply floating point	ME	6C
Multiply floating Register	MER	2C
Store	ST	50
Store floating point	STE	60
Subtract floating point	SE	6B
Subtract floating Register	SER	2B
Test byte	TBT	74

Table 3.

LOGICAL OPERATOR: OR, AND, NOT, TLT, TGT

The logical operators give constants 0.0 (false) or 1.0 (true) as a result of logical operations. This is used in the IF-construction. A few examples follow:

```
   A OR B


   CE  0.5, A
   BL  L1
   CE  0.5, B
   BL  L1
   LER Rx, 0.0
   BS  L2
L1:LER Rx, 1.0
L2:
```

```
   IF A OR B THEN ...


   CE  0.5, A
   BL  L1
   CE  0.5, B
   BL  L1
   B   J1
L1:
```

```
   A AND B

   CE  0.5, A
   BNL L1
   CE  0.5, B
   BNL L1
   LER Rx, 1.0
   BS  L2
L1:LER Rx, 0.0
L2:

   IF A AND B THEN ...

   CE  0.5, A
   BNL L1
   CE  0.5, B
   BNL L1
   BS  L2
L1:B   J1
L2:

   NOT A

   CE  0.5, A
   BP  L1
   LER Rx, 1.0
   BS  L2
L1:LER Rx, 0.0
L2:
```

```
   IF NOT A THEN ...

   CE  0.5, A
   BPS L1
   B   J1
L1:

   A < B

   LE  Rx, A
   CE  Rx, B
   BLS L1
   LER Rx, 0.0
   BS  L2
L1:LER Rx, 1.0
L2:

   IF A<B THEN ...

   LE  Rx, A
   CE  A, B
   BLS L1
   B   J1
L1:
```

The code for TGT (>) is the same as TLT exept that BLS will be BGS.

ARITHMETIC BINARY CALCULATIONS: ADD, MUL, SUB, DIV

ADD and SUB are symmetric (commutative) and are grouped together. If both operands are in registers AER or MER instructions are used else, if one of the operands is in a register AE or ME instructions are used, else one of the operands is loaded in register then AE or ME instructions are used.

```
R1 + R2          AER R1, R2

A + R1           AE  R1, A

A + B            LE  Rx, A
                 AE  Rx, B
```

SUB and DIV are also grouped together. If both operands are registers SER or DER instructions are used, else if the first operand is not in a register, it is loaded in a register and SE or DE instructions are then used.

```
R1 - R2              SER  R1, R2

R1 - B               SE   R1, B

A - B                LE   Rx, A    SE  Rx, B
```

RAI - RAISE A NUMBER

The mathematical procedure from the runtime library for raising a real base to a real number is used. The two operands are loaded in floating registers R14 and R12 respectively.

```
A ^ B          LE  R14, A
               LE  R12, B
               BAL R15, IPOWR
               LER Rx, R14
```

JUMP INSTRUCTIONS: JMPF, JMP

Since the displacement may exceed 15 (halfword) a B instruction is used instead of the shorter instruction BS. JMPF is the conditional branch of an IF-THEN-ELSE expression. If the condition is not the result of a logical operation, the following code is generated:

```
   IF A THEN ...

   CE  0.5, A
   BL  L1
   B   J1
L1:
```

Since JMP marks the end of the first branch block in an IF-THEN-ELSE an instruction to load the operand on the top of the stack in a register is issued when it is in a variable.

FETCH and DEPOS

FETCH does not generate any code. DEPOS generates instructions to store the operand in a memory cell.

```
A = B              L  Rx, B
                   ST Rx, A
```

FUNC -library function call

Some of the functions are computed directly in code (ABS, SIGN, MIN, MAX, MOD), the rest are calls to runtime library routines, in which case the operands are loaded into registers R14 and R12 respectively. The result is returned in register R14.

```
SIN(A)          one argument function call

LE  R14, A
BAL R15, .SIN
LER Rx, R14


ATAN2(Rx, A)    two argument function call

LER  R14, Rx
LE   R12, A
BAL  R15, .ATAN2
LER  Rx, R14
```

Inline coded functions:

MAX(A, B) (BLS is changed to BPS in MIN)

```
   LE  Rx, A
   CE  Rx, B
   BLS L1
   LE  Rx, B
   BS  L2
L1:LE  Rx, A
L2:
```

SIGN(A)

```
   LE  Rx, A
   CER 0.0, Rx
   BLS L1
   BPS L2
   LER Rx, 0.0
   BS  L3
L1:LER Rx, 1.0
   BS  L3
L2:LER Rx, 0.0
   SER Rx, 1.0
L3:
```

```
   ABS(A)

   CE  0.0, A
   BLS L1
   LER Rx, 0.0
   SE  Rx, A

   MOD(A, B)

   LE  Rx1, A
   LER Rx2, Rx1
   DE  Rx2, B
   FXR Rx, Rx2
   FLR Rx2, Rx
   ME  Rx2, B
   SER Rx1, Rx2
```

CALL -calling external routines(SYSTS)

Integer values ISYST and IPART are known at compile time.

```
   LIS  Rx, ISYST
   ST   Rx, ISYSAD
   LIS  Rx, IPART
   ST   Rx, IPARAD
   BAL  R15, .SYSTS
   TBT  0.0, ISTOP
   BES  L1
   BR   R2
L1:
```

CODE EXAMPLES:

The examples listed earlier will generate the following assembler and absolute codes.

```
  ASSEMBLER CODE                          ABS. CODE

     Y1 = IF A<B THEN C ELSE 2

    LE      R2, A                         6020 4002 9C68
    CE      R2, B                         6920 4002 9C70
    BLS     4                             2184
    B       L1                            4300 4003 0DCA
    LE      R2, C                         6840 4002 9C74
    B       L2                            4300 4003 0DD0
L1: LE      R2, 2                         6820 4002 A10C
L2: STE     R2, Y1                        6020 4002 9C78

     Y2 = IF A AND B>C THEN C ELSE IF NOT C THEN A ELSE B

    LE      R2, B                         6820 4002 9C70
    CE      R2, C                         6920 4002 9C74
    BPS     3                             2123
    LER     R2, 0                         2820
    BS      2                             2302
    LER     R2, 1.                        282A
    CER     .5, R2                        2982
    BP      L1                            4220 4003 0E00
```

```
   CE      .5, A                    6880 4002 9C68
   BP      L1                       4220 4003 0E00
   BS      4                        2304
L1:B       L2                       4300 4003 0E12
   LE      R2, C                    6820 4002 9C74
   B       L3                       4300 4003 0E32
L2:CE      .5, C                    6980 4002 9C74
   BPS     4                        2124
   B       L4                       4300 4003 0E23
   LE      R2, A                    6820 4002 9C68
   B       L3                       4300 4003 0E32
L4:LE      R2, B                    6820 4002 9C70
L3:STE     R2, Y2                   6020 4002 9C7C
```

Y3 = (-B)/(A + B - 2)*(-C)

```
   LER     R2, 0                    2820
   SE      R2, B                    6B20 4002 9C70
   LE      R4, B                    6840 4002 9C70
   AE      R4, A                    6A40 4002 9C68
   SE      R4, 2                    6B40 4002 A10C
   DER     R2, R4                   2D24
   LER     R4, 0                    2840
   SE      R4, C                    6B40 4002 9C74
   MER     R2, R4                   2C24
   STE     R2, Y3                   6020 4002 9C84
```

```
      Y4 = IF A>B AND C THEN TAN(A) ELSE B&(0.2*A) + LN(B)

      LE      R2, A                    6820 4002 9C68
      CE      R2, B                    6920 4002 9C70
      BPS     3                        2123
      LER     R2, 0                    2820
      BS      2                        2302
      LER     R2, 1.                   282A
      CE      .5, C                    6980 4002 9C74
      BP      L1                       4220 4003 0F44
      CER     .5, R2                   2982
      BP      L1                       4220 4003 0F44
      BS      4                        2304
L1:   B       L2                       4300 4003 0F5E
      LE      R14, A                   68E0 4002 9C68
      BAL     R15, .TAN                41F0 4002 2A8E
      LER     R2, R14                  282E
      B       L3                       4300 4003 0F8A
L2:   LE      R2, A                    6820 4002 9C68
      ME      R2, 0.2                  6C20 4002 A100
      LE      R14, B                   68E0 4002 9C70
      LER     R12, R2                  28C2
      BAL     R15, .IPOWER             41F0 4002 2160
      LER     R2, R14                  282E
      LE      R14, B                   68E0 4002 9C70
      BAL     R15, .LN                 41F0 4002 2D60
      LER     R4, R14                  284E
      AER     R2, R4                   2A24
L3:   STE     R2, Y4                   6020 4002 9C94
```

I.5 - REFERENCES

[1] - Elmquist H.,"SIMNON- An interactive simulation program for nonlinear system", Users manual, TFRT 7502, 1975.

[2] - Elmquist H.,"SIMNON- An interactive simulation program, Implementation", TFR, 1978.

[3] - Essebo T.,"SIMNON- A code generator for VAX 11", 1980.

[4] - PERKIN ELMER 8/32 Reference Manual S29-640R02

[5] - VAX-11 Architecture Handbook(1979-1980), Digital Equipment Corporation.

[6] - Sahba M.,"A compiler for SIMNON", Internal Report, Department of Electrical Engineering, Imperial College, EE/IC/CON/83.21, U.K.