DESIGN OF NON-LINEAR CONTROL SYSTEMS
VIA
MATHEMATICAL PROGRAMMING

by

GU DA-WEI

A Thesis submitted for the Degree of

Doctor of Philosophy

May 1985

Control Section

Department of Electrical Engineering

Imperial College of Science and Technology

University of London

ACKNOWLEDGEMENTS

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

I am greatly indebted to Shanghai Jiao Tong University, China, for the grant which supported my study in Imperial College.

I would like to thank my wife, WenYa, for her support during these years.

I would also like to thank Professor D.Q.Mayne and Mrs. J.Mayne for their hospitality which made my stay in England a pleasant time.

I thank Dr. A.Heunis, Dr. M.Sahba and Mr. F. Pereira for their help during the preparation of this thesis.

DESIGN OF NON-LINEAR CONTROL SYSTEMS

VIA

MATHEMATICAL PROGRAMMING

ABSTRACTS

Two kinds of mathematical programming problems which arise in the design of non-linear control systems are discussed in this thesis. Control design is naturally formulated as an optimization problem in which the constraints may be conventional, non-differentiable (e.g. constraints on singular values for robustness or eigenvalues for stability) or infinite-dimensional (e.g. time domain constraints on step responses or frequency domain constraints for performance) and the design variables may be finite dimensional (e.g. parameters of a controller) or infinite dimensional (e.g. a nonlinear control law). The first problem is concerned with the choice of a control law to satisfy stability and performance constraints; the constraints and the design variables are infinite dimensional. The problem can be posed as choosing a control law and a Lyapunov function to satisfy stability and performance constraints or to minimize a criterion function subject to these constraints. The second problem is concerned with the satisfaction of performance constraints in the time domain for all disturbances lying in a specified class of functions; this problem also arises in the design

of robust circuits. This problem can be posed as choosing a finite dimensional design vector so that the output satisfies one set of constraints in the time domain for ALL inputs satisfying another set of constraints in the time domain.

Algorithms for these problems are proposed in this thesis. They are based on the outer approximation approach. Sets of simplices are created during computation to replace the continuum regions and piecewise linear continuous functions (or piecewise quadratic functions) are employed to approximate the solution functions. Feasibility and convergence of the algorithms are established.

Applications of the algorithm to the design of non-linear control systems are discussed. Conditions, in Lie bracket form, for the existence of an observer for a non-linear system and conditions for the stability of the composite system incorporating the observer and the non-linear control law are presented.

To illustrate the algorithm a non-linear controller is designed for a simple non-linear system. Simplices are created adaptively; this reduces computation effort compared with standard finite element methods. The method of decomposing the state space permits the employment of a constraint dropping scheme.

CONTENTS

CHAPTER ONE INTRODUCTION

1.1 Control system design and optimization problems

It is now well-known that design of control systems can lead naturally to the formulation of a constrained optimization problem and any solution to the optimization problem will characterise an acceptable design [1]. The constraints may be conventional, non-differentiable or infinite-dimensional. Here are some examples.

(i) Design of linear feedback systems[2].

Fig.1 is the block diagram of a linear control system. By the usual conventions, G(s) denotes the plant transfer function. K(s,z) is the transfer function of the controller. Y(s,z) and R(s) are the output and input, respectively. Thus, the output of the closed-loop system is

$$Y(s,z) = \{ I + G(s)K(s,z)\}^{-1}G(s)K(s,z)R(s).$$

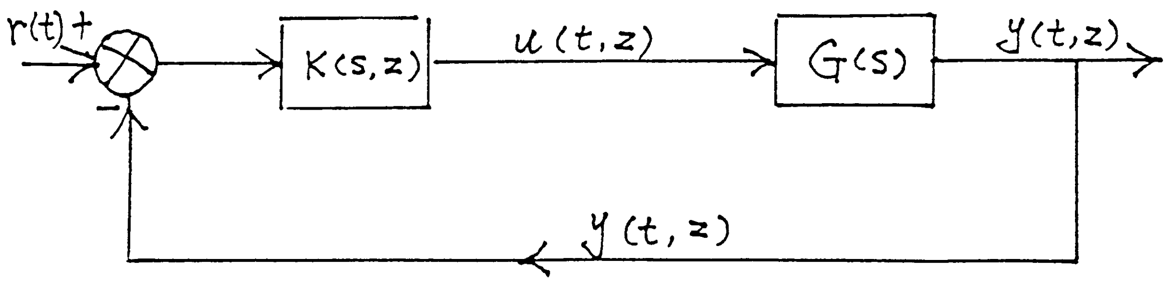


Figure 1

Suppose that $R(s) = 1/s$ and

$$K(s,z) = z_1 + z_2 s^{-1} .$$

Hence, $K(s,z)$ represents a proportional-plus-integral controller. The design criteria can be specified as a set of inequalities. For instance,

$$\phi_1(z) \leqslant c_1 \qquad (1.1)$$

$$\phi_2(z) \leqslant c_2 \qquad (1.2)$$

where ϕ_1 and ϕ_2 denote risetime and overshoot of the output $y(t)$, respectively. c_1 and c_2 are constants in accordance with engineering considerations and with due regard to the physical system. Moreover, because of physical limitations, the parameters z of the controller, $z = [z_1, z_2]^T$, may be restricted as

$$\phi_3(z) \leqslant c_3 \qquad (1.3)$$

$$\phi_4(z) \leqslant c_4 \qquad (1.4)$$

where,

$$\phi_3(z) = Az,$$

$$\phi_4(z) = Bz,$$

A and B are constant matrices.

The design problem is equivalent to finding a vector z such that inequalities (1.1) ~ (1.4) are satisfied.

(ii) Stabilization of a linear system[1].

Suppose we are given a linear time-invariant system in state-space form:

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx \\ x(0) = x_0 \end{cases}$$

where, A,B and C are, respectively, nxn,nxm and pxn real matrices. We choose the feedback controller as

$$u = -Kx + v,$$

where v is the reference input.

Thus the dynamic equation of the system is

$$\dot{x} = (A-BK)x + Bv.$$

The design variable z is an $n \cdot m$-dimensional vector consisting of all entries in matrix K. For stability of the closed-loop system, a z should be chosen to satisfy the following inequality:

$$\phi(z) < 0,$$

where, $\phi(z) \triangleq \max_i \{Re(\lambda_i)\}$, λ_i, $i = 1,...,n$, is the eigenvalue of matrix A-BK. Obviously, λ_i depends on z.

The constraint here is non-differentiable.

(iii) Design of envelope-constrained filters[3].

The problem here is to choose the weighting function w of a digital filter to process a given input pulse s corrupted by noise such that the output error is minimized subject to the constraint that the resulting output pulse ψ satisfies an envelope constraint, i.e., $\psi(t) \in [a(t), b(t)]$ for all $t \in T \triangleq [0, t_1]$.

This problem is relevant to pulse compression in radar systems, waveform equilization, channel equalization for communications and deconvolution of seismic and medical ultrasonic data.

(iv) Design of earthquake resistant structures[3].

The objective here is to design structures, such as steel-framed multilevel buildings that can resist earthquakes. The design considerations include the constraint that the displacements of structure elements, in response to a specified input, should be limited in magnitude at all times in a certain interval.

Let z denote the design variable. Then the constraint may be summarized as the following inequality:

$$\phi(z,t) \leqslant 0, \quad \text{for all } t \in T,$$

where, T is the time interval.

When the input is parameterized by a parameter vector α, for instance, the input r is

$$r(t;\alpha) = (\alpha_0+\alpha_1 t)H(t),$$

$$\alpha = [\alpha_0,\alpha_1]^T$$

and α lies in a subset $\Omega \subset R^2$, then the optimization problem is to find a z such that

$$\phi(z,t;\alpha) \leqslant 0 \ , \quad \text{for all } t \in T \text{ and all } \alpha \in \Omega \ .$$

Moreover, when there is some cost criterion for the structure, it is clearly a function of z. Hence the problem is now to find the vector z which minimizes the cost function subject to

$$\phi(z,t;\alpha) \leqslant 0, \text{ for all } t \in T, \ \alpha \in \Omega.$$

The last two examples are semi-infinite programming problems. They are characterised by finite-dimensional variables and infinite-dimensional constraints. Recent research shows that they arise in control system design, circuit design and other fields[4].

1.2 Semi-infinite programming algorithms

Semi-infinite programming problems have been considered in the mathematical programming literature in recent years. Several kinds of algorithms for solving problems of satisfying infinite constraints or for minimizing a cost function subject to those constraints have been proposed. Two of them, due to Mayne, Polak and Trahan[5,6,7], will be introduced in this section. For the sake of simplicity, we ignore the conventional constraints and restrict the number of infinite-dimensional constraints to one. Thus the feasible-point-finding problem P(A) and the optimal-point-finding problem P(B) are, respectively, as:

P(A): Find a $z \in R^n$, such that

$$\phi(z,\alpha) \leqslant 0, \quad \text{for all } \alpha \in \mathcal{A}.$$

P(B): Minimize $f(z)$

subject to

$$\phi(z,\alpha) \leqslant 0, \quad \text{for all } \alpha \in \mathcal{A}.$$

(i) Descent algorithms.

This type of algorithms deal with the case when $\mathcal{A}$ is an interval (i.e., closed and bounded subset) of R . The following assumptions are made:

(a) ϕ is continuously differentiable;

(b) f is continuously differentiable;

(c) For each z, $\phi(z,\cdot)$ has only a finite number of local maxima in $\mathcal{A}$;

(d) For all z in F^C, $0 \notin co\{\nabla_z \phi(z,\alpha), \alpha \in \mathcal{A}_0(z)\}$,

where, F^C and $\mathcal{A}_0(z)$ are as defined shortly below.

For all z, we define

$$\psi(z) \triangleq \max\{\phi(z,\alpha) \mid \alpha \in \mathcal{A}\}$$

and

$$\psi_+(z) \triangleq \max\{0, \psi(z)\} .$$

The feasible set F is defined by

$$F \triangleq \{ z \in R^n \mid \psi(z) \leq 0 \}$$

$$= \{ z \in R^n \mid \psi_+(z) = 0 \}.$$

F^C is the complement of F.

For any z in R^n, all $\varepsilon \geq 0$, let the ε-most-active constraint set $\mathcal{A}_\varepsilon(z) \subset \mathcal{A}$ be defined by

$$\mathcal{A}_\varepsilon(z) \triangleq \{\alpha \in \mathcal{A} \mid \phi(z,\alpha) \geq \psi_+(z) - \varepsilon\} .$$

A subset of $\mathcal{A}_\varepsilon(z)$ is defined by

$$\tilde{\mathcal{A}}_\varepsilon(z) \triangleq \{\alpha \in \mathcal{A}_\varepsilon(z) \mid \alpha \text{ is a local maximizer of } \phi(z,\cdot) \text{ on } \mathcal{A}\}.$$

From Assumption (c), $\tilde{\mathcal{A}}_\varepsilon(z)$ is finite for all z and all $\varepsilon \geq 0$.

For problem P(A), a search direction $\tilde{p}_\varepsilon(z)$ is chosen by solving

$$\tilde{\theta}_\varepsilon(z) = \min_{p \in P} \max_{\alpha \in \tilde{\mathcal{A}}_\varepsilon(z)} \{ \langle \nabla_z \phi(z,\alpha), p \rangle \}, \qquad (1.5)$$

where, $p \triangleq \{ p \subset R^n \mid \|p\| \leq 1\}$.

For problem P(B), a search direction $\tilde{p}_\varepsilon(z)$ is chosen by solving

$$\tilde{\theta}_\varepsilon(z) = \min_{p \in P} \max \{\langle \nabla f(z), p\rangle - \gamma\psi_+(z);\ \langle \nabla_z \phi(z,\alpha), p\rangle, \alpha \in \tilde{\mathcal{A}}_\varepsilon(z)\}$$

(1.6)

where, γ is a positive constant.

The step-length is obtained using Armijo rule. Thus, we have the following algorithm for problem P(A).

Algorithm (A):

Data: $z_0 \in R^n$, $\varepsilon_0 \in (0,\infty)$, $\beta \in (0,1)$.

Step 0: Set $i = 0$;

$\varepsilon = \varepsilon_0$.

Step 1: Compute $\tilde{\mu}_\varepsilon(z_i), \tilde{\theta}_\varepsilon(z_i)$ by (1.5), and obtain $\tilde{p}_\varepsilon(z_i)$.

Step 2: If $\tilde{\theta}_\varepsilon(z_i) > -\varepsilon$, set $\varepsilon = \varepsilon/2$ and go to Step 1.

Step 3: If $z \in F$, stop;

Else, compute the largest $\lambda \in \{1, \beta, \beta^2, \ldots\}$ such that

$$\psi(z_i + \lambda \tilde{p}_\varepsilon(z_i)) - \psi(z_i) \leqslant -\lambda\varepsilon/2.$$

Step 4: Set $i = i + 1$ and go to Step 1.

Algorithm (B) for problem P(B) is similar to Algorithm (A) except that $\tilde{\theta}_\varepsilon(z)$ and $\tilde{p}_\varepsilon(z)$ are computed from (1.6) and Step 3 is replaced by Step 3':

Step 3': If $\psi(z_i) > 0$, compute the largest $\lambda \in \{1, \beta, \beta^2, \ldots\}$ such that

$$\psi(z_i + \lambda \tilde{p}_\varepsilon(z_i)) - \psi(z_i) \leqslant -\lambda\varepsilon/2;$$

If $\psi(z_i) \leqslant 0$, compute the largest $\lambda \in \{1, \beta, \beta^2, \ldots\}$ such that

$$f(z_i + \lambda \tilde{p}_\varepsilon(z_i)) - f(z_i) \leqslant -\lambda\varepsilon/2$$

$$\psi(z_i + \lambda \tilde{p}_\varepsilon(z_i)) \leqslant 0 .$$

The convergence of these algorithms has been established. To make them implementable, $\psi(z)$ and $\hat{\mathcal{A}}_{\varepsilon}(z)$ are only approximately evaluated, the accuracy being increased automatically to ensure convergence.

(ii) Outer approximation algorithms.

The assumptions of continuously differentiable ϕ and f are still made. $\mathcal{A}$ is no longer restricted to one dimension; it may be a compact subset of R^p.

In these algorithms problem P(A) and P(B) are replaced by an infinite sequence of conventional problems in which $\mathcal{A}$ is replaced by an infinite sequence $\{\mathcal{A}_i\}$ of suitably chosen finite (discrete) subsets of $\mathcal{A}$. The corresponding feasible sets $\{F_i\}$ are defined for all i by

$$F_i \triangleq \{ z \mid \phi(z,\alpha) \leqslant 0 \text{ for all } \alpha \in \mathcal{A}_i \}.$$

At iteration i, $\mathcal{A}_{i+1}$ is formed by adding to $\mathcal{A}_i$ the α which (approximately) solves the global maximization problem $\max\{\phi(z,\alpha) \mid \alpha \in \mathcal{A}\}$ and discarding some elements of $\mathcal{A}_i$ which are judged to be unnecessary. An algorithm, for problem P(A), which has proved successful, is as follows:

Algorithm (C)

Data: $\mathcal{A}_0$ (a discrete subset of $\mathcal{A}$), $\delta \in (0,1)$, $K >> 1$.

Step 0: Set $i = 0$.

Step 1: Compute a z_i in F_i .

Step 2: Compute $\psi(z_i)$ and an $\alpha_i \in \mathcal{A}$ such that

$$\phi(z_i,\alpha_i) = \psi(z_i);$$

If $\psi(z_i) \leqslant 0$, stop .

Step 3 Set

$$\mathcal{A}_{i+1} = \{\alpha_j \in \mathcal{A}_i \cup \{\alpha_i\} \mid \phi(z_j, \alpha_j) \geq K(\delta^j - \delta^i), \quad j = 0, \ldots, i\} .$$

Step 4: Set $i = i + 1$ and go to Step 1.

For problem P(B), an algorithm is easily obtained by replacing Step 1 in Algorithm (C) with the following

Step 1': Compute a z_i to solve

$$\min \{ f(z) \mid z \in F_i \} .$$

Also, the implementable versions of these algorithms utilizing an approximate, but progressively more precise computation of $\psi(z_i)$ and α_i have been proposed.

1.3 Recent results on non-linear systems theory

Non-linear systems constitute a large and important class of control system models for practical purposes. The non-linear features may originate inherently or intentionally. In either case, the nonlinearities make the analysis and design of systems much more difficult than in linear case. Because of the well-developed knowledge about linear systems, the idea to find out the relationship between certain kinds of non-linear systems and linear systems is obviously always attractive. In control systems, the following non-linear systems form is popular and of great interest:

$$\dot{X}(t) = f[x(t)] + g[x(t)]u(t) \qquad (1.7)$$

where, f and g are C^{∞} vector fields on some subset $\Omega \subset R^{n}$, Ω contains the origin and $f(0)=0$. Here we consider the single-input single-output situation only for the sake of convenience.

Krener[8] has used state transformation without feedback to transform (1.7) into a linear system. Meyer and Cicolani[9] considered such transformations for block triangular systems. Brockett[10] obtained sufficient conditions, using both state transformation and feedback, for a real-analytic system of the form (1.7) to be locally equivalent to a linear system. Recently, Hunt, Su and Meyer[11] obtained necessary and sufficient conditions for the local transformation of system (1.7) into a controllable linear system, and sufficient conditions for the global case. In this section, we will very briefly introduce their results. Firstly we define three kinds of Lie derivatives.

If f and g are C^∞ vector fields on R^n, the Lie bracket of f and g is defined as

$$[f,g] = (\partial g/\partial x)f - (\partial f/\partial x)g ,$$

where, $\partial g/\partial x$ and $\partial f/\partial x$ are nxn Jacobian matrices. The Lie bracket [f,g] is also a vector field on R^n. We define Lie brackets [f,[f,g]], etc., successively as follows

$$(ad^1 f,g) = [f,g]$$

$$(ad^2 f,g) = [f,[f,g]] = [f,(ad^1 f,g)]$$

$$\vdots$$

$$(ad^k f,g) = [f,(ad^{k-1} f,g)].$$

A set of C^∞ vector fields $\{f_1, f_2, \cdots, f_r\}$ on R^n is called involutive if there exist C^∞ functions γ_{ijk} such that

$$[f_i, f_j](x) = \sum_{k=1}^{r} \gamma_{ijk}(x) f_k(x), \quad 1 \leqslant i,j \leqslant r, i \neq j.$$

Let f be a C^∞ vector field and h a C^∞ function, h: $R^n \rightarrow R$, with gradient dh on R^n. Then the Lie derivative of h with respect to f is

$$L_f(h) = \langle dh, f \rangle,$$

where $\langle \cdot , \cdot \rangle$ denotes the duality between one-forms and vector-fields. This duality is easily understood if we set

$$\langle dx_i, \partial/\partial x_j \rangle = 0, \quad i \neq j$$
$$= 1, \quad i = j .$$

If f is a C^∞ vector field on R^n and ω is a C^∞ one-form on R^n, i.e., $\omega = \omega_1 dx_1 + \ldots + \omega_n dx_n$, ω_i being a C^∞ function, i=1,...,n, then the Lie derivative of ω with respect to f is defined as

$$L_f(\omega) = ((\partial\omega/\partial x)^* f)^* + \omega(\partial f/\partial x),$$

where * denotes transpose and $\partial\omega/\partial x$ and $\partial f/\partial x$ are Jacobian matrices.

The above three kinds of Lie differentiations are related by a Leibnitz-type formula

$$L_f\langle\omega,g\rangle = \langle L_f(\omega),g\rangle+\langle\omega,[f,g]\rangle$$

with f and ω as before and g a C^∞ vector field. Also $dL_f(h)=L_f(dh)$ with h a C^∞ function.

A C^∞ transformation $T = (T_1,T_2,\ldots,T_n,T_{n+1})$ from R^{n+1} to R^{n+1} is ideal for system (1.7) if it has the following properties:

1) $T(0) = 0$

2) $T_1,\ldots,T_n$ are functions of $x_1,\ldots,x_n$ only and T maps Ω in R^n into $(T_1,\ldots,T_n)$ space with a nonsingular Jacobian matrix

3) T_{n+1} is a function on $x_1,\ldots,x_n,u$ which can be inverted as a function of u

4) T satisfies

$$\begin{aligned}\dot T_1 &= T_2\\ \dot T_2 &= T_3\\ &\ \vdots\\ \dot T_n &= T_{n+1}\end{aligned} \qquad (1.8)$$

5) T is one-to-one (with $(T_1,\ldots,T_n)$ being one-to-one on Ω).

It is obvious that under such a mapping T, system (1.7) is transformed into a linear controllable system in integrator form (1.8), where T_{n+1} is taken as control.

From Hunt, Su and Meyer, we have the following results.

Theorem 1.1:

Necessary and sufficient conditions for the local (at the origin) existence of a mapping T above are:

1) the controllability matrix $\{g,[f,g],\ldots,(ad^{n-1}f,g)\}$ is nonsingular in some neighbourhood of the origin in R^n (with variables $x_1, x_2, \ldots, x_n$)

2) the set of vector fields $\{g,[f,g],\ldots,(ad^{n-2}f,g)\}$ is involutive in some neighbourhood of the origin in R^n.

For such a map in global situation, they gave the following theorem.

Theorem 1.2:

Assume that the controllability matrix of system (1.7) is nonsingular on R^n, the set $\{g,[f,g],\ldots,(ad^{n-2}f,g)\}$ is involutive on R^n, and the noncharacteristic matrix satisfies the ratio condition on R^n (see their paper for definitions.) Then there exists a C^∞ transformation $T=(T_1,\ldots,T_n,T_{n+1})$ on R^{n+1} with the properties mentioned above.

These results provide a sound, theoretic base for the analysis and design of a large class of non-linear systems.

1.4 Contribution of the thesis

In this thesis, the design of non-linear control systems is discussed. We show that these design problems give rise to two new kinds of mathematical programming problems. The first kind is an optimization problem with infinite-dimensional constraints and infinite-dimensional variables. This arises from designing a controller for stability as well as performance and robustness, design of observers and design of regulators. In the second kind of mathematical programming problems, the constraints must be satisfied for an infinite number of functions over a continuum although the design variable is finite-dimensional. Such a problem arises in the design of a controller which ensures that the output of the system satisfies certain constraints in time domain for a class of input functions specified by hard constraints. It also arises in the design of robust circuits. In this thesis, the former problem is discussed in more detail. Some discussions are restricted to the non-linear systems dealt with by Hunt, Su and Meyer. Conditions, in Lie bracket form, for the existence of an observer for a non-linear system and conditions for the stability of the composite system incorporating the observer and the non-linear control law are obtained.

Algorithms for these mathematical programming problems are proposed in this thesis. They are based on the outer approximation approach. Sets of simplices are created adaptively, during the computation, to replace the continuum region and piecewise linear continuous functions are employed to approximate solution functions. The feasibility and convergence of these algorithms are established.

Design problems which result in the first kind of mathematical programming problems are discussed in Chapter 2.

In Chapter 3, algorithms for this kind of problem are proposed and analysed.

An example illustrating the algorithms for the first kind of mathematical programming problem is presented in Chapter 4. It is concerned with the design of a stability regulator to a 2-dimensional non-linear system.

Chapter 5 deals with the second kind of mathematical programming problem. Algorithms are presented and analysed.

In the Appendix, a dynamic programming approach to the optimal control problem is briefly introduced. It shows that the optimal value function may be computed approximately using an algorithm for the first kind of mathematical programming problem discussed in the thesis.

CHAPTER TWO SOME NON-LINEAR CONTROL SYSTEM DESIGN OBJECTIVES

2.1 Stability constraints

Stability is the most important constraint in the design of dynamic systems. Here we only involve ourselves with stability in the sense of Lyapunov.

Suppose we are given a control system in state-space description as follows:

$$\begin{cases} \dot{x}(t) = f[x(t),t] , & t \geqslant 0 \\ x(0) = x_0 . \end{cases} \qquad (2.1)$$

The state x_e is called an equilibrium state of system (2.1) if $f(x_e,t) = 0$, for all t. Without loss of generality, we always may assume $x_e = 0$, the origin of the state space.

The equilibrium state is called stable if for any $\varepsilon>0$, there exists $\delta(\varepsilon)>0$, such that

$$\|x(t;x_0,0)\| < \varepsilon,$$

$$\text{for any } x_0 \in B_{\delta(\varepsilon)}(0), \text{ any } t \geqslant t_0,$$

where, $B_{\delta(\varepsilon)}(0)$ is the open ball centered at the origin with radius $\delta(\varepsilon)$. Let $x(t;x_0,0)$ denote the solution of (2.1) at time t with initial time 0 and initial state x_0.

The equilibrium state is called convergent if there exists $\delta>0$, such that

$$x(t;x_0,0) \longrightarrow 0, \quad \text{as } t \longrightarrow \infty,$$

$$\text{for any } x_0 \in B_\delta(0).$$

The origin is called asymptotically stable if it is both stable and convergent.

In control system design we are more interested in asymptotical stability. Hence, in the rest of the paper, we discuss asymptotical stability only and simply use "stable (stability)" instead of "asymptotically stable (asymptotical stability)".

The well-known Lyapunov stability theorem follows.

Theorem 2.1:

Consider the system (2.1) with the origin as an equilibrium state. If there exists an open subset G of R^n which contains the origin and a function V(x,t) such that

1) $\psi_1(\|x\|) \leqslant V(x,t) \leqslant \psi_2(\|x\|)$,

for all $x \in G, t \geqslant 0$;

2) $\dot{V}(x,t) \leqslant -\psi_3(\|x\|)$,

for all $x \in G, t \geqslant 0$,

where, for $i=1,2,3$, $\psi_i(0)=0$ and $\psi_i(\|x\|)$ are continuous, strictly monotonically-increasing with $\|x\|$, $x \in G$, then, the origin is a stable state of system (2.1).

The function V is usually called a Lyapunov function. The region G *contains* a domain of attraction of the origin. When $G = R^n$, the origin is called globally stable. The Lyapunov theorem for global stability is almost the same as the above theorem except G is replaced by R^n and the condition $\psi_1(\|x\|) \longrightarrow \infty$ as $\|x\| \longrightarrow \infty$ is added.

The Lyapunov stability theorems gives an approach to establishing the stability of either a linear or a non-linear system. In linear systems, it is relatively easy to construct a Lyapunov function. Suppose a linear, time-invariant system is described by

$$\begin{cases} \dot{x}(t) = Ax(t) , & t \geq 0 \\ x(0) = x_0 . \end{cases} \qquad (2.2)$$

A prospective Lyapunov function V is defined by

$$V(x) = x^T P x \qquad (2.3)$$

where P is a positive definite matrix to be chosen. The following theorem is available.

Theorem 2.2:

If matrix A is stable (i.e., all the eigenvalues of A lie in the open left half complex plane), then for an arbitrary positive definite matrix Q, there exists a positive definite matrix P satisfying the matrix equation

$$PA + A^T P = -Q \qquad . \qquad (2.4)$$

Thus in case that the matrix A in system (2.2) is stable, we can arbitrarily choose a positive definite Q and obtain a positive definite matrix P either by the formula

$$P = \int_0^\infty e^{A^T t} Q e^{At} dt$$

or by solving a set of linear equations from (2.4). Then function (2.3) is a Lyapunov function for system (2.2).

The simplicity of the linear case suggests an approach for the non-linear case; the system is linearised around an equilibrium point and a controller designed for the resultant linear system. This is called Lyapunov's first method. In Lyapunov's second method, a Lyapunov function for the non-linear system is directly constructed. Now let us consider the kind of non-linear system mentioned in Section 1.3. We rewrite it here.

$$\dot{x}(t) = f[x(t)] + g[x(t)]u(t) \qquad (2.5)$$

where, $f(0) = 0$.

The objective is to design a regulator for (2.5) such that the closed-loop system is stable at the origin, that is to find a function $r: R^n \longrightarrow R$ of state x, satisfying $r(0) = 0$ such that the following system

$$\dot{x} = f(x) + g(x)r(x) \qquad (2.6)$$

is stable at the origin. We try to apply Lyapunov's second method. The function V is unknown and because of the non-linearity of system it might be very complicated. Suppose that the system (2.5) satisfies the assumptions of theorem in Section 1.3. Then, from Hunt, Su and Meyer's result, there exist transformations $\Gamma: R^n \longrightarrow R^n$ and $\delta: R^n \times R \longrightarrow R$, $\Gamma(x)=z$, $\delta(x,u)=w$ such that under these transformations, system (2.5) is in form:

$$\dot{z} = \begin{pmatrix} 0 & 1 & & & \\ & 0 & 1 & & \\ & & \ddots & \ddots & \\ & & & 0 & 1 \\ & & & & 0 \end{pmatrix} z + \begin{pmatrix} 0 \\ \vdots \\ \vdots \\ 0 \\ 1 \end{pmatrix} w \qquad (2.7)$$

Thus, a feedback law $w = k^T z$ can be easily chosen such that the closed-loop system is stable at the origin, as in the linear case discussed above, with a Lyapunov function $\tilde{V}$ defined by

$$\tilde{V}(z) = (1/2)z^T Pz \qquad (2.8)$$

where P is a positive definite matrix, decided partly by k and obtained from (2.4). From (2.5), we have

$$(d/dt)\tilde{V}(z(t)) = z^T(t)P\dot{z}(t)$$

$$= (\Gamma(x))^T P \Gamma_x (f(x)+g(x)u)$$

$$= (\Gamma(x))^T P \Gamma_x (f(x)+g(x)r(x)),$$

where, $r(x) = u = \delta^{-1}(w;x) = \delta^{-1}(k^T\Gamma(x);x)$.

From Lyapunov's theorem, in order to make the closed-loop system stable we should require, for some ψ_3,

$$(\Gamma(x))^T P \Gamma_x (f(x)+g(x)r(x)) \leq -\psi_3(\|x\|) , \qquad (2.9)$$

$$\text{for all } x \in R^n .$$

The invertibility of transformation Γ also requires that the Jacobian matrix Γ_x is nonsingular everywhere in R^n.

This shows us that there do exist Lyapunov functions and feedback control laws for system (2.5). Furthermore, it encourages us to contruct a Lyapunov function V and a feedback control law r directly (whenever their existence has been established) since those transformations are difficult to obtain analytically or numerically. Hence the stability constraint is that V and r (both from R^n into R) must satisfy (for some $\psi_i, i = 1,2,3$, defined in Lyapunov's theorem) the following inequalities;

$$\begin{cases} V(x) \geq \psi_1(\|x\|) \\ V(x) \leq \psi_2(\|x\|) \\ \dot{V}(x) = V_x(f(x)+g(x)r(x)) \\ \qquad \leq -\psi_3(\|x\|) , \end{cases} \qquad (2.10)$$

$$\text{for all } x \in R^n.$$

In the case when the stability property of system (2.5) is required only in some subset Ω (containing the origin) of R^n, V and r usually should be considered over a larger subset $\tilde{\Omega}$, where

$$\bar{\Omega} \subset \tilde{\Omega} ,$$

$$\bar{\Omega} \triangleq \{x \in R^n \mid V(x) \leq \sup_{x \in \Omega} V(x)\} ,$$

and (2.10) should be satisfied for all x in $\tilde{\Omega}$.

In practice, we may apply Lyapunov's first method to construct a Lyapunov function and a control law since these might be good initial values and good behavior around the origin.

2.2 Performance and robustness constraints

In the design of control systems, a controller is sought to satisfy performance constraints as well as stability constraints. In the last section, we discussed stability constraints in the sense of Lyapunov. In this section, we consider performance constraints.

Performance criteria can be naturally expressed as semi-infinite constraints as we saw in Chapter 1 (for instance, example (iv) in Section 1.1.) There, however, some knowledge about the controller is available, because at least it has been parameterized by finite-dimensional design variables. If, unfortunately, nothing is known a priori about the structure of the controller, then a direct construction of the controller might be worth trying, since in many cases the controller itself is a function of the state and, possibly, time. For instance, let us consider a performance constraint for tracking error. Suppose the instantaneous tracking error due to an input is $e(t,c(x(t));x_0)$, where $c\colon R^n \longrightarrow R^1$ is the controller function. The performance constraint is assumed to be

$$|e(t,c(x(t));x_0)| \leq a(t;x_0) \;,\qquad \text{for all } t \geq 0,$$

where, a is a specified function and x_0 is an initial state. Then, it is required to obtain a control law c, satisfying (in addition to stability and other constraints)

$$|e(t,c(x(t));x_0)| \leq a(t;x_0),\qquad \text{for all } t \in T, \text{ all } x_0 \in \Omega,$$

where, T is the time interval interested, Ω is some subset in R^n.

If the input is parameterized by vector α and/or the system is parameterized by some vector θ, where α and θ lie in specified sets $\mathcal{A}$ and Θ, respectively, then the robust controller must satisfy (in addition to other constraints)

$$|e(t,c(x);x_0,\alpha,\theta)| \leqslant a(t;x_0,\alpha,\theta),$$

$$\text{for all } t \in T,\ x_0 \in \Omega,\ \alpha \in \mathcal{A},\ \theta \in \Theta.$$

2.3 Design of observers for non-linear systems

Often in design of control systems it is necessary to construct estimates of state variables which are not available by direct measurement. The problem of observer design in linear systems is well studied and nice results are available. Take as an example the simplest case a linear, time-invariant system. (For sake of simplicity, we consider single input and single output case only.) Suppose we are given a system as follows.

$$\begin{cases} \dot{x} = Ax + bu \\ y = c^T x \end{cases} \qquad . \tag{2.11}$$

Then an observer can be constructed as in Fig.2[12]

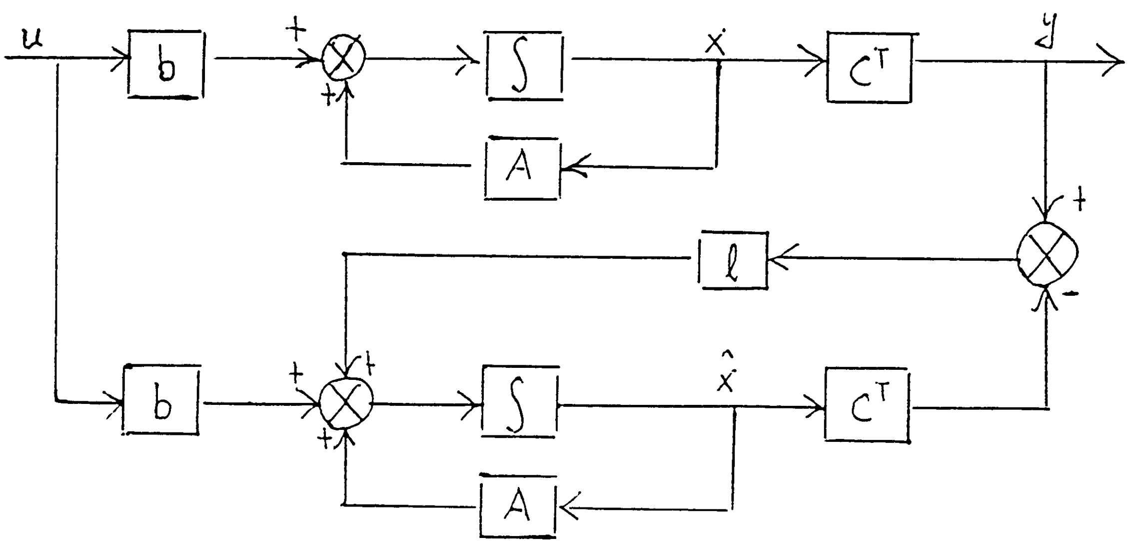


Figure 2

Obviously, the estimate $\hat{x}$ of the state x is obtained by

$$\dot{\hat{x}} = A\hat{x} + bu + l(y - c^T\hat{x})$$

$$= (A - lc^T)\hat{x} + bu + ly \quad .$$

The error e between x and $\hat{x}$ satisfies the following equation:

$$\dot{e} = (A - lc^T)e \quad .$$

The linear system theory tells us that when the system (2.11) is observable, i.e., the matrix $[c, A^T c, \ldots, (A^T)^{n-1} c]^T$ is nonsingular, l may be chosen so that $(A - lc^T)$ has arbitrary eigenvalues. Thus, we can choose l such that all the eigenvalues of $(A - lc^T)$ lie in the open left half plane. Hence, the error $e(t) \longrightarrow 0$, as $t \longrightarrow \infty$.

In non-linear system case, situation is complicated and no uniform approach is available. Let us first consider non-linear systems in an observable canonical form[13]:

$$\begin{cases} \dot{z} = E_n z + w(z_n, u) \\ y = c^T z \end{cases} \qquad (2.12)$$

where,

$$E_n = \begin{bmatrix} 0 & & & 0 \\ 1 & 0 & & \\ & \ddots & \ddots & \\ 0 & & 1 & 0 \end{bmatrix}, \quad w(z_n, u) = \begin{bmatrix} W_1(z_n, u) \\ \vdots \\ W_n(z_n, u) \end{bmatrix}.$$

$$c^T = [\,0, \ldots, 0, 1\,].$$

In this case we can construct an observer as

$$\dot{\hat{z}} = E_n \hat{z} + w(y, u) + \hat{k}(y - c^T \hat{z}).$$

As in linear case, the error $e_1 \triangleq z - \hat{z}$ satisfies

$$\dot{e}_1 = (E_n - \hat{k} c^T) e_1 .$$

The constant parameters $\hat{k}$ may be chosen to ensure that $e_1(t) \longrightarrow 0$, as $t \longrightarrow \infty$.

Now let us consider the more general non-linear system (2.5) with output y as follows:

$$\begin{cases} \dot{x} = f(x) + g(x) u \\ y = h(x) \end{cases} \qquad (2.13)$$

where, f and g are C^∞ vector fields, h is C^∞ function.

We then have the following result concerning the existence condition of a transformation which transforms system

(2.13) into (2.12).

Theorem 2.3:

For system (2.13), if there exists a C^∞ vector field r satisfying the following two groups of equations:

$$(G1)\quad \begin{cases} \langle dh,\ r\rangle = 0 \\ \langle dh,\ [f,r]\rangle = 0 \\ \quad\vdots \\ \langle dh, (ad^{n-2} f, r)\rangle = 0 \\ \langle dh, (ad^{n-1} f, r)\rangle = (-1)^{n-1} \end{cases},$$

$$(G2)\quad \begin{cases} [g,\ r] = 0 \\ [g,\ [f,r]] = 0 \\ \quad\vdots \\ [g, (ad^{n-2} f, r)] = 0 \end{cases},$$

where dh is the gradient of h, then there exists a non-singular transformation ϕ of x (also $\psi = \phi^{-1}$) such that under this transformation, system (2.13) is in the form of (2.12).

Proof:

Suppose r satisfies (G1) and (G2)

Let ϕ, a vector field from R^n into R^n, be the solution of the following equation

$$(\frac{\partial \phi}{\partial x})^T (r,\ -[f,r],\ \ldots,\ (-1)^{n-1}(ad^{n-1} f, r)) = I \qquad (1)$$

Let $z = \phi(x)$. (i.e., $z_i = \phi_i(x)$, $i = 1,\ldots,n$)

From (1), $\frac{\partial \phi}{\partial x}$ is nonsingular, hence there exists $\psi: R^n \longrightarrow R^n$, such that $\psi = \phi^{-1}$.

Consequently,

$$\frac{\partial\phi}{\partial x}\frac{\partial\psi}{\partial z} = I \tag{ii}$$

$$\left(\frac{\partial}{\partial z_k}\left(\frac{\partial\phi}{\partial x}\right)\right)\frac{\partial\psi}{\partial z} + \frac{\partial\phi}{\partial x}\left(\frac{\partial}{\partial z_k}\left(\frac{\partial\psi}{\partial z}\right)\right) = 0$$

$$\frac{\partial}{\partial z_k}\left(\frac{\partial\phi}{\partial x}\right) = -\frac{\partial\phi}{\partial x}\left(\frac{\partial}{\partial z_k}\left(\frac{\partial\psi}{\partial z}\right)\right)\frac{\partial\phi}{\partial x}$$

$$= -\frac{\partial\phi}{\partial x}\left(\frac{\partial}{\partial z}\left(\frac{\partial\psi}{\partial z_k}\right)\right)\frac{\partial\phi}{\partial x}$$

$$= -\frac{\partial\phi}{\partial x}\left(\frac{\partial}{\partial x}\left(\frac{\partial\psi}{\partial z_k}\right)\right). \tag{iii}$$

Also, from (ii) and (i), we have

$$\frac{\partial\psi}{\partial z_1} = r$$

$$\frac{\partial\psi}{\partial z_2} = -[f,r] = -[f,\frac{\partial\psi}{\partial z_1}]$$

$$\vdots \tag{iv}$$

$$\frac{\partial\psi}{\partial z_n} = (-1)^{n-1}(ad^{n-1} f, r) = -[f,\frac{\partial\psi}{\partial z_{n-1}}]$$

For $k = 1, \ldots, n-1$,

$$\frac{\partial}{\partial z_k}\left(\frac{\partial\phi}{\partial x}f\right) = \left(\frac{\partial}{\partial z_k}\left(\frac{\partial\phi}{\partial x}\right)\right)f + \frac{\partial\phi}{\partial x}\frac{\partial f}{\partial z_k}$$

$$= -\frac{\partial\phi}{\partial x}\left(\frac{\partial}{\partial x}\left(\frac{\partial\psi}{\partial z_k}\right)\right)f + \frac{\partial\phi}{\partial x}\left(\frac{\partial f}{\partial x}\frac{\partial\psi}{\partial z_k}\right)$$

(using (iii))

$$= \frac{\partial\phi}{\partial x}[f,\frac{\partial\psi}{\partial z_k}]$$

$$= \frac{\partial\phi}{\partial x}\frac{\partial\psi}{\partial z_{k+1}}$$

$$= \frac{\partial\phi}{\partial z_{k+1}}$$

$$= [0,\ldots,0,1,0,\ldots,]^T \quad .$$
$$\uparrow$$
$$(k+1)$$

Thus,

$$\frac{\partial\phi}{\partial x}f = [0,z_1,\ldots,z_{n-1}]^T + \tilde{w}(z_n)$$

$$= E_n z + \tilde{w}(z_n) \qquad (v)$$

where, E_n is defined in (2.12), $\tilde{w}$ is an n-dimensional vector function of z_n only.

For k = 1, ..., n-1, similar to the derivation of (v) and by the equations (G2), we have

$$\frac{\partial\psi}{\partial z_k}\left(\frac{\partial\phi}{\partial x}g\right) = -\frac{\partial\phi}{\partial x}\left[g,\frac{\partial\psi}{\partial z_k}\right]$$

$$= [0,\ldots,0]^T,$$

then,

$$\frac{\partial\phi}{\partial x}g = \tilde{v}(z_n). \qquad (vi)$$

Thus, from $z = \phi(x)$, we have

$$\dot{z} = \frac{\partial\phi}{\partial x}\dot{x}$$

$$= \frac{\partial\phi}{\partial x}(f + gu)$$

$$= \frac{\partial\phi}{\partial x}f + \frac{\partial\phi}{\partial x}gu$$

$$= E_n z + (\tilde{w}(z_n)+\tilde{v}(z_n)u)$$

$$= E_n z + w(z_n,u). \qquad (vii)$$

Furthermore, (G1) and (iv) yield

$$\frac{\partial h}{\partial z_k} = (h_x)\frac{\partial \psi}{\partial z_k} = 0, \quad k = 1,\ldots,n-1$$

$$\frac{\partial h}{\partial z_n} = (h_x)\frac{\partial \psi}{\partial z_n} = 1.$$

This implies

$$z_n = h(x).$$

Hence, the theorem is proven.

Q E.D.

Also, from the proof above, we can obtain

$$\frac{\partial}{\partial z_n}(w(z_n,u)) = (-1)^n\frac{\partial \phi}{\partial x}((ad^n f,r) + [g,(ad^{n-1} f,r)]u),$$

so that

$$\frac{\partial \psi}{\partial z}(\frac{\partial}{\partial z_n}(w(z_n,u))) = (-1)^n((ad^n f,r) + [g,(ad^{n-1} f,r)]u) \qquad \text{(viii)}$$

This relationship will be later used in the design of observer.

By the Leibnitz-type formula about Lie differentiation (on page 12) it is easy to show that a sufficient condition for the existence of an r satisfying (G1) is the nonsingularity of the following matrix

$$\{dh,\ L_f(dh),\ L_{f^2}(dh),\ \ldots,\ L_{f^{n-1}}(dh)\}$$

The above result gives a constructive method for designing an observer. For the transformed system (2 12) of

(2.13), the observer may be chosen as

$$\dot{\hat{z}} = E_n\hat{z} + w(\hat{z}_n, u) + \hat{k}(y-\hat{z}_n) \quad .$$

The error e_1 thus satisfies

$$\dot{e}_1 = E_n e_1 + w(z_n, u) - w(\hat{z}_n, u) - \hat{k}(y-\hat{z}_n)$$

$$= E_n e_1 + \frac{\partial w}{\partial z_n}\Big|_{\hat{z}_n, u} c^T e_1 - \hat{k}c^T e_1 + O(\|c^T e_1\|^2)$$

$$= (E_n + \frac{\partial w}{\partial z_n}\Big|_{\hat{z}_n, u} c^T - \hat{k}c^T)e_1 + O(\|c^T e_1\|^2) \quad .$$

Thus, let $\hat{k} = \frac{\partial w}{\partial z_n}\Big|_{\hat{z}_n, u} + l$, with l a constant vector such that $(E_n - lc^T)$ is stable .

For the system (2.13), the observer may be

$$\dot{\hat{x}} = f(\hat{x}) + g(\hat{x})u + k(\hat{x}, u)(y - h(\hat{x})).$$

Then, under the transformation Φ,

$$\dot{\hat{z}} = E_n\hat{z} + w(\dot{z}_n, u) + \psi_z^{-1}k(\hat{x}, u)(y-h(\hat{x})).$$

Thus, $k(\hat{x}, u) = \psi_z(\frac{\partial w}{\partial z_n}\Big|_{\hat{z}_n, u} + l)$,

where, ψ_z and $\frac{\partial w}{\partial z_n}\Big|_{\hat{z}_n, u}$ are available from (iv) and (viii) above evaluated at $\hat{x}$, $\hat{z}_n = h(\hat{x})$.

Since (G1) and (G2) are high-dimensional partial differential equations, an analytic solution is difficult. Numerical methods are required This shows that the observer design problem for a non-linear control system can be again turned into a mathematical programming problem of finding a function to satisfy a set of constraints

2.4 Design of regulator and observer

In this section, we consider the problem of designing a regulator for system (2.12), for the case when the states can not be obtained directly; it is assumed that the origin is a stable equilibrium in sense of Lyapunov. The system equations with regulator r are as follows:

$$\begin{cases} \dot{x}(t) = f(x(t)) + g(x(t))r(\hat{x}(t)) \\ y(t) = h(x(t)) \end{cases} \tag{2.14}$$

The dynamic equation of the observer is:

$$\dot{\hat{x}}(t) = f(\hat{x}(t)) + g(\hat{x}(t))r(\hat{x}(t)) + k(\hat{x}(t),y(t)) \tag{2.15}$$

Hence, the composite system is:

$$\begin{pmatrix} \dot{x}(t) \\ \dot{\hat{x}}(t) \end{pmatrix} = \begin{pmatrix} f(x(t)) + g(x(t))r(\hat{x}(t)) \\ f(\hat{x}(t)) + g(\hat{x}(t))r(\hat{x}(t)) + k(\hat{x}(t),h(x(t))) \end{pmatrix}$$

$$\begin{pmatrix} x(0) \\ \hat{x}(0) \end{pmatrix} = \begin{pmatrix} x_0 \\ \hat{x}_0 \end{pmatrix} \tag{2.16}$$

The design objective is to find r and k such that for any pair of initial states $(x_0,\hat{x}_0)$ in $R^n \times R^n$, the trajectories of system (2 16) are bounded and go to the origin eventually.

First, we give a lemma concerning general non-linear differential equations with perturbations.

Lemma 2.1:

Given the following systems:

$$\begin{cases} \dot{x}(t) = f(x(t)) \\ x(0) = x_0 \end{cases} , \tag{2.17}$$

$$\begin{cases} \dot{y}(t) = f(y(t)) + u(t) \\ y(0) = x_0 \end{cases} \qquad , \qquad (2.18)$$

where, $t \geq 0$, $x_0 \in R^n$.

Suppose there exists a scalar function $V: R^n \longrightarrow R$ for system (2.17) such that

i) $V(x) \geq d_1\|x\|$, for all x in R^n, where d_1 is a positive constant

ii) $V(0) = 0$

iii) $V_x(x)f(x) \leq -d_2V(x)$, for all $x \in R^n$, where d'_2 is a positive constant

iv) There exist positive constants d_3 and D_4, such that

$$\|\nabla V(x)\| \leq d_3V(x)$$

for all x in the set $\{z \in R^n | \|z\| > D_4\}$.

Then, if the function $u: R_+ \longrightarrow R^n$ satisfies

$$\int_0^\infty \|u(t)\|dt < +\infty ,$$

the trajectory of system (2.18) is bounded.

Proof:

Consider function V along trajectory of system (2.18),

$$\begin{aligned} \dot{V}(x(t)) &= V_x(x(t))[f(x(t))+u(t)] \\ &= V_x(x(t))f(x(t)) + V_x(x(t))u(t) \\ &\leq -d_2V(x(t)) + V_x(x(t))u(t). \end{aligned}$$

Multiplying by $e^{(d_2t)}$, yields

$$e^{(d_2t)}\dot{V}(x(t)) + d_2V(x(t))e^{(d_2t)} \leq e^{(d_2t)}[V_x(x(t))u(t)].$$

Thus,

$$V(x(t)) \leqslant e^{-(d_2 t)} \int_0^t e^{(d_2 \tau)} [V_x(x(\tau))u(\tau)] d\tau + e^{-(d_2 t)} V(x_0)$$

$$\leqslant \int_{T_1} e^{-d_2(t-\tau)} \|\nabla V\| \|u\| d\tau \quad +$$

$$\int_{T_2} e^{-d_2(t-\tau)} \|\nabla V\| \|u\| d\tau + e^{-(d_2 t)} V(x_0)$$

where, $T_2 \triangleq \{\tau \in [0,t] \mid \|y(\tau)\| > D_4\}$, $T_1 \triangleq [0,t] - T_2$.

Denote $d = \max \{\|\nabla V(z)\| \mid z \in R^n, \|z\| \leqslant D_4\}$.

Thus, at any time t,

$$V(x(t)) \leqslant d_3 \int_0^t \|u(\tau)\| V(x(\tau)) d\tau + d \int_0^t \|u(\tau)\| d\tau + e^{-(d_2 t)} V(x_0)$$

$$\leqslant d_3 \int_0^t \|u(\tau)\| V(x(\tau)) d\tau + C(t),$$

where, $C(t) = d \int_0^t \|u(\tau)\| d\tau + e^{-(d_2 t)} V(x_0)$.

From the Gronwall inequality, we have

$$V(x(t)) \leqslant C(t) + \int_0^t d_3 \|u(\tau)\| C(\tau) e^{\int_\tau^t d_3 \|u(\gamma)\| d\gamma} d\tau.$$

By our assumption on u, the right side is bounded by some constant independent of time t. Thus V is bounded along trajectories of system (2.18). By assumption (i) on V, the trajectories of system (2.18) are bounded.

Q.E.D.

Hence, the design of regulator and observer should satisfy the following assumptions, as based on the discussions in Sections 2.1 and 2.3 and the lemma above.

Assumption A1 (For system (2.5))

1) The n-dimension vector fields f and g satisfy

$f(0)=0$, $g(0)=0$;

2) g is bounded, i.e., there exists a positive constant a such that $\|g(x)\| \leq a$, for all x in R^n.

Assumption A2 (For controller r)

1) r is a uniformly Lipschitz continuous function from R^n into R, i.e., there exists a positive constant b_1 such that

$|r(x_1)-r(x_2)| \leq b_1\|x_1-x_2\|$,

for all x_1, x_2 in R^n;

2) There exists a scalar function V_1: $R^n \longrightarrow R$ satisfying

(i) $V_1(x) \geq b_2\|x\|$, for all $x \in R^n$;

(ii) $V_1(0) = 0$;

(iii) $[\nabla_x V_1(x)]^T [f(x)+g(x)r(x)] \leq -b_3V_1(x)$, for all $x \in R^n$;

(iv) For any given $\delta>0$, there exists a $b_4(\delta)>0$ such that

$[\nabla_x V_1(x)]^T[f(x)+g(x)r(x)] \leq -b_4(\delta)\|\nabla_x V_1(x)\|$;

for all $x \in \{z \in R^n \mid \|z\|>\delta\}$;

(v) There exist b_5 and B_6, such that

$\|\nabla_x V_1(x)\| \leq b_5 V_1(x)$,

for all $x \in \{z \in R^n \mid \|z\|>B_6\}$;

where, b_2, b_3, b_4, b_5, B_6 are all positive constants.

Assumption A3 (For observer function $k(\cdot, h(\cdot))$)

1) $k(\cdot, h(\cdot))$ is a function from $R^n \times R^n$ into R^n satisfying $k(x,h(x)) = 0$, for all x in R^n;

2) $\|k(x_1,h(x_2))\| \leq c_1\|x_1-x_2\|$, for all x_1,x_2 in R^n;

3) There exists a scalar function $V_2: R^n \longrightarrow R$ satisfying (here, we denote the variable of V_2 as e):

(i) $V_2(e) \geq c_2\|e\|$, for all e in R^n;

(ii) $V_2(0) = 0$;

(iii) $[\nabla_e V_2(e)]^T[f(e+z)-f(z)+(g(e+z)-g(z))r(z)-k(z,h(e+z)]$

$$\leq -c_4[V_2(e)]^{c_3}, \text{ when } V_2(e)<1$$

$$< 0 \quad , \quad \text{otherwise}$$

for all $e \neq 0$ in R^n, all z in R^n;

where c_1,c_2,c_3, and c_4 are all positive constants.

We have the following theorem.

Theorem 2.4:

If system (2.5), function r and k satisfy Assumptions A1, A2 and A3 , respectively, then for any pair of initial states $(x_0,\hat{x}_0)$, the origin is a stable equilibrium point of system (2.16) in sense of Lyapunov.

Proof:

Let $e = x - \hat{x}$,

then

$$\begin{aligned}\dot{e} &= f(x)-f(\hat{x})+[g(x)-g(\hat{x})]r(\hat{x})-k(\hat{x},h(x)) \\ &= f(e+\hat{x})-f(\hat{x})+[g(e+\hat{x})-g(\hat{x})]r(\hat{x})-k(\hat{x},h(e+\hat{x})) \; .\end{aligned}$$

Let W_1, W_2, be defined by

$W_1(t)=[\nabla_x V_1(x)]^T[f(x)+g(x)r(x)]$

$W_2(t)=[\nabla_e V_2(e)]^T[f(e+\hat{x})-f(\hat{x})+[g(e+\hat{x})-g(\hat{x})]r(\hat{x})-k(\hat{x},h(e+\hat{x}))]$

where, x, $\hat{x}$ and e stand for x(t), $\hat{x}(t)$ and e(t), respectively.

Define the function $V(x,\hat{x})$ for system (2.16) by

$$V(x,\hat{x}) = V_1(x) + V_2(e)$$
$$= V_1(x) + V_2(x-\hat{x}) .$$

Obviously, $V(x,\hat{x})$ is positive for any $(x,\hat{x})$ and

$$\dot{V}(x(t),\hat{x}(t)) = \dot{V}_1 + \dot{V}_2$$
$$= [\nabla V_1(x)]^T[f(x)+g(x)r(\hat{x})]+[\nabla V_2(e)]^T[f(x)+g(x)r(\hat{x})- f(\hat{x})-g(\hat{x})r(\hat{x})-k(\hat{x},h(x)]$$
$$= W_1(t) + W_2(t) + [\nabla V_1(x)]^T g(x)[r(\hat{x})-r(x)] .$$

By Assumption A2, the following system (2.19) is stable at origin,

$$\dot{\hat{x}} = f(\hat{x}) + g(\hat{x})r(\hat{x}) \qquad (2.19)$$

By Assumption A3, e is bounded and

$$e(t) \longrightarrow 0 , \text{ as } t \longrightarrow \infty .$$

And from Assumptions 2),3)/(i), 3)/(iii) in A3,

$$\int_0^\infty \|-k(\hat{x}(t),h(x(t)))\|dt < \infty .$$

Thus, by Lemma 2.1, $\hat{x}$ in system (2.16) is bounded. And so is x in system (2.16) because of the boundness of e. There exists M such that $\|x(t)\| < M$, for any $t \geq 0$.

By Assumptions 2) in A1, 1) in A2 ,

$$\|[\nabla V_1(x)]^T g(x)(r(\hat{x})-r(x))\| \leq \|\nabla V_1(x)\|\|g(x)\|\|r(\hat{x})-r(x)\|$$
$$\leq ab_1\|\nabla V_1(x)\|\|e\| .$$

For any given $\delta>0$, in annulus $\delta<\|x\|<M$,

by Assumption 2)/(iv) in A2, there is $b_4>0$, such that

$$W_1(t) \leq -b_4\|\nabla V_1(x(t))\| < 0 .$$

Since $\|e(t)\| \longrightarrow 0$, as $t \longrightarrow \infty$, there exists finite T, such that

$$W_1(t) + ab_1\|\nabla V_1(x(t))\|\cdot\|e(t)\| < 0 , \text{ when } t > T .$$

Hence, when $t > T$, $\dot{V}(x(t)) < 0$, for all $x\in\{z\in R^n \mid \delta<\|z\|<M\}$.

Thus, x is forced into the disc $\|x\| \leqslant \delta$.

Since the arbitrariness of δ, $x \longrightarrow 0$, as $t \longrightarrow \infty$.

So is $\hat{x}$, since $e = x - \hat{x} \longrightarrow 0$, $t \longrightarrow \infty$.

This completes the proof of the stability of system (2.16).

Q.E.D.

Thus, the design objective here once more turns into the selection of functions r, k and V_1, V_2 which have to satisfy a set of constraints over R^n or a subset of R^n.

CHAPTER THREE MATHEMATICAL PROGRAMMING PROBLEM WITH INFINITE DESIGN VARIABLES AND CONSTRAINTS

3.1 The problems

As we saw in Chapter 2, many non-linear control system design problems can be expressed as a type of mathematical programming problem of finding a function satisfying an infinite number of constraints or minimizing a cost function subject to an infinite number of constraints. The former is referred to as the feasibility problem and the latter as the constrained optimization problem. To be more specific, such problems in control system design often take the following forms, (we ignore other kinds of constraints in design problems and concentrate on this new type of constraint).

$\tilde{P}1$ (the feasibility problem):

Given a continuum subset Ω of R^n and a function f: $\Omega \times R^m \times R^{n\times m} \longrightarrow R^p$, find a continuously differentiable function ϕ: $\Omega \longrightarrow R^m$, such that the following inequality is satisfied

$$f(x,\phi(x),\nabla_x \phi(x)) \leqslant 0, \qquad \text{for all } x \in \Omega.$$

It is easy to see that the stability design constraints (2.10) described in Sec.2.1 falls into this category with $m=2$ and $p=3$.

$\tilde{P}2$ (the constrained optimization problem):

Given a continuum subset Ω of R^n, a function f: $\Omega \times R^m \times R^{n\times m} \longrightarrow R^p$ and a functional f^0. $C_m^{(1)}(\Omega) \longrightarrow R$, where

$C_m^{(1)}(\Omega) \triangleq \{\phi : \Omega \longrightarrow R^m$ is continuously differentiable $\}$, find a function $\phi^* \in C_m^{(1)}(\Omega)$ such that

$$f^0(\phi^*) = \inf\{ f^0(\phi) \mid \phi \in C_m^{(1)}(\Omega) \text{ and } f(x,\phi(x),\nabla_x\phi(x)) \leqslant 0 , \text{ for all } x \in \Omega \}.$$

As we said in Introduction, the approach proposed in this paper is to use piecewise linear continuous functions to approximate solution functions. In control system design problems, piecewise continuously differentiable solution functions are usually acceptable. Hence, we are concerned with the following problems which are different from $\tilde{P}1$ and $\tilde{P}2$ in the definition of ϕ.

P1:

Given a continuum subset Ω of R^n and a function f: $\Omega \times R^m \times R^{n\times m} \longrightarrow R^p$, find a function ϕ: $\Omega \longrightarrow R^m$, which is a piecewise linear continuous function or is the accumulation point (under the norm of Def.3.7 below) of a sequence of piecewise linear continuous functions, such that the following inequality is satisfied

$$f(x,\phi(x),\nabla_x\phi(x)) \leqslant 0, \quad \text{for all } x \in \Omega,$$

where, the value of $\nabla_x\phi(\cdot)$ at the discontinuous points of $\nabla_x\phi(\cdot)$ takes the local average value.

P2:

Given a continuum subset Ω of R^n, a function f: $\Omega \times R^m \times R^{n\times m} \longrightarrow R^p$, and a functional f^0: $\tilde{C}_m^{(1)}(\Omega) \longrightarrow R$, where $\tilde{C}_m^{(1)}(\Omega) \triangleq \{ \phi : \Omega \longrightarrow R^m$ is continuous and its first order derivative is a piecewise continuous function}, find a function $\phi^* \in \tilde{C}_m^{(1)}(\Omega)$ such that

$$f^0(\phi^*) = \inf\{ f^0(\phi) \mid \phi \in \tilde{C}_m^{(1)}(\Omega) \quad , \text{ and}$$
$$f(x,\phi(x),\nabla_x\phi(x)) \leqslant 0 \ , \text{ for all } x \in \Omega \},$$

where, the value of $\nabla_x\phi(\cdot)$ at discontinuous points of $\nabla_x\phi(\cdot)$ takes the local average value.

The definition of the solution is not precise here. We will make it more detailed in Sec.3.5.

The relationship, i.e., the feasibility and optimality, between $\tilde{P}1$, $\tilde{P}2$ and P1, P2 will be discussed along with the proceeding. For the sake of simplicity, we assume m=1 and p=1. This assumption will hold through this chapter.

3.2 Algorithms for Problem P1

The main algorithm proposed below for problem P1 is of the outer-approximation type. At every iteration, there are two sets of discrete points in Ω. One is the set of nodal points, ω^i, at which the constraints must be satisfied. The other is the set of auxiliary points, Δ^j, which helps to find the desirable values at nodal points. We use some sub-algorithm to find the values at both ω^i and Δ^j such that the piecewise linear continuous function obtained satisfies constraints with regard only to points in ω^i. If this is successful within certain iterations in the sub-algorithm, we go to next step that is to find, approximately, the "worst" point in Ω in the sense that the constraints are mostly violated at this point. Then we add the "worst" point to the nodal set and go to the next iteration. If we are unable to find the desirable piecewise linear continuous function within certain iterations in the execution of the sub-algorithm, we increase the points in the auxiliary point set and try again. That is the basic idea of Main-algorithm 1 stated below.

First, there are some definitions.

Definition 3.1:

For a given discrete subset ω of Ω, where Ω is a continuum subset of R^n, $\tilde{s}(\omega)$ is a set of nonoverlapping simplices whose vertex set is ω. $S(\omega)$ is the class of all such $\tilde{s}(\omega)$.

Definition 3.2:

For discrete sets $\omega^1 \subset \omega^2$, a set of simplices $\tilde{s}(\omega^2)$ is said to contain a set of simplices $\tilde{s}(\omega^1)$ if $\tilde{s}(\omega^i) \in S(\omega^i)$, $i = 1, 2$, respectively, and every simplex in $\tilde{s}(\omega^1)$ is either a simplex in $\tilde{s}(\omega^2)$ or is the union of some simplices in $\tilde{s}(\omega^2)$.

Definition 3.3:

For a given discrete subset ω of Ω, a function class $L(\omega)$ is called the Piecewise Linear Continuous Function Class over ω iff $L(\omega)$ satisfies

$$L(\omega) \triangleq \{ l \mid \text{There exists a set of simplices } \tilde{s}(\omega) \text{ on which } l \text{ is continuous and is linear on every simplex in } \tilde{s}(\omega). \}$$

The subset of $L(\omega)$ consisting of functions specified over $\tilde{s}(\omega)$ is denoted by $L(\omega, \tilde{s}(\omega))$.

Definition 3.4:

For a given simplex s in R^n, a side of s is called the largest side of s if its measurement defined by

$$\sum_{1 \leqslant i < j \leqslant n} \| x^j - x^i \|_n$$

exceeds (or, equals) the measurements of all other sides, where $\{x^1, x^2, \cdots, x^n\}$ is the set of vertices of this side and $\| \cdot \|_n$ is the Euclidean norm in R^n.

Definition 3.5:

For a given simplex s in R^n, the dividing-point x is the centroid of the largest side of s.

Now we present our main-algorithm for problem P1.

Main-algorithm 1:

Data: $\omega^0 = \{x^1, \ldots, x^{n_0}\} \subset \Omega$ and $\tilde{s}(\omega^0) = \{s^1, \ldots, s^{m_0}\}$, as

defined in Def.3.1 such that $U\tilde{s}(\omega^0)$ (the union of the elements of $\tilde{s}(\omega^0)$), is an approximation to Ω;

Integer: $it > 1$;

Logical variable: CASE;

Initial function: $l_0 \in L(\omega^0, \tilde{s}(\omega^0))$ as defined in Def.3.3.

Step 0: Set $i = 0$;

Set $\omega^{-1,0} = \omega^0$;

Set $\omega^{0,0} = \omega^0$.

Step 1: Set $j = 0$;

Set Δ^0 = empty set;

Construct $\tilde{s}(\omega^{i,0})$ such that it contains $\tilde{s}(\omega^{i-1,0})$.

Step 2: Let Δ^{j+1} be the dividing-point set of $\tilde{s}(\omega^{i,j})$;

Set $\omega^{i,j+1} = \omega^{i,j} \cup \Delta^{j+1}$;

Construct $\tilde{s}(\omega^{i,j+1})$ containing $\tilde{s}(\omega^{i,j})$;

Set $l_{i,j+1} = l_i$;

Set $j = j + 1$;

Let $n_{i,j}$ be the cardinal number of $\omega^{i,j}$.

Step 3: Find (using Sub-algorithm 1.1) the piecewise linear continuous function $l_{i,j}$ satisfying

$$f(x,\ l_{i,j}(x),\ \nabla_x l_{i,j}(x)) \leqslant 0,$$
$$\text{for all } \ x \in \omega^{i};$$

If a solution is found (i.e. CASE = true),

go to Step 4;

Else (i.e. CASE = false),

set $l_i = l_{i,j}$ and

go to Step 2.

Step 4: Find the "worst" point x^* in $U\tilde{s}(\omega^{i,j})$ such that

$$f(x^*,\ l_{i,j}(x^*),\ \nabla_x l_{i,j}(x^*)) =$$
$$\max_{x \in U\tilde{s}(\omega^{i,j})} \quad f(x,\ l_{i,j}(x),\ \nabla_x l_{i,j}(x))$$

Step 5: If $f(x^*,\ l_{i,j}(x^*),\ \nabla_x l_{i,j}(x^*)) \leqslant 0$,

set $l_i = l_{i,j}$ and stop.

Step 5: Set $\omega^{i+1} = \omega^{i} \cup \{x^*\}$;

Set $l_{i+1} = l_{i,j}$;

Set $\omega^{i+1,0} = \omega^{i,j}$;

Set $i = i+1$;

Go to Step 1.

The following is a sub-algorithm called in Step 3 of Main-algorithm 1 to determine $l_{i,j}$.

Sub-algorithm 1.1

Comment 1 :

This sub-algorithm determines a function $l_{i,j} \in L(\omega^{i,j}, \tilde{s}(\omega^{i,j}))$ satisfying

$$f(x,\ l_{i,j}(x),\ \nabla_x l_{i,j}(x)) \leqslant 0, \quad \text{for all } x \in \omega^i ,$$

when such an $l_{i,j}$ exists and can be found within certain number of iterations (e.g., j*it).

Comment 2:

What we want to do now is to find a finite-dimensional vector satisfying a finite number of constraints. Many algorithms are available. The following[14] is just one of them. Roughly speaking, this algorithm employs a minimum norm search direction p, at a present variable z, such that the first-order approximation of constraints at z+p is the most descending. It uses an Armijo rule to choose the step length. Some modification has been made here since the existence of solution from the sub-algorithm is not guaranteed in our case.

For a piecewise linear continuous function $l \in L(\omega^{i,j}, \tilde{s}(\omega^{i,j}))$, we define the vector z_l by

$$z_l \triangleq [\ldots, l(x^k), \ldots] \in R^{n_{i,j}},$$

where, $x^k \in \omega^{i,j}$ and $\{x^k\}$ is arranged in certain order. On the other hand, a vector $z \in R^{n_{i,j}}$ defines a piecewise linear continuous function $l_z \in L(\omega^{i,j}, \tilde{s}(\omega^{i,j}))$.

Consequently, we define a set of constraints, for vector z, by

$$g^j(z) \triangleq f(x^j, l(x^j), \nabla_x l(x^j)), \quad j \in \{1, \ldots, n_1\},$$

where, n_1 is the cardinal number of set ω^1.

Comment 3:

Since such sub-algorithm often employs the gradients of constraints, we assume that f is differentiable in z. This is possible in many design problems (e.g., in stability design) and in case l is a piecewise linear continuous function.

We define a function ψ: $R^{n_{i,j}} \longrightarrow R$ by

$$\psi(z) \triangleq \max \{g^j(z) \mid j=1,\ldots,n_1\}.$$

Also, for all z, p in $R^{n_{1,j}}$, we define the first-order approximation to $\psi(z+p)$ by

$$\tilde{\psi}(z,p) \triangleq \max \{ g^j(z)+g_z^j(z)p \mid j=1,\ldots,n_1\}.$$

Let the functions ψ^0, $\psi_\varepsilon{}^0$: $R^{n_{1,j}} \longrightarrow R$ be defined, for all $\varepsilon \geq 0$, by

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

$$\tilde{\psi}_\varepsilon{}^0(z) \triangleq \max \{\psi^0(z) \quad , -\varepsilon\},$$

where, $P \triangleq \{ p \in R^{n_{1,j}} \mid \|p\|_\infty \leq L\}$,

L is some suitably chosen large number.

The algorithm is as follows.

Sub-algorithm 1.1:

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

Step 0: Set $z_0 = z_{1_1}$;

Set $\varepsilon = \varepsilon' \psi(z_0)$;

Set $k = 0$.

Step 1: If $\psi(z_k) \leqslant 0$,

set CASE = true ;

Set $l_{i,j} = l_{z_k}$, stop.

Step 2: If $k > j*it$,

set CASE = false;

Set $l_{i,j} = l_{z_k}$, stop.

Step 3: Compute $\hat{\psi}^0(z_k) = \min \{ \hat{\psi}(z_k, p) \mid p \in P \}$.

Step 4: Compute

$$p_k = p_\varepsilon(z_k) = \operatorname{argmin}\{\|p\| \mid \hat{\psi}(z_k, p) \leqslant \hat{\psi}_\varepsilon{}^0(z_k)\}.$$

Step 5: Compute λ, the largest number in $\{1, \beta, \beta^2, \ldots\}$

such that

$$\psi(z_k+\lambda p_k)-\psi(z_k) \leqslant \lambda[\hat{\psi}_\varepsilon{}^0(z_k)-\psi(z_k)]/2.$$

Step 6: Set $z_{k+1} = z_k+\lambda p_k$;

Set $k = k+1$;

Go to Step 1.

Main-algorithm 1 together with Sub-algorithm 1.1 is a complete algorithm for problem P1. Yet it is, however, a conceptual algorithm in the sense that the Step 4 in Main-algorithm 1 requires the solution of a global optimization problem. To make it implementable, we propose the following sub-algorithm.

Sub-algorithm 1.2:

Comment:

This sub-algorithm is to find, approximately, the "worst" point in Ω. It is of grid type and replaces Step 4 in Main-algorithm 1.

Data: $\tilde{s}(\omega^{i,j})$ as before;

$m_{i,j}$: the number of simplices in $\tilde{s}(\omega^{i,j})$;

$l_{i,j}$: a function in $L(\omega^{i,j},\tilde{s}(\omega^{i,j}))$;

$G(i)$: a monotonically increasing integer function

Step 1: Order the $m_{i,j}$ simplices in $\tilde{s}(\omega^{i,j})$, from 1 to $m_{i,j}$;
Choose, equally spread or randomly, $G(i)$ points in every simplex in $\tilde{s}(\omega^{i,j})$ and order the points (e.g., lexicographically) in each simplex from 1 to $G(i)$.

Step 2: Set $k = 1$.

Step 3: In Simplex k, find, x_k^*, the worst point among $G(i)$ points, i.e.,

$$f(x_k^*,\ l_{i,j}(x_k^*),\ \nabla_x l_{i,j}(x_k^*)) = \max_x f(x,\ l_{i,j}(x),\ \nabla_x l_{i,j}(x)).$$

Step 4: If $k < m_{i,j}$, set $k = k+1$;
Go to Step 3.

Step 5: Among the above $m_{i,j}$ points, find the worst point x^*, i.e.,

$$f(x^*, l_{i,j}(x^*), \nabla_x l_{i,j}(x^*)) = \max\{f(x_k^*, l_{i,j}(x_k^*), \nabla_x l_{i,j}(x_k^*)), \quad k = 1, \ldots, m_{i,j}\}.$$

As is pointed out in Section 3.1, in many cases a piecewise linear continuous function is acceptable for control system design problems, provided it satisfies the design constraints. Hence we do not employ a precision criterion for the size of the final simplices in algorithms solving problem P1, when only Main-algorithm 1 and sub-algorithm 1.1 are employed (i.e., if the exact or nearly exact maximization in Step 4 is carried out via some alternatives, analytic or numerical, to Sub-algorithm 1.2). If Sub-algorithm 1.2 is used, however, the size of simplices and the value of G should be taken into account. Since the corresponding modification is similar to that in Main-algorithm 2, we leave discussion on this point later.

3.3 Analysis of the algorithms in Sec.3.2

In this section, the properties of the algorithms proposed in the last section for problem P1 will be discussed.

Definition 3.6:

Given a continuum subset Ω in R^n, the piecewise continuously differentiable function class on Ω, $\tilde{C}^{(1)}(\Omega)$, is defined by:

$$\tilde{C}^{(1)}(\Omega) \triangleq \{\phi: \Omega \longrightarrow R \text{ is a continuous function and its first order derivative is a piecewise continuous function}\}.$$

And the piecewise linear continuous function class on Ω, $L(\Omega)$, is defined by:

$$L(\Omega) \triangleq \{\phi: \Omega \longrightarrow R \text{ is a piecewise linear continuous function }\}.$$

Definition 3.7:

For a function ϕ in $\tilde{C}^{(1)}(\Omega)$, the norm of ϕ is defined by:

$$\|\phi\|_\infty \triangleq \sup_{x\in\Omega} (|\phi(x)| + \|\nabla_x \phi(x)\|_n)$$

where, $\|\cdot\|_n$ is the Euclidean norm in R^n space. For those points x, at which $\nabla_x\phi(\cdot)$ is discontinuous, $\nabla_x\phi(x)$ is defined to be the local average of all relevant derivative values.

In most cases we consider a closed, bounded Ω. Then the 'sup' in the definition of norm may be replaced by 'max'. Also we suppose Ω can be covered by a set of simplices, i.e., $\Omega = \tilde{\cup} s(\omega)$ Thus, $L(\omega)$ is a subset of $\tilde{C}^{(1)}(\Omega)$.

Lemma 3.1:

$\tilde{C}^{(1)}(\Omega)$ is a normed linear space of functions with norm $\|\cdot\|_\infty$ under Def.3.7 .

Proof: (Elementary)

Definition 3.8:

For problem P1, the feasible set Ψ is defined as:

$$\Psi \triangleq \{\phi \in \tilde{C}^{(1)}(\Omega) \mid f(x, \phi(x), \nabla_x \phi(x)) \leqslant 0, \text{ for all } x \in \Omega\}.$$

We give the following assumption for f in problem P1.

Assumption 3.1:

The function f in problem P1 is assumed to satisfy the following inequality: for all x_1, x_2 y_1, y_2, z_1, z_2,

$$|f(x_1, y_1, z_1) - f(x_2, y_2, z_2)| \leqslant K(\|x_1 - x_2\|_n + |y_1 - y_2| + \|z_1 - z_2\|_n),$$

where,

$x_1, x_2 \in \Omega_x$, Ω_x is a bounded subset in R^n,

$y_1, y_2 \in \Omega_y$, Ω_y is a bounded subset in R,

$z_1, z_2 \in \Omega_z$, Ω_z is a bounded subset in R^n,

K is a constant, $0 < K < +\infty$, related to $\Omega_x, \Omega_y, \Omega_z$ and f.

Actually, in Comment 3 of Sub-algorithm 1.1, we assume that f is differentiable. Hence it satisfies this Lipschitz condition.

Lemma 3.2:

In normed space $\tilde{C}^{(1)}(\Omega)$, if a sequence $\{\phi_i\}$ converges to ϕ^*, as $i \longrightarrow \infty$, then $\bigcup_i \{\phi_i(x) \mid x \in \Omega\}$ and $\bigcup_i \{\nabla_x \phi_i(x) \mid x \in \Omega\}$ must be bounded.

Proof:

Since $\phi_i \longrightarrow \phi^*$, as $i \longrightarrow \infty$, in $\tilde{C}^{(1)}(\Omega)$,

$$\max_{x\in\Omega} (|\phi_i(x)-\phi^*(x)| + \|\nabla_x\phi_i(x)-\nabla_x\phi^*(x)\|_n)$$

$$= \|\phi_i - \phi^*\|_\infty \longrightarrow 0, \text{as } i \longrightarrow \infty.$$

Obviously, there is a $B\in(0,\infty)$, such that for any i,

$$\max_{x\in\Omega} (|\phi_i(x)-\phi^*(x)|+\|\nabla_x\phi_i(x)-\nabla_x\phi^*(x)\|_n) \leqslant B.$$

Since $\phi^*\in\tilde{C}^{(1)}(\Omega)$ and Ω is bounded, so are

$\{\phi^*(x)|x\in\Omega\}$ and $\{\nabla_x\phi^*(x)|x\in\Omega\}$.

Consequently, we have positive numbers B_1 and B_2 such that, for any i,

B_1 is a bound for $\{\phi_i(x)|x\in\Omega\}$,

B_2 is a bound for $\{\nabla_x\phi_i(x)|x\in\Omega\}$.

For any vector z in $\bigcup_i\{\phi_i(x)|x\in\Omega\}$, there exists an i, such that

$z \in \{\phi_i(x)|x\in\Omega\}$,

thus, $\|z\| \leqslant B_1$.

The same argument holds for $\bigcup\{\nabla_x\phi_i(x)|x\in\Omega\}$.

Hence, the lemma is derived.

Q.E.D.

Lemma 3.3:

If f in problem P1 satisfies Assumption 3.1, then Ψ, the set of feasible functions, is closed.

Proof:

Let $\{\phi_i\}$ be a convergent sequence in Ψ, i.e.,

$\phi_i \longrightarrow \phi^*$, as $i\longrightarrow \infty$.

From Lemma 3.2, we know the sets

$\bigcup_i\{\phi_i(x): x \in \Omega\}$ and $\bigcup_i\{\nabla_x\phi_i(x): x \in \Omega\}$

are bounded. Thus, from Assumption 3.1, there exists a K such that for any $x \in \Omega$,

$$|f(x,\phi_i(x),\nabla_x\phi_i(x))-f(x,\phi^*(x),\nabla_x\phi^*(x))|$$
$$\leqslant K(|\phi_i(x)-\phi^*(x)|+\|\nabla_x\phi_i(x)-\nabla_x\phi^*(x)\|).$$

Since $\|\phi_i-\phi^*\|_\infty \longrightarrow 0$, as $i \longrightarrow \infty$,

$$|f(x,\phi_i(x),\nabla_x\phi_i(x))-f(x,\phi^*(x),\nabla_x\phi^*(x))| \longrightarrow 0,$$
$$\text{as } i \longrightarrow \infty, \text{ for all } x \in \Omega.$$

Since $\phi_i \in \Psi$, for every i,

$$f(x,\phi_i(x),\nabla_x\phi_i(x)) \leqslant 0, \quad \text{for all } x \in \Omega,$$

we have

$$f(x,\phi^*(x),\nabla_x\phi^*(x)) \leqslant 0, \quad \text{for all } x \in \Omega.$$

Thus,

$$\max_{x\in\Omega} f(x,\phi^*(x),\nabla_x\phi^*(x)) \leqslant 0,$$

i.e. $\phi^* \in \Psi$.

Q.E.D.

Definition 3.9:

A real-valued, nonnegative function d_Ω: $\tilde{C}^{(1)}(\Omega) \longrightarrow R$ (measures the "distance" from ϕ to ψ) is defined by

$$d_\Omega(\phi) \triangleq \max\{0, \max_{x\in\Omega} f(x,\phi(x),\nabla_x\phi(x))\}.$$

Theorem 3.1:

Assume that f in problem P1 satisfies Assumption 3.1, that $\{\phi_i\}$ is a sequence in $\tilde{C}^{(1)}(\Omega)$ converging to ϕ^* in $\tilde{C}^{(1)}(\Omega)$, and that $d_\Omega(\phi_i) \longrightarrow 0$, as $i \longrightarrow \infty$. Then, $\phi^* \in \Psi$.

Proof:

From the assumption that $d_\Omega(\phi_i) \longrightarrow 0$, as $i \longrightarrow \infty$, we obtain

$$\max_{x\in\Omega} f(x,\phi_i(x),\nabla_x\phi_i(x)) < \varepsilon_i,$$

where, $\varepsilon_i > 0$ and $\varepsilon_i \longrightarrow 0$, as $i \longrightarrow \infty$.

From Assumption 3.1, there exists (similar in the proof of Lemma 3.3) a K such that

$$|f(x,\phi_i(x),\nabla_x\phi_i(x))-f(x,\phi^*(x),\nabla_x\phi^*(x))|$$
$$\leqslant K\|\phi_i-\phi^*\|_\infty ,$$

for all $x \in \Omega$ and all $i \geqslant 0$.

Thus, for all $x \in \Omega$ and all $i \geqslant 0$,

$$f(x,\phi^*(x),\nabla_x\phi^*(x)) \leqslant K\|\phi_i-\phi^*\|_\infty+f(x,\phi_i(x),\nabla_x\phi_i(x))$$

$$\leqslant K\|\phi_i-\phi^*\|_\infty + \max_{x\in\Omega} f(x,\phi_i(x),\nabla_x\phi_i(x))$$

$$< K\|\phi_i-\phi^*\|_\infty + \varepsilon_i .$$

As $i \longrightarrow \infty$, $\|\phi_i-\phi^*\|_\infty \longrightarrow 0, \varepsilon_i \longrightarrow 0$ so that

$$f(x,\phi^*(x),\nabla_x\phi^*(x)) \leqslant 0, \quad \text{for all } x \in \Omega.$$

Hence,

$$\max_{x\in\Omega} f(x,\phi^*(x),\nabla_x\phi^*(x)) \leqslant 0 ,$$

That is $d_\Omega(\phi^*) = 0,$

$\phi^* \in \Psi.$

Q.E.D.

Now we prove the following main theorem.

Theorem 3.2:

For problem P1, suppose Ψ is not empty and f satisfies Assumption 3.1. Let sequence $\{l_i\}$ be generated by Main-algorithm 1 and Sub-algorithm 1.1 in 3 2. Then, if the sequence is finite, the last point lies in Ψ; if the sequence is infinite, then any accumulation point of it is in Ψ.

Proof:

The finite case is trivial.

In the infinite case, since Ψ is not empty and the class of piecewise linear continuous functions is dense in the class of continuous functions, the algorithm does not jam between Step 2 and Step 3 in Main-algorithm 1 .

Suppose $\{l_{n_i}\}$ is a sub-sequence of $\{l_i\}$ and $\{l_{n_i}\}$ converges to l^*, as $i \longrightarrow \infty$. As in the proof of Lemma 3.3, by Assumption 3.1, there exists a K such that

$$|f(x,l_{n_i}(x),\nabla_x l_{n_i}(x))-f(x,l_{n_j}(x),\nabla_x l_{n_j}(x))|$$
$$\leqslant K(|l_{n_i}(x)-l_{n_j}(x)|+\|\nabla_x l_{n_i}(x)-\nabla_x l_{n_j}(x)\|_n),$$

for all x in Ω and all $i,j \geqslant 0$.

By construction, ω^{n_i} is a proper subset of ω^{n_j}, for $i < j$.

Let x_i^* satisfy

$$f(x_i^*,l_{n_i}(x_i^*),\nabla_x l_{n_i}(x_i^*)) = \max_{x\in\Omega} f(x,l_{n_i}(x),\nabla_x l_{n_i}(x)) .$$

By construction,

$$f(x_i^*,l_{n_i}(x_i^*),\nabla_x l_{n_i}(x_i^*)) > 0, \quad \text{for all } i \geqslant 0.$$

Also, by construction, $x_i^* \in \omega^{n_j}$, and

$$f(x_i^*,l_{n_j}(x_i^*),\nabla_x l_{n_j}(x_i^*)) \leqslant 0,$$
$$\text{for all } i \geqslant 0, \text{ all } j > i.$$

Thus,

$$\|l_{n_j}-l_{n_i}\|_\infty \geqslant |l_{n_j}(x_i^*)-l_{n_i}(x_i^*)| + \|\nabla_x l_{n_j}(x_i^*)-\nabla_x l_{n_i}(x_i^*)\|_n$$

$$\geqslant K^{-1}|f(x_i^*,l_{n_i}(x_i^*),\nabla_x l_{n_i}(x_i^*))-f(x_i^*,l_{n_j}(x_i^*),\nabla_x l_{n_j}(x_i^*))|$$

$$= K^{-1}(f(x_i^*,l_{n_i}(x_i^*),\nabla_x l_{n_i}(x_i^*))-f(x_i^*,l_{n_j}(x_i^*),\nabla_x l_{n_j}(x_i^*)))$$

$$\geqslant K^{-1} f(x_i^*, l_{n_i}(x_i^*), \nabla_x l_{n_i}(x_i^*))$$

$$= K^{-1} d_\Omega(l_{n_i}) \ .$$

But, since $l_{n_i} \longrightarrow l^*$, as $i \longrightarrow \infty$, and

$$K^{-1} d_\Omega(l_{n_i}) \leqslant \| l_{n_j} - l_{n_i} \|_\infty,$$

for all i,j such that $j > i$,

the right side also tends to zero as $i \longrightarrow \infty$.

Thus, $d_\Omega(l_{n_i}) \longrightarrow 0$, as $i \longrightarrow \infty$.

From Theorem 3.1, we have

$$l^* \in \Psi \ .$$

Q.E.D.

We notice that an infinite sequence of piecewise linear continuous functions does not always have an accumulation point in the class of the piecewise continuously differentiable functions. However, by construction and the continuity, the "distance" between the functions constructed and the feasible function set hopefully tends to zero.

Now we consider Main-algorithm 1 together with Sub-algorithm 1.1 and Sub-algorithm 1.2.

Definition 3.10:

For a function l in $\tilde{C}^{(1)}(\Omega)$, define

$$M(l) \triangleq \max_{x\in\Omega} f(x,l(x),\nabla_x l(x)).$$

Definition 3.11:

Similarly, we define, for any subset Ω_i of Ω,

$$\hat{M}_{\Omega_i}(l) \triangleq \max_{x\in\Omega_i} f(x,l(x),\nabla_x l(x)).$$

Definition 3.12: (cf. Def.3.9)

Define a real-valued, nonnegative distance function $\hat{d}_{\Omega_i} : \tilde{C}^{(1)}(\Omega) \longrightarrow R$ by

$$\hat{d}_{\Omega_i}(l) \triangleq \max\{0, \hat{M}_{\Omega_i}(l)\} ,$$

where, Ω_i is a subset of Ω.

Now we state a theorem which parallels Thm. 3.1.

Theorem 3.3:

Suppose f in problem P1 satisfies Assumption 3.1, $\{l_i\}$ is a sequence in $\tilde{C}^{(1)}(\Omega)$ converging to l in $\tilde{C}^{(1)}(\Omega)$ and $\{\Omega_i\}$ is a sequence of subsets of Ω. If $\hat{d}_{\Omega_i}(l_i) \longrightarrow 0$, as $i \longrightarrow \infty$, and, for all $i \geq 0$, $\hat{M}_{\Omega_i}(l_i) \geq M(l_i)-\gamma_i$, where $\gamma_i \geq 0$ and $\gamma_i \longrightarrow 0$, as $i \longrightarrow \infty$, then $l \in \Psi$.

Proof:

As in the proof of Thm. 3.1, since $\hat{d}_{\Omega_i}(l_i) \longrightarrow 0$, as $i \longrightarrow \infty$, we may deduce that

$$\hat{M}_{\Omega_i}(l_i) < \varepsilon_i,$$

where $\varepsilon_i > 0$ and $\varepsilon_i \longrightarrow 0$, as $i \longrightarrow \infty$.

Also, since $\{l_i\}$ converges to l, by Assumption 3.1, there exists a K such that for any x in Ω and any i,

$$|f(x,l_i(x),\nabla_x l_i(x))-f(x,l(x),\nabla_x l(x))| \leq K\|l_i-l\|_\infty.$$

Thus,

$$f(x,l(x),\nabla_x l(x)) \leqslant K\|l_i - l\|_\infty + f(x,l_i(x),\nabla_x l_i(x))$$

$$\leqslant K\|l_i - l\|_\infty + M(l_i)$$

$$\leqslant K\|l_i - l\|_\infty + \hat{M}_{\Omega_i}(l_i) + \gamma_i$$

$$< K\|l_i - l\|_\infty + \varepsilon_i + \gamma_i \; .$$

Allowing i to tend to infinity, we obtain

$$f(x,l(x),\nabla_x l(x)) \leqslant 0, \quad \text{for all } x \text{ in } \Omega.$$

Hence, $l \in \Psi$.

Q.E.D.

From the construction of Main-algorithm 1 and Sub-algorithm 1.2, we can see that the number of simplices over Ω and the number of points in every simplex are both increasing as the computation proceeds. Hence it is obvious that the functions generated by these algorithms satisfy the assumptions regarding $\hat{M}$ and M. Therefore, we can have the following theorem.

Theorem 3.4:

For problem P1, suppose Ψ is not empty and f satisfies Assumption 3.1. Let sequence $\{l_i\}$ be generated by Main-algorithm 1 together with Sub-algorithms 1.1 and 1.2. Then, if the sequence is infinite, any accumulation point of it is in Ψ.

Proof:

The proof is almost the same as the proof in Theorem 3.2

except using $\hat{M}_{\Omega_l}(l_l)$ in the place of $\max_{x\in\Omega} f(x, l(x), \nabla_x l(x))$, using $\hat{d}$ instead of d and employing Thm. 3.3 instead of Thm. 3.1.

Q.E.D.

3.4 Algorithm for problem P2

For the constrained optimization problem P2, the following main-algorithm is proposed.

For a set of simplices, $\tilde{s}(\omega^l)$, we use d^l to denote the size of $\tilde{s}(\omega^l)$ (e.g., d^l is the largest volume of the simplices in $\tilde{s}(\omega^l)$).

Main-algorithm 2:

Data: ω^0, $\tilde{s}(\omega^0)$, $L(\omega^0, \tilde{s}(\omega^0))$ as in Main-algorithm 1;

$\delta > 0$.

Step 0: Set $l = 0$;

Set $\tilde{s}(\omega^{-1}) = \tilde{s}(\omega^0)$.

Step 1: Construct $\tilde{s}(\omega^l)$ containing $\tilde{s}(\omega^{l-1})$;

Compute d^l - the size of $\tilde{s}(\omega^l)$;

Find x^l - the dividing-point of the simplex with d^l.

Step 2: Solve problem $P(\omega^l)$:

$$P(\omega^l): \min \{ f^0(l) \mid l \in L(\omega^l, \tilde{s}(\omega^l)); \quad f(x, l(x), \nabla_x l(x)) \leqslant 0, \text{ for all } x \in \omega^l \}$$

to obtain l_l

Step 3: Find $x^* \in U\tilde{s}(\omega^l)$ such that

$$x^* = \underset{x}{\operatorname{argmax}} \{ f(x, l_l(x), \nabla_x l_l(x)) \mid x \in U\tilde{s}(\omega^l) \}$$

Step 4: If $f(x^*, l_i(x^*), \nabla_x l_i(x^*)) \leqslant 0$, go to Step 5;

Else, set $\omega^{i+1} = \omega^i \cup \{x^*\}$,

$i = i+1$,

go to Step 1

Step 5: If $d^i < \delta$, stop;

Else, set $\omega^{i+1} = \omega^i \cup \{ x^i \}$,

$i = i+1$,

go to Step 1.

Step 3 in Main-algorithm 2 can be replaced by Sub-algorithm 1.2 as in Main-algorithm 1. Step 2 is now a minimization problem with finite variables and finite constraints. Hence available algorithms can be used. The point is that the feasible set of l in Step 2 might be empty because of the "coarseness" of the simplices even though the existence of a solution to problem P2 is guaranteed. Thus modifications, such as use of the auxiliary nodal set Δ^j as in Main-algorithm 1, should be made. The modified algorithm is as follows.

Main-algorithm 2':

Data: ω^0, $\tilde{s}(\omega^0)$, $L(\omega^0, \tilde{s}(\omega^0))$ as in Main-algorithm 1;

$\delta > 0$.

Step 0: Set $i = 0$;

Set $j = 0$;

Set $\tilde{s}(\omega^{-1}) = \tilde{s}(\omega^{0})$;

Set Δ^{0} = empty set;

Set $\tilde{S}(\omega^{0,-1}) = \tilde{s}(\omega^{0})$.

Step 1: Construct $\tilde{s}(\omega^{i})$ containing $\tilde{s}(\omega^{i-1})$;

Compute d^{i} - the size of $\tilde{s}(\omega^{i})$;

Find x^{i} - the dividing-point of the simplex with d^{i};

Let $\omega^{i,j} = \omega^{i} \cup \Delta^{j}$.

Step 2: Let $\omega^{i,j} = \omega^{i,j} \cup \Delta^{j}$;

Construct $\tilde{s}(\omega^{i,j})$ containing $\tilde{s}(\omega^{i,j-1})$.

Step 3: Solve problem $P(\omega^{i,j})$:

$$P(\omega^{i,j}): \quad \min \{ f^{0}(l) \mid l \in L(\omega^{i,j}, \tilde{s}(\omega^{i,j}));\ f(x, l(x), \nabla_{x} l(x)) \leqslant 0, \text{ for all } x \in \omega^{i} \}$$

to obtain l_{i};

If the feasible function set is empty,

let Δ^{j+1} be the dividing-point set of $\tilde{s}(\omega^{i,j})$,

set $j = j+1$ and go to Step 2.

Step 4: Find $x^{*} \in \cup \tilde{s}(\omega^{i,j})$ such that

$$x^{*} = \underset{x}{\operatorname{argmax}} \{ f(x, l_{i}(x), \nabla_{x} l_{i}(x)) \mid x \in \cup \tilde{s}(\omega^{i,j}) \}$$

Step 5: If $f(x^{*}, l_{i}(x^{*}), \nabla_{x} l_{i}(x^{*})) \leqslant 0$, go to Step 6;

Else, set $\omega^{i+1} = \omega^{i} \cup \{x^{*}\}$,

set $i = i+1$,

set $j = 0$,

set Δ^0 = empty set, and

go to Step 1.

Step 6: If $d^i < \delta$, stop;

Else, set $\omega^{i+1} = \omega^i \cup \{x^i\}$,

set $i = i+1$,

set $j = 0$,

set Δ^0 = empty set, and

go to Step 1.

3.5 Analysis of algorithm in Sec.3.4

For the constrained optimization problem $\tilde{P}_2$, a bound assumption is needed.

Assumption 3.2:

We assume that the set of feasible function in $\tilde{P}2$ is contained in a bounded subset, $\hat{C}^{(1)}(\Omega)$, of $\tilde{C}^{(1)}(\Omega)$, i.e., there exists a finite positive number B such that

$$\|\phi\|_\infty \leqslant B, \qquad \text{for all } \phi \in \hat{C}^{(1)}(\Omega).$$

This assumption is often achieved in control system designs since the control law we want to construct usually is restricted in magnitude.

In this section, we always restrict the feasible set (Def.3.8) on $\hat{C}^{(1)}(\Omega)$.

Definition 3.13:

The feasible set for problem $P(\omega^{i,j})$ in Step 3 of Main-algorithm 2' is defined as follows

$$\Psi(\omega^{i,j}) \triangleq \{l \in L(\omega^{i,j}, \tilde{s}(\omega^{i,j})) \mid f(x, l(x), \nabla_x l(x)) \leqslant 0, \text{ for all } x \in \omega^i\}.$$

Similarly, we define

$$\Psi(\omega^{i}) \triangleq \{l \in L(\omega^{i}, \tilde{s}(\omega^{i})) \mid f(x, l(x), \nabla_x l(x)) \leqslant 0, \text{ for all } x \in \omega^i\}.$$

Clearly, for all $j \geqslant 0$,

$$\Psi(\omega^i) \subset \Psi(\omega^{i,j}).$$

By a solution l to Problem P2, we mean one of the two situations as follows:

(1) There is a set of simplices, $\tilde{s}(\omega^1)$, satisfying the precision requirement δ in Main-algorithm 2 or 2'; l is such that

$$f^0(l) = \inf\{f^0(\psi) \mid \psi \in \Psi(\omega^1)\},$$

and

$$f(x,l(x),\nabla_x l(x)) \leq 0, \qquad \text{for all} \quad x \in \Omega;$$

(2) There is a sequence of simplex sets, $\{\tilde{s}(\omega^i)\}$, such that

$\tilde{s}(\omega^i)$ meets the precision requirement for all i;

$l \in \hat{C}^{(1)}(\Omega)$ and is such that

$$f^0(l) \leq f^0(\psi), \quad \text{for all } \psi \in \Psi(\omega^i), \quad \text{for all } i,$$

and

$$f(x,l(x),\nabla_x l(x)) \leq 0, \qquad \text{for all} \quad x \in \Omega.$$

Lemma 3.4:

Suppose S is a simplex in R^n whose vertices are $\{x^1, \ldots, x^{n+1}\}$. Let l_1, l_2 be two linear scalar functions on S, and vectors β, ζ be defined by the values of l_1, l_2 at the vertices of S, i.e.,

$$\beta = \begin{pmatrix} \beta_1 \\ \cdot \\ \cdot \\ \cdot \\ \beta_{n+1} \end{pmatrix} = \begin{pmatrix} l_1(x^1) \\ \cdot \\ \cdot \\ \cdot \\ l_1(x^{n+1}) \end{pmatrix}, \quad \zeta = \begin{pmatrix} \zeta_1 \\ \cdot \\ \cdot \\ \cdot \\ \zeta_{n+1} \end{pmatrix} = \begin{pmatrix} l_2(x^1) \\ \cdot \\ \cdot \\ \cdot \\ l_2(x^{n+1}) \end{pmatrix}$$

Then, on S, $\| l_1 - l_2 \|_\infty = \| \beta - \zeta \|_\infty + \| b_1 - b_2 \|_n$,

where, $\| l_1 - l_2 \|_\infty$ defined as in Def.3.7, and

$$\| \beta - \zeta \|_\infty \triangleq \max\{ | \beta_j - \zeta_j | \;\big|\; j = 1, \ldots, n+1 \}$$

and b_1, b_2 are, respectively, the gradients of l_1, l_2 on S.

Proof:

Let $l_1(x) = a_1 + b_1^T x$, $l_2(x) = a_2 + b_2^T x$,

$$y^i \triangleq x^{i+1} - x^1, \quad i = 1, \ldots, n.$$

Choose j such that

$$| \beta_j - \zeta_j | = \| \beta - \zeta \|_\infty .$$

For all $x \in S$, there exist $\alpha_1, \ldots, \alpha_n \geq 0, \sum_{i=1}^{n} \alpha_i \leq 1$,

such that $x = x^1 + \sum_{i=1}^{n} \alpha_i y^i$.

Thus, for all x in S,

$$l_1(x) = a_1 + b_1^T (x^1 + \sum_{i=1}^{n} \alpha_i y^i)$$

$$= a_1 + b_1^T x^1 + \sum_{i=1}^{n} \alpha_i b_1^T (x^{i+1} - x^1)$$

$$= \beta_1 + \sum_{i=1}^{n} \alpha_i \beta_{i+1} - \sum_{i=1}^{n} \alpha_i \beta_1$$

$$= \beta_1 (1 - \sum_{i=1}^{n} \alpha_i) + \sum_{i=1}^{n} \alpha_i \beta_{i+1} .$$

Similarly, $l_2(x) = \zeta_1 (1 - \sum_{i=1}^{n} \alpha_i) + \sum_{i=1}^{n} \alpha_i \zeta_{i+1}$.

Then, for all x in S,

$$| l_1(x) - l_2(x) | = | (\beta_1 - \zeta_1)(1 - \sum_{i=1}^{n} \alpha_i) + \sum_{i=1}^{n} \alpha_i (\beta_{i+1} - \zeta_{i+1}) |$$

$$\leqslant |\beta_1-\zeta_1|(1-\sum_{i=1}^{n}\alpha_i)+\sum_{i=1}^{n}\alpha_i|\beta_{i+1}-\zeta_{i+1}|$$

$$\leqslant |\beta_j-\zeta_j|$$

$$= \|\beta-\zeta\|_\infty ,$$

so that

$$|l_1(x)-l_2(x)| + \|b_1-b_2\|_n \leqslant \|\beta-\zeta\|_\infty + \|b_1-b_2\|_n .$$

Hence,

$$\|l_1-l_2\|_\infty \leqslant \|\beta-\zeta\|_\infty + \|b_1-b_2\|_n .$$

On the other hand, it is obvious that

$$\|\beta-\zeta\|_\infty + \|b_1-b_2\|_n \leqslant \|l_1-l_2\|_\infty ,$$

since $x^j \in S$.

Hence, the conclusion of the lemma holds.

Q.E.D.

Proposition 3.1:

For every $i > 0$, if $\Psi(\omega^{i,j})$ is not empty, then it is a sequentially compact set (i.e., every infinite sequence of functions in $\Psi(\omega^{i,j})$ has, at least, one convergent sub-sequence).

Proof:

Suppose $\Psi(\omega^{i,j})$ is not empty.

Let E be a "cubic" subset of the finite-dimensional space $R^{n_{i,j}} \times R^{m_{i,j} \times n}$:

$$E = \{y \in R^{n_{i,j}} \times R^{m_{i,j} \times n} \mid \|y\|_\infty \leqslant B\},$$

where, $n_{i,j}$ is the cardinality of $\omega^{i,j}$, $m_{i,j}$ is the number of simplices in $\tilde{s}(\omega^{i,j})$, B is the bound in Assumption 3.2.

Define a map Γ from $L(\omega^{i,j}, \tilde{s}(\omega^{i,j}))$ into E by

$$\Gamma(l) = [\theta,\ \eta]^T ,$$

where, θ is a vector in $R^{n_{i,j}}$, of which the components are just the value of l at each point in $\omega^{i,j}$; η is a vector in $R^{m_{i,j}\cdot n}$, of which the components (every n in a group) are the gradients of l in each simplex in $\tilde{s}(\omega^{i,j})$.

Suppose $\{l_j\}$ is an infinite sequence in $\Psi(\omega^{i,j})$, there is hence a corresponding sequence of vectors $\{[\theta_j,\eta_j]^T\}$ in E. Since E is a closed and bounded subset of $R^{n_{i,j}}\times R^{m_{i,j}\times n}$, there exists a subsequence $\{[\theta_{p_j},\eta_{p_j}]^T\} \in E$,

$$[\theta_{p_j},\eta_{p_j}]^T \longrightarrow [\theta^*,\eta^*]^T \in E,$$

$$\text{as } j \longrightarrow \infty.$$

Form a piecewise linear, continuous function l^* on Ω, i.e. $l^* \in \hat{C}^{(1)}(\Omega)$, with θ^* as its value at points in $\omega^{i,j}$ (having the same structure as the elements of $L(\omega^{i,j},\tilde{s}(\omega^{i,j}))$).

By an obvious extension of Lemma 3.4, we have , for all j,

$$\|l_{p_j}-l^*\|_\infty \leqslant \|[\theta_{p_j},\eta_{p_j}]^T-[\theta^*,\eta^*]^T\|_\infty .$$

Since the right side tends to zero as $j \longrightarrow \infty$,

$$l_{p_j} \longrightarrow l^*, \qquad \text{as } j \longrightarrow \infty.$$

Now, for any $x \in \omega^i$, by Assumption 3.1, there exists a K such that

$$|f(x,l_{p_j}(x),\nabla_x l_{p_j}(x))-f(x,l^*(x),\nabla_x l^*(x))|$$

$$\leqslant K(|l_{p_j}(x)-l^*(x)|+\|\nabla_x l_{p_j}(x)-\nabla_x l^*(x)\|_\infty)$$

$$\leqslant K\|l_{p_j}-l^*\|_\infty .$$

Thus, as $j \longrightarrow \infty$,

$$f(x,l_{p_j}(x),\nabla_x l_{p_j}(x)) \longrightarrow f(x,l^*(x),\nabla_x l^*(x)) ,$$

$$\text{for all } x \in \omega^i.$$

It follows that, for all $x \in \omega^i$,

$$f(x,l^*(x),\nabla_x l^*(x)) \leqslant 0,$$

since for every j,

$$f(x, l_{p_j}(x), \nabla_x l_{p_j}(x)) \leqslant 0, \quad \text{for all} \quad x \in \omega^i .$$

Thus,

$$l^* \in \Psi(\omega^{i,j}) \ .$$

Hence $\Psi(\omega^{i,j})$ is sequentially compact.

Q.E.D.

Since the set of piecewise linear continuous functions is dense in the set of continuous functions, for every $i \geqslant 0$, $\Psi(\omega^{i,j})$ will be not empty for a finite j. (We assume that Ψ, as defined in Def.3.8 with the restriction in $\hat{C}^{(1)}(\Omega)$, is not empty.) Hence we have the following obvious theorem.

Theorem 3.5:

The problem $P(\omega^{i,j})$ in Main-algorithm 2' is solvable, i.e., the minimum is achievable, for some finite j.

For Main-algorithm 2', we have the following result.

Theorem 3.6:

Suppose Ψ, restricted in $\hat{C}^{(1)}(\Omega)$, is not empty, Assumptions 3.1 and 3.2 are satisfied and f^0 is continuous. Let $\{l_i\}$ be a sequence generated by Main-algorithm 2'. Then, if $\{l_i\}$ is finite, the last point is an approximation solution to problem P2 with satisfactory precision δ; if $\{l_i\}$ is infinite, then any accumulation point of this sequence solves problem P2.

Proof:

The finite case is trivial.

In the infinite case, let l^* be an accumulation point of $\{l_i\}$. For the sake of convenience, we assume

$$l_i \longrightarrow l^*, \quad \text{as} \quad i \longrightarrow \infty.$$

Since $\{l_i\}$ is infinite, δ is a given number, it is clear that, after a finite number of iterations, the computation does not enter Step 6 (otherwise the execution will stop because of the decreasing of d^i). Hence we can assume that the size of $\tilde{s}(\omega^i)$ is less that δ for all i. From the discussion earlier about problem P1, $l^* \in \Psi$ (see Thm.3.2).

For a function $\psi \in \Psi(\omega^i)$, there is a j (depending on i) such that

$$f^0(l_i) = \min\{f^0(l) \mid l \in \Psi(\omega^{i,j})\} \leqslant f^0(\psi).$$

By construction, for all $k > i$, as $i \longrightarrow \infty$,

$$f^0(l_k) \leqslant f^0(l_i).$$

Thus, $f^0(l_k) \leqslant f^0(\psi)$, for all $k > i$.

Allowing k to tend to infinity yields

$$f^0(l^*) \leqslant f^0(\psi).$$

Hence, l^* is a solution to problem P2.

Q.E.D.

Actually, the solution generated by Main-algorithm 2' lies in the piecewise linear continuous function class, which is a subspace of the piecewise continuously differentiable function space. This is different from the solution to problem $\tilde{P}2$, where a continuously differentiable function is required. Discussion on the optimality of solutions subject to these different spaces is left to Section 3.6.

3.6 A sequence approximation theory

As we stated earlier, the original problem $\tilde{P}_2$ is to seek an optimal function in the space of continuously differentiable functions. But the procedures proposed in the above sections are carried out in the class of piecewise linear continuous functions which does not include the class of continuously differentiable functions nor vice verse. Can the solution obtained by Main-algorithm 2' really solve the original problem $\tilde{P}2$? In this section, this question will be considered in a broader situation.

Assumption 3.3:

Let C be a closed subset of a Banach space, E is a subset of C, and $\{C_i\}$ is a family of subsets in C satisfying

$$C_i \subset C_{i+1}, \quad i = 1,2,...$$

$$\overline{\bigcup_{i=1}^{\infty} C_i} = C;$$

Also, $\{E_i\}$ is a family of subsets in E satisfying

$$E_i \subset E_{i+1}, \quad i = 1,2,...$$

$$\overline{\bigcup_{i=1}^{\infty} E_i} = E,$$

where, for any set S, $\bar{S}$ denotes the closure of S.

Definition 3.14:

Let E be a subset of C, and let both E and C be elements of a normed space (called the base space). Let $f: E \longrightarrow R$ be a global Lipschitz continuous functional on E. A functional $\tilde{f}$ is called a Lipschitz Bounded Extension Functional (LBEF) of f, if $\tilde{f}$ is defined on C and there exists a positive finite K such that

$$\tilde{f}(\phi) = f(\phi), \quad \text{for all } \phi \text{ in } E,$$

$$|\tilde{f}(\phi_1)-\tilde{f}(\phi_2)| \leqslant K\|\phi_1-\phi_2\|, \quad \text{for all } \phi_1, \phi_2 \text{ in } C,$$

where, $\|\cdot\|$ is the norm in the base space.

Now we consider the following optimization problem.

$$P(E): \quad \inf \{ f(\psi) \mid \psi \in E \} .$$

Definition 3.15:

For any functional f: $E \longrightarrow R$, a function ψ^* in E is called a solution of problem $P(E)$ if

$$f(\psi^*) = \inf\{f(\psi) \mid \psi \in E \}.$$

Definition 3.16:

Let f be a functional on E, f: $E \longrightarrow R$. $\{E_i\}$ is a family of subsets in E and $\{\varepsilon_i\}$ is a positive sequence converging to zero. A function $\psi_i^* \in E_i$ is called a solution of optimization problem $P(E_i, \varepsilon_i)$ if it satisfies

$$f(\psi_i^*) \leqslant \inf\{f(\psi) \mid \psi \in E_i\} + \varepsilon_i .$$

Lemma 3.5:

Let E be a subset of a closed subset in a Banach space, $\{E_i\}$ a family of subsets of E satisfying Assumption 3.3. Let ψ^* and $\{\psi_i^*\}$ be solutions for $P(E)$ and $P(E_i, \varepsilon_i)$, respectively. If the functional f is continuous, then

$$f(\psi_i^*) \longrightarrow f(\psi^*), \quad \text{as } i \longrightarrow \infty,$$

and any accumulation point in E of $\{\psi_i^*\}$, if there exists, solves $P(E)$.

Proof:

Since E_i increasingly approaches (Assum.3.3) E, there exists a sequence $\{\hat{\psi}_i\}$ such that $\hat{\psi}_i \in E_i$, for every i and

$$\hat{\psi}_i \longrightarrow \psi^*, \quad \text{as } i \longrightarrow \infty.$$

It is also obvious that $\{\inf\{f(\psi_i)|\psi_i \in E_i\}\}$ is decreasing monotonically with i and is bounded below by $f(\psi^*)$.

Thus, from

$$\inf_{\psi_i \in E_i} f(\psi) \leqslant f(\hat{\psi}_i),$$

we have

$$f(\psi^*) \leqslant \lim_i \left(\inf_{\psi_i \in E_i} f(\psi_i) \right)$$

$$\leqslant \lim_i f(\hat{\psi}_i)$$

$$= f(\psi^*) ,$$

where, the last equality comes from the continuity of f.

Hence,

$$\lim_i \left(\inf_{\psi_i \in E_i} f(\psi_i) \right) = f(\psi^*) .$$

Since

$$f(\psi^*) \leqslant f(\psi_i^*) \leqslant \inf_{\psi_i \in E_i} f(\psi_i)+\varepsilon_i ,$$

allowing i to tend to infinity yields

$$f(\psi^*) \leqslant \lim_i \inf(f(\psi_i^*))$$

$$\leqslant \lim_i \sup(f(\psi_i^*))$$

$$\leqslant \lim_i \left(\inf_{\psi_i \in E_i} f(\psi_i)+\varepsilon_i \right)$$

$$= f(\psi^*),$$

that is

$$f(\psi_i^*) \longrightarrow f(\psi^*), \quad \text{as } i \longrightarrow \infty.$$

Finally, if there exists an accumulation point $\hat{\psi}$ of $\{\psi_i^*\}$ such that

$$\psi_{i_k}^* \longrightarrow \hat{\psi}, \quad \text{as } k \longrightarrow \infty,$$

from the continuity of f, we have

$$f(\hat{\psi}) = \lim_k f(\psi_{i_k}^*) \ .$$

It follows that

$$f(\hat{\psi}) = f(\psi^*) \ ,$$

since

$$f(\psi_i^*) \longrightarrow f(\psi^*), \quad \text{as } i \longrightarrow \infty.$$

Q.E.D.

Suppose C, E, $\{C_i\}$, $\{E_i\}$ satisfy Assumption 3.3, f is a continuous functional on E and $\tilde{f}$ is an LBEF of f on C. Let $\{\phi_i^*\}$ be a sequence of solutions for problems $\{P(C_i,\varepsilon_i)\}$ and ψ^* solves P(E).

Definition 3.17:

For every $\phi_i \in C_i$, the distance between ϕ_i and E_i is defined by

$$d^0(\phi_i, E_i) \triangleq \inf\{\|\phi_i - \psi\| \mid \psi \in E_i\} \ .$$

Definition 3.18:

For every C_i, define

$$\eta_i \triangleq \sup_{\phi_i \in C_i} d^0(\phi_i, E_i).$$

Then the following theorem is available.

Theorem 3.7:

If η_i tends to zero as i tends to infinity, then

$$\tilde{f}(\phi_i^*) \longrightarrow f(\psi^*), \quad \text{as } i \longrightarrow \infty,$$

and, any accumulation point $\hat{\phi}$ of $\{\phi_i^*\}$, if it exists, solves problem P(E) in the sense that

$$\tilde{f}(\hat{\phi}) = \inf\{f(\psi) \mid \psi \in E\} .$$

Proof:

It is clear that f and $\tilde{f}$ are continuous.

Assume that ϕ^* solves P(C) and $\{\psi_i^*\}$ solve $\{P(E_i)\}$, for all i, i.e.

$$\tilde{f}(\phi^*) = \inf\{\tilde{f}(\phi) \mid \phi \in C\},$$

$$f(\psi_i^*) \leqslant \inf\{f(\psi) \mid \psi \in E_i\} + \varepsilon_i,$$

where, $\{\varepsilon_i\}$ is a sequence given in Def.3.16.

By Lemma 3.5, we have

$$\lim_i \tilde{f}(\phi_i^*) = \tilde{f}(\phi^*),$$
$$\lim_i f(\psi_i^*) = f(\psi^*).$$

For every ϕ_i^* there exists a $\hat{\psi}_i \in E_i$ such that

$$\|\phi_i^* - \hat{\psi}_i\| \leqslant 2\eta_i,$$

where, η_i is as defined in Def.3.18.

Hence, from Definition 3.14,

$$|\tilde{f}(\phi_i^*) - f(\hat{\psi}_i)| \leqslant K\|\phi_i^* - \hat{\psi}_i\| \leqslant 2K\eta_i.$$

Letting i tend to infinity yields

$$\tilde{f}(\phi^*) = \lim_i \tilde{f}(\phi_i^*) = \lim_i f(\hat{\psi}_i)$$

$$\geqslant \lim_i f(\psi_i^*) = f(\psi^*) \geqslant \tilde{f}(\phi^*).$$

Thus,

$$\lim_i \tilde{f}(\phi_i^*) = f(\psi^*) .$$

Finally, let $\{\phi^*_{i_k}\}$ be a subsequence of $\{\phi^*_i\}$ which converges to a function $\Phi \in C$, i.e.,

$$\phi^*_{i_k} \longrightarrow \Phi, \qquad \text{as } k \longrightarrow \infty.$$

Then, from the continuity of $\tilde{f}$,

$$\tilde{f}(\Phi) = \lim_k \tilde{f}(\phi^*_{i_k})$$

$$= \lim_i \tilde{f}(\phi^*_i)$$

$$= f(\psi^*) \; .$$

That is,

$$\tilde{f}(\Phi) = \inf\{f(\psi) \mid \psi \in E\}.$$

Q.E.D.

The theorem shows the accumulation point Φ of $\{\phi^*_i\}$, if it exists, really solves problem P(E).

Finally, we state the following obvious corollary.

Corollary 3.1:

The sequence $\{ \tilde{f}(\phi^*_i) \}$ is a monotonically decreasing sequence.

From these results and the properties of function classes we employed in algorithms, we are able to conclude that the solutions generated by Main-algorithm 2 do solve problem $\tilde{P}2$ in sense of the statements in Thm.3.7.

3.7 Independent simplex construction scheme

In the design of control systems it may be desirable to have a set of "regular" simplices over the region of state space on which the control law is defined. Suppose such a set is $\{S^1, S^2, S^3, \ldots\}$, in which S^i is a set of simplices and S^{i+1} "contains" S^i in sense that every simplex in S^i is the union of some simplex(es) in S^{i+1}. (Cf. Def.3.2.) For instance, S^{i+1} might be made by dividing each simplex in S^i into n by the "dividing-point" (Cf. Def.3.5). In such cases, Main- algorithm 1 and Main-algorithm 2 are still valid with some slight modifications. The following proto-type algorithm is a modified version of Main-algorithm 1 with Sub-algorithm 1.1, which shows the necessary alterations.

Problem:

Find a piecewise linear continuous function l defined on region Ω such that

$$f(x, l(x), \nabla_x l(x)) \leqslant 0, \quad \text{for all } x \in \Omega ,$$

where, the value of $\nabla_x l$ at points of discontinuity is defined to be the local average value.

Without loss of generality, we suppose Ω is the union of S^1, where S^1 is a set of simplices, and denote P^1 as the set of nodal points in S^1.

Main-algorithm 3:

Data: $\delta > 0$;

Set $\{k_1, k_2, \ldots\}$ of positive integers satisfying

$k_i \longrightarrow \infty$, as $i \longrightarrow \infty$;

$\{S^1, S^2, \ldots\}$ and $\{P^1, P^2, \ldots\}$ as specified above.

Step 0: Set $i = 1$

Step 1: Find a piecewise linear continuous function l_i on US^i such that

$$f(x, l_i(x), \nabla_x l_i(x)) \leqslant 0, \quad \text{for all } x \in P^i,$$

using an iterative algorithm;

If the iteration number reaches k_i and a solution l_i has not been found, stop the execution, set

$$i = i + 1,$$

and go to the iterative algorithm again;

Otherwise, go to Step 2.

Step 2: Find an $x^* \in P^{i+1}$ such that

$$x^* = \text{argmax}\{f(x, l_i(x), \nabla_x l_i(x)) \mid x \in P^{i+1}\};$$

Compute d^i -- the of S^i

(Cf. Main-algorithm 2 or 2').

Step 3: If $f(x^*, l_i(x^*), \nabla_x l_i(x^*)) \leqslant 0$, go to Step 4;

Else, set $i = i+1$, and go to Step 1.

Step 4: If $d^i < \delta$, stop;

Else, set $i = i+1$, and go to Step 1.

Main-algorithm 3 is, in fact, a special case of Main-algorithm 1 together with Sub-algorithm 1.1 and Sub-algorithm 1.2. Thus Theorem 3.4 remains valid, i.e., any accumulation point of an infinite sequence generated by the algorithm will be feasible.

Main-algorithm 3 is linked more closely to the conventional finite element method. The "distance" between the functions constructed by this algorithm and the feasible set for P1 is more likely to tend to zero.

3.8 Constraint construction scheme

Algorithms proposed in Sec.3.2, Sec.3.4 and Sec.3.7 are of the outer approximation type. The drawback of such algorithms is that the number of constraints will grow quickly with iteration number i. Hence some constraint dropping scheme has to be considered.

We state it with the feasibility problem P1. That is

P1: Find a piecewise continuous differentiable function ϕ on Ω, where $\Omega \subset R^n$, such that

$$f(x, \phi(x), \nabla_x \phi(x)) \leqslant 0, \quad \text{for all } x \in \Omega .$$

Definition 3.19:

Define a functional $\tilde{f}$: $L(\Omega) \longrightarrow R$ by

$$\tilde{f}(\phi) = \max \{f(x, \phi(x), \nabla_x \phi(x)) \mid x \in \Omega\}.$$

To specify the dependence of $\tilde{f}$ on the maximal point in Ω, it can also be denoted as

$$\bar{f}_\phi(x^*) \triangleq \tilde{f}(\phi),$$

where,

$$x^* \in \{\operatorname{argmax}\{f(x, \phi(x), \nabla_x \phi(x)) \mid x \in \Omega\}.$$

The constraint for problem P1 is now in the following form:

$$\bar{f}_\phi(x^*) \leqslant 0.$$

The following constraint construction scheme can be proposed.

Constraint construction scheme:

(1) Given a positive constant K and a positive, decreasing sequence $\{\varepsilon_i\}$ with $\varepsilon_i \longrightarrow 0$, as $i \longrightarrow \infty$.

(2) At Iteration i with function ϕ_i, compute

$$x^i \in \{\text{argmax } \{f(x,\phi_i(x),\nabla_x\phi_i(x)) \mid x \in \Omega\}\}.$$

(3) Define $\bar{\Omega}_i \triangleq \omega^i \cup \{x^i\}$;

Construct $\omega^{i+1} = \bar{\Omega}_i - \Omega_i$,

where,

$$\Omega_i \triangleq \{x^j \in \bar{\Omega}_i \mid \tilde{f}_{\phi_j}(x^j) \leqslant K(\varepsilon_j-\varepsilon_i),\ j = 0,1,\ldots,i\}.$$

Apart from the specified restriction in (3), the construction of the discrete subset ω^i is arbitrary in the sense that other points x lying in Ω, not specifically covered by the scheme, can be added to the set ω^i.

This procedure can be employed in Step 6 of Main-algorithm 1. The following theorem shows that Thm.3.2 still remains valid when Main-algorithm 1 and Sub-algorithm 1.1 are carried out using the above constraint construction scheme.

Theorem 3.8:

For problem P1, suppose Ψ is not empty, f satisfies Assumption 3.1 and functional $\tilde{f}$ defined in Def.3.19 is continuous. Let sequence $\{l_i\}$ be generated by Main-algorithm 1, Sub-algorithm 1.1 together with the above constraint construction scheme. Then, if the sequence is finite, the last point lies in Ψ; if the sequence is infinite, then any accumulation point is in Ψ.

Proof:

The finite case is trivial.

In the infinite case, for sake of simplicity, suppose that

$$l_i \longrightarrow l^*, \text{ as } i \longrightarrow \infty.$$

(1) If there is a set $J \subset \{0,1,\ldots\}$ of infinite cardinality such that

$$\tilde{f}(l_{i_j}) < K\varepsilon_{i_j}, \quad \text{for all } j \text{ in } J,$$

then since $\varepsilon_{i_j} \longrightarrow 0$ as $j \xrightarrow{J} \infty$ and $\tilde{f}$ is continuous,

$$\tilde{f}(l^*) = \lim_j \tilde{f}(l_{i_j}) \leqslant 0.$$

Hence $l^* \varepsilon \Psi$.

(2) If $\tilde{f}(l_i) < K\varepsilon_i$, for only a finite subset I of $\{0, 1, \ldots\}$,

then, there exists an $N_0 > 0$ such that after N_0 iterations

$$f(x^j, l_i(x^j), \nabla_x l_i(x^j)) \leqslant 0, \quad \text{for all } N_0 \leqslant j < i,$$

and

$$l_i \longrightarrow l^*, \quad \text{for } i \geqslant N_0 \text{ and } i \longrightarrow \infty.$$

Hence, $\tilde{f}(l^*) \leqslant 0$ can be established as in the proof of Theorem 3.2.

Q.E.D.

In case that the maximization step in Main-algorithm 1 is obtained approximately (i.e., algorithms like Sub-algorithm 1.2 are used), it is easy to prove that the constraint construction scheme still can be employed provided

$$|\tilde{f}_i(l_i)-\tilde{f}(l_i)| \longrightarrow 0,$$
$$\text{as } i \longrightarrow \infty,$$

where, the approximation maximum value at i^{th} iteration with function l_i is denoted as $\tilde{f}_i(l_i)$.

It should be noted that with this constraint dropping scheme in operation, the set of simplices $\tilde{s}(\omega^i)$ no longer necessarily contains $\tilde{s}(\omega^j)$, for $j < i$. Hence relative steps in Main-algorithm 1 will have small changes. This might cause poor performance inconvenience in computation.

This kind of constraint construction scheme can be used for problem P2 in algorithms proposed in Set.3.4 or the algorithm in Sec.3.7. The situations are more or less the same since the scheme retains the feasibility of solutions.

3.9 Reduction of variables

In order to improve the efficiency of algorithms proposed in above sections, it is always beneficial to employ all available a priori knowledge to reduce the computation variables. For instance, in the design of non-linear control systems, if the feedback control law can be parameterized by a finite-dimensional design vector, then the computation can be considerably reduced. Suppose the feedback control law is parameterized by a vector θ, i.e., $u=r(x;\theta)$, $\theta\in\Theta$. The problem now is to choose a θ^* in Θ such that there exists a Lyapunov function V satisfying (cf. Sec.2.1)

$$V_x(f(x)+g(x)r(x;\theta^*)) < 0, \text{ for all } x\in\Omega,\ x\neq 0$$

$$V_x(f(0)+g(0)r(0;\theta^*)) = 0 .$$

Hence, at every iteration of procedure, a positive piecewise linear continuous function l_i of x, $l_i: R^n \longrightarrow R$ such that

$$\min_{\theta\in\Theta} \max_{x\in\omega^i} (\nabla_x l_i(x)(f(x)+g(x)r(x;\theta))) \leqslant 0,$$

where, ω^i is the nodal set at iteration i, must be found.

Suppose n_i is the number of points in ω^i, then the number of variables in i^{th} iteration is reduced from $2n_i$ to n_i+p, where p is the dimension of vector θ. Efficiency can be increased considerably.

CHAPTER FOUR AN EXAMPLE

4.1 Control system equations and design objective

In this chapter, we will consider a design example for a 2-dimensional single-input and single-output control system.

We are given a non-linear control system Σ_1 as follows:

$$\Sigma_1: \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} \sin x_2 \\ 0 \end{bmatrix} + \begin{bmatrix} \cos x_1 \\ 1 \end{bmatrix} u, \tag{4.1}$$

where, u is the control.

The design objective here is to find a feedback control law $u = \gamma(x_1, x_2)$ such that the closed-loop system Σ_2 below is stable at the origin in sense of Lyapunov over region Ω, where

$$\Sigma_2: \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} \sin x_2 \\ 0 \end{bmatrix} + \begin{bmatrix} \gamma(x_1, x_2)\cos x_1 \\ \gamma(x_1, x_2) \end{bmatrix}, \tag{4.2}$$

and

$$\Omega = \begin{bmatrix} -\pi/2 < x_1 < \pi/2 \\ -\pi/4 \leq x_2 \leq \pi/4 \end{bmatrix} \subset R^2 .$$

First, let us check the properties of system Σ_1. Following the notations used in ChapI and ChapII, the non-linear system equation is

$$\dot{x} = f(x) + g(x)u \quad .$$

The vector fields f and g are, respectively,

$$f(x) = \begin{bmatrix} \sin x_2 \\ 0 \end{bmatrix} , \qquad (4.3)$$

$$g(x) = \begin{bmatrix} \cos x_1 \\ 1 \end{bmatrix} . \qquad (4.4)$$

Thus,

$$[f,g] = (\partial g/\partial x)f - (\partial f/\partial x)g$$

$$= \begin{bmatrix} -\sin x_1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \sin x_2 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 & \cos x_2 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \cos x_1 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} -\sin x_1 \sin x_2 - \cos x_2 \\ 0 \end{bmatrix} .$$

The controllability matrix is

$$\left[g, [f,g]\right] = \begin{bmatrix} \cos x_1 & -\sin x_1 \sin x_2 - \cos x_2 \\ 1 & 0 \end{bmatrix} ,$$

which is non-singular in Ω. Also, $\{g\}$ is, trivially, involutive.

Hence, from Thm.1.1 in Sec.1.3, Σ_1 is equivalent to a linear controllable system in integrator form. Thus, there do exist feedback control laws such that the system Σ_2 is stable at the origin in sense of Lyapunov over Ω. This establishes the existence of the function γ and V such that the closed loop system is stable and V is a Lyapunov function.

In the light of the design philosophy demonstrated in Sec.2.1, two scalar functions, γ and V, defined on Ω, will be constructed directly. Satisfaction of the following constraints on γ and V ensures the closed loop system stable.

$$
\begin{cases}
V(x_1, x_2) > 0 , & (4.5) \\
V_x \cdot \begin{bmatrix} \sin x_2 + \gamma(x_1, x_2)\cos x_1 \\ \gamma(x_1, x_2) \end{bmatrix} < 0 , & (4.6) \\
\quad \text{for all } (x_1, x_2) \in \Omega, \quad (x_1, x_2) \neq 0 & \\
V(0,0) = 0 , & \\
\gamma(0,0) = 0 . &
\end{cases}
$$

The controller should be designed to satisfy both stability and performance constraints. In the interest of simplicity we ignore performance constraints.

4.2 Computation procedure

We propose to use a computation method to obtain the functions V and γ defined on Ω. We start with a larger region $\tilde{\Omega}$ defined by:

$$\tilde{\Omega} = \left\{ \begin{matrix} -\pi/2 \leqslant x_1 \leqslant \pi/2 \\ -\pi \leqslant x_2 \leqslant \pi \end{matrix} \right\} \subset R^2,$$

and satisfying

$$\Omega \subset \tilde{\Omega}.$$

In fact, the region $\tilde{\Omega}$ should be as large as possible, provided that the existence of V and γ on $\tilde{\Omega}$ is guaranteed; a larger region facilitates the determination of a level contour of Lyapunov function V which encloses Ω.

Following the approach proposed in Chap.III, sets of simplices and piecewise linear, continuous functions over these simplices will be constructed on $\tilde{\Omega}$.

Suppose the vertices of a simplex are p^1, p^2 and p^3 and that the origin does not lie in this simplex. The coordinates of vertices are, respectively, (p_{11}, p_{12}), (p_{21}, p_{22}), and (p_{31}, p_{32}). Also, the values of the function V on these vertices are y^i, $i = 1,2,3$, correspondingly. Then, from the formula,

$$V(x_1, x_2) = a_1 + b_1^T \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

where, a_1 is a scalar, b_1 is a 2-dim vector, $b_1 = (b_{11}, b_{12})^T$, it follows that:

$$y^i = a_1 + b_1^T \begin{bmatrix} p_{i1} \\ p_{i2} \end{bmatrix}, \qquad i = 1,2,3,$$

$$y^2 - y^1 = b_1^T(p^2-p^1),$$

$$y^3 - y^1 = b_1^T(p^3-p^1),$$

where,

$$p^2 - p^1 = \begin{bmatrix} p_{21} \\ p_{22} \end{bmatrix} - \begin{bmatrix} p_{11} \\ p_{12} \end{bmatrix} = \begin{bmatrix} p_{21}-p_{11} \\ p_{22}-p_{12} \end{bmatrix},$$

$$p^3 - p^1 = \begin{bmatrix} p_{31} \\ p_{32} \end{bmatrix} - \begin{bmatrix} p_{11} \\ p_{12} \end{bmatrix} = \begin{bmatrix} p_{31}-p_{11} \\ p_{32}-p_{12} \end{bmatrix}.$$

So,

$$b_1^T = [y^2-y^1, y^3-y^1][p^2-p^1, p^3-p^1]^{-1}.$$

b_1 is the slope of function V on this simplex.

Since any point $p = (x_1, x_2)^T$ in the simplex can be expressed as

$$p = p^1 + \alpha_1(p^2-p^1) + \alpha_2(p^3-p^1)$$

$$= (1-\alpha_1-\alpha_2)p^1 + \alpha_1 p^2 + \alpha_2 p^3,$$

where,

$$0 \leqslant \alpha_1, \alpha_2 \leqslant 1,$$

$$\alpha_1 + \alpha_2 \leqslant 1,$$

the value of V at p is

$$V(x_1, x_2) = (1-\alpha_1-\alpha_2)y^1 + \alpha_1 y^2 + \alpha_2 y^3.$$

It is obvious that $V(x_1, x_2)$ will remain positive if y^1, y^2, y^3 are positive.

Similarly, suppose the values of function γ at p^1, p^2, p^3 are z^1, z^2, z^3, respectively. γ satisfies

$$\gamma(x_1, x_2) = a_2 + b_2^T \begin{bmatrix} x_1 \\ x_2 \end{bmatrix},$$

where,

$$b_2 = [b_{21}, b_{22}]^T ,$$

on this simplex. Thus,

$$b_2^T = [z^2-z^1, z^3-z^1][p^2-p^1, p^3-p^1]^{-1},$$

and the value of γ at point $p=[x_1, x_2]^T$ in this simplex is

$$\gamma(x_1, x_2) = (1-\alpha_1-\alpha_2)z^1 + \alpha_1 z^2 + \alpha_2 z^3,$$

if

$$p = p^1 + \alpha_1(p^2-p^1) + \alpha_2(p^3-p^1) ,$$

where, α_1 and α_2 are restricted as above.

Thus, the constraints (4.5) and (4.6) for p^1, p^2, p^3 with regard to this simplex are:

$$\begin{cases} y^1 > 0 \\ y^2 > 0 \\ y^3 > 0 \\ b_{11}(\sin p_{12} + z^1\cos p_{11}) + b_{12}z^1 < 0 \\ b_{11}(\sin p_{22} + z^2\cos p_{21}) + b_{12}z^2 < 0 \\ b_{11}(\sin p_{32} + z^3\cos p_{31}) + b_{12}z^3 < 0 , \end{cases}$$

where,

$$b_{11} = q_{11}(y^2-y^1) + q_{21}(y^3-y^1)$$

$$b_{12} = q_{12}(y^2-y^1) + q_{22}(y^3-y^1),$$

$$\begin{bmatrix} q_{11} & q_{12} \\ q_{21} & q_{22} \end{bmatrix} = \begin{bmatrix} p_{21}-p_{11} & p_{31}-p_{11} \\ p_{22}-p_{12} & p_{32}-p_{12} \end{bmatrix}^{-1} .$$

For an arbitrary point $p=[x_1, x_2]^T$ in this simplex, when

$$p = p^1 + \alpha_1(p^2-p^1) + \alpha_2(p^3-p^1),$$

the following inequality should be satisfied,

$$b_{11}(\sin x_2 + y\cos x_1) + b_{12}z < 0 ,$$

where,

$$x_1 = (1-\alpha_1-\alpha_2)p_{11}+\alpha_1 p_{21}+\alpha_2 p_{31}$$

$$x_2 = (1-\alpha_1-\alpha_2)p_{12}+\alpha_1 p_{22}+\alpha_2 p_{32},$$

$$y = (1-\alpha_1-\alpha_2)y^1+\alpha_1 y^2+\alpha_2 y^3,$$

$$z = (1-\alpha_1-\alpha_2)z^1+\alpha_1 z^2+\alpha_2 z^3.$$

Varying (α_1,α_2) in the domain $\{(\alpha_1,\alpha_2) \mid 0\leqslant\alpha_1, 0\leqslant\alpha_2, \alpha_1+\alpha_2 \leqslant 1\}$ gives different points in the simplex. In the present example, the following rule is set for (α_1,α_2), (in Step 4 of Main-algorithm 1),

$$\alpha_1 = i/10, \quad i = 0,\ldots, 10;$$

$$\alpha_2 = j/10, \quad j = 0,\ldots, 10-i .$$

In a net of simplices a nodal point should satisfy the constraints involved with all adjacent simplices.

The system Σ_1 is linearized at the origin yielding the linearized system Σ_L:

$$\Sigma_L: \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u$$

or, in vector form,

$$\Sigma_L: \quad \dot{x} = \tilde{A}x + bu,$$

where,

$$x = [x_1, x_2]^T,$$

$$\tilde{A} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix},$$

$$b = [1,\ 1]^T .$$

The parameterization of the feedback control law is chosen to be:

$$u = k^T x = [k_1, k_2] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}.$$

Thus,

$$\dot{x} = \tilde{A}x + bk^T x$$

$$= \begin{bmatrix} k_1 & k_2+1 \\ k_1 & k_2 \end{bmatrix} x$$

$$\triangleq Ax .$$

The selection of $k_1 = -5$, $k_2 = 1$ makes A a stable matrix, since then

$$A = \begin{bmatrix} -5 & 2 \\ -5 & 1 \end{bmatrix}$$

and has eigenvalues $-2+i$ and $-2-i$.

Let Q be a positive definite matrix, say,

$$Q = \begin{bmatrix} 40 & 0 \\ 0 & 40 \end{bmatrix}$$

Solving the following matrix Lyapunov equation

$$A^T P + PA = -Q$$

yields the positive definite matrix P,

$$P = \begin{bmatrix} 31 & -27 \\ -27 & 34 \end{bmatrix}$$

Thus, a suitable Lyapunov function V_L for system Σ_L might be chosen as

$$V_L = (1/2)x^T Px ,$$

since

$$\dot{V}_L = (1/2)x^T(A^T P+PA)x$$

$$= -(1/2)x^T Qx$$

$$= -20(x_1^2 + x_2^2)$$

< 0 , for all $x \neq 0$.

Then, an attractive region around the origin, with regard to this Lyapunov function V_L and the feedback control law $u = -5x_1 + x_2$, is obtained via computation. Meanwhile, the values of V_L and this $u(x_1, x_2)$ serve as initial values in $\tilde{\Omega}$ and the computations which were described in the last chapter are then carried out. The results are given in the next section.

As the computation proceeds, the number of variables and the number of constraints increases. In this example the region $\bar{\tilde{\Omega}}$ is decomposed into several regions in order to make better use of the computer facility available and computation is carried out separately in each region. Of course, the values of variables along shared boundaries should agree.

4.3 Computation results and simulations

The simplices constructed are in Figure 3. The corresponding data are as follows.

	Coordinates	Lyapunov function	Controller function
(In the first quadrant)			
P1	(0, 0)	0.	0.
P2	(1.571, 0)	3.052e+1	2.372e-4
P3	(1.571, .5447)	3.550964e+1	-1.149135e+1
P4	(1.571, .8978)	4.200947e+1	-1.554873e+1
P5	(1.571, 1.270)	4.720882e+1	-1.809163e+1
P6	(1.571, 1.571)	5.734558e+1	-1.235701
P7	(1.571, 3.142)	8.824875e+1	-6.184629e+1
P8	(0, 3.142)	6.106750e+1	-1.236891e+2
P9	(0, 1.571)	3.053375e+1	-6.184455e+1
P10	(.7855, .7855)	2.961961e+1	-0.9503145
P11	(1.449, 1.384)	4.436309e+1	-1.829056e+1
P12	(1.275, 1.270)	4.632621e+1	-1.130028
(In the second quadrant)			
P13	(-1.571, 3.142)	8.741590e+1	-2.427157e+2
P14	(-1.571, 1.571)	5.669063e+1	-1.810794e+2
P15	(-1.571, .3886)	1.719193e+1	-6.096947
P16	(-1.571, .0849)	0.7737252	-8.242046e+3
P17	(-1.571, 0)	4.574e-2	-4.846e+3
P18	(-1.028, .1143)	9.928853e-1	-7.776366e+3

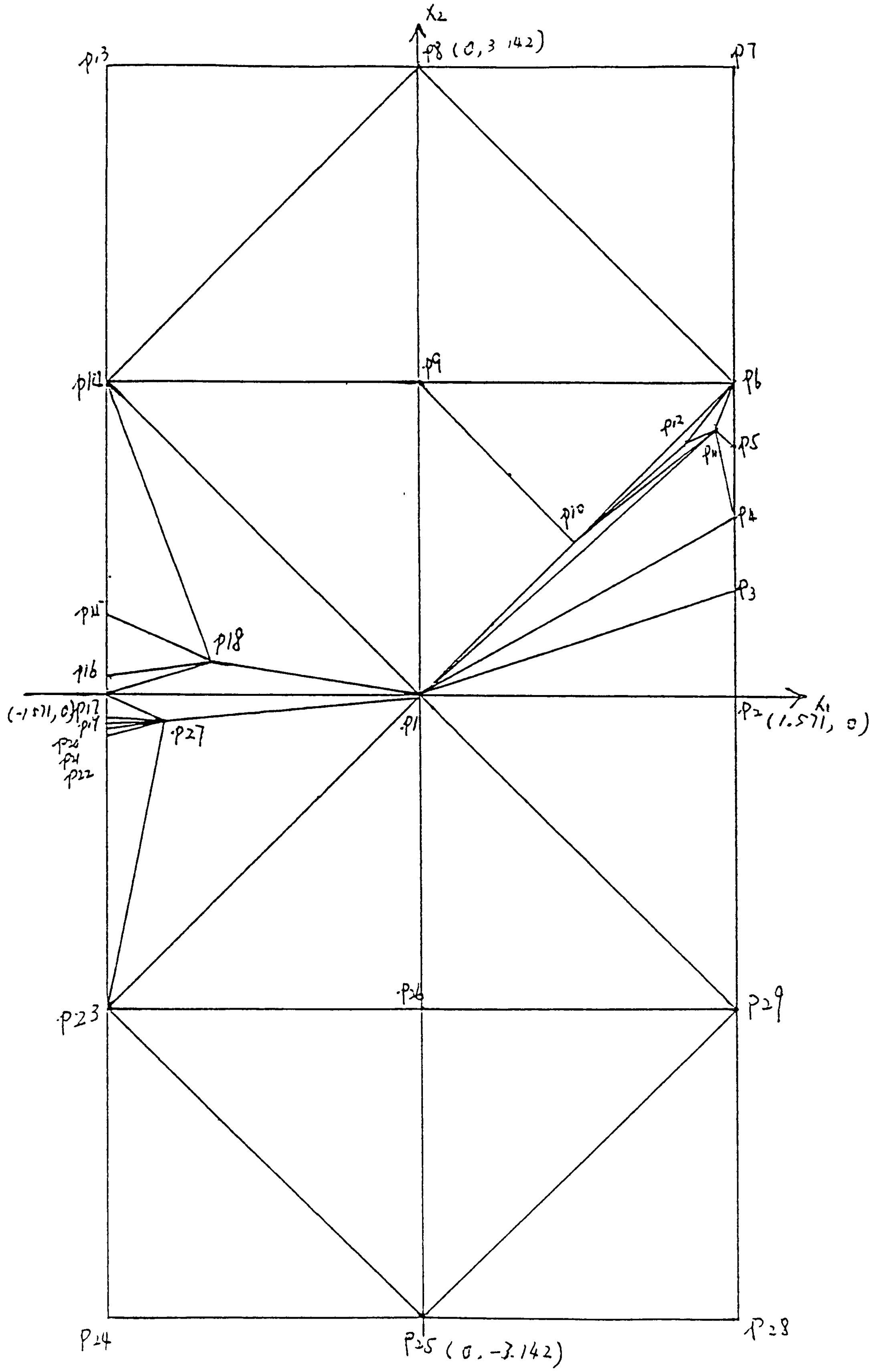
(In the third quadrant)

P19	(-1.571, -.1020)	9.150498e-3	-1.016303e+3
P20	(-1.571, -.1029)	9.70411e-3	-1.08560e+3
P21	(-1.571, -.1088)	9.61703e-3	5.73200
P22	(-1.571, -.1617)	0.349339e-1	8.587715e+2
P23	(-1.571, -1.571)	7.291926e-2	9.366053e+2
P24	(-1.571, -3.142)	1.766527e+5	8.934137e+2
P25	(0, -3.142)	5.227220e+2	2.207211
P26	(0, -1.571)	2.613610e+2	1.1036055
P27	(-1.309, -.1190)	2.87446e-2	-1.10536e-2

(In the fourth quadrant)

P28	(1.571, -3.142)	5.153777e+2	4.594650e+1
P29	(1.571, -1.571)	2.545794e+2	4.465063e+1

Some simulations are shown in Figure 4. Figure 5 is the overlapped diagram.



X2
p8 (0, 3.142)
p13
p7
p12
p9
p6
p12
p5
p4
p10
p3
p15
p18
p16
(-1.571, 0) p17
p19
p20
p21
p22
p27
p1
p2 (1.571, 0)
X1
p23
p26
p29
p24
p25 (0, -3.142)
p28

Figure 3

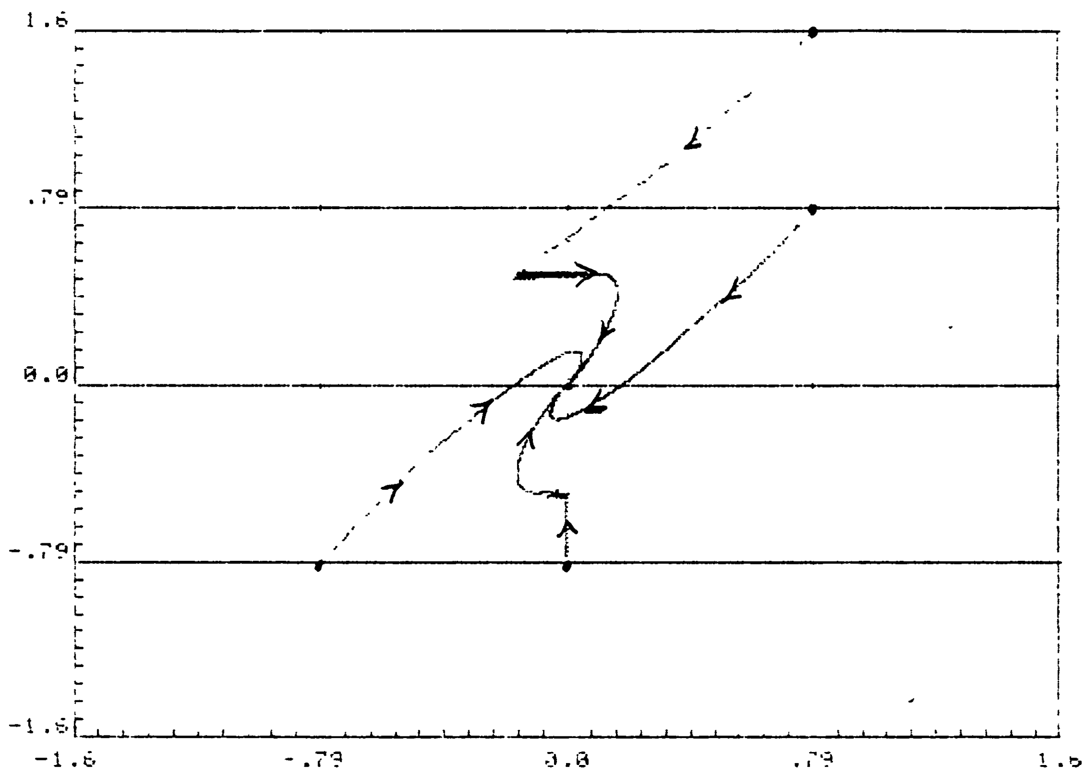


Figure 4

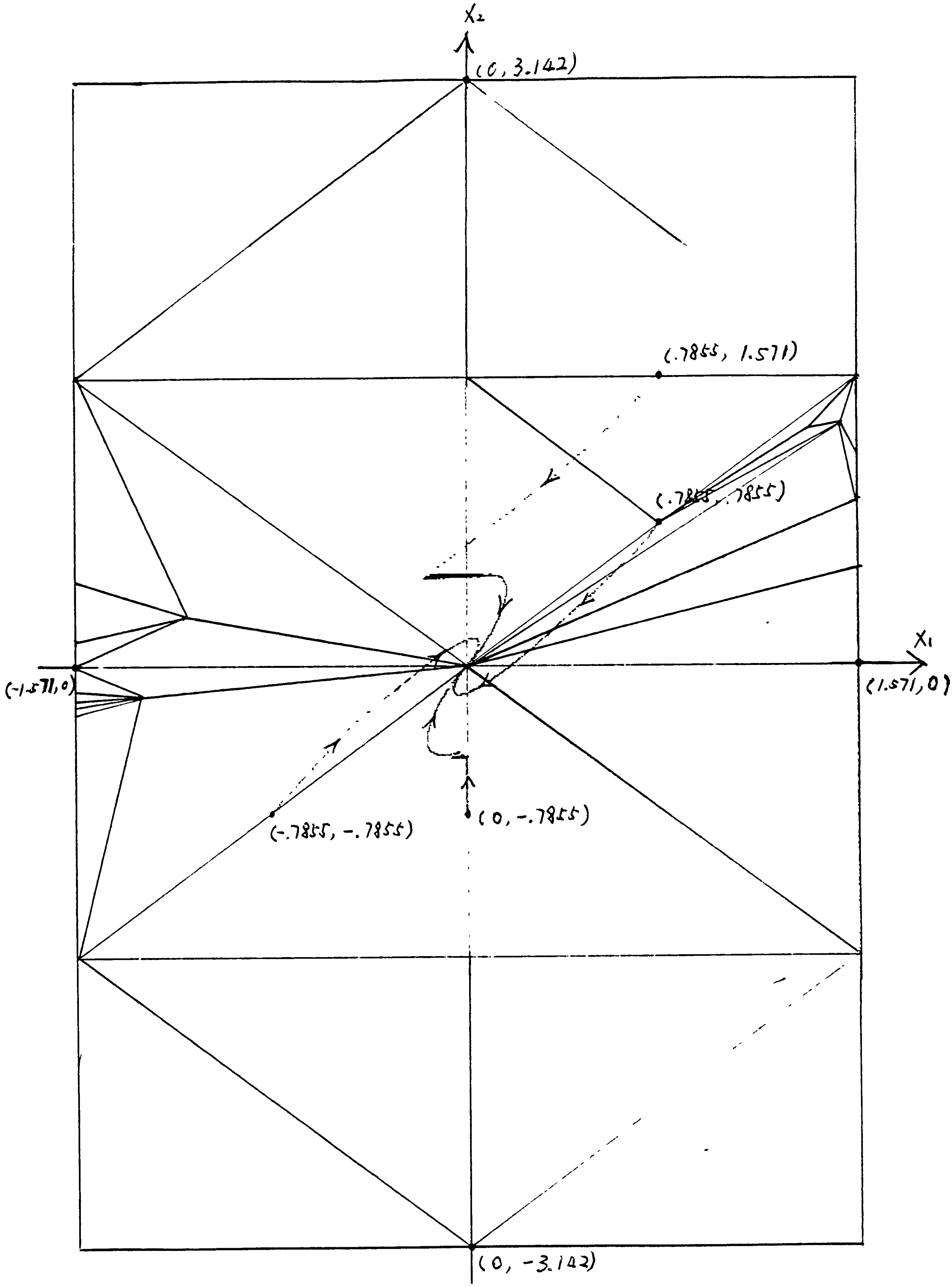


Figure 5: The simulation results.

CHAPTER FIVE MATHEMATICAL PROGRAMMING PROBLEM WITH INFINITE FUNCTION CONSTRAINTS IN THE TIME DOMAIN

5.1 The introduction

In this chapter we consider a different mathematical programming problem. In this problem the design variables are finite in number, but the constraints must be satisfied for every function in a specified class and every time in a specified interval. Only the feasibility problem will be discussed. The abstract problem form is as follows.

$\tilde{P}3$:

Given a subset Θ of R^r, find a $\theta \in \Theta$ such that the following inequality holds:

$$f(t,u(t);\theta) \leqslant 0,$$

$$\text{for all } u \in U, \text{ all } t \in T,$$

where, U is a set of functions,

$T \triangleq [0,t_1]$ is a time interval,

f is a function of t, u and θ.

We will consider a problem of control system design which can be formulated as $\tilde{P}3$. In the next section the design problem will be introduced and a prototype algorithm will be proposed. This algorithm reveals the basic idea of our approach. The analysis is relatively easy because the prototype algorithm is simple. The design problem will be specified in more detailed in Sec.5.3. A detailed algorithm will be described and analysed. Sec.5.4 will introduce two possible approaches towards the global maximization problem in the main-algorithm proposed in Sec.5.3. In Sec.5.5 a constraint dropping scheme will be discussed.

For the sake of convenience only the single input/single output case will be discussed in this chapter.

5.2 A prototype algorithm

In this section we will propose a prototype algorithm for a control system design problem.

The continuous system $\Sigma(\theta)$, $\theta \in \Theta \subset R^r$, is defined by

$$\Sigma(\theta) \qquad \begin{cases} \dot{x}(t) = f(x(t), u(t); \theta) \\ y(t) = h(x(t)), \qquad t \in T . \end{cases}$$

There are two classes of functions: C_1 and C_2. C_1 is the class of admissible input functions and C_2 is the class of desired output functions.

The control system design problem requires the determination of a θ in the design parameter set Θ such that for any function u in C_1 the corresponding output function y always lies in C_2.

Suppose that when $\theta \in \Theta$ and an input function $u \in C_1$ is given we can solve $\Sigma(\theta)$ and obtain the output function y. Thus we can write

$$y(t) = y(t; u, \theta),$$

or, simply,

$$y = y(u, \theta),$$

if no ambiguity arises.

The problem is now in the form:

P3: Find a $\theta \in \Theta$, such that

$$y(u, \theta) \in C_2, \qquad \text{for all } u \in C_1 .$$

Furthermore, some real-valued functional $\xi(y)$ of y can usually be defined to decide whether y lies in C_2 or not. For instance, when C_2 is given by

$C_2 \triangleq \{ y$ is continuous $| \; y_1(t) \leq y(t) \leq y_2(t)$, a.e. $t \in T\}$,

we can define a non-negative ξ by

$$\xi(y) \triangleq \int_0^{t_1} \max\{0, y(t)-y_2(t), y_1(t)-y(t)\}dt .$$

Thus, $\xi(y) = 0$ denotes $y \in C_2$; otherwise, $y \notin C_2$.

Also it is clear that we can define $\hat{\eta}$: $C_1 \times \Theta \longrightarrow R$ by

$$\hat{\eta}(u,\theta) \triangleq \xi(y(u,\theta)).$$

Hence, letting C denote C_1, the problem becomes

PP:

Find a $\theta \in \Theta$, such that

$$\hat{\eta}(u,\theta) = 0, \quad \text{for all } u \in C.$$

For any subset $C^{(i)}$ of C we define the following problem:

$P_{C^{(i)}}$: Find a $\theta \in \Theta$, such that

$$\hat{\eta}(u,\theta) = 0, \quad \text{for all } u \in C^{(i)}.$$

For a given $\theta \in \Theta$, we can define the following maximization problems:

M: Given a $\theta \in \Theta$, find the "worst" u^* in C,

in the sense that

$$\hat{\eta}(u^*,\theta) = \sup\{\hat{\eta}(u,\theta) \mid u \in C\}.$$

Similarly, if $D^{(i)}$ is a set of functions (not necessarily a subset of C), we define:

$M_{D^{(i)}}$: Given a $\theta \in \Theta$, find the "worst" u^* in $D^{(i)}$, i.e.,

$$\hat{\eta}(u^*, \theta) = \sup\{\hat{\eta}(u, \theta) \mid u \in D^{(i)}\}.$$

For the sake of convenience, we define

η_C, $\eta_{D^{(i)}}$: $\Theta \longrightarrow R$ by

$$\eta_C(\theta) \triangleq \sup\{\hat{\eta}(u, \theta) \mid u \in C\},$$

$$\eta_{D^{(i)}}(\theta) \triangleq \sup\{\hat{\eta}(u, \theta) \mid u \in D^{(i)}\}.$$

$M_{D^{(i)}}$ and $\eta_{D^{(i)}}$ will be useful in the next section.

Now we are able to describe the following prototype algorithm, which is of the outer approximation type, for the control system design problem.

Prototype-algorithm:

Data: Choose a subset $C^{(0)} \subset C$

Step 0: Set $i = 0$

Step 1: Solve $P_{C^{(i)}}$ to obtain θ_i

Step 2: For θ_i, solve M to obtain u^*

Step 3: If $\eta_C(\theta_i) = 0$, stop;

Else, go to Step 4

Step 4: Set $C^{(i+1)} = C^{(i)} \cup \{u^*\}$,

$i = i+1$,

where, u^* is obtained from problem M in Step 2;

Go to Step 1.

To explore the properties of this prototype algorithm, the following assumptions are made first.

Assumption 5.1:

The design parameter set Θ is compact.

Assumption 5.2:

$\hat{\eta}$ is continuous in u and in θ, separately.

Assumption 5.3:

For u in C, there is a constant K such that for any θ_1, $\theta_2 \in \Theta$,

$$|\hat{\eta}(u,\theta_1)-\hat{\eta}(u,\theta_2)| \leqslant K\|\theta_1-\theta_2\|_r .$$

The first assumption is natural and can be obtained from physical systems. Assumption 5.2 is quite reasonable since the function f in system $\Sigma(\theta)$ is always continuous differentiable and usually the output depends continuously on input and design parameters. Assumption 5.3 is a Lipschitz condition for $\hat{\eta}$ in θ. It can be obtained by the output of the control system usually being differentiable in the design parameters.

Now we define the feasible set F of problem PP as.

$$F \triangleq \{\theta \in \Theta \mid \eta_C(\theta) = 0\}.$$

Lemma 5.1: F is closed.

Proof:

Suppose $\theta_i \longrightarrow \theta^*$, as $i \longrightarrow \infty$, and

$$\theta_i \in F, \quad \text{for all } i.$$

For any $u \in C$, since $\hat{\eta}(u,\theta_i) = 0$ and $\hat{\eta}(.,.)$ is continuous in the second variable, we have

$$\hat{\eta}(u,\theta^*)=0.$$

Thus $\eta_C(\theta^*) = 0$,

i.e.,

$$\theta^* \in F.$$

Q.E.D.

Lemma 5.2:

Suppose the infinite sequence $\{\theta_i\}$ satisfies $\theta_i \longrightarrow \theta^*$, and $\eta_C(\theta_i) \longrightarrow 0$, as $i \longrightarrow \infty$.

Then, $\theta^* \in F$.

Proof:

From the hypothesis in the lemma, for all u in C, there exists a sequence $\{\varepsilon_i\}$ satisfying, for all $i \geqslant 0$,

$$\varepsilon_i \geqslant 0 \quad \text{and}$$

$$\varepsilon_i \longrightarrow 0, \quad \text{as } i \longrightarrow \infty,$$

such that

$$\hat{\eta}(u,\theta_i) \leqslant \varepsilon_i .$$

Since $\hat{\eta}$ is continuous in θ, allowing i to tend to infinity in the above inequality yields

$$\hat{\eta}(u,\theta^*) = 0,$$

since the right side of the inequality tends to zero and the left side is non-negative.

Thus $\eta_C(\theta^*) = 0$,

i.e.,

$$\theta^* \in F.$$

Q.E.D.

Theorem 5.1:

Suppose F is not empty.

If the prototype algorithm generates a finite sequence in Θ, then the last point lies in F.

If the algorithm generates an infinite sequence, then any accumulation point of the sequence will be in F.

Proof:

The finite case is trivial.

In the infinite case, suppose $\theta_{n_i} \longrightarrow \theta^*$, as $i \longrightarrow \infty$.

For any i, let u_i^* be such

$$\hat{\eta}(u_i^*, \theta_{n_i}) = \eta_C(\theta_{n_i}) .$$

For any $j > i$, we have a constant K from Assumption 5.3 such that

$$|\hat{\eta}(u_i^*, \theta_{n_i}) - \hat{\eta}(u_i^*, \theta_{n_j})| \leqslant K\|\theta_{n_i} - \theta_{n_j}\|_r .$$

By construction:

$$u_i^* \in C^{(n_j)}, \quad \text{if } i < j.$$

Thus,

$$\hat{\eta}(u_i^*, \theta_{n_j}) = 0$$

and, for all i, j such that $i < j$,

$$K\|\theta_{n_i}-\theta_{n_j}\|_r \geqslant \hat{\eta}(u_i^*,\theta_{n_i}) = \eta_C(\theta_{n_i}).$$

Since $\theta_{n_i} \longrightarrow \theta^*$, as $i \longrightarrow \infty$, the left side of the above inequality tends to zero.

Hence, $\eta_C(\theta_{n_i}) \longrightarrow 0$, as $i \longrightarrow \infty$.

From Lemma 5.2, we have

$$\theta^* \in F .$$

Since Θ is compact from Assumption 5.1, there at least exists one accumulation point. This completes the proof.

Q.E.D.

5.3 The main-algorithm

In this section we consider the control system design problem in detail. First we specify the input and output function classes as hard constraints: the input function class C_1 and the output function class C_2 are, respectively, defined by:

$$C_1 \triangleq \{ u \in C^0(T) \mid u_1(t) \leq u(t) \leq u_2(t), \text{ a.e. } t \in T\}$$

and

$$C_2 \triangleq \{ y \in C^0(T) \mid y_1(t) \leq y(t) \leq y_2(t), \text{ a.e. } t \in T\},$$

where, $C^0(T)$ is the space of continuous scalar functions defined on T, and u_1, u_2, y_1 and y_2 are continuous functions defined on the time interval T and satisfying

$$u_1(t) < u_2(t) \ , \qquad \text{a.e.} \quad t \in T,$$

$$y_1(t) < y_2(t) \ , \qquad \text{a.e.} \quad t \in T.$$

The basic idea of our approach is to use a discretized time interval instead of the continuous time interval and to use piecewise linear continuous functions instead of continuous functions. We introduce the following definitions first.

Definition 5.1:

A discrete subset τ^1 of $T \triangleq [0,t_1]$ is defined by

$$\tau^1 \triangleq \{\tau^1_1, \ldots, \tau^1_{n_1}\},$$

where,

$$0 = \tau^1_1 < \ldots < \tau^1_{n_1} = t_1 \ .$$

Definition 5.2:

For any subset $\tau^i = \{\tau_1^i, \ldots, \tau_{n_i}^i\}$ of T and $\omega^i = \{\omega_1^i, \ldots, \omega_{n_i}^i\} \in R \times \ldots \times R$, ($n_i$ times), u_{τ^i,ω^i} denotes a piecewise linear continuous input function satisfying

$$u_{\tau^i,\omega^i}(\tau_k^i) = \omega_k^i \quad , k = 1, \ldots, n_i.$$

Definition 5.3:

For the continuous system $\Sigma(\theta)$, the discrete subset τ^i of T and vector ω^i, the discrete output y_{τ^i,ω^i} is defined by

$$y_{\tau^i,\omega^i}(\tau_k^i) = y(\tau_k^i), \qquad k = 1, \ldots, n_i,$$

where, y is the output of system Σ with the input function u_{τ^i,ω^i}.

Def.5.3 shows that we use a piecewise linear continuous function as the input function in the continuous system $\Sigma(\theta)$ and consider the output only at the discrete time points.

Definition 5.4:

For an output y_{τ^i,ω^i} defined above, we define a measurement $\varphi(y_{\tau^i,\omega^i})$ of y_{τ^i,ω^i}, which measures the deviation of y_{τ^i,ω^i} from the function class C_2, as follows

$$\varphi(y_{\tau^i,\omega^i}) \triangleq$$

$$\sum_{k=1}^{n_i-1} \max\{0,\ y_{\tau^i,\omega^i}(\tau_k^i) - y_2(\tau_k^i),$$

$$y_1(\tau^i_k)-y_{\tau^i,\omega^i}(\tau^i_k)\}(\tau^i_{k+1}-\tau^i_k).$$

Definition 5.5:

For a discrete subset τ^i of T, the largest length of the sub-intervals of τ^i is defined by

$$\Delta_i \triangleq \max\{\tau^i_{k+1} - \tau^i_k \mid k = 1, \ldots, n_i-1\}.$$

Further, we define an adding point σ_i of τ^i is the mid-point of the sub-interval corresponding to Δ_i (If there are more than one such intervals, choose any one.)

Definition 5.6:

For a discrete subset τ of T and a point δ in T, suppose

$$\tau = \{\tau_1, \ldots, \tau_n\},$$

$$0 \leqslant \tau_1 < \tau_2 < \ldots < \tau_n \leqslant t_1,$$

we define an addition operation $\blacktriangle$ of τ and $\{\delta\}$ by joining them together in order, i.e.

$$\tau \blacktriangle \{\delta\} = \{\tau_1, \ldots, \tau_k, \delta, \tau_{k+1}, \ldots, \tau_n\},$$

$$\text{if } \tau_k < \delta < \tau_{k+1},\ 1 \leqslant k \leqslant n-1;$$

$$= \tau, \text{ if } \delta \text{ coincides with one point in } \tau.$$

Definition 5.7:

For a discrete subset τ^i of T, we define an input range set $\underline{\Omega}^i$ by

$$\underline{\Omega}^i \triangleq \Omega^1 \times \ldots \times \Omega^{n_i},$$

where, $\Omega^k = [u_1(\tau^i_k), u_2(\tau^i_k)]$,

$$k \in \{1, \ldots, n_i\}.$$

We now have all the ingredients for the main-algorithm low. Step 1 and Step 2 in the main-algorithm construct a piecewise linear continuous input function lying in C_1. In Step 3, a design parameter θ is found to satisfy the input function set up to present. The newly "worst" input function's value at the present discrete time subset is determined in Step 4. Stopping rule includes not only satisfactory "worst" case but also the fineness of the discretized time interval, which is the number ε in the following algorithm.

Main-algorithm 4:

Data: The time interval $T \triangleq [0, t_1]$;

Continuous functions u_1, u_2, y_1 and y_2;

Compact design parameter set $\Theta \subset R^r$;

Positive number $\varepsilon < 1$;

Positive sequence $\{\beta_i\}$, $\beta_i \longrightarrow 0$, as $i \longrightarrow \infty$.

Step 0: Choose a discrete subset τ^0 of T;

Construct an input range set $\underline{\Omega}^0$;

Construct $\omega^0 = (\omega^0_1, \ldots, \omega^0_{n_0}) \in \underline{\Omega}^0$

to satisfy

$$\omega^0_k = [u_1(\tau^0_k) + u_2(\tau^0_k)]/2, \quad \text{for all } k \in \{1, \ldots, n_0\};$$

Set $i = 0$.

Step 1: Form $u^i = u_{\tau^i, \omega^i}$;

Compute Δ_i.

Step 2: Check whether u^i lies in C_1 by computing

$$\psi(u^i) \triangleq \max_{t \in T} \{0,\ u^i(t)-u_2(t),\ u_1(t)-u^i(t)\};$$

If $\psi(u^i) = 0$, go to Step 3;

Else, compute

$$\delta \in \{\operatorname{argmax}_{t \in T} \{0,\ u^i(t)-u_2(t),\ u_1(t)-u^i(t)\}\};$$

Let $\tilde{\omega} = [u_1(\delta) + u_2(\delta)]/2$;

Set $\tau^i = \tau^i \blacktriangle \{\delta\}$;

Set $n_i = n_i + 1$;

Form ω^i by adding $\tilde{\omega}$ to ω^i (in the place that δ occupies in τ^i);

Go to Step 1.

Step 3: Find a vector $\theta_i \in \Theta$ such that for every u^j, $j \in \underline{i} \triangleq \{0, \ldots, i\}$, the corresponding y_{τ^i, ω^j} lies, approximately, in C_2, in the sense

that $\varphi(y_{\tau^i, \omega^j}) = 0$.

Step 4: Determine the "worst" vector $\hat{\omega}$ in $\underline{\Omega}^i$ to satisfy

$$\varphi(y_{\tau^i, \hat{\omega}}) = \max_{\omega \in \underline{\Omega}^i} \varphi(y_{\tau^i, \omega}).$$

Step 5: If $\varphi(y_{\tau^i, \hat{\omega}}) \leqslant \beta_i$, go to Step 6;

Else, set $\omega^{i+1} = \hat{\omega}$;

Set $n_{i+1} = n_i$;

Set $\tau^{i+1} = \tau^{i}$;

Set $i = i + 1$;

Go to Step 1.

Step 6: If $\Delta_i < \varepsilon$, stop;

Else, find an adding point σ of τ^i;

Set $n_{i+1} = n_i$;

Set $\tau^{i+1} = \tau^i \blacktriangle \{\sigma\}$;

Form ω^{i+1} by adding $u_{\tau^i,\hat{\omega}}(\sigma)$

to $\hat{\omega}$ in the corresponding place

with σ in τ^{i+1}, and set

$i = i + 1$;

Go to Step 1.

Step 3 in the above algorithm requires the determination of a finite-dimensional variable, (the design parameter θ), to satisfy a finite number of constraints. Hence, Step 3 is solvable. In Step 4, a finite dimensional global maximization problem involved. This problem will be discussed in the next section. The properties of Main-algorithm 4, however, are discussed here.

Since the class of piecewise linear continuous functions is dense in the class of continuous functions and the stopping rule in the algorithm includes the satisfactory fineness of the discrete time interval, we may define a "relaxed" feasible set $\tilde{F}$ as the set consisting of all points satisfying the stopping rule in Main-algorithm 4. We admit that $\tilde{F}$ is acceptable for problem P3 in Sec.5.2.

Let us compare Main-algorithm 4 with the Prototype-algorithm presented in the last section. The set C in the Prototype-algorithm is now, in the Main-algorithm 4, the class of piecewise linear continuous functions defined on T bounded by u_1 and u_2. The only essential difference between these two algorithms is that in Step 4 of Main-algorithm 4 we solve problem $M_{D^{(i)}}$ to obtain the "worst" point $\hat{\omega}$ instead of solving M in Step 2 of Prototype-algorithm to obtain the "worst" point u^*, where $D^{(i)}$ is a class of piecewise linear continuous functions and is not necessarily a subset of C. Hence, we state the following modified algorithm.

Modified prototype-algorithm:

Data: Choose a subset $C^{(0)} \subset C$.

Step 0: Set $i = 0$.

Step 1: Solve $P_{C^{(i)}}$ to obtain θ_i.

Step 2: For θ_i, solve $M_{D^{(i)}}$ to obtain $\hat{u}_i$, satisfying

$$\hat{\eta}(\hat{u}_i, \theta_i) = \max \{ \hat{\eta}(u, \theta_i) \mid u \in D^{(i)} \}.$$

Step 3: If $\eta_C(\theta_i) = 0$, i.e., $\theta_i \in F$, stop;

Else, go to Step 4.

Step 4: Construct $u_i^* \in C$ from $\hat{u}_i$ (as in Step 2 of Main-algorithm 4).

Step 5: Set $C^{(i+1)} = C^{(i)} \cup \{u_i^*\}$;

Set $i = i + 1$;

Go to Step 1.

Suppose C and $D^{(i)}$, $i = 0, 1, \ldots,$ are in a normed space. We have the following result.

Theorem 5.2:

Suppose F is not empty.

If the modified prototype-algorithm generates a finite sequence in Θ, then the last point of the sequence lies in F.

If the algorithm generates an infinite sequence such that

$$| \hat{\eta}(\hat{u}_i, \theta_i) - \eta_C(\theta_i) | \longrightarrow 0,$$

and $\| \hat{u}_i - u_i^* \| \longrightarrow 0$, as $i \longrightarrow \infty$,

then any accumulation point of $\{\theta_i\}$ will be in F.

First, we need an assumption similar to Assumption 5.3.

Assumption 5.4:

For θ in Θ, there exists a constant L such that

$$| \hat{\eta}(u_1, \theta) - \hat{\eta}(u_2, \theta) | \leq L \| u_1 - u_2 \|,$$

for any u_1, u_2 in the normed space.

Proof of Thm.5.2:

The conclusion for the finite case follows from the stopping rule.

In the infinite case, suppose $\theta_{n_i} \longrightarrow \theta^*$, as $i \longrightarrow \infty$.

Then there is an L such that

$$| \hat{\eta}(u_i^*,\theta_i)-\eta_C(\theta_i) | \leqslant |\hat{\eta}(u_i^*,\theta_i)-\hat{\eta}(\hat{u}_i,\theta_i)|+|\hat{\eta}(\hat{u}_i,\theta_i)-\eta_C(\theta_i)|$$

$$\leqslant L\|u_i^*-\hat{u}_i\|+|\hat{\eta}(\hat{u}_i,\theta_i)-\eta_C(\theta_i)|.$$

The right side of the inequality tends to zero, as $i \longrightarrow \infty$, because of the assumptions of the theorem. We have

$$|\hat{\eta}(u_i^*,\theta_i)-\eta_C(\theta_i)| \longrightarrow 0, \quad \text{as } i \longrightarrow \infty.$$

(*)

By Assumption 5.3 there exists a constant K such that

$$|\hat{\eta}(u_i^*,\theta_{n_i})-\hat{\eta}(u_i^*,\theta_{n_j})| \leqslant K\|\theta_{n_i}-\theta_{n_j}\|_r,$$

for all i,j, $i < j$.

By construction:

$$u_i^* \in C^{(n_j)}, \qquad \text{if } i < j.$$

Thus,

$$\hat{\eta}(u_i^*,\theta_{n_j}) = 0$$

and

$$K\|\theta_{n_i}-\theta_{n_j}\|_r \geqslant \hat{\eta}(u_i^*,\theta_{n_i}),$$

for all i,j, $i < j$.

Since $\theta_{n_i} \longrightarrow \theta^*$, as $i \longrightarrow \infty$, the left side of the above inequality tends to zero.

Hence, $\hat{\eta}(u_i^*,\theta_{n_i})$ tends to zero, as $i \longrightarrow \infty$,

since $\hat{\eta}(u_i^*,\theta_{n_i}) \geqslant 0$, for all i.

Furthermore, $\eta_C(\theta_{n_i}) \longrightarrow 0$, as $i \longrightarrow \infty$, because of (*).

From Lemma 5.2, we have

$$\theta^* \in F .$$

Since Θ is compact from Assumption 5.1, there at least exists one accumulation point. This completes the proof.

Q.E.D.

In the Prototype-algorithm and its modified version we employ the non-negative scalar function η_C to denote the "distance" of θ to the feasible set F. Execution stops when this value is zero. In the Main-algorithm 4 the stopping rule consists of a non-negative scalar function φ (defined in Step 4) and the fineness of the discrete time interval. The feasible set $\tilde{F}$ consists of all θ satisfying stopping rule. As a matter of fact, we can set $\{\beta_i\}$ in such a way that it goes to zero slowly before the fineness requirement on discrete time interval meets and very fast thereafter. Then Lemma 5.2 and Thm.5.2 still hold if we replace F and η_C by $\tilde{F}$ and φ, respectively. When the functions f and h in system $\Sigma(\theta)$ are smooth enough Assumptions 5.1 - 5.4 are satisfied as well as the hypothesis in Theorem 5.2. Hence, it is easy to see that the following theorem is a direct consequence of Thm.5.2. We conclude this section with Theorem 5.3.

Theorem 5.3:

Suppose F is not empty.

If the Main-algorithm 4 generates a finite sequence $\{\theta_i\}$

in Θ, then the last point lies in $\tilde{F}$.

If the algorithm generates an infinite sequence $\{\theta_i\}$, then any accumulation point of the sequence lies in F. Since Θ is assumed to be compact, the accumulation point always exists.

5.4 The global maximization in Main-algorithm 4

In Step 4 of Main-algorithm 4 (presented in the last section) a global maximization problem has to be solved, the determination of a vector $\hat{\omega}$ in the set $\underline{\Omega}^i \triangleq \Omega^1 \times \dots \times \Omega^{n_i}$ such that

$$\varphi(y_{\tau^i,\hat{\omega}}) = \max_{\omega \in \underline{\Omega}^i} \varphi(y_{\tau^i,\omega}) ,$$

where, φ is defined in Def.5.4 by

$$\varphi(y_{\tau^i,\omega}) = \sum_{k=1}^{n_i-1} \max\{0, y_{\tau^i,\omega}(\tau^i_k)-y_2(\tau^i_k),\ y_1(\tau^i_k)-y_{\tau^i,\omega}(\tau^i_k)\}(\tau^i_{k+1}-\tau^i_k),$$

and,

$$y_{\tau^i,\omega}(\tau^i_k) = y(\tau^i_k),$$

where, y is the output of system $\Sigma(\theta)$ with input function $u_{\tau^i,\omega}$, a piecewise linear, continuous function.

This is an optimization problem with a finite-dimensional design variable and a non-differentiable cost function. We propose below two approaches to this problem. The first approach employs a Monte-Carlo method to produce interpolation input functions. It is as follows.

Sub-algorithm 5.1:

Data: A set $\{G(i)\}$ of positive integers;

A discrete subset τ^i of T;

Intervals $\Omega^{i,k} \triangleq [u_1(\tau^i_k),\ u_2(\tau^i_k)]$,

$k = 1, \dots, n_i, \quad i = 0, 1, 2, \dots .$

Step 0: Set $j = 1$,

$k = 1$

Step 1: Generate, randomly, a number σ_k between 0 and 1

Step 2: Set $\omega_k^j = (1-\sigma_k)u_1(\tau_k^i) + \sigma_k u_2(\tau_k^i)$;

If $k = n_i$,

set $\omega^j = [w_1^j, \ldots, w_{n_i}^j]$,

and go to Step 3;

Else, set $k = k + 1$,

and go to Step 1.

Step 3: If $j = G(i)$, go to Step 4;

Else, set $j = j + 1$;

Set $k = 1$;

Go to Step 1.

Step 4: Solve

$$\max \{\varphi(y_{\tau^i,\omega}) \mid \omega \in \{\omega^1, \ldots, \omega^{G(i)}\}\}$$

to obtain the "worst" $\hat{\omega}$.

Thus, the maximization step in Main-algorithm 4 at i^{th} iteration is carried out over all $\{\omega^1, \ldots, \omega^{G(i)}\}$. This situation is included in Theorem 5.2, when $\{G(i)\}$ is a strictly monotonically increasing sequence as $i \longrightarrow \infty$.

The second approach is based on Bertsekas' approximation method for non-differentiable optimization[15]. For the sake of convenience, we use the following simple notation to illustrate the method.

We define $\gamma[g(\omega)]$ as a simple kink if

$$\gamma[g(\omega)] = \max\{ 0, g(\omega)\},$$

where, $\omega \in \Omega \subset R^s$, and g is a real-valued function on R^s.

Some other kinds of non-differentiable terms can also be expressed by the simple kinks. For instance,

$$\max\{ g_1, g_2, \ldots, g_m\}$$
$$= g_1 + \gamma[g_2-g_1+\gamma[\ldots+\gamma[g_{m-1}-g_{m-2}+\gamma[g_m-g_{m-1}]] \ldots]].$$

The basic idea of the approach is to approximate every simple kink by a smooth function and solve the resulting differentiable problem by conventional methods. The following two-parameter approximation $\tilde{\gamma}[g(\omega), b, c]$ is often used for a simple kink $\gamma[g(\omega)]$,

$$\tilde{\gamma}[g(\omega), b, c] = \begin{cases} g(\omega)-(1-b)^2/2c & \text{if} \quad (1-b)/c \leqslant g(\omega) \\ bg(\omega)+(1/2)c[g(\omega)]^2 & \text{if} \quad -b/c \leqslant g(\omega) < (1-b)/c \\ -b^2/2c & \text{if} \quad g(\omega) < -b/c, \end{cases}$$

where, b and c are parameters with

$$0 \leqslant b \leqslant 1, \quad 0 < c.$$

Hence, if the function g is differentiable then the function $\tilde{\gamma}[g(\omega), b, c]$ above is also differentiable with respect to ω. Its gradient is given by

$$\nabla\tilde{\gamma}[g(\omega), b, c] = \begin{cases} \nabla g(\omega) & \text{if} \quad (1-b)/c \leqslant g(\omega) \\ [b+cg(\omega)]\nabla g(\omega) & \text{if} \quad -b/c \leqslant g(\omega) < (1-b)/c \end{cases}$$

$$\Big\{ \; 0 \qquad\qquad \text{if} \quad g(\omega) < -b/c.$$

The parameter c controls the accuracy of the approximation. The parameter b determines whether the approximation is more accurate for positive or negative values of the argument $g(\omega)$. This kind of approximation is closely related to penalty and multiplier methods for constrained optimization. Bertsekas proved that when the non-differentiability of the cost function is exclusively due to the presence of simple kink terms, the number of kink terms might be arbitrary but finite, then the solution to the approximation problem will converge to the true solution to the original problem as $c \to \infty$. Also, he proposed some updating formula for the multiplier b in order to improve the computational efficiency.

In our case, since $y_1(t) < y_2(t)$, a.e. $t \in T$, the cost function might be rewritten as

$$\varphi(y_{\tau^1,\omega}) = \sum_{k=1}^{n_i-1} \max\{\, 0,\ [y_{\tau^1,\omega}(\tau_k^i) - y_2(\tau_k^1)](\tau_{k+1}^1 - \tau_k^1)\} +$$

$$\max\{\, 0,\ [y_1(\tau_k^1) - y_{\tau^1,\omega}(\tau_k^1)](\tau_{k+1}^1 - \tau_k^1)\}.$$

Thus, it is a form involved only with simple kinks convenient for applying Bertsekas' method. However, we notice, the special difficulty here is that $[y_{\tau^1,\omega}(\tau_k^1) - y_2(\tau_k^1)](\tau_{k+1}^1 - \tau_k^1)$ and $[y_1(\tau_k^1) - y_{\tau^1,\omega}(\tau_k^1)](\tau_{k+1}^1 - \tau_k^1)$ are not direct functions of ω. They are composite mappings from R^{n_1} into R, via the piecewise linear continuous function space and continuous function space. Hence, the differentiability should be understood in sense of Frechet differentiability.

5.5 Constraint construction scheme

A constraint construction scheme similar to that in Sec.3.8 is proposed below.

Definition 5.8:

Define a scalar function $\mu: \Theta \longrightarrow R$ by,

for every θ_i generated by Main-algorithm 4,

$$\mu(\theta_i) \triangleq \varphi(y_{\tau^i,\hat{\omega}}),$$

where, $\varphi(y_{\tau^i,\hat{\omega}})$ is defined in Step 4 of Main-algorithm 4.

We notice that μ is continuous in θ since φ is continuous in $y_{\tau^i,\hat{\omega}}$ and $y_{\tau^i,\hat{\omega}}$ is continuous in θ.

The following is the scheme.

Constraint construction scheme:

(In cooperation with Main-algorithm 4)

(1) Given a positive constant k and a positive, decreasing sequence $\{\varepsilon_i\}$ with $\varepsilon_i \longrightarrow 0$, as $i \longrightarrow \infty$.

(2) Construct $\underline{\hat{i}} \triangleq \{0 \leqslant j \leqslant i \mid \mu(\theta_j) > k(\varepsilon_j - \varepsilon_i)\}$.

(3) Replace $\underline{i}$ in Step 3 of Main-algorithm 4 by $\underline{\hat{i}}$ constructed in (2).

We have the following result.

Theorem 5.4:

Suppose $\tilde{F}$ is not empty.

Suppose the Main-algorithm 4 is employed together with the above constraint construction scheme.

Then, if the algorithm generates a finite sequence in Θ, the last point lies in $\tilde{F}$.

If the algorithm generates an infinite sequence, then any accumulation point of the sequence will be in $\tilde{F}$.

Proof:

The finite case is obvious.

In the infinite case, for sake of simplicity, we suppose that

$$\theta_i \longrightarrow \theta^*, \quad \text{as } i \longrightarrow \infty.$$

(1) If there is a set $J \subset \{0, 1, \ldots,\}$ of infinite cardinality such that

$$\mu(\theta_{i_j}) < k\varepsilon_{i_j}, \quad \text{for all } j \in J,$$

then (since $\varepsilon_{i_j} \longrightarrow 0$ and $\theta_{i_j} \longrightarrow \theta^*$ as $j \longrightarrow \infty$, μ is non-negative and is continuous in θ) we have

$$\mu(\theta^*) = 0.$$

As is pointed out in Sec.5.3 (see Theorem 5.3) the requirement on the fineness of discrete time interval will be satisfied in a finite number of iterations, hence $\theta^* \in \tilde{F}$.

(2) If the set J of j such that

$$\mu(\theta_j) < k\varepsilon_j$$

has finite cardinality,

then, there exists an integer $N_0 > 0$ such that after N_0 iterations every input function constructed will remain in the constraint set. Thus the conclusion in Theorem 5.3 can be applied here.

Q.E.D.

CHAPTER SIX CONCLUSION

Many control system designs may be naturally expressed as mathematical programming problems of which the constraints may be conventional (e.g., constraints on controller parameters), non-differentiable (e.g., stability constraints on eigenvalues or robustness constraints on the singular values of certain transfer function matrices), or infinite dimensional (e.g., constraints on time or frequency responses to ensure stability, performance and robustness). In this thesis, two more types of optimization problems which also arise in the design of non-linear control systems are presented and discussed. The first one is the optimization with infinite dimensional constraints and parameters. This often occurs when it is required to find a controller which cannot be parameterized by finite-dimensional design variables (e.g., to find a Lyapunov function and a non-linear control law to ensure the stability of a closed-loop non-linear system). The prototype problem considered is that of finding a function ϕ: $\Omega \longrightarrow R^m$, where Ω is a compact subset of R^n, to satisfy the inequality

$$f(x,\phi(x),\nabla_x\phi(x)) \leqslant 0, \text{ for all } x \in \Omega, \quad (6.1)$$

where, f: $\Omega \times R^m \times R^{n\times m} \longrightarrow R^p$ is given; or finding ϕ to minimize a cost function subject to this inequality constraint. Main-algorithm 1, 2, 2', and 3 are proposed to this kind of problems. All of them are of the outer approximation type.

Main-algorithm 1 is for the feasibility problem (i.e., finding a function ϕ satisfying (6.1)). The function ϕ is approximated using finite element methodology by piecewise linear continuous functions. The number of simplices is increased adaptively during computation. At each iteration, the set of nodal points at which the constraint must be satisfied is augmented by adding in the "worst" point (i.e., the most violence of constraint) in Ω. A set of auxiliary points are used to obtain a presently feasible piecewise linear continuous function. The convergence is proved when an accumulation point exists. An example, which concerns with the design of a stability regulator to a 2-dimensional non-linear system, is presented. It is noticed that, however, the simulations from some initial states are not satisfactory in the example. This is supposed due to the absence of performance criteria in the constraints.

Main-algorithm 2 and 2' are for the constrained optimization problem (i.e., finding a Φ to minimize a cost function subject to the constraint (6.1)). They follow the same basic methodology as in Main-algorithm 1. However, in these two algorithms a criterion is set for the fineness of simplices. This device prevents the algorithms from stopping at a "coarse" piecewise linear continuous function which satisfies the present optimality and constraint (6.1).

Main-algorithm 3 is for the purpose that a "regular" shape of simplex set is preferred (in the interest of convenience of practising) in the above mathematical programmings. In this case, the methodology approaches more closely to the conventional finite element method.

The second mathematical programming problem is the optimization with constraints which should be satisfied by an infinite number of functions on a time or frequency domain.

Only the feasibility problem of this kind is discussed in this thesis. It occurs in the design of a controller such that the output of the closed-loop system satisfies hard constraints in a time domain for a class of input functions. It also arises in the design of robust circuits. In this thesis, we consider such problem with finite-dimensional design variables. The prototype problem of this kind is that of finding a finite-dimensional design variable θ which lies in a compact admissible set Θ such that the output function y satisfies

$$y(t;u,\theta) \leqslant 0, \qquad \text{for all} \quad t \in T, \text{ all } u \in U,$$

where, T is a time interval, u is an input function on T, U is the class of admissible input functions, y is the output decided by u and θ.

Main-algorithm 4 is proposed for the second mathematical programming. It is also of the outer approximation type. The time domain T is discretized adaptively and the input functions are replaced by piecewise linear continuous functions.

Convergence analysis is made for the above algorithms. Constraint dropping schemes are proposed to reduce the amount of computation.

The work which has been done in this thesis is only a first attempt towards these two mathematical programmings. By no means It is complete. The computation is demanding. An infinite sequence of piecewise linear continuous functions does not always have an accumulation point in the class of continuously differentiable (or, piecewise continuously differentiable) functions. However, we believe that it is worthy to explore more powerful algorithms towards these mathematical programming problems, since the design methodology (i.e., constructing a control law directly) presented in the work is attractive in the design of non-linear control systems and, also, since such mathematical programmings arise frequently in other engineering fields.

REFERENCES

1 MAYNE, D.Q., 'Control System Design via Mathematical Programming', in 'Design of Modern Control Systems', (BELL, D.J., P.A.COOK and N.MUNRO, ed.), Peter Peregrinus Ltd., (1982), pp.144-158

2 ZAKIAN, V. and U.AL-NAIB, 'Design of Dynamical and Control Systems by the Method of Inequalities', Proc. IEE, Vol.120, No.11, (1973), pp. 1421-1427

3 MAYNE, D.Q., E.POLAK and A.SANGIOVANNI-VINCENTELLI, 'Computer-Aided Design via Optimization: A Review', Automatica, Vol.18, No.2, (1982), pp. 147-154

4 E.POLAK, 'Semi-Infinite Optimization in Engineering Design', in K.O.KORTANEK, ed.), Springer-Verlag, (1983), pp.236-248

5 MAYNE, D.Q. and E.POLAK, 'An Algorithm for Optimization Problems with Functional Inequality Constraints', IEEE Trans. Auto. Control, AC-21, (1976), pp.184-193

6 MAYNE, D.Q., E.POLAK and R.TRAHAN, 'Combined Phase I, Phase II Methods of Feasible Directions', Mathematical Programming, Vol.17, (1979), pp. 61-73

7 MAYNE, D.Q., E.POLAK and R.TRAHAN, 'An Outer Approxima-

tion Algorithm for Computer Aided Design Problems', J. of Optimization Theory and Applications, Vol.28, (1979), pp. 231-252

8 KRENER, A.J., 'On the Equivalence of Control Systems and the Linearization of Nonlinear Systems', SIAM J. Contr., Vol.11, (1973), pp. 670-676

9 MEYER, G. and L.CICOLANI, 'A Formal Structure for Advanced Flight Control Systems', NASA TN D-7940, 1975

10 BROCKETT, R.W., 'Feedback Invariants for Nonlinear Systems', IFAC Congr., Helsinki, Finland, 1978

11 HUNT, L.R., R.SU and G.MEYER, 'Global Transformations of Nonlinear Systems', IEEE Trans. Auto. Control, AC-28, No.1, (1983), pp. 24-31

12 KAILATH, T., 'Linear Systems', Prentice-Hall, (1980), Chap.4

13 BESTLE, D. and M.ZEITZ, 'Canonical Form Observer Design for Non-Linear Time-Variable Systems', INT. J. Control, Vol.38, No.2, (1983), pp. 419-431

14 MAYNE, D.Q. and M.SAHBA, 'An Efficient Algorithm for Solving Inequalities', JOTA, Vol.45, No.3, (1985), pp. 407-423

15 BERTSEKAS, D.P., 'Nondifferentiable Optimization via Approximation', in 'Nondifferentiable Optimization' (BALINSKI, M.L. and P.WOLFE, ed.) North-Holland, (1975), pp. 1-25

APPENDIX A NUMERICAL METHOD IN OPTIMAL CONTROL PROBLEM

A.1 Optimal control problem and dynamic programming

In many control problems it is natural to want to control the system so that a given performance index is minimized. Thus the optimal control problem makes up an interesting and well-studied, yet still developing, field in control system theory. The following is a simple type of optimal control problem in Lagrange form.

$$(OP): \qquad \text{Minimize} \int_0^1 L(x(t),t,u(t))dt$$

Subject to

$$\dot{x}(t) = f(x(t),t,u(t)) \qquad \text{a.e.} \quad t \in [0,1]$$

$$x(0) = x_0$$

$$x(1) = x_1$$

$$u(t) \in \Omega \qquad \text{a.e.} \quad t \in [0,1]$$

$$\{(x(t),t) \mid t \in [0,1]\} \subset \mathcal{A} .$$

For problem (OP), the feasible control set $F_{(x_0,t_0)}$ for an initial pair (x_0,t_0), where $0 \leq t_0 < 1$, consists of all piecewise continuous functions u such that $u(t) \in \Omega$, a.e. $t \in [t_0,1]$ and the corresponding trajectory $x(t;x_0,t_0)$ reaches x_1 at t=1 with $(x(t),t) \in \mathcal{A}$, a.e. $t \in [t_0,1]$.

The pair (x,u) is an admissible process when u is a feasible control and x is the corresponding trajectory.

For an initial pair (x_0,t_0) and a feasible control u, define that the performance function of u is

$$J(x_0,t_0,u) \triangleq \int_{t_0}^{1} L(x(t),t,u(t))dt \; .$$

The value function for problem (OP), when the initial time is t_0, is defined as

$$V(x_0,t_0) \triangleq \min \{J(x_0,t_0,u) \mid u \in F_{(x_0,t_0)}\}.$$

An admissible process which is corresponding to the value function is called an optimal process.

The most significant aspect of this kind of optimal theory concerns the following Hamilton-Jacobi equation:

$$\phi_t(x,t) + \max_{\upsilon\in\Omega} \{\phi_x(x,t)f(x,t,\upsilon)-L(x,t,\upsilon)\} = 0. \tag{A.1}$$

We have the following theorem.

Theorem A.1:

Suppose $W(x,t)$ is a $C^{(1)}$ solution of the Hamilton-Jacobi equation (A.1), $W: \Omega \longrightarrow R$, and suppose $(\bar{x},\bar{u})$ is an admissible process such that

$$W_t(\bar{x}(t),t)+W_x(\bar{x}(t),t)f(\bar{x}(t),t,\bar{u}(t))-L(\bar{x}(t),t,\bar{u}(t)) = 0$$

$$\text{a.e. } t \in [t_0, 1] \; . \tag{A.2}$$

Then $(\bar{x},\bar{u})$ is an optimal process.

The approach to finding the optimal process using Hamilton-Jacobi equation (A.1) is usually referred to as the Dynamic Programming Method.

A.2 A dual problem

Now let us specify $t_0 = 0$. Suppose $W(x,t)$ is a $C^{(1)}$ solution of equation (A.1) and (x,u) is any admissible process. Then

$$W(x_1,1)-W(x_0,0) = \int_0^1 \frac{d}{dt}W(x(t),t)dt$$
$$= \int_0^1 \{W_t(x(t),t)+W_x(x(t),t)f(x(t),t,u(t))\}dt$$
$$\leqslant \int_0^1 L(x(t),t,u(t))dt. \qquad \text{(A.3)}$$

If $(\bar{x},\bar{u})$ is an admissible process satisfying (A.2), then

$$W(x_1,1)-W(x_0,0) = \int_0^1 W_t(\bar{x}(t),t)+W_x(\bar{x}(t),t)f(\bar{x}(t),t,\bar{u}(t))dt$$

$$= \int_0^1 L(\bar{x}(t),t,\bar{u}(t))dt .$$

Thus $(\bar{x},\bar{u})$ is an optimal process and $W(x_1,1)-W(x_0,0)$ is the minimum cost, i.e.,

$$V(x_0,0) = W(x_1,1)-W(x_0,0).$$

Since (A.3) is valid for any $C^{(1)}$ function W which satisfies (A.1) provided the set of feasible control is not empty, this implies that

$$V(x_0,0) \geqslant \sup\{W(x_1,1)-W(x_0,0)\}$$

where, the supremum is taken over all $C^{(1)}$ functions W which satisfy the Hamilton-Jacobi equation (A.1).

Theorem (A.1) gives sufficient conditions for optimality. When some convexity hypothesis is made. i.e.

$$u \longmapsto (L(x,t,u),\ f(x,t,u))$$

is convex for all $(x,t)\in\Omega$,

the optimality conditions becomes necessary as well[11]. Furthermore, the conclusions of Thm A.1 remains valid when the Hamilton-Jacobi equation is relaxed to inequality[11]

$$\phi_t(x,t)+\phi_x(x,t)f(x,t,u)-L(x,t,u) \leqslant 0. \tag{A.4}$$

Hence, we have the following duality principle.

Theorem A.2:

Suppose the convexity hypothesis holds, then

$$V(x_0,0) = \sup\{W(x_1,1)-W(x_0,0)\}$$

where the supremum is taken over all $C^{(1)}$ functions W which satisfy the Hamilton-Jacobi inequality (A.4).

A.3 A construction method

With a further hypothesis called "strong calmness" on the system (This includes the requirement that

$$\lim_{\hat{x}_1 \longrightarrow x_1} \inf \frac{\eta(\hat{x}_1)-\eta(x_1)}{\|\hat{x}_1 - x_1\|} > -\infty,$$

where η is the minimum cost expressed as a function of terminal constrained state x_1), we have the following existence theorem[iii].

Theorem A.3:

Suppose the convexity hypothesis holds and $f(x,t,u)$, $L(x,t,u)$ are continuous in (x,t,u) and Lipschitz continuous in x, uniformly in (t,u). Suppose the strong calmness hypothesis also holds. Then there exists a Lipschitz function $W(x,t)$ such that the Hamilton-Jacobi inequality (A.4) is satisfied and an admissible process (x,u) is optimal if and only if

$$\frac{d}{dt}W(x(t),t) = L(x(t),t,u(t)) \qquad \text{a.e. } t \in [0,1].$$

The function $W(x,t)$ can be taken as a solution to the dual problem:

$$\text{Maximize } \{ W(x_1,1) - W(x_0,0) \}$$

over Lipschitz continuous functions $W: \Omega \longrightarrow R$ such that (A.4) is satisfied. The value of the dual problem is the minimum cost.

Hence, the optimal control problem (OP) can be turned into a kind of optimization problem which is

Maximize $W(x_1,1) - W(x_0,0)$

subject to Lipschitz continuous functions $W: \mathcal{A} \longrightarrow R$

$$W_t(x,t)+W_x(x,t)f(x,t,u) \leqslant L(x,t,u)$$

$$\text{for all } u \in \Omega$$

$$\text{a.e. } (x,t) \in \mathcal{A} .$$

This is obviously the mathematical programming problem which we discussed in Chapter 3. Thus the algorithms proposed there can be applied to this optimal control problem.

References:

i FLEMING, W.H. and R.W.RICHEL, 'Deterministic and Stochastic Optimal Control', Springer-Verlag, (1975)

ii VINTER, R.B. and R.M.LEWIS, 'A Necessary and Sufficient Condition for Optimality of Dynamic Programming Type, Making No a Priori Assumptions on the Controls', SIAM J. Control and Optimization, Vol.16, No.4, (1978)

iii CLARKE, F.H , 'The Generalized Problem of Bolza', SIAM J. Control and Optimization, Vol.14, (1976), pp. 682-699