1981

Stability analysis of complex dynamical systems: some computational methods

Narotham Reddy Sarabudla
Iowa State University

8128855

SARABUDLA, NAROTHAM REDDY

STABILITY ANALYSIS OF COMPLEX DYNAMICAL SYSTEMS: SOME COMPUTATIONAL METHODS

Iowa State University PH.D. 1981

University
Microfilms
International 300 N. Zeeb Road, Ann Arbor, MI 48106

PLEASE NOTE:

In all cases this material has been filmed in the best possible way from the available copy. Problems encountered with this document have been identified here with a check mark _√_.

1. Glossy photographs or pages ____
2. Colored illustrations, paper or print ____
3. Photographs with dark background ____
4. Illustrations are poor copy ____
5. Pages with black marks, not original copy ____
6. Print shows through as there is text on both sides of page ____
7. Indistinct, broken or small print on several pages _√_
8. Print exceeds margin requirements ____
9. Tightly bound copy with print lost in spine ____
10. Computer printout pages with indistinct print ____
11. Page(s) ________ lacking when material received, and not available from school or author.
12. Page(s) ________ seem to be missing in numbering only as text follows.
13. Two pages numbered ________. Text follows.
14. Curling and wrinkled pages ____
15. Other ____________________________

Stability analysis of complex dynamical systems:

Some computational methods

by

Narotham Reddy Sarabudla

A Dissertation Submitted to the

Graduate Faculty in Partial Fulfillment of the

Requirements for the Degree of

DOCTOR OF PHILOSOPHY

Department: Electrical Engineering
Major: Electrical Engineering

Approved:

Signature was redacted for privacy.

In Charge of Major Work

Signature was redacted for privacy.

For the Major Department

Signature was redacted for privacy.

For the Graduate College

Iowa State University
Ames, Iowa

1981

TABLE OF CONTENTS

I. INTRODUCTION

It is well-known that many dynamical systems encountered in engineering and physical sciences can be modeled by systems of ordinary differential equations and that such differential equations are frequently nonlinear. For the engineers and scientists involved in modeling, analysis and design of such dynamical systems, stability is a fundamental problem. The concept of stability has its origin in mechanics where the position of equilibrium (rest) of a rigid body is considered to be stable, if it returns to its original position of equilibrium after a small disturbance. The nonlinear differential equations representing the dynamical system, in general, are so complex that they cannot be solved analytically in a closed form. Thus, in order to ascertain the qualitative behavior of an equilibrium point of a dynamical system alternative methods of analysis are required.

In 1892, A.M. Lyapunov, a Russian mathematician proposed two methods known as Indirect (First) method and Direct (Second) method to investigate the stability of dynamical systems represented by differential equations. Although Lyapunov's stability theory (see [1]) is playing an important role in modern control theory, his work went unnoticed for a long time. Subsequently, much work has been done on the extension and application of his work to systems represented by difference equations, stochastic differential equations, functional differential equations, Volterra integro-differential equations, partial differential equations and hybrid systems which are appropriately modeled by a mixture of different types of equations.

Of particular importance is the Direct method, which has its origin in energy considerations, and through which one can ascertain the qualitative properties of the equilibrium of a dynamical system without any knowledge of its solution trajectories. In the analysis and design by Lyapunov's Direct method, the concept of asymptotic stability and the domain of attraction of an equilibrium is invaluable. In the literature, the domain of attraction is also referred to as the region of attraction or as the asymptotic stability region of an equilibrium. The domain of attraction of an equilibrium is a region with the equilibrium point in its interior and having the property that the trajectories of the system starting at any point within this region will eventually approach the equilibrium point. To determine an estimate for the domain of attraction of an equilibrium by the Lyapunov's Direct method, we need to find a scalar function such that this function and its time derivative along the solutions of the differential equation of the system in question satisfy certain conditions in some neighborhood of the given equilibrium point. (Without loss of generality, the equilibrium point will be assumed to be the origin of R^n.) Although the Direct method of Lyapunov is a powerful tool available for the analysis and design of nonlinear dynamical systems, it suffers from several drawbacks. Notably, there is no universal and systematic method available which tells us how to find the required Lyapunov function. Although converse Lyapunov theorems have been established, these results provide in general no clue (except in the case of linear equations) as how to construct Lyapunov functions. In addition, finding a Lyapunov function successfully does not necessarily mean that it is necessarily the best choice in a particular problem. The prevailing opinion among researchers is that even if applicable, it often requires excessive computation time to find a reasonably satisfactory Lyapunov function.

Generally, the degree of the above difficulties associated with the Lyapunov theory increases as a system becomes large and complex. For this reason, to analyze the qualitative behavior of large-scale dynamical systems, it is frequently convenient to view such systems as interconnected systems composed of several lower order free subsystems. The analysis of such large-scale systems can often be accomplished in terms of the properties of the simpler subsystems and in terms of the properties of the interconnecting structure.

This dissertation is devoted to the development of systematic, efficient and fast algorithms to estimate the domain of attraction of general and large-scale nonlinear dynamical systems.

II. NOTATION

The symbol i denotes $\sqrt{-1}$. Let U and V be arbitrary sets. If u is an element of U we write $u \varepsilon U$. If U is a subset of V, we write $U \subset V$ and we denote the boundary of U by ∂U. If W is a convex polyhedral region, then the elements of the set E(W) denotes its extreme vertices and $\mathbf{K}[W] = W \cup \partial W$ denotes its convex hull.

Let R denote the real line, let $R^+ = [0,\infty)$, let E^n denote the Euclidean n-space and let C^n denote the set of n-tuples of complex numbers. If $x = \text{col}[x_1, x_2, \ldots, x_n] \varepsilon E^n$, then $x^T = [x_1, x_2, \ldots, x_n]$ denotes the transpose of x. If $x,y \varepsilon E^n$, then $x > y$ (resp., $x \geq y$) denotes $x_i > y_i$ (resp., $x_i \geq y_i$), $i = 1, 2, \ldots, n$. The symbol $|\cdot|$ denotes a vector norm on E^n. Also, the symbol $\|\cdot\|$ is used to denote the matrix norm induced by some vector norm. If f is a function or mapping of a set X into a set Y, we write $f : X \rightarrow Y$.

Unless otherwise specified, matrices are usually assumed to be real and we denote them by upper case letters. If $A=[a_{ij}]$ is an arbitrary $n \times n$ matrix, then A^T denotes the transpose of A. An eigenvalue of A is identified as $\lambda(A)$ and Re $\lambda(A)$ denotes the real part of $\lambda(A)$. If all eigenvalues of A happen to be real we write $\lambda_{max}(A)$ and $\lambda_{min}(A)$ to denote the largest and smallest eigenvalues of A, respectively. We call a real $n \times n$ matrix $A = [a_{ij}]$ an M-matrix (Minkowski matrix) if $a_{ij} \leq 0$ for all $i \neq j$ and if all principal minors of A are positive. The symbol I is used to denote an identity matrix. If $\underline{B}$ is a set of matrices, we let $\underline{B}'$ denote its multiplicative semi-group.

The time derivative of a variable (e.g., $\frac{dx}{dt}$) is expressed by a dot over the variable (e.g., $\dot{x}$). If $v:R^n \rightarrow R$, then $\nabla v(x)$ denotes its gradient and $\nabla v(x)^T$ is the transpose of the gradient. We use $Dv(x)$ to denote the total time derivative of $v(x)$ along the trajectories of a dynamical system. We define $C(r) = \left\{ x \varepsilon E^n: |x| < r \right\}$ and $\overline{C(r)} = \left\{ x \varepsilon E^n: |x| \leq r \right\}$ for some $r > 0$ as open and closed balls, respectively.

A comparison function $\Psi:R^+ \rightarrow R^+$ belongs to class K (i.e., $\Psi \varepsilon K$) if $\Psi(0) = 0$ and if $\Psi(r_1) > \Psi(r_2)$ whenever $r_1 > r_2$. If $\Phi \varepsilon K$ and if in addition $\lim_{r \rightarrow \infty} \Phi(r) = \infty$, then we say that Φ belongs to class KR (i.e., $\Phi \varepsilon KR$).

III. STABILITY THEORY

The objective of this chapter is to present the necessary background material needed in subsequent chapters and also to present, in general terms, some of the algorithms available for the estimation of domain of attraction of an equilibrium. In section A, the Indirect method and the Direct method of Lyapunov will be presented. Since the literature on these two methods is enormous and the material on this subject can be found in several standard texts (see, e.g., [2], [3]), only the basic results required in subsequent chapters will be presented. In section B, some significant results on the constructive stability due to Brayton and Tong [4], [5] will be presented. Their constructive algorithm determines the stability, instability and global asymptotic stability of nonlinear dynamical systems having a single equilibrium point. For dynamical systems which are asymptotically stable but not necessarily globally asymptotically stable (e.g., a dynamical system having more than one equilibrium point), their results will be modified to develop a systematic and fast algorithm to estimate the domain of attraction of such systems. This modification, which is not entirely straightforward, will be discussed in Chapter V. In section C, some of the existing algorithms for the estimation of the domain of attraction of an equilibrium will be summarized. In addition, some observations concerning their usefulness will be presented.

A. Lyapunov Stability Theory

In this chapter and in chapters IV and V, we consider dynamical systems described by ordinary differential equations of the form

$$\dot{x} = f(x) \tag{1}$$

where $t \in R^+$, and $f: E^n \to E^n$. We assume that f is sufficiently smooth to ensure the existence of unique solutions $x(t, x_0, t_0)$ of (1), for any $x_0 \in E^n$ and for all $t \geq t_0$, where $t_0 \in R^+$ and $x_0 = x(t_0, x_0, t_0)$. Henceforth, we refer to t as "time" and to x_0 as an initial point. Without loss of generality, we assume that $t_0 = 0$. Let $x = 0$ be an isolated equilibrium point of (1) so that $f(0) = 0$. The point $x = 0$ is referred to as an isolated equilibrium point if there exists a constant $r' > 0$ such that $C(r') \subset E^n$ contains no equilibrium points of (1) other than $x = 0$ itself. Next, a short summary of Lyapunov's Indirect (First) method will be presented.

1. Indirect method

By this method, under certain conditions, one can draw conclusions about the stability behavior of an equilibrium point of (1) by studying the stability behavior of a linear system. In this method each equilibrium point, if there is more than one, is investigated separately. If an equilibrium point is not at the origin, it can always be transferred to the origin by an appropriate coordinate transformation. In addition to the existing assumptions on f, we assume that f is continuously differentiable with respect to all x_i, $i = 1, \ldots, n$. The first step in this method is to expand f(x) in a Taylor series about the equilibrium x = 0 and to separate the linear terms, which yields

$$\dot{x} = Jx + f_1(x). \tag{2}$$

Here $J = [\frac{\partial f(x)}{\partial x}]_{x=0}$ is the Jacobian evaluated at x = 0, and $f_1(x)$ consists of higher order terms in the components of x. We call the equation

$$\dot{x} = Jx \tag{3}$$

the first approximation (see Hahn [3, pp. 120-122]) to the nonlinear system (1). In the literature system (3) is also referred to as the linearized system of nonlinear system (1). Lyapunov showed that if all the eigenvalues of matrix J have negative real parts then the equilibrium x = 0 of (1) is asymptotically stable, and if at least one eigenvalue of matrix J has a positive real part then the equilibrium point x = 0 of (1) is unstable. Even though x = 0 of (1) may be found to be asymptotically stable by this method, this method fails to yield any information on the extent of asymptotic stability. In order to estimate the extent of asymptotic stability we need to use Lyapunov's Direct (Second) method, which we will present next.

2. Direct method

This is a powerful method to ascertain the stability properties of an equilibrium point of a dynamical system represented by (1) without any knowledge of its solutions. In addition, this method also yields information on the extent of stability and thus assumes particular importance in applications. This method is based on the concept of energy in a physical system. In such a system the stored energy is a positive function, and the rate of change of such stored energy along the motions of the physical system determines the stability of an equilibrium point. But for a complex dynamical system described by (1), it will be difficult to find an expression for energy. Thus, Lyapunov showed that if a scalar function v(x) with properties similar to those of energy can be found, then the sign definiteness of the time derivative of v(x) with respect to (1) given by

$$\dot{v}_{(1)}(x) = \sum_{i=1}^{n} \frac{\partial v}{\partial x_i}(x) \bullet f_i(x) = \nabla v(x)^T \bullet f(x) \tag{4}$$

will determine the stability of an equilibrium of (1). The function $v: E^n \to R$ which is assumed to be continuously differentiable with respect to all of its arguments, is referred to as a Lyapunov function in the literature. In the principal Lyapunov stability results for the equilibrium $x = 0$ of (1), such v-functions are characterized as being positive definite, negative definite, decrescent, radially unbounded, and so forth. Refer, e.g., to [2, pp. 14-20] for a summary of the principal Lyapunov stability results and for the definitions of these concepts. Assume that Ω (where $\Omega \subset E^n$) is an arbitrary domain with the equilibrium $x = 0$ of (1) in its interior. Next, we present three important theorems of the Direct method.

<u>Theorem 1</u>. For all $x \in \Omega$, if there exists a positive definite function v(x) such that $\dot{v}_{(1)}(x)$ is negative semidefinite, then the equilibrium $x = 0$ of (1) is <u>stable</u>.

Theorem 2. For all $x \varepsilon \Omega$, if there exists a positive definite function $v(x)$ such that $\dot{v}_{(1)}(x)$ is negative definite, then the equilibrium $x = 0$ of (1) is asymptotically stable.

Theorem 3. For all $x \varepsilon E^n$, if there exists a positive definite and radially unbounded function $v(x)$ such that $\dot{v}_{(1)}(x)$ is negative definite, then the equilibrium $x = 0$ of (1) is asymptotically stable in the large.

Let Ω be the domain given in Theorem 2. It can be shown that if a domain D defined by

$$D = \left\{ x \varepsilon E^n : v(x) \leq d,\ d > 0 \right\} \tag{5}$$

is entirely contained in Ω, then this domain D will be contained in the actual domain of attraction of $x = 0$ and as such D can be used as an estimate of the domain of attraction of $x = 0$ for (1). If in particular, $d > 0$ is the largest constant such that $D \subset \Omega$ is true, then D will be the largest estimate of the domain of attraction which one can obtain by this particular Lyapunov function $v(x)$. In this dissertation, an effort will be made to develop systematic and fast algorithms to estimate the best possible domain of attraction of a nonlinear dynamical system.

B. Constructive Stability Results

In two papers [4] and [5], Brayton and Tong present an algorithm to construct a Lyapunov function to prove the stability, instability and global asymptotic stability of the equilibrium $x = 0$ of dynamical systems represented by (1). Their algorithm is applicable to systems having only one equilibrium point. The basic idea involved in their method is to generate for (1) an appropriate set of matrices. This will be accomplished by linearizing the system (1), applying Euler's method and utilizing the convexity properties of the set of matrices. First, they established the concepts of stability, instability and asymptotic stability of a set of matrices and then they utilized these concepts to deduce the stability, instability and global asymptotic stability of the equilibrium $x = 0$ of (1). Some of their results, which we require in subsequent chapters, will be presented next. Refer to [4], [5] for further details concerning these results and for the notation used herein.

A set $\underline{A}$ of $n \times n$ matrices is stable if for every neighborhood of the origin $U \subset C^n$, there exists another neighborhood of the origin V, such that for each $M \varepsilon \underline{A}'$ (where $\underline{A}'$ denotes the multiplicative semigroup generated by $\underline{A}$), $MV \subseteq U$. It can be shown (see [5]) that the following statements are equivalents:

a) $\underline{A}$ is stable.

b) $\underline{A}'$ is bounded.

c) There exists a bounded neighborhood of the origin $W \subset C^n$ such that $MW \subseteq W$ for every $M \varepsilon \underline{A}$. Furthermore, W can be chosen to be convex and real.

d) There exists a vector norm $\| \cdot \|_W$ such that $\| Mx \|_W \leq \| x \|_W$ for all $M \varepsilon \underline{A}$ and all $x \varepsilon C^n$.

Since the statements (c) and (d) are related (see [6, Ch. 2]) by

$$\| x \|_W \equiv \inf \left\{ \alpha \;\middle|\; \alpha \geq 0,\ x \varepsilon \alpha W \right\} ,$$

it follows that $\|x\|_W$ defines a Lyapunov function for $\underline{A}$, i.e., it defines a Lyapunov function w with the property

$$w(Mx) \leq w(x) \quad \text{for all } M \in \underline{A} \text{ and } x \in C^n.$$

Note that a set of complex matrices $\underline{A} = \{M_j = M_{j1} + iM_{j2} : M_{j1} \text{ and } M_{j2} \text{ are real}\}$ can always be mapped into the real set $\sigma\underline{A}$ which is stable if and only if $\underline{A}$ is stable, where

$$\sigma\underline{A} = \left\{ \begin{bmatrix} M_{j1} & -M_{j2} \\ M_{j2} & M_{j1} \end{bmatrix} \right\}$$

We call a set of matrices $\underline{A}$ asymptotically stable if there exists a $\rho > 1$ such that $\rho\underline{A}$ is stable. The following statements are equivalent:

a) $\underline{A}$ is asymptotically stable.

b) There exists a convex, balanced, and polyhedral neighborhood of the origin W and positive $\gamma < 1$ such that, for each $M \in \underline{A}$, we have $MW \subseteq \gamma W$.

c) $\underline{A}$ is stable and there exists K such that for all $M \in \underline{A}'$, $|\lambda_i(M)| \leq K < 1$ (where $\lambda_i(M)$ denotes the i-th eigenvalue of M).

Note that if $\underline{A}$ is stable, then $\gamma\underline{A}$ is asymptotically stable for all positive $\gamma < 1$.

In [4] and [5], a constructive algorithm is given to determine whether a set of m n x n matrices $\underline{A} = \{M_0, \ldots, M_{m-1}\}$ is stable by starting with an initial polyhedral neighborhood of the origin W_0 and by defining subsequent regions W_{k+1} by

$$W_{k+1} \triangleq \kappa[\bigcup_{j=0}^{\infty} M_{k'}^{j} W_k], \text{ where } k' \triangleq (k-1) \bmod m.$$

Now $\underline{A}$ is stable if and only if $W^* = \left\{ \bigcup_{k=0}^{\infty} W_k \right\}$ is bounded. Note that $W^* = \kappa[\bigcup MW_0, M \in \underline{A}']$. Since all extreme points z of W_{k+1} are of the form $z = M_i^j u$, where u is an extreme point of W_k, we need only deal with the extreme points of W_k in order to obtain

$$W_{k+1} = \kappa[M_{k'}^{j} u : u \in E(W_k)]$$

where $E(W_k)$ denotes the set of extreme points of W_k. Clearly, the new extreme points of $E(W_{k+1})$ are images of $E(W_k)$. If $|\lambda(M_{k'})| < 1$ for $M_{k'} \in \underline{A}$, then there exists $J_{k'}$ such that

$$\kappa[\bigcup_{j=0}^{\infty} M_{k'}^{j} W_k] = \kappa[\bigcup_{j=0}^{J_{k'}} M_{k'}^{j} W_k].$$

Since W_k is a bounded neighborhood of the origin, the $J_{k'}$ can be recognized since

$$M_{k'} \kappa[\bigcup_{j=0}^{J_{k'}} M_{k'}^{j} W_k] \subseteq \kappa[\bigcup_{j=0}^{J_{k'}} M_{k'}^{j} W_k].$$

Thus W_{k+1} will be formed in a <u>finite number of steps</u>, since W_k is expressed as the convex hull of a finite set of points.

In [4], the following <u>instability</u> stopping criterion is also established: $\underline{A}$ is unstable if there exists a k such that $\left\{ \partial W_0 \cap \partial W_k \right\} = \emptyset$. For additional (and improved) instability stopping criteria, refer to [5].

In practice, W_0 is chosen as simple as possible, i.e., it is chosen as the region defined by

$$E(W_0) = \left\{ w_i = (x_{i1}, \ldots, x_{in}) \in E^n : x_{ii} = 1,\ x_{ij} = 0 \text{ if } i \neq j,\ i = 1, \ldots, n \right\}.$$

Note that W_0 determined in this way is simple in two senses: It is symmetric; of all symmetric polyhedral regions, it possesses a minimal number of extreme points (2n).

The following algorithm incorporates the above discussion. (For a more complete version, see [4] and [5].)

<u>Step 1</u>:

a. If the set $\underline{S}$ is a set of complex $n \times n$ matrices, let $\underline{S} = \sigma\underline{S}$ and set n to 2n.

b. Form the vertex set $E(W_0) = \left\{ w_i : x_{ii} = 1,\ x_{ij} = 0 \text{ if } i \neq j,\ i = 1, \ldots, n \right\}$, where $w_i = (x_{i1}, \ldots, x_{in}) \in E^n$.

c. Set $k = 0$.

<u>Step 2</u>: Form the new vertex set $E(W_{k+1})$.

<u>Step 3</u>: Exit unstable if $E(W_0) \cap E(W_{k+1}) = \emptyset$.

<u>Step 4</u>:

a. Set $W_F = W_k$.

b. Exit stable if $E(W_{k+1}) \subseteq \kappa[W_F]$.

<u>Step 5</u>: Set $k = k+1$ and go to Step 2.

In [5] it is also shown that if a set of matrices $\underline{A}$ with a finite set of extreme matrices $E(\underline{A})$ is asymptotically stable, then the constructive algorithm will terminate "stable" in a finite number of steps. We have no way of knowing, by means of the constructive algorithm alone, that $\underline{A}$ is asymptotically stable at the termination of the algorithm. However, we can show that $\underline{A}$ is asymptotically stable by guessing a $\rho > 1$ but sufficiently close to 1 and then proving $\rho\underline{A}$ stable by using the constructive algorithm.

The applicability of the above results (concerning the stability of matrices) in the stability analysis of the equilibrium $x = 0$ of (1) is given in [4] and [5]. The basic idea involved in their method is to view the system (1) as a "linear system":

$$\dot{x} = M(x) \cdot x \tag{6}$$

where $M(x)$ is a matrix. Application of Euler's method yields

$$x_{n+1} = x_n + h_n M(x_n) x_n = (I + h_n M(x_n)) x_n \tag{7}$$

where h_n is current step size (i.e., $h_n \equiv t_{n+1} - t_n$). Let $\underline{S}$ be a set of matrices with the property that for all $x \in E^n$, there exists $M \in \underline{S}$ such that $f(x) = Mx$. Now the equation (7) can be written as

$$x_{n+1} = (I + h_n M) x_n \qquad M \in \underline{S}. \tag{8}$$

Set $\underline{S}$ is not unique, but should preferably be chosen to be minimal. For example, we may let $\underline{S}$ be the set of Jacobians, $J(x)$. Now define a set of matrices $\underline{A}$ by

$$\underline{A} \equiv \left\{ I + h_n M : 0 \leq h_n \leq h', \ M \in \underline{S} \right\}. \tag{9}$$

If $\underline{A}$ is stable (asymptotically stable) then the equilibrium $x = 0$ of (1) is stable (globally exponentially stable).

In general, the set $\underline{A}$ defined by (9) is an infinite set. However, the following result (see Appendix A) reduces the analysis to finite sets:

Let $\underline{A}$ be a set of matrices and let $E(\underline{A})$ be the set of extreme matrices of $\underline{A}$. Then $\mathcal{K}[\underline{A}]$ is stable if and only if $E(\underline{A})$ is stable. Thus, if $E(\underline{A})$ is finite, the analysis is in terms of a finite number of sets. For further details, see [4] and [5].

C. Domain of Attraction

For a dynamical system defined by (1), suppose that $x = 0$ is an asymptotically stable equilibrium point. In a real situation, even without any reference input to the dynamical system, the trajectory of such a dynamical system constantly shifts in an arbitrary neighborhood of the equilibrium point due to ever-present internal and external disturbances (e.g., changes in component values, noise). Hence, a designer will be interested in estimating a region around the equilibrium point such that the trajectories of the disturbed system originating from any point in such a region will eventually approach the equilibrium point. The estimation of such a region gives the designer an idea about the magnitude of allowable disturbances, allowable responses, or both. In the Lyapunov stability theory, such a region corresponds to the domain of attraction. A literature survey indicates that the methods based on results by Krasovskii, Zubov, Lurè, and results using quadratic Lyapunov functions can be utilized to find an estimate for the domain of attraction. This section will be devoted to a discussion of some of these methods.

In Krasovskii's method (see, e.g., [7], [8]), a quadratic function in f-space (where f is defined in (1)), $v(x) = f(x)^T Bf(x)$, will be selected as a Lyapunov function candidate and the elements of the B matrix for such a function will be picked arbitrarily. Now the time derivative of $v(x)$ with respect to (1) will be computed and then the domain defined by $D = \left\{x \in E^n : v(x) = f(x)^T Bf(x) \leq d,\ d > 0,\ \dot{v}_{(1)}(x) < 0\right\}$ will be an estimate of the domain of attraction that can be obtained by this method for the equilibrium point $x = 0$. In [9] and [10], this method has been utilized to find an estimate for the domain of attraction of chemical reactor systems. The existing literature on the Krasovskii method was reviewed in [11]. In [12] and [13] the elements of the B matrix are varied systematically to determine the largest possible estimate for the domain of attraction through this method. However, it was observed that the variation of the elements of matrix B significantly increases the complexity of computations. A literature survey indicates that this method gives consistently conservative results and also this conservatism persists even with the variation of the elements of the B matrix. Moreover, the degree of conservatism increased with an increase in the dimension of the system.

V.I. Zubov [14] has shown that the Lyapunov function which gives an estimate for the domain of attraction of an equilibrium point $x = 0$ for (1) satisfies a certain partial differential equation. If such a partial differential equation can be solved in closed form, then the Lyapunov function obtained uniquely defines the exact domain of attraction of the system. However, this is rarely true since the right-hand side of the partial differential equation is unknown, as observed in [15], [16] and [17]. Thus, a convergent power series solution is used to find the exact domain of attraction. Unfortunately, the convergence of the power series to the exact domain of attraction is not uniform, and also the higher order approximation does not necessarily produce a larger estimate for the domain of attraction. A literature survey indicates that this method requires extensive computations and excessive computer memory. Szegö [18] generalized Zubov's method such that the right-hand side of the differential equation in his method has a well defined form. This generalization of Zubov's method and a certain transformation of variables have been used in [19] to find an estimate for the domain of attraction of power system problems. In [20] and [21], the authors used Lie series to solve the Zubov's partial differential equation and their results indicate a good convergence of solution to the exact domain of attraction if such a domain is closed and bounded. The right-hand side of Zubov's partial differential equation was chosen to be a quadratic function in [22] for a two dimensional system and then a systematic procedure was presented to maximize the size of the domain of attraction by varying the elements of the quadratic function.

The Popov criterion [23] provides sufficient conditions for global asymptotic stability of the equilibrium point of a certain class of nonlinear systems. The nonlinearities $\phi(\sigma)$ in this class satisfy a sector condition of the form $0 < \frac{\phi(\sigma)}{\sigma} < k$ for all $\sigma \neq 0$ and $\phi(0) = 0$. In [23], [24] and [25] some methods have been presented for the determination of the Lurè type of Lyapunov function to prove the Popov criterion. In practice, the nonlinearity $\phi(\sigma)$ satisfies the Popov sector conditions only over a finite or semi-infinite range of the argument and thus it is of importance to find an estimate for the domain of attraction of the equilibrium point. In [26],

[27] and [28] an estimate for the domain of attraction was obtained by utilizing the Luré type of Lyapunov function. Such Lyapunov functions are extensively used in the stability analysis of power systems. For a comprehensive survey on this subject, refer to [29] and [30].

In [31] a systematic algorithm using a quadratic Lyapunov function to find an estimate for the domain of attraction is presented and then a method is given to maximize the size of such an estimate. The behavior of optimal k-th degree and 2k-th degree quadratic Lyapunov functions was presented in [32]. The algorithm of [31] has been used in [33] and [34] for the power system stability analysis and also the computational difficulties encountered in the adaptation of the algorithm have been discussed.

Most of the existing algorithms to determine an estimate for the domain of attraction were developed before 1975. In a 1975 review paper [29] on the application of Lyapunov stability theory to power system transient stability analysis, the existing computational difficulties and also the need for continued research to develop systematic and fast algorithms to estimate greater stability regions was emphasized. With the ever-increasing size and complexity of systems, this task assumes particular significance. In the next two chapters, two systematic and fast algorithms to find an estimate for the domain of attraction will be presented.

IV. ANALYSIS BY QUADRATIC LYAPUNOV FUNCTIONS

In this chapter quadratic Lyapunov functions are used to determine estimates for the domain of attraction of the equilibrium $x = 0$ for (1), and an optimization procedure is employed to maximize the size of such estimates. Existing results which utilize quadratic Lyapunov functions, and which differ significantly from the algorithm presented herein, are given in [31] and [32]. This chapter consists of five sections. In section A, the algorithm is developed, in section B the algorithm is summarized, in section C the results obtained by the present algorithm are compared with the existing ones, and in section D several specific examples are presented and discussed. In section E, the effects of changes in the eigenvalues of the Jacobian matrix on the estimated domain of attraction are discussed.

A. Development of the Algorithm

We begin by choosing in particular a quadractic Lyapunov function of the form

$$v_0(x) = x^T B_0 x, \qquad B_0 = B_0^T \tag{10}$$

where B_0 is a real, positive definite $n \times n$ matrix. Let Ω_0 be some domain with $x = 0$ in its interior. Suppose that

$$\dot{v}_{0(1)}(x) = 2f^T(x)\, B_0 x < 0$$

for all $x \in \Omega_0$ and $x \neq 0$, and

$$\dot{v}_{0(1)}(0) = 0,$$

and in addition, suppose that $d_0 > 0$ is the largest value for which the set D_0 determined by

$$D_0 = \left\{x \in E^n : v_0(x) \leq d_0\right\}$$

is contained in Ω_0. Then clearly, D_0 is contained in the domain of attraction of the equilibrium $x = 0$ for (1).

In order to find a good estimate of the domain of attraction of $x = 0$ for (1) via quadratic Lyapunov functions, using the <u>initial choice</u> v_0 given in (10), we seek positive definite $n \times n$ matrices B_i, $i = 1, 2, \ldots$, such that the sets D_i given by

$$D_i = \left\{x \in E^n : v_i(x) = x^T B_i x \leq d_i,\ \ \dot{v}_{i(1)}(x) = 2f^T(x) B_i x < 0 \right.$$
$$\left. \text{for all } x \neq 0 \text{ and } \dot{v}_{i(1)}(0) = 0 \right\} \tag{11}$$

satisfy the condition

$$D_0 \subseteq D_1 \subseteq D_2 \subseteq \cdots$$

where in (11) the largest possible d_i are chosen. The set of matrices B_i will be found by using an optimization procedure based on a direct search method. To employ an optimization procedure we need an optimization functional and this functional will be established next, making use of certain known facts.

Recall that if $B = B^T = [b_{ij}]$ is a positive definite $n \times n$ matrix, then for fixed $d > 0$, the equation $v(x) = x^T Bx = d$ will determine an ellipsoid in E^n, and all the eigenvalues of B, $\lambda_1(B), \ldots, \lambda_n(B)$, are real and positive. Furthermore, it follows from the principal-axis theorem (see [35, p. 120]) that in this case the k-th principal axis of the ellipsoid is equal to $\sqrt{d/\lambda_k(B)}$, $k = 1, \ldots, n$. Thus, the hypervolume enclosed by the ellipsoid $x^T Bx = d$ is proportional to

$$1 \Big/ \left[\prod_{i=1}^{n} \lambda_i(B)\right] .$$

This suggests the use of certain types of functions, specified below, in an optimization procedure to determine a matrix B which yields the maximum volume enclosed by the ellipsoid $x^T Bx = d$, subject to certain constraints to be specified later. Let

$$\alpha_1(B) = \sum_{i=1}^{n} \lambda_i(B) \tag{12}$$

and

$$\alpha_2(B) = \prod_{i=1}^{n} \lambda_i(B). \tag{13}$$

The functions $\alpha_1(B)$ and/or $\alpha_2(B)$ can be used as functionals in optimization procedures to maximize the volume enclosed by $v(x) = x^T Bx = d$. It turns out that the minimization procedure involving (12) results in a uniform minimization of all eigenvalues of B, while a minimization procedure involving (13) emphasizes the minimization of the smallest eigenvalue of B.

Since the minimization of $\alpha_1(B)$ is computationally more efficient than the minimization of $\alpha_2(B)$, we will in the present algorithm always first obtain an <u>initial</u> <u>estimate</u> for the domain of attraction by minimizing $\alpha_1(B)$. However, when the trace of B contains elements with disproportionate values, the eccentricity of the ellipsoid determined by $v(x) = x^T Bx$ is large. This is also true for the contours in the parameter space (for the elements b_{ij} of B), determined by $\alpha_1(B) = c$, respectively, $\alpha_2(B) = c$ (see [36, pp. 53-55]). In such cases, the minimization process of $\alpha_1(B)$ is very inefficient after a certain number of iterations and has to be terminated, since the step sizes (in the parameter space for B) become exceedingly small and no further significant progress in the minimization of $\alpha_1(B)$ can be realized. Experience has shown (refer to the examples given in section D of this chapter) that under such circumstances, the <u>estimate</u> for the domain of attraction can be improved significantly by restarting the optimization process by minimizing $\alpha_2(B)$ in place of $\alpha_1(B)$ in the remaining iterations. The reason for such an improvement lies in the fact that usually, the step sizes of the algorithm in the parameter space will be increased after the restart, and also, the orientation and eccentricity of the final ellipsoid, which determines the estimate, is such as to increase the final estimate for the domain of attraction (refer to the figures of the examples in section D of this chapter). Note that since the Lyapunov function $v(x) = x^T Bx$ is required to be positive definite, we must satisfy the constraints

$$\lambda_i(B) > 0, \qquad i = 1, \ldots, n. \tag{14}$$

To obtain the <u>initial Lyapunov function</u> $v_0(x) = x^T B_0 x$, we make use of the Jacobian matrix $J(x) = [\partial f(x)/\partial x]$ of f, evaluated at $x = 0$. Specifically, we obtain B_0 by solving the Lyapunov matrix equation for B_0

$$J(0)^T B_0 + B_0 J(0) = -C,$$

where C is a symmetric, positive definite, arbitrarily chosen matrix. In practice, we usually choose $C = I$, and we solve the equation

$$J(0)^T B_0 + B_0 J(0) = -I$$

for B_0, where I denotes the $n \times n$ identity matrix.

Using Rodden's method (see [37]), we now find the points in E^n such that

$$\left.\begin{aligned} &\dot{v}_{0(1)}(x) = 0 \\ &\frac{\nabla v_0(x)}{|\nabla v_0(x)|} - \left(\frac{\nabla v_0^T(x)}{|\nabla v_0(x)|} \, \frac{\nabla \dot{v}_{0(1)}(x)}{|\nabla \dot{v}_{0(1)}(x)|} \right) \frac{\nabla \dot{v}_{0(1)}(x)}{|\nabla \dot{v}_{0(1)}(x)|} = 0. \end{aligned}\right\} \quad (15)$$

Let $x^1, x^2, \ldots, x^{\ell_T} \in E^n$ denote all points which satisfy (15) and let

$$d_0 = \min_{j=1, \ldots, \ell_T} \left\{ v_0(x^j) \right\} .$$

Suppose that $x^t = x^k$ is a point from $\left\{x^1, \ldots, x^{\ell_T}\right\}$ for which $v_0(x)$ attains its minimum d_0. We will call such a point a <u>tangent point</u> between the locus of points determined by $\dot{v}_{0(1)}(x) = 0$ and the locus of points determined by $v_0(x) = d_0$. (A typical example of a tangent point, when

$n = 2$, is shown in Figure 1.) The estimated domain of attraction of $x = 0$ for (1), using v_0, is now

$$D_0 = \left\{ x \in E^n : v_0(x) = x^T B_0 x \leq d_0 \right\} .$$

To improve this estimate (making use of quadratic Lyapunov functions), we now rephrase our problem in terms of a parameter optimization problem which we state next: given B_0, find positive definite $n \times n$ matrices B_k, $k = 1, 2, \ldots$, so as to minimize $\alpha_1(B_k)$, respectively, $\alpha_2(B_k)$, subject to the constraints

$$\left.\begin{array}{ll} \text{(a)} & \lambda_i(B_k) > 0, \qquad\qquad i = 1, \ldots, n \\ \text{(b)} & -2f^T(x)B_k x > 0, \quad x \in \partial D_{k-1} = \left\{ x \in E^n : x^T B_{k-1} x = d_{k-1} \right\} \\ & \text{and} \quad x \in \partial D_k = \left\{ x \in E^n : x^T B_k x = d_k \right\}, \; k = 1, 2, \ldots . \end{array}\right\} \quad (16)$$

Rosenbrock's constrained optimization algorithm (see [38] and [39, pp. 386-396]) which provides an acceleration in the direction as well as in the distance of search was found to be well-suited for the implementation of the above parameter minimization problem. In Appendix A, Rosenbrock's optimization algorithm is summarized.

The functionals α_1 and α_2 given by (12) and (13), respectively, and the positive definite constraints given in (14), are phrased in terms of the eigenvalues $\lambda_1, \ldots, \lambda_n$ of B. The explicit evaluation of these eigenvalues of each optimization search point is very costly, especially in high dimensional problems, and it can lead to accuracy problems as well.

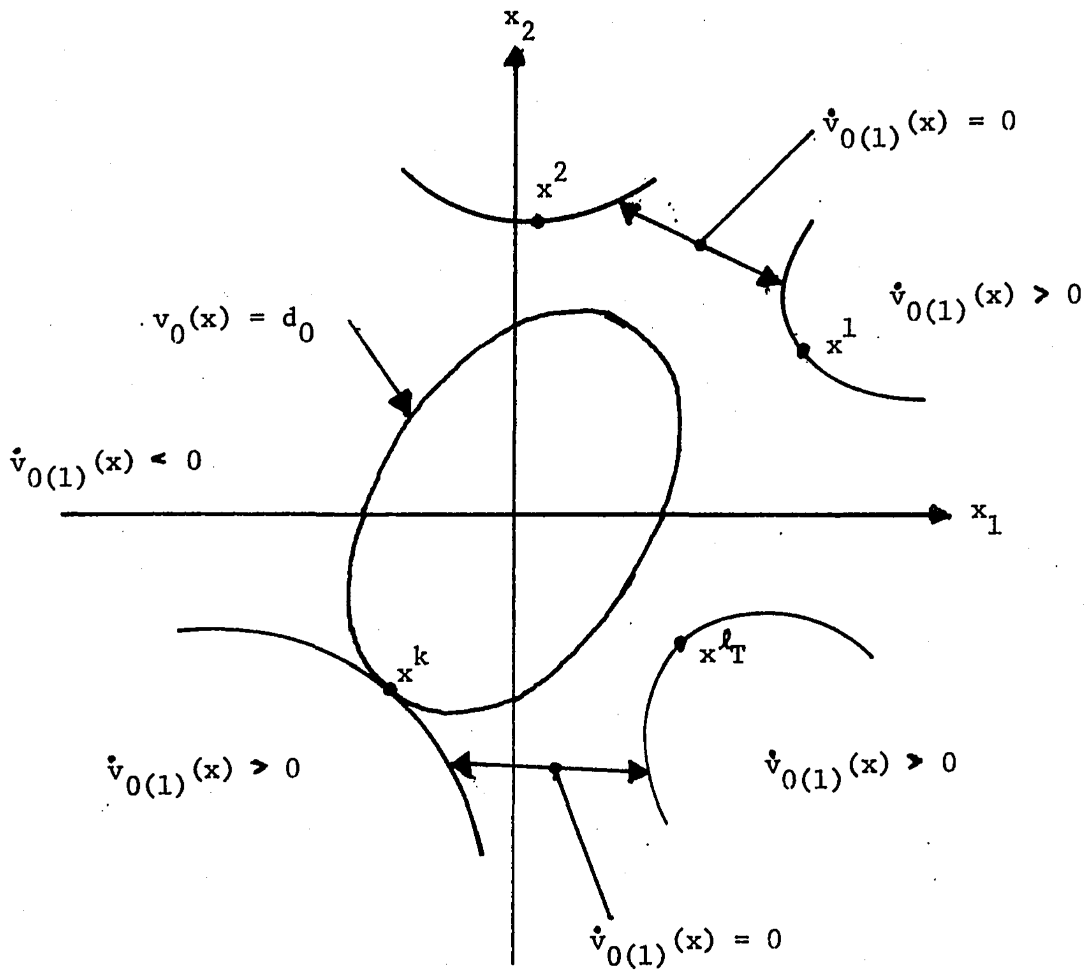


Figure 1. Example of a tangent point $x^t = x^k$

These difficulties can be circumvented by noting that conditions (12), (13) and (14) can be replaced by the equivalent conditions

$$\alpha_1(B) = \text{trace } B \tag{17}$$

$$\alpha_2(B) = \det B \tag{18}$$

$$\det B_i > 0, \qquad i = 1, \ldots, n, \tag{19}$$

respectively, where $\det B_i$ denotes the i-th principal minor of B. It was precisely the *simplicity* and *efficiency* of computation of these quantities which motivated the choice of the functionals α_1 and α_2 in the first place.

As seen in (16), we need to check the negative definiteness of $\dot{v}_{k(1)}(x)=2f^T(x)\, B_k x$ over the boundaries of D_{k-1} and D_k. Since f is assumed to be smooth, it suffices to make this check at a sufficiently fine grid on ∂D_{k-1} and ∂D_k. A method which was found to be very useful in generating such a grid will be presented next. (Our discussion is phrased in terms of the boundary ∂D_0. The procedure for ∂D_k is identical.) We employ the relation

$$x = \sum_{i=1}^{n} \beta_i x_i, \quad \beta_i \geq 0, \quad \sum_{i=1}^{n} \beta_i = 1. \tag{20}$$

For discussion purposes, we let in the following n = 2. The procedure for n > 2 involves obvious modifications. We choose the points $x_1 = (1,0)$ and $x_2 = (0,1)$. Then (20) assumes the form

$$x = \beta_1 x_1 + (1 - \beta_1)x_2, \qquad 0 \leq \beta_1 \leq 1. \tag{21}$$

Choosing appropriate values for β_1, we can now construct any desired grid on the line $x_1 + x_2 = 1$ in the first quadrant, as shown in Figure 2. By using proper signs for the other quadrants, we can similarly set up appropriate grids for the lines in the remaining quadrants, as indicated in Figure 2. Noting that the quadratic function $v_0(x) = x^T B_0 x$ is homogeneous, we have for any constant $c \geq 0$,

$$v(cx) = c^2 v(x). \tag{22}$$

From this it follows that for any point determined by (21) (respectively by (20)), say x'_a, as shown in Figure 2, there is a point $x_a = c \cdot x'_a$ on the locus of points determined by $v_0(x) = d_0$, where $c = \left(d_0 / v_0(x'_a) \right)^{1/2}$. Let the number of points on the locus $v_0(x) = d_0$ generated in this way be equal to m. To ensure that the region determined by $v_0(x) \leq d_0$ belongs to the domain of attraction, we add to the above grid points, x^t, as well as their negative values, $-x^t$. (The latter are added, since frequently there is a symmetry about the origin for the points determined by $\dot{v}_{0(1)}(x) = 0$ and $v_0(x) = d_0$.) Thus, we end up with a total of $\ell \geq m + 2$ points for the grid of the locus $v_0(x) = d_0$.

It turns out that in actual computer implementations, $\dot{v}_{0(1)}(x)$ evaluated at a tangent point x^t may be zero, or slightly positive, or slightly negative. To ensure that $\dot{v}_{0(1)}(x)$ be negative definite over the region determined by $v_0(x) \leq d_0$, it may be necessary to somewhat reduce the constant c in (22). This is verified by means of a search method.

Finally, the constrained optimization procedure used in the present algorithm to find the best possible estimate of the domain of attraction,

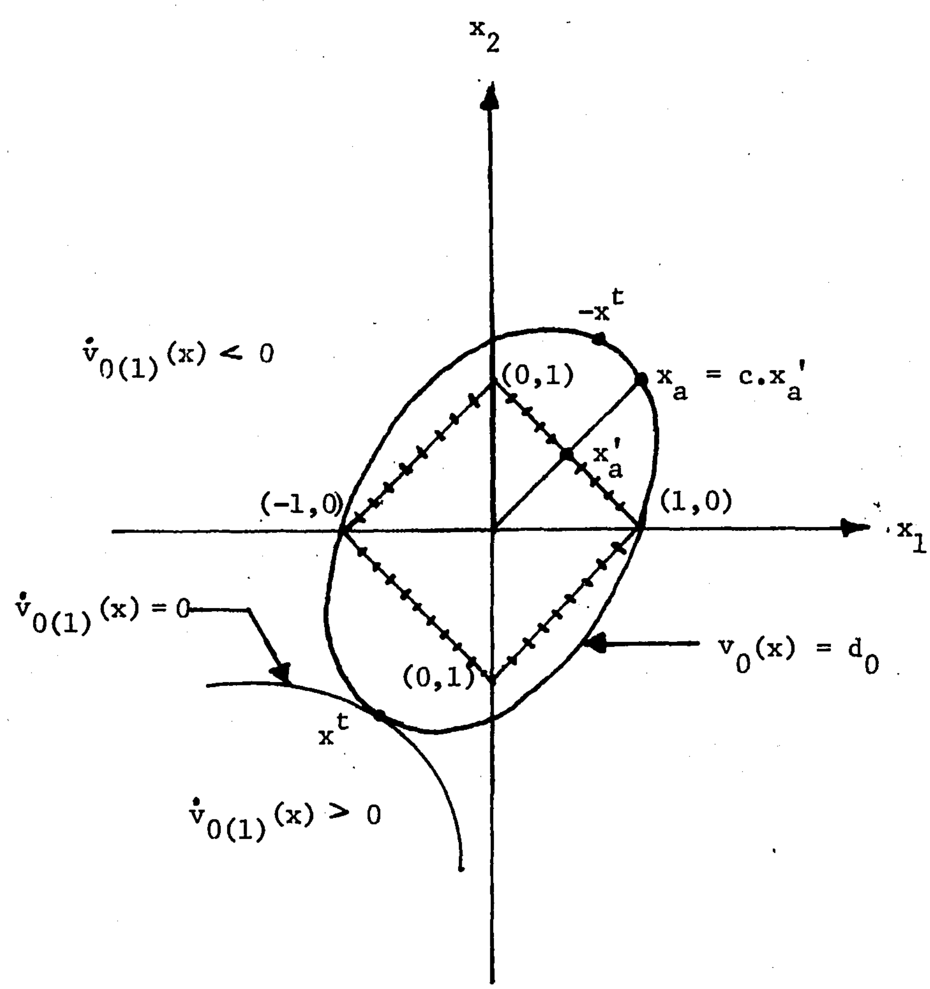


Figure 2. Generation of a grid on the locus $v_0(x) = d_0$

is terminated when any one of the following three convergence conditions is satisfied.

(c-1) The number of search points exceeds a prespecified number, say $50n(n+1)/2$.

(c-2) When minimizing $\alpha_1(B)$ = trace B, if the completed stages in Rosenbrock's optimization procedure is greater or equal to s_{max} (usually $s_{max} = 2$ when $n = 2$), then minimize $\alpha_2(B)$ = det B.

(c-3) If no improvement can be realized for the domain of attraction after $14n(n+1)/2$ search points over a preceding successful search point.

Condition (c-1) is based on the computational experience of Rosenbrock (see [36, pp. 67-68]). For 13 specific examples, which will be presented later, an estimate for the domain of attraction was obtained solely by minimizing the functional $\alpha_1(B)$. Analysis of these results indicates that in most of the problems significant improvement was achieved within two stages (for $n = 2$) only. Also, we know that the minimization of functional $\alpha_2(B)$ = det (B) will increase the eccentricity of the final estimate and hence the condition (c-2) was used. The condition (c-3) was used to terminate the minimization procedure when the step sizes become too small. In the beginning of the algorithm the step sizes are set to 0.05 and if a search point is unsuccessful, then the step sizes are reduced by a factor of 0.5. If there are $14n(n+1)/2$ consecutive failures in all directions then the step sizes become so small (usually 3×10^{-6} or less) that the procedure may not be able to recover, and hence condition (c-3) is used.

B. Summary of the Algorithm

We now summarize the preceding discussion in the following algorithm:

Step 1: Evaluate $J(0) = \left(\frac{\partial f}{\partial x}(x)\right)\Big|_{x=0}$ and solve for B_0 using the matrix Lyapunov equation $J(0)^T B_0 + B_0 J(0) = -I$. This yields the initial Lyapunov function $v_0(x) = x^T B_0 x$.

Step 2: Using Rodden's method, find all points in E^n satisfying $\dot{v}_{0(1)}(x) = 0$, and

$$\frac{\nabla v_0(x)}{|\nabla v_0(x)|} - \left(\frac{\nabla v_0(x)^T}{|\nabla v_0(x)|} \frac{\nabla \dot{v}_{0(1)}(x)}{|\nabla \dot{v}_{0(1)}(x)|}\right) \frac{\nabla \dot{v}_0(x)}{|\nabla \dot{v}_{0(1)}(x)|} = 0.$$

Let $\{x^1, \ldots, x^{\ell_T}\}$ denote all such points and evaluate $d_0 = \min_{j=1, \ldots, \ell_T} \{v_0(x^j)\}$. Determine from these points the tangent points, x^t, and their negative values, $-x^t$.

Step 3: Determine m grid points for the locus determined by $v_0(x) = d_0$ and add to these all tangent points, x^t, and the corresponding points, $-x^t$. This yields a total of ℓ points.

Step 4: Using a direct search method, find as large of a constant c as possible, $0 < c \leq 1.0$, such that for all $j = 1, \ldots, \ell$, $\dot{v}_{0(1)}(cx^j) < 0$, $j = 1, \ldots, \ell$. Set d_0 equal to $c^2 d_0$ and set x^j equal to cx^j, $j = 1, \ldots, \ell$. The initial domain of attraction, D_0, is now given by

$$D_0 = \{x \in E^n : v_0(x) = x^T B_0 x \leq d_0\}.$$

Step 5: Starting with the initial matrix B_0, use Rosenbrock's constrained parameter optimization algorithm to find a final matrix B_M so as to minimize trace B_k, subject to the following constraints:

(i) $\det(B_k)_i > 0, \quad i = 1, \ldots, n; \; k = 1, \ldots, M;$

(ii) $-2f(x^j)^T B_k x^j > 0, \; x^j \in \partial D_{k-1} = \{x^j \in E^n : (x^j)^T B_{k-1} x^j = d_{k-1}\},$

(iii) $-2f(cx^j)^T B_k (cx^j) > 0, \; cx^j \in \partial D_k = \{cx^j \in E^n : (cx^j)^T B_k (cx^j) = d_k\}$, for some $c \geq 1.0$ and $D_{k-1} \subseteq D_k$, $k = 1, \ldots, M$.

If the convergence criterion (c-3) is met, exit.

If the convergence criterion (c-2) is met, go to Step 6.

If not, continue with Step 5.

Step 6: Starting with the initial matrix B_M, use Rosenbrock's algorithm to find a final matrix B_F so as to minimize $\det(B_k)$, subject to the following constraints:

(i) $\det(B_k)_i > 0, \; i = 1, \ldots, n; \; k = M+1, \ldots, F-1, F.$

(ii) $-2f(x^j)^T B_k x^j > 0, \; x^j \in \partial D_{k-1} = \{x^j \in E^n : (x^j)^T B_{k-1} x^j = d_{k-1}\}$, $k = M+1, \ldots, F.$

(iii) $-2f(cx^j)^T B_k (cx^j) > 0, \; cx^j \in \partial D_k = \{cx^j \in E^n : (cx^j)^T B_k (cx^j) = d_k\}$, for some $c \geq 1.0$ and $D_{k-1} \subseteq D_k$, $k = M+1, \ldots, F.$

If any of the convergence criteria (c-1), (c-3) are met, exit.

If not, continue with Step 6.

C. Discussion

The present algorithm was motivated by the work of Davison and Kurak [31]; however, their algorithm differs significantly from the present one, and in the following, the two methods will be contrasted.

In [31], use is made of quadratic Lyapunov functions of the form $x^T Bx$. First, the maximum eigenvalue of B is minimized (to reduce the eccentricity of the locus $v(x) = x^T Bx = d$), and then the product of the eigenvalues of B is minimized to obtain the best possible estimate of the domain of attraction of $x = 0$ for (1). At every optimization search point (using Rosenbrock's procedure), the eigenvalues of B are evaluated explicitly by means of the Jacobi transformation method (see [40] and [35, p. 127]), in order to compute the minimization functional and to check the constraints on the eigenvalues. In [31] and [35, p. 127], it was concluded that the required amount of computation time to implement the Jacobi transformation method is proportional to n^3, where n is the order of the system. When the accuracy of the eigenvalues is critical (as is usually the case), the required amount of computation time can increase significantly. Also, the accuracy of the computed eigenvalues can affect the functional to be minimized. In [35, p. 127], it was also observed that the approximate number of multiplications and divisions required by the Jacobi transformation method is equal to $2n^3 \log (^1/\varepsilon)$, where ε = (sum of squares of off diagonal terms in matrix B at the k-th rotation/sum of squares of off-diagonal terms in matrix B at the beginning of the algorithm), and it is usually equal to 1.0×10^{-5}. Hence, the total number of multiplications and divisions required by the Jacobi transformation method is approximately $10n^3$. In the present algorithm the principal minor determinants are computed by the Gaussian elimination procedure and then the sign definiteness of such principal minor determinants determines the sign definiteness of the eigenvalues of matrix B. The total amount of multiplications and divisions required by this method (Gaussian elimination) is $(n^3 - n)/2$. This alternative method does not encounter any accuracy problems. For small n, the computation time required by the alternative method increases approximately in a linear fashion, and for the large n the alternative method is approximately 20 times faster than the Jacobi transformation method.

In [31], N_n directional vectors are used to check the negative definiteness of $\dot{v}_{(1)}(x)$ by employing p points along every directional vector and by using these $p \cdot N_n$ points as a grid to span an appropriate region in E^n containing the origin. For example, in the case of two-dimensional systems, 200 points are used in [31] to form a "regular" grid at each iteration of the optimization procedure. At the end of the optimization procedure, a "fine" grid is used in [31] to verify that $\dot{v}_{(1)}(x)$ is actually negative definite in the estimated domain of attraction. If it turns out that $\dot{v}_{(1)}(x)$ is not negative definite on the fine grid, then the number of points in the original regular grid is increased by increasing N_n (the number of directional vectors) and p (the number of points along every direction vector) and the optimization procedure is repeated. In generating the grids in [31], the user must make an a priori guess of the extent of the domain of attraction. Also, if the eccentricity of the ellipsoid represented by $x^T Bx = 1$ in [31] is large then the regular grid may be too coarse and one might incorrectly conclude that $\dot{v}_{(1)}(x)$ is negative definite in the region $\{x \varepsilon E^2 : v(x) \leq d\}$ as shown in Figure 3. Such difficulties seem to have arisen in one of the examples given in [34]. Difficulties along such lines were not encountered by the present algorithm.

The algorithm proposed by Bingulac [41] (Appendix C) was used to expand the Lyapunov matrix equation (encountered in step 1 of the present

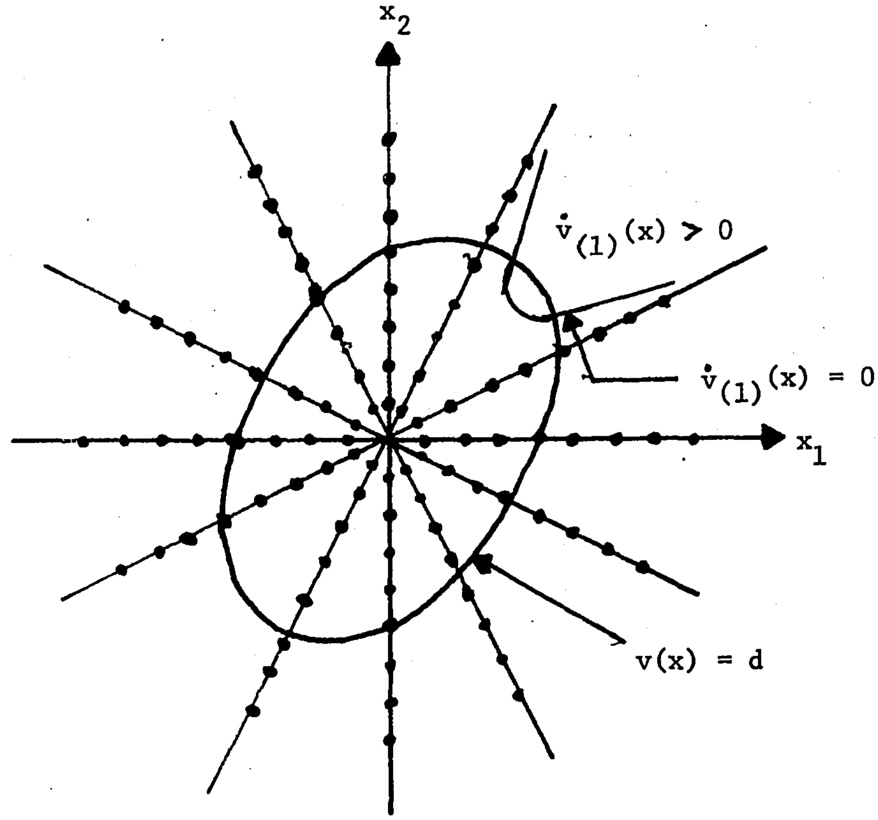


Figure 3. An example of incorrect estimation of domain of attraction encountered in [34] using the algorithm of [31]

algorithm) into a system of linear algebraic equations and then the LEQT2F subroutine available in IMSL subroutine package was used to solve the algebraic equations to determine the Lyapunov matrix. In step 2 of the present algorithm, the subroutine ZSYSTM was used to solve the simultaneous nonlinear equations.

D. Examples

The present algorithm was coded in Fortran WATFIV on the ITEL AS/6 system. Thirteen specific examples were considered, including examples from [31], [32], [34], as well as some examples from [3] and [42], and some others. The average CPU time required for these examples is 3.03 seconds. A comparison of the domains of attraction obtained by the algorithm of [31] with those obtained by the present method, for the examples provided in [31], was made. The estimates obtained by these two methods seem to be essentially the same. However, for the reasons given above, the present method appears to be significantly more efficient than the method of [31].

To demonstrate the applicability of the present algorithm to specific cases, we consider the following specific examples:

Example 1

$$\dot{x}_1 = -x_2$$

$$\dot{x}_2 = x_1 - x_2(1 - x_1^2)$$

Example 2

$$\dot{x}_1 = -x_1 + 2x_1^2 x_2$$

$$\dot{x}_2 = -x_2$$

Example 3

$$\dot{x}_1 = x_1(x_1^2 + x_2^2 - 1) - x_2$$

$$\dot{x}_2 = x_1 + x_2(x_1^2 + x_2^2 - 1)$$

Example 4

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = -2x_1 - 0.5x_2 - x_1^2$$

Example 5

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = 0.301 - \sin(x_1 + 0.4136) + 0.138 \sin 2(x_1 + 0.4136) - 0.279x_2$$

Example 6

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = 0.234 - 0.0633 \sin(x_1 + 0.0405) - 0.582 \sin(x_1 + 0.4103) - 0.07143x_2$$

Example 7

$$\dot{x}_1 = -2x_1(1 - x_1) + 0.1x_1x_2$$

$$\dot{x}_2 = -2x_2(9 - x_2) + 0.1(x_1 + x_2)$$

Example 8

$$\dot{x}_1 = -2x_1 + x_1x_2$$

$$\dot{x}_2 = -x_2 + x_1x_2$$

Example 9

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = -x_1 - x_2 + x_1^3$$

<u>Example 10</u>

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = -x_1 - 4x_2 + 1/4(x_2 - 0.5x_1)(x_2 - 2x_1)(x_2 + 2x_1)(x_2 + x_1)$$

<u>Example 11</u>

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = -x_1(1 - x_1^2) - x_2(1 - x_2^2)$$

<u>Example 12</u>

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = -x_1(1 + x_2) - x_2(1 - x_2^2)$$

<u>Example 13</u>

$$\dot{x}_1 = 6x_2 - 2x_2^2$$

$$\dot{x}_2 = -10x_1 - x_2 + 4x_1^2 + 2x_1x_2 + 4x_2^2$$

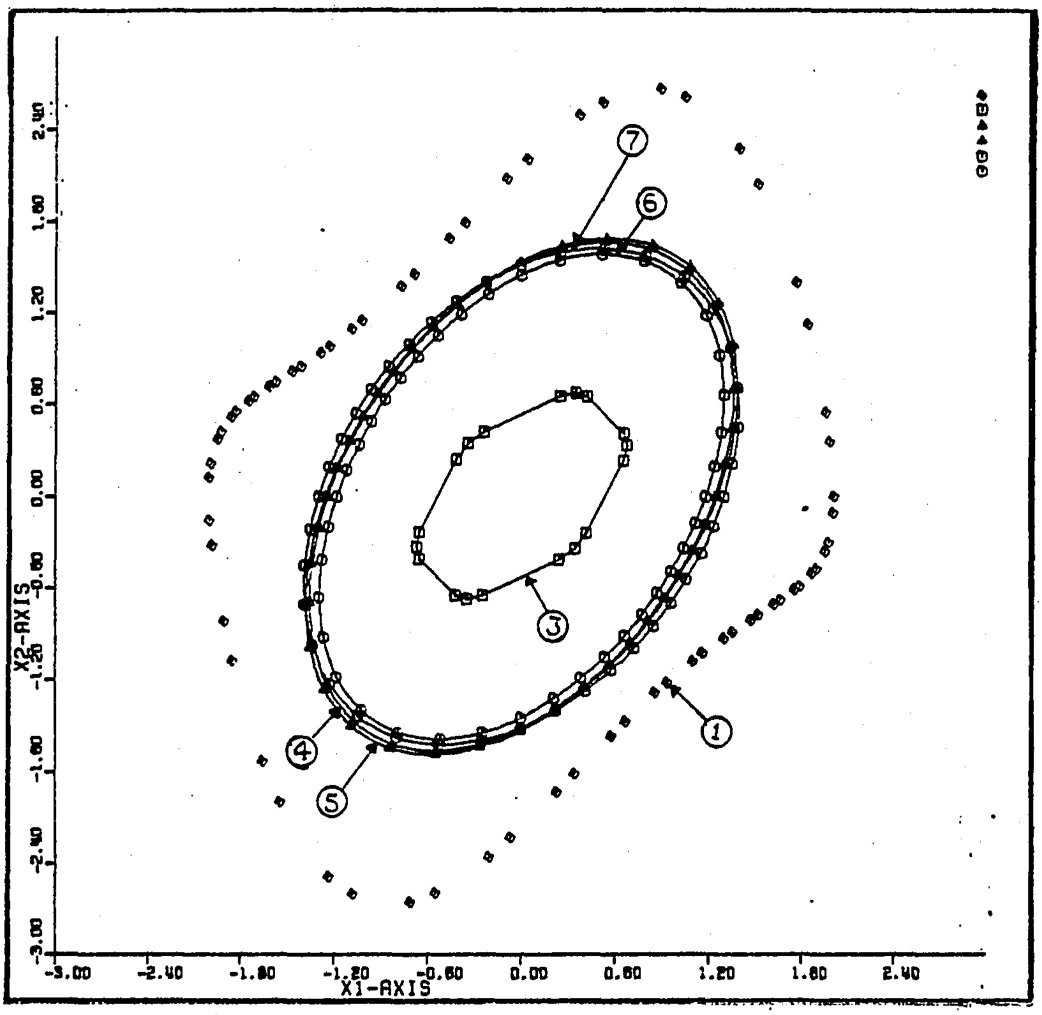
Note that all of these examples have an isolated equilibrium at the origin $x^T = (x_1, x_2) = 0$.

For all of these examples, the estimates for the domains of attraction were obtained by the present method and also by the norm Lyapunov function method. The latter method will be discussed in the next chapter. These estimates are depicted in Figures 4 through 16. In all of these figures, the initial estimate used for the present method, the final estimate obtained by the present method, the estimate obtained through the norm Lyapunov function, and some points of the actual stability boundary obtained by integrating the system equations will be presented.

Figure 4. Estimates of domain of attraction for Example 1

Legend for Figures 4 through 16:

1. Points in actual stability boundary (obtained by Runge-Kutta integration of the system differential equations).
2. Points outside the domain of attraction (obtained by Runge-Kutta integration of the system differential equations).
3. Estimate of the domain of attraction obtained via the Norm Lyapunov Function Algorithm (Chapter V).
4. Initial estimate of the domain of attraction obtained via the Quadratic Lyapunov Function Algorithm (Chapter IV).
5. Final estimate of the domain of attraction via the Quadratic Lyapunov Function Algorithm (Chapter IV).
6. Initial estimate of the domain of attraction obtained via the Quadratic Lyapunov Function Algorithm of reference [31].
7. Final estimate of the domain of attraction obtained via the Quadratic Lyapunov Function Algorithm of reference [31].


X2-AXIS
X1-AXIS
-3.00
-2.40
-1.80
-1.20
-0.60
0.00
0.60
1.20
1.80
2.40
1
3
4
5
6
7

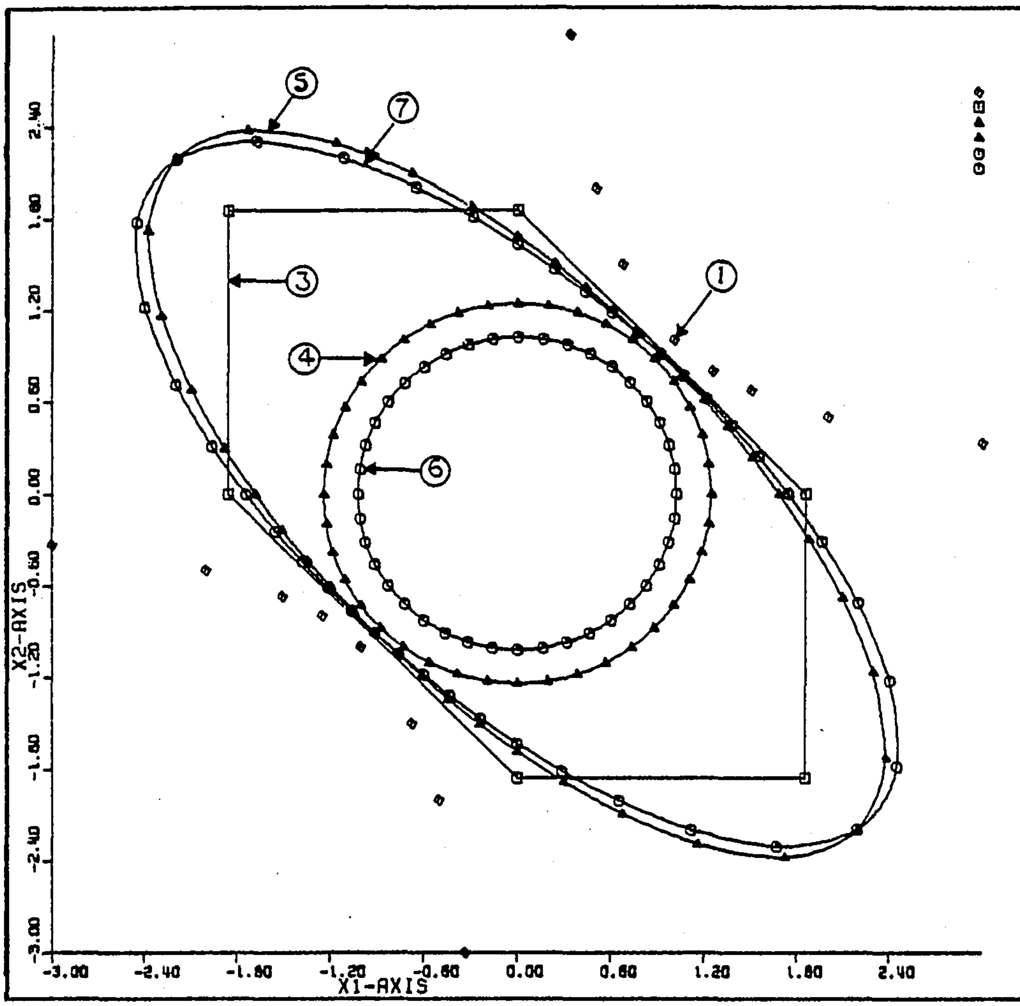


Figure 5. Estimates of domain of attraction for Example 2

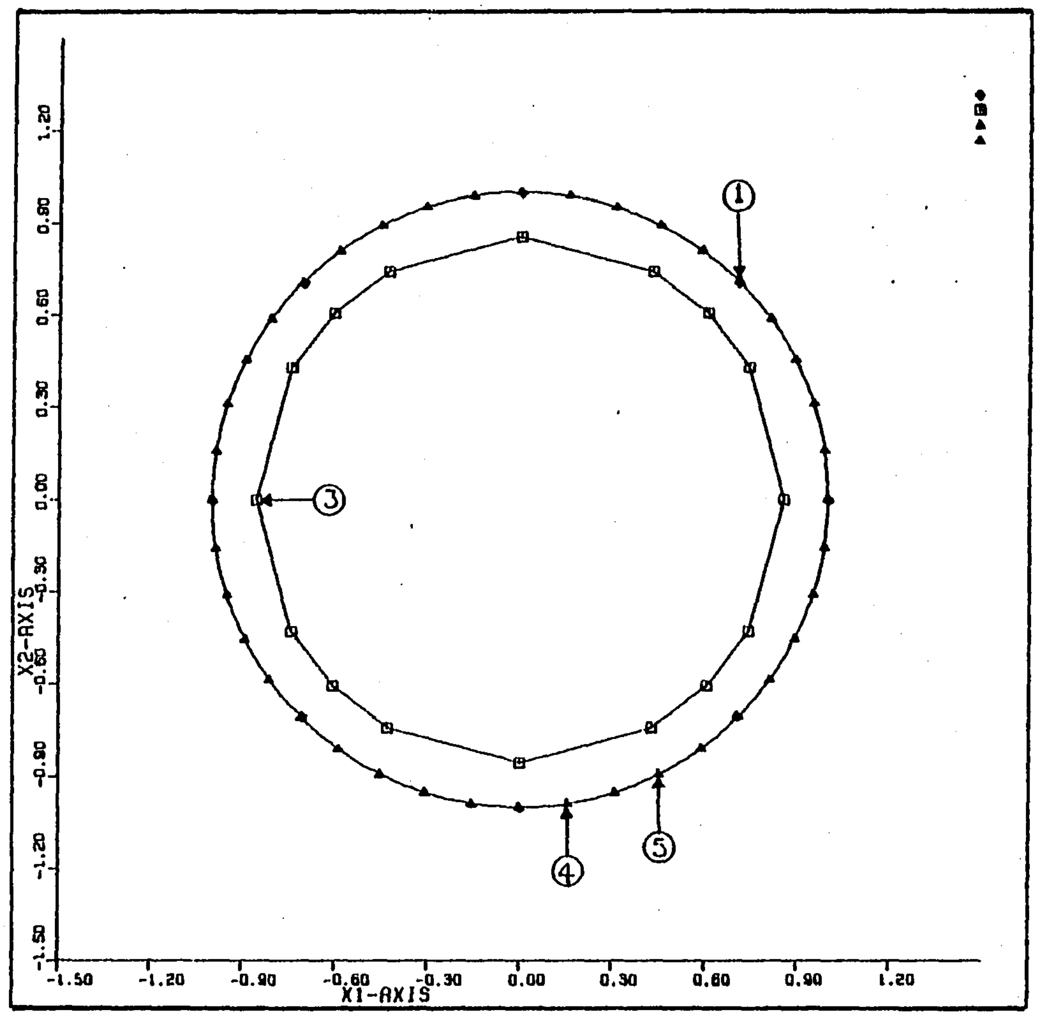


Figure 6. Estimates of domain of attraction for Example 3

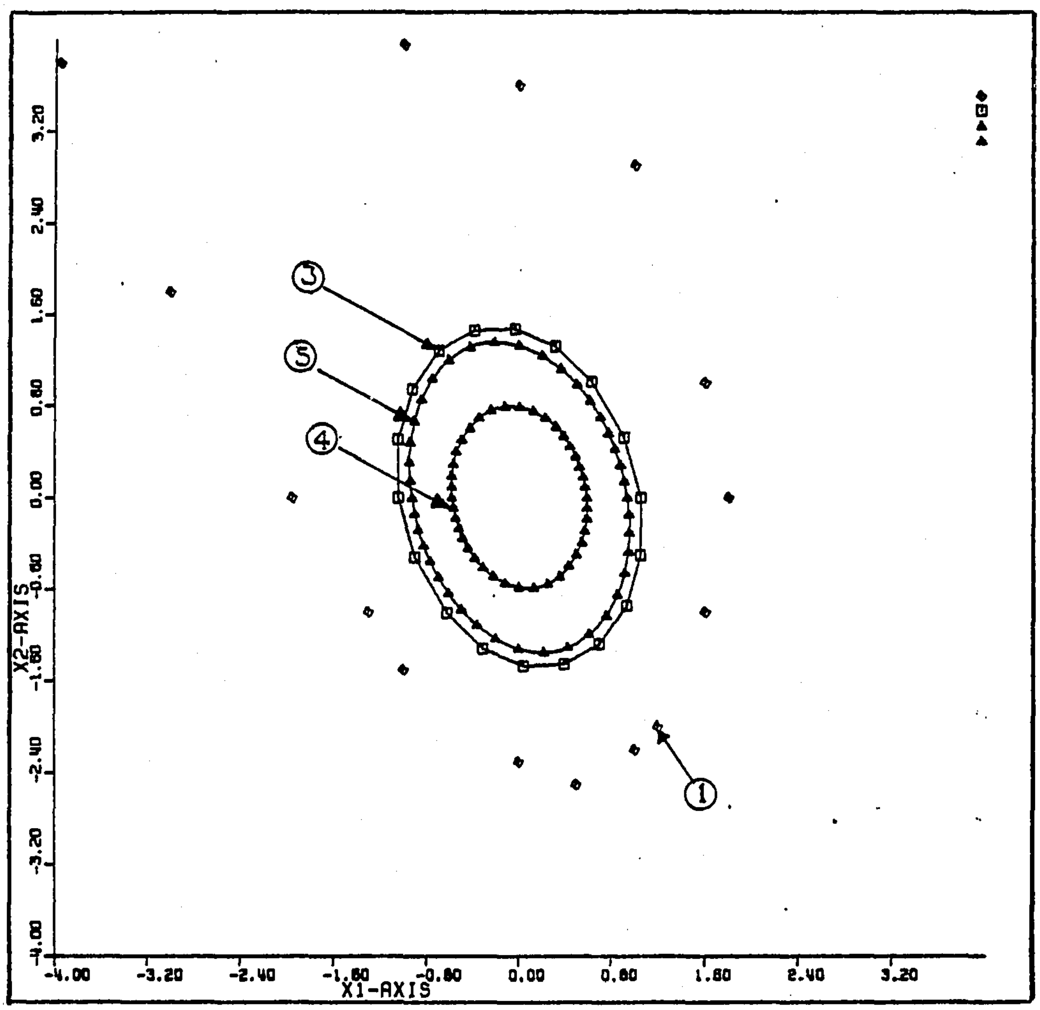


Figure 7. Estimates of domain of attraction for Example 4

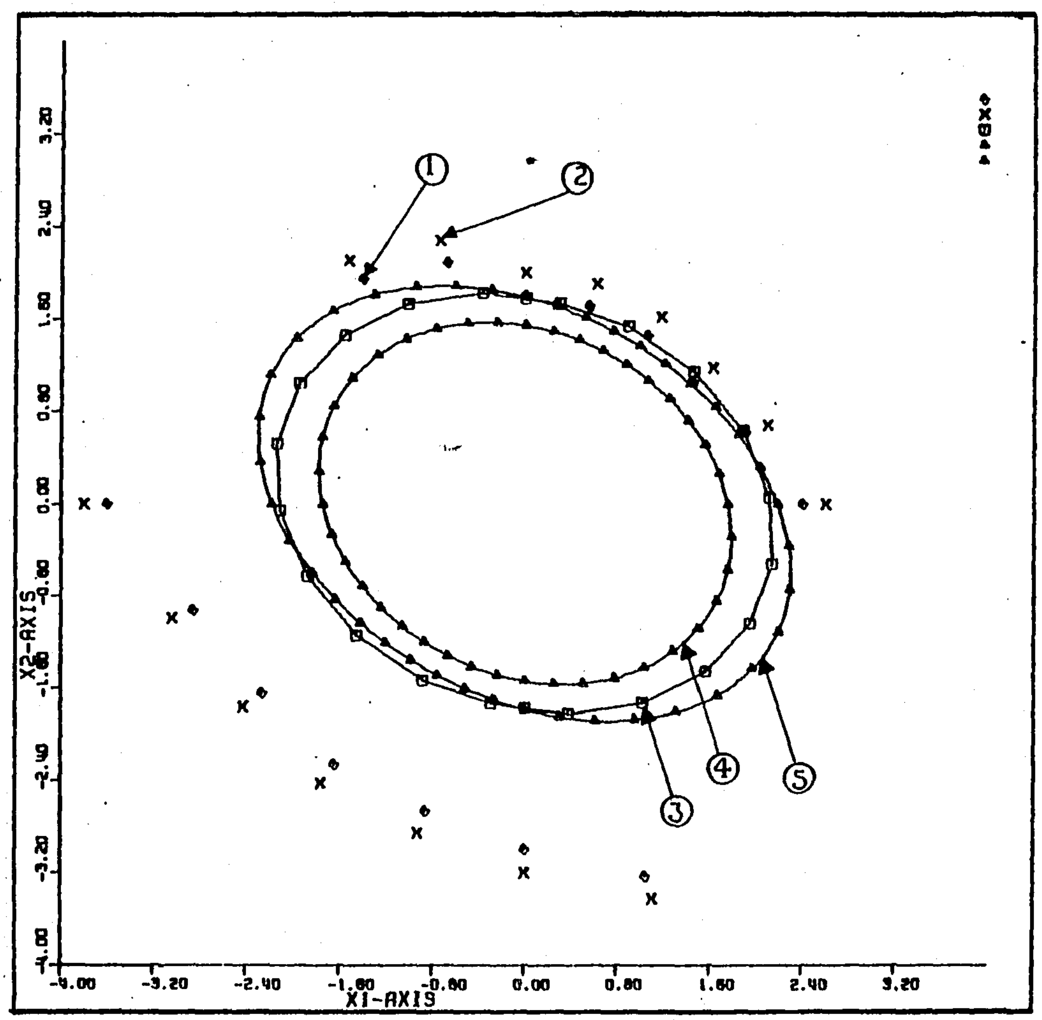


Figure 8. Estimates of domain of attraction for Example 5

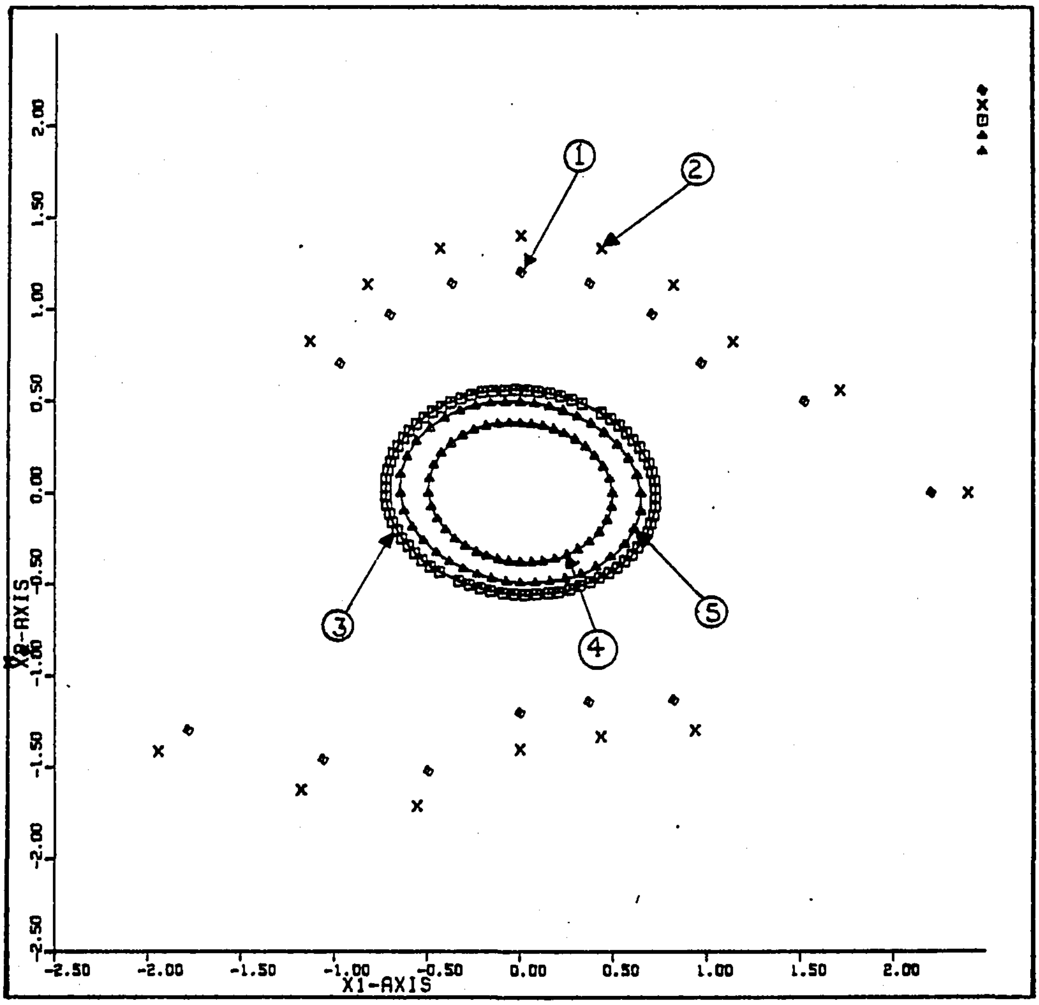


Figure 9. Estimates of domain of attraction for Example 6

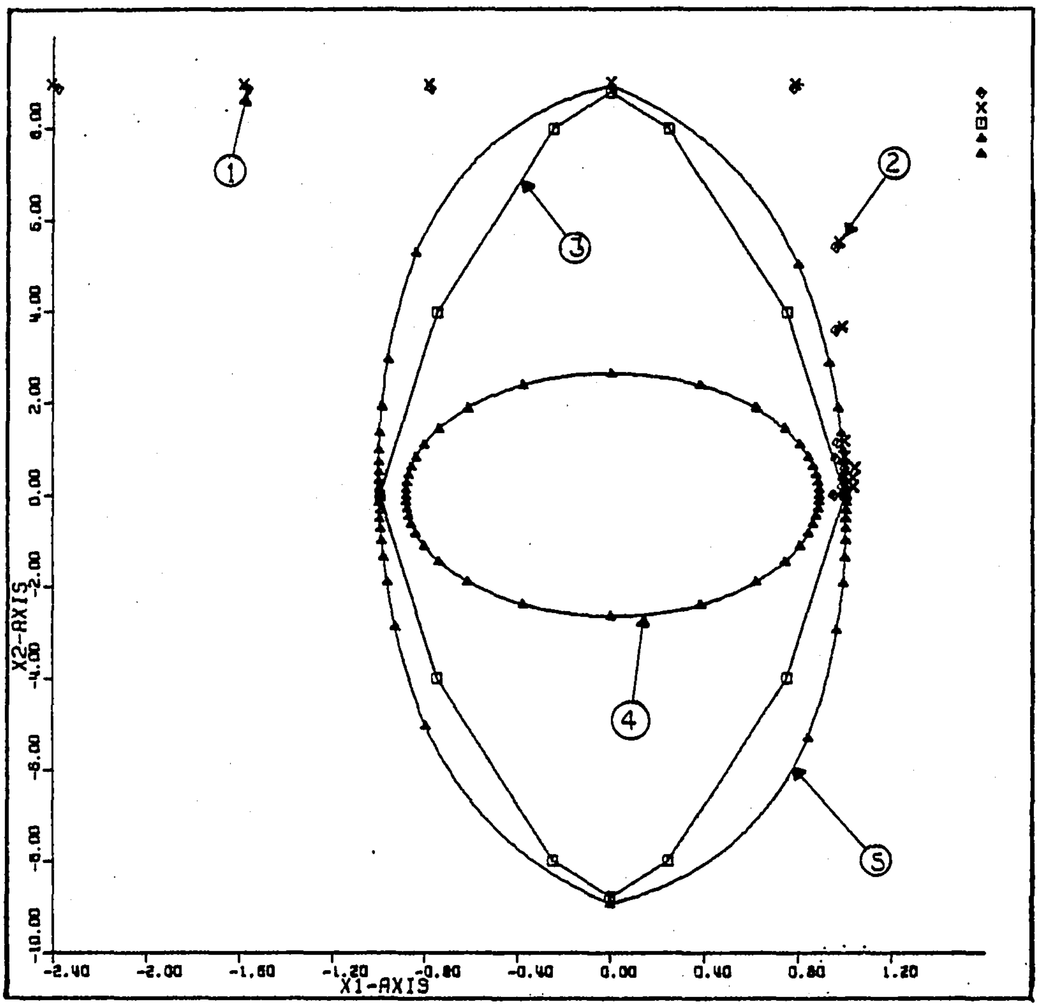


Figure 10. Estimates of domain of attraction for Example 7

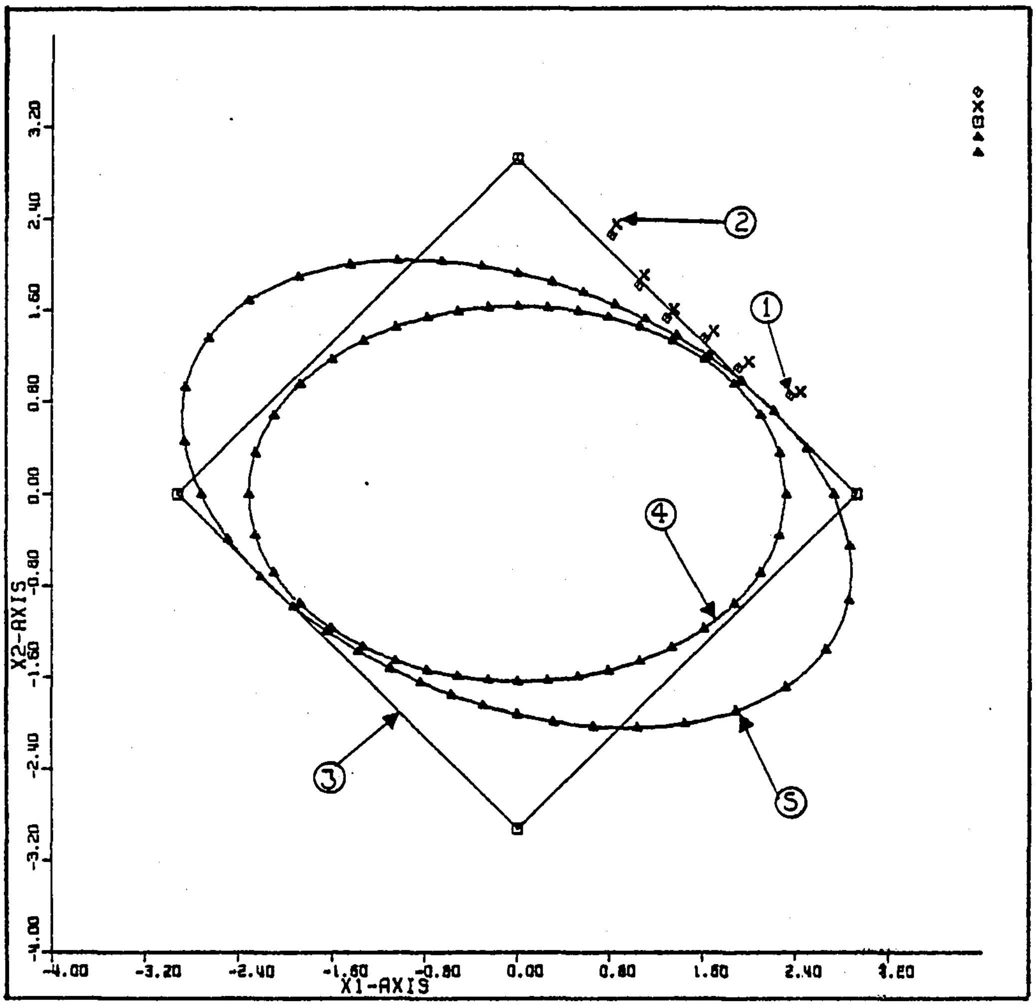


Figure 11. Estimates of domain of attraction for Example 8

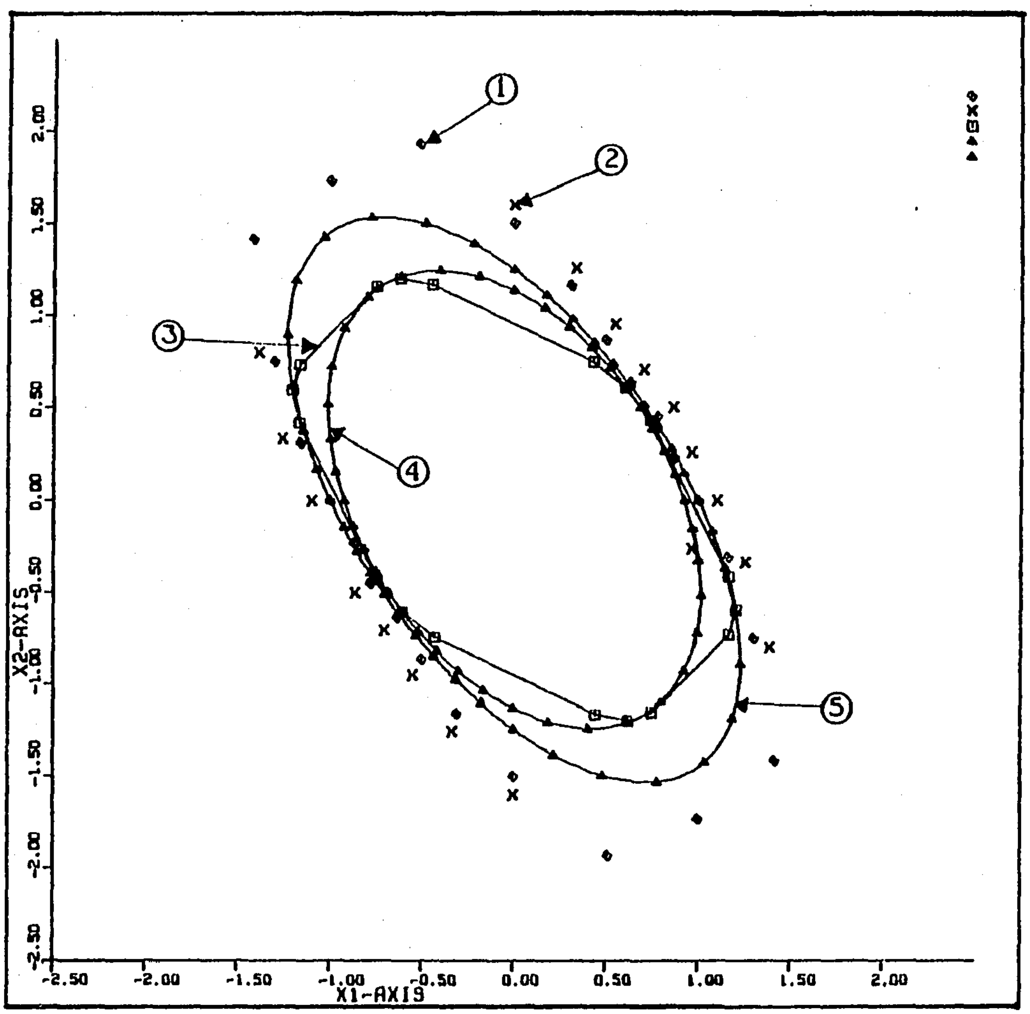


Figure 12. Estimates of domain of attraction for Example 9

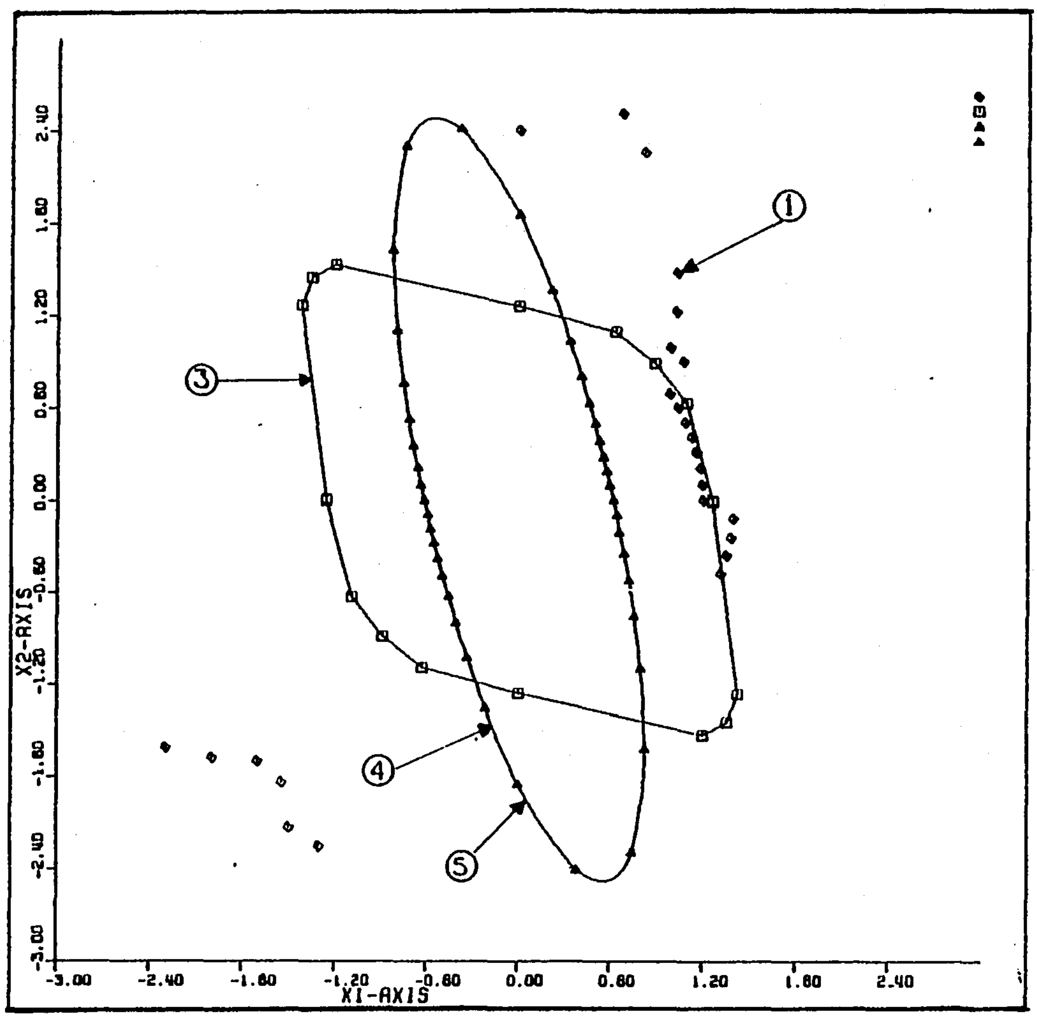


Figure 13. Estimates of domain of attraction for Example 10

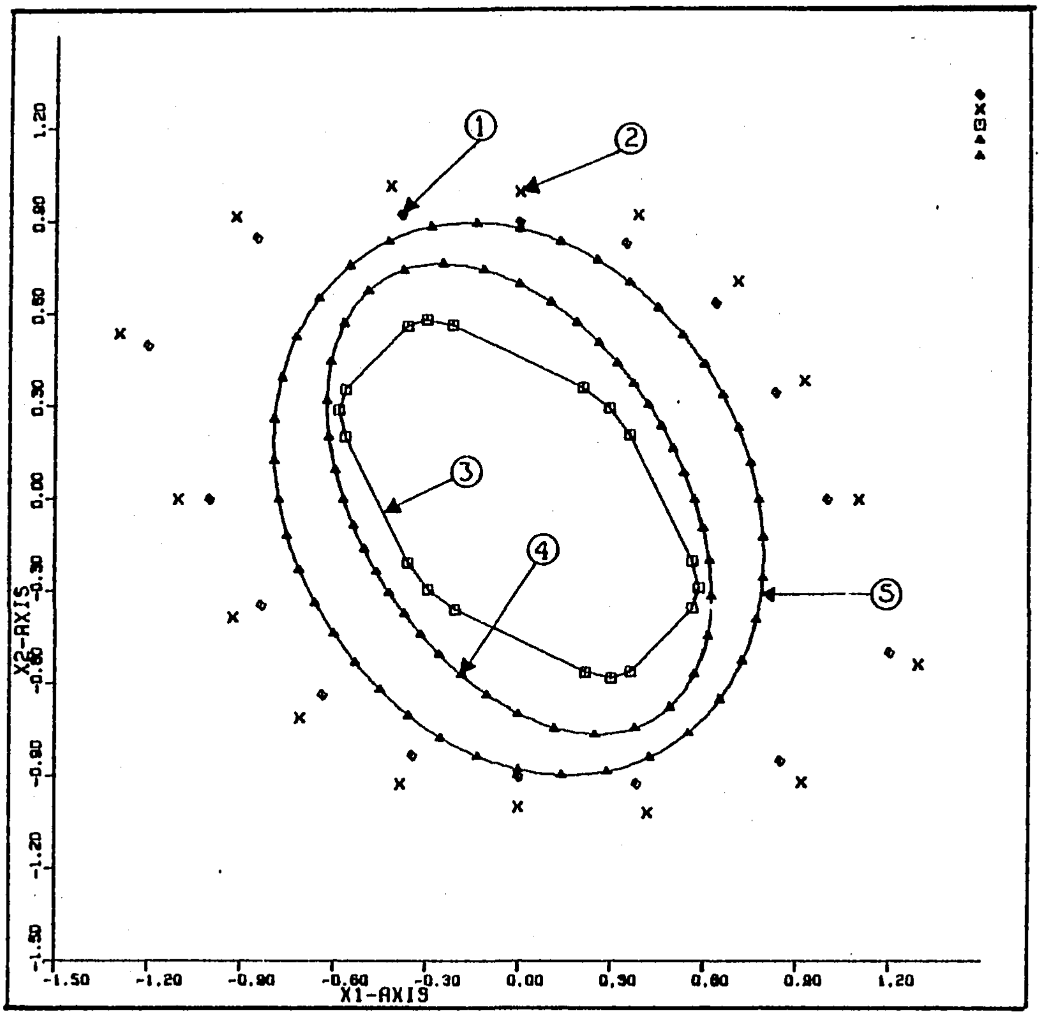


Figure 14. Estimates of domain of attraction for Example 11

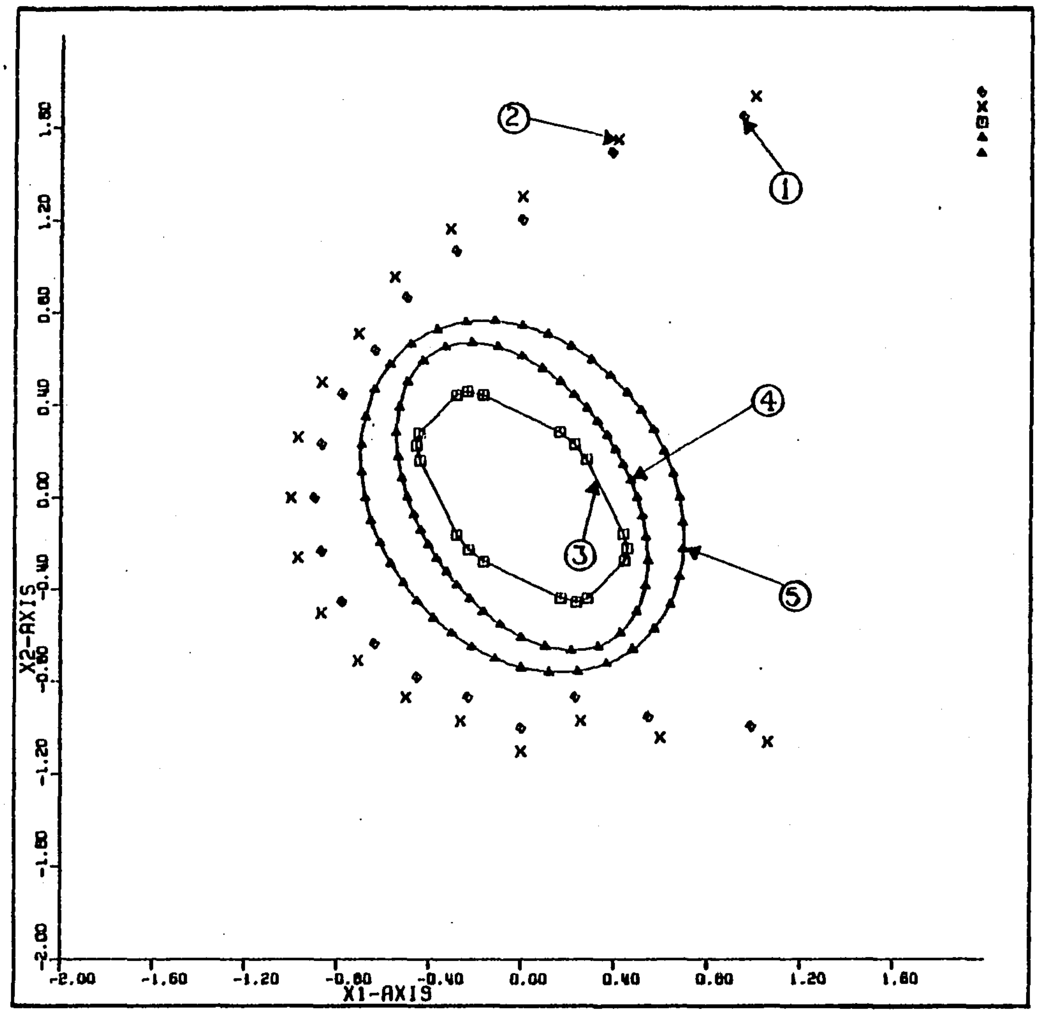


Figure 15. Estimates of domain of attraction for Example 12

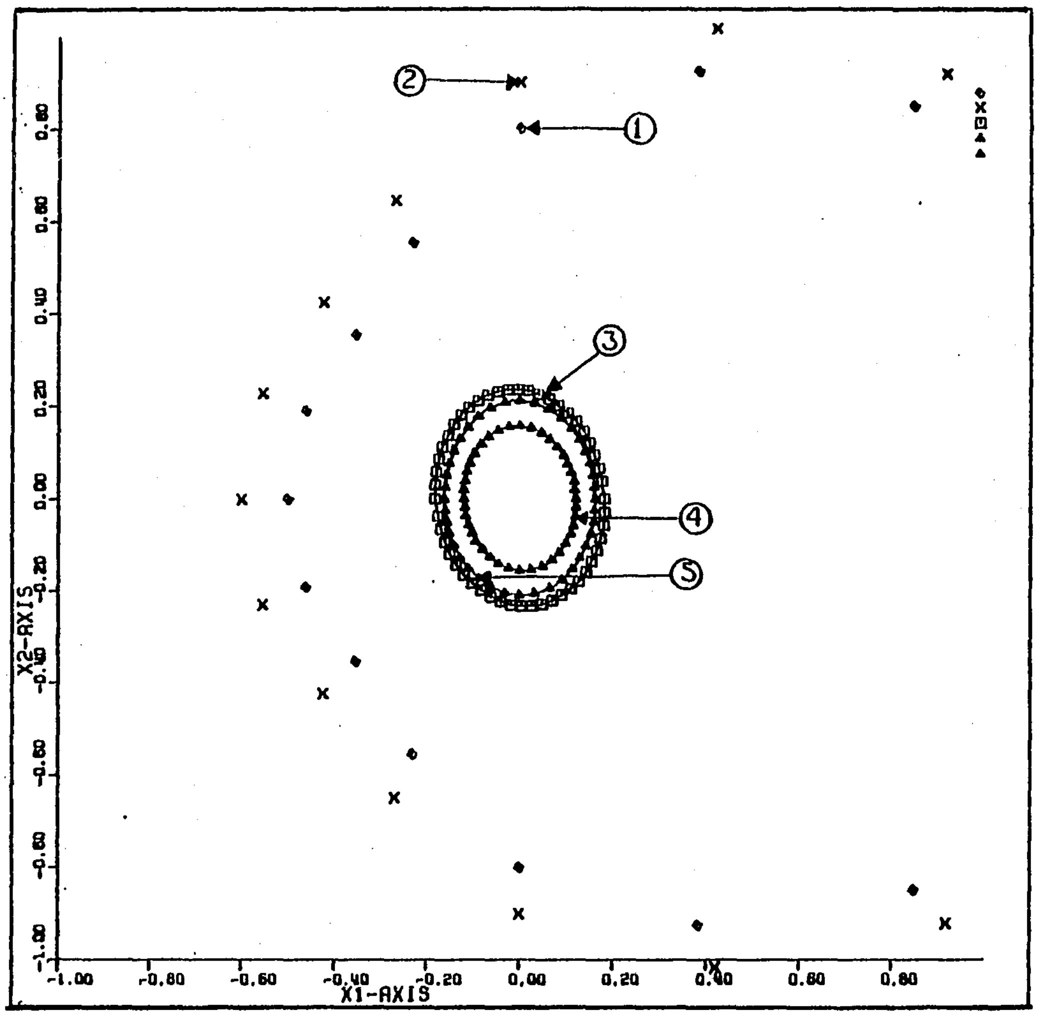


Figure 16. Estimates of domain of attraction for Example 13

Examples 1 and 2 were also considered in [31]. As can be seen from Figures 4 and 5, the results obtained by the method in [31] and by the present method are essentially identical.

The estimates for the domain of attraction for examples 5 and 6, obtained by the present method, are given in Figures 8 and 9. These examples were considered in [33], with some difficulties, using the method of [31].

Examples 8 through 13 were previously considered by Shields and Storey [32]; however, they provide no estimates of the domain of attraction for these examples in [32], and thus no comparisons could be made. In the algorithm of [32], which differs significantly from the present algorithm, it is necessary to solve a system of nonlinear algebraic equations and then to compute an approximate minimization function at every optimization search point. This method seems to require more computational effort than other existing methods, including the method in [31].

For the 13 examples, the average CPU time required to complete all the computations of the present algorithm was found to be 3.03 seconds. For the same 13 problems, the average number of search points, S_A, required for convergency was computed. The Jacobi transformation method program listed in [40] was used S_A times to compute the eigenvalues explicitly, and it was found that the required CPU time to be 1.78 seconds. In the algorithm of [31], it was noted that the major part of the CPU time will be used for the verification of $\dot{v}_{(1)}(x)$ sign definiteness. If we were using their algorithm, then the remaining 1.25 seconds of CPU time, which is less than that needed for the evaluation of eigenvalues, will be insufficient to solve the Lyapunov matrix equation, to create regular and fine grid in n-dimensional space and to verify $\dot{v}_{(1)}(x)$ during each optimization search point at the regular and fine grid. The above comparison, in terms of CPU times, demonstrates the efficiency of the present algorithm.

E. Relationship Between the Eigenvalues of Jacobian and Lyapunov Matrices

Let us choose a quadratic Lyapunov function of the form

$$v(x) = x^T Bx, \qquad B = B^T \tag{23}$$

where B is a real positive definite $n \times n$ matrix. The quadratic Lyapunov functions were used by some authors to find an estimate for the domain of attraction but so far no one has presented any results concerning the effect of changes in the eigenvalues of the J matrix (or equivalently, the "time constant" of the system) on the estimated domain of attraction. In [43], it was stated that $\eta = [- \frac{\dot{v}(x)}{v(x)}]$ and η^{-1} correspond to the largest "time constant" related to the changes in the Lyapunov function v(x). This "time constant" may be regarded as a figure of merit of the system and in [44] it is stated that this "time constant" is about half the conventional "time constant" defined for the system (1). In this section we derive a relationship between the eigenvalues of J and B matrices, and then we make some conclusions, based on this relationship and computational experience, about the effect of changes in eigenvalues of the J matrix on the estimated domain of attraction.

One possible choice for the matrix B in equation (23) can be obtained by solving the Lyapunov matrix equation

$$J^T B + BJ = -I. \tag{24}$$

The above equation can also be written in the form

$$J^T + BJB^{-1} = -B^{-1}. \tag{25}$$

Recalling that Trace (J^T) = Trace (J) and Trace (BJB^{-1}) = Trace (J), we obtain from (25)

$$2 \text{ Trace } (J) = - \text{ Trace } (B^{-1}). \tag{26}$$

Furthermore, Trace (J) = $\sum_{i=1}^{n} \lambda_i(J)$ and $\lambda_i(B^{-1}) = \lambda_i^{-1}(B)$ and thus the equation (26) can be written as

$$2 \sum_{i=1}^{n} \lambda_i(J) = - \sum_{i=1}^{n} \lambda_i^{-1}(B). \tag{27}$$

Without loss of generality, assume that the element J_{nn} of matrix $J = [J_{ij}]$ is equal to $-\mu$, where $\mu > 0$ is a parameter, and all other elements are constants. Now the equation (27) can be written as

$$2\beta + 2\mu = \sum_{i=1}^{n} \lambda_i^{-1}(B) \tag{28}$$

where $\beta = -(J_{11} + \dots + J_{n-1,n-1}) \geq 0$ is a constant. Since β is a constant in equation (28), we deduce that for any change in μ (equivalently, for any change in eigenvalues of J) there will be a corresponding change in the eigenvalues of matrix B. The examples 4 and 5 given in section D of this chapter and which conform to the following state space form

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= f_2(x_1,x_2) \end{aligned} \tag{29}$$

were used to gain some computational experience. These examples are widely used in the literature and both examples have a saddle point in the proximity of the asymptotically stable equilibrium point $x = 0$. In examples 4 and 5, assume that the element J_{22} of Jacobian matrix $J = [J_{ij}]$ is equal to $-\mu$, where $\mu > 0$ is a parameter. Under the above conditions, for the examples 4 and 5, the equation (28) can be written as

$$2\mu = \frac{1}{\lambda_1(B)} + \frac{1}{\lambda_2(B)} \cdot \tag{30}$$

For examples 4 and 5, the parameter μ has been varied over some closed interval $[\mu_1, \mu_2]$ where μ_1 is a sufficiently small positive constant, usually 0.01, and μ_2 is a large positive constant in comparison with μ_1. When the parameter μ was increased progressively from $\mu = \mu_1$ to $\mu = \mu_2$, it was found that the largest "time constant" which is also the largest eigenvalue of matrix B, achieved its minimum at some μ_m and the minimum eigenvalue of B decreased continuously. In the interval $[\mu_1, \mu_m]$, the estimated domain of attraction D_a for some $\mu = \mu_a$ was found to be a subset of the estimated domain of attraction D_b for some $\mu = \mu_b$ whenever $\mu_a < \mu_b$. And also as the value of μ approached μ_m, it was observed that the asymptotic stability boundary described by $v(x) = d$ will approach the actual stability boundary. In the interval $[\mu_m, \mu_2]$, the eccentricity of the estimated domain of attraction increased rapidly as the μ is increased from μ_m to μ_2. This is due to the decrease of the $\lambda_{min}(B)$ and increase of the $\lambda_{max}(B)$. The estimated domains of attraction for the various values of μ, for the problems 4 and 5, are depicted in Figures 17 and 18.

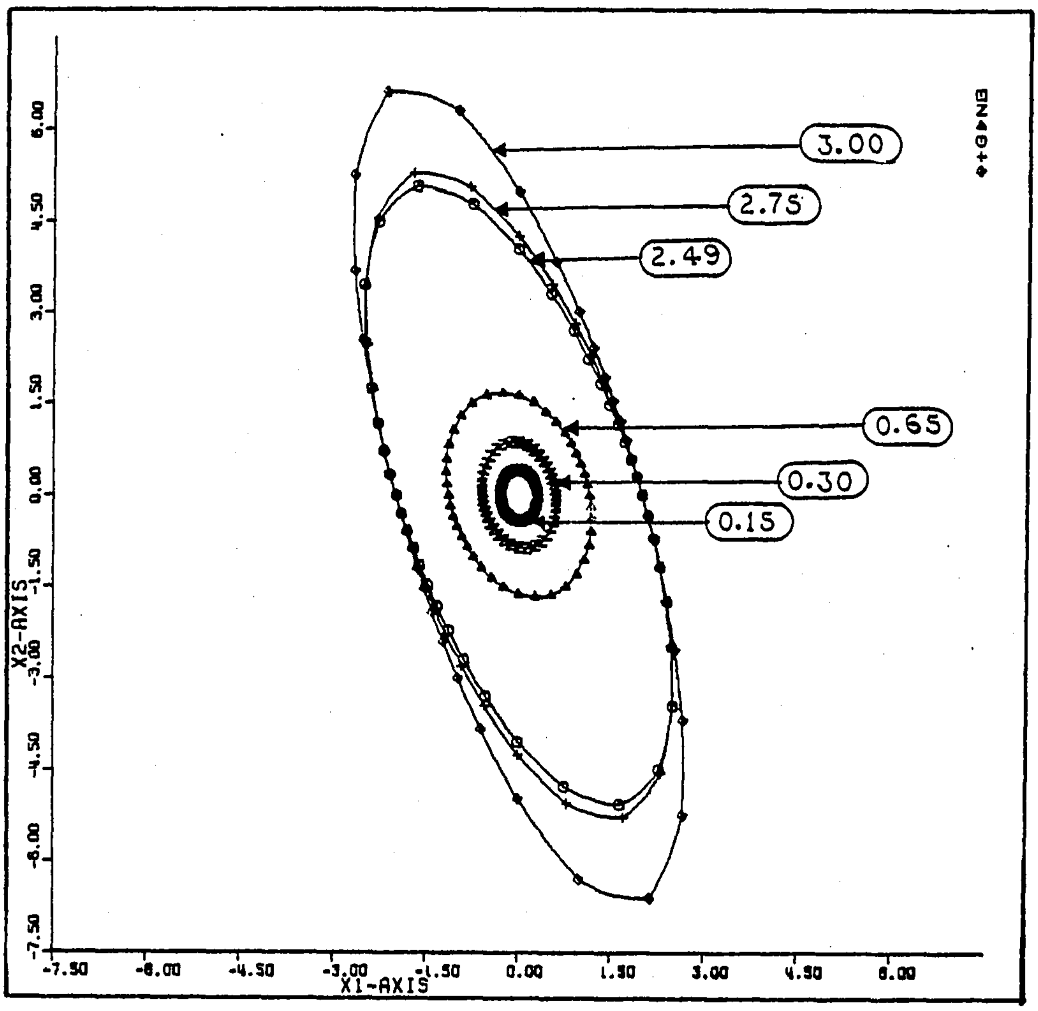


Figure 17. Final estimates of domain of attraction obtained (using the quadratic Lyapunov function algorithm) for Example 4 for various values of J_{22} of Jacobian matrix $J = [J_{ij}]$

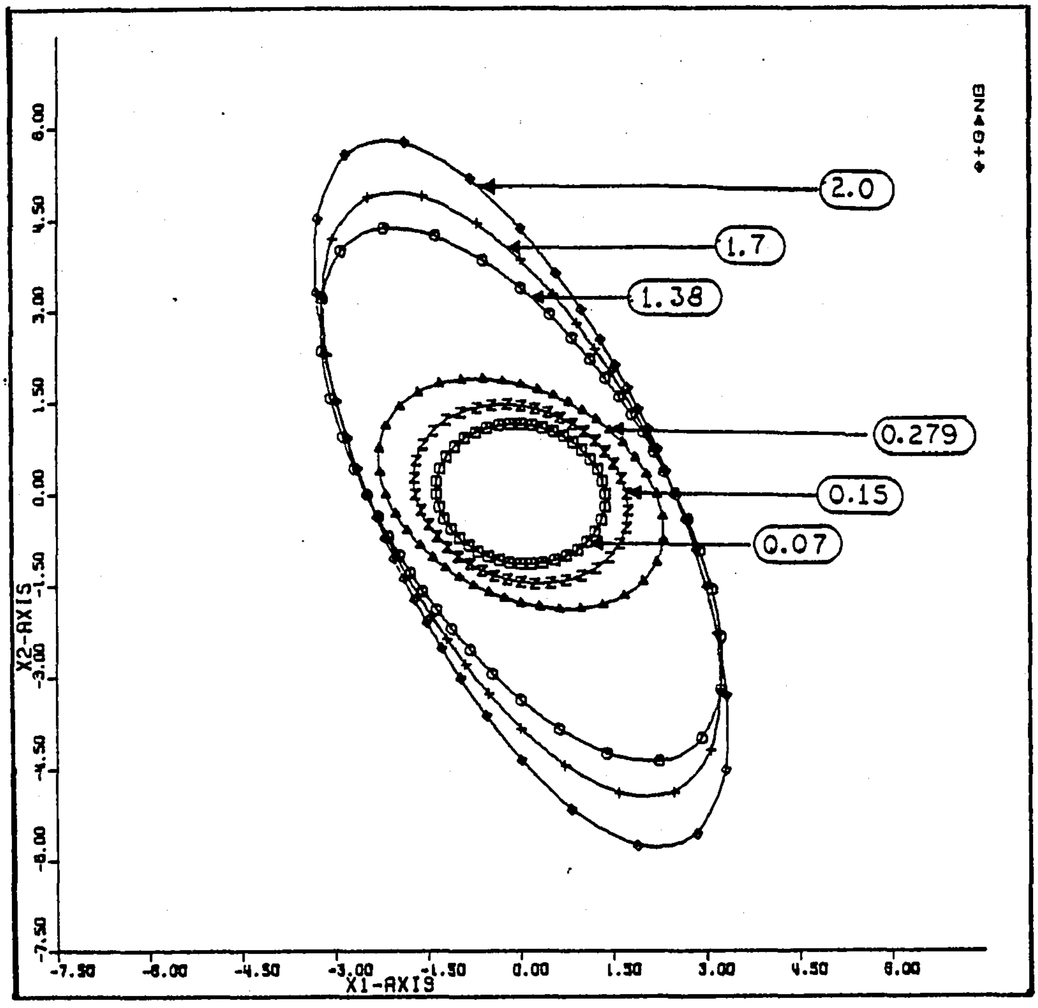


Figure 18. Final estimates of domain of attraction obtained (using the quadratic Lyapunov function algorithm) for Example 5 for various values of J_{22} of Jacobian matrix $J = [J_{ij}]$

V. ANALYSIS BY NORM LYAPUNOV FUNCTIONS

A short summary of results on the constructive stability theory due to Brayton and Tong [4], [5] was presented in section III.B. In this chapter, we modify their results to develop a new algorithm to determine an estimate of the domain of attraction of $x = 0$ for (1). An analysis of numerous examples shows that the present method is significantly more efficient in terms of computation time than the methods discussed in Chapter IV and section III.C. Also, the present method extends the applicability of the results of [4] and [5] to the cases when (1) has more than one equilibrium (i.e., the present method yields a procedure to establish asymptotic stability of $x = 0$ which need not be global). This chapter is divided into three sections. In section A, the algorithm is developed; in section B, the algorithm is summarized; and in section C, the results obtained by this algorithm for the 13 specific examples (considered in section IV.D) are compared with the existing ones.

A. Development of the Algorithm

The first step in the present method involves the linearization of equation (1) about the equilibrium point $x = 0$. (Note that an equilibrium point not located at the origin can be transferred to the origin by an appropriate coordinate transformation.) This yields the equation

$$\dot{x} = Jx + f_1(x) \tag{31}$$

where $J = \frac{\partial f}{\partial x}(x)\Big|_{x=0}$ denotes the Jacobian matrix evaluated at $x = 0$ and $f_1(x)$ consists of higher order terms in the components of x. If the real parts of the eigenvalues of J are negative, then the equilibrium $x = 0$ of the linearization of (1) given by

$$\dot{x} = Jx \tag{32}$$

is asymptotically stable (in fact, $x = 0$ of the linearized system (32) is globally asymptotically stable). Furthermore, from Lyapunov's Indirect method (presented in section III.A) we can deduce that the equilibrium $x = 0$ of (1) is locally asymptotically stable. Now our problem, as we stated earlier, is to determine an estimate of the domain of attraction of $x = 0$ for (1). To this end we apply Euler's method to equation (32) to obtain the difference equation

$$x_{n+1} = (I + h_n J)x_n \tag{33}$$

where h_n denotes the current computation step size (i.e., $h_n = t_{n+1} - t_n$). Next we form an infinite set of matrices defined by

$$\underline{A} = \left\{ I + h_n J\colon\ 0 \leq h_n \leq h',\ J \in \underline{S}_J \right\}. \tag{34}$$

In general, we assume that J may depend on a parameter $p \in R^k$, i.e., $J = J(p)$, and p is permitted to vary over all allowable values to generate the set $\underline{S}_J$. When J is independent of parameters, then the set $\underline{S}_J$ will consist of a single matrix. Furthermore, if there are no parameters in the matrix J, then the set of extreme matrices of $\underline{A}$ is given by

$E(\underline{A}) = \{I, I + h'J\}$. Recall that $\underline{A} \subseteq \mathcal{K}[I, I + h'J]$, and also recall that if $(I + h'J)$ is stable then so is the set $\{I, I + h'J\}$. Therefore, when there are no parameters in the matrix J, the stability analysis need to be done with a single matrix $(I + h'J)$ only. For further details, refer to Theorem 5.2 in [5].

We next use the constructive algorithm presented in section III.B along with the set $E(\underline{A})$ to determine a final convex body W_F from an initial convex body W_0. For any initial point in W_F a solution for (33) will approach the origin with increasing t and the same observation is true for (32). Since the equilibrium $x = 0$ of (32) is globally asymptotically stable, the above observation is true even if we multiply the extreme vertices of W_F by some constant c, $0 < c < \infty$. Thus, the norm defined by

$$\|x\|_{W_F} = \inf \left\{ \alpha \,\middle|\, \alpha \geq 0,\ x \in \alpha W_F \right\}$$

is a Lyapunov function (see Figure 19) for (32).

To estimate the domain of attraction of $x = 0$ for (1), we pick $v(x) = \|x\|_{W_F}$ as a Lyapunov function, and then we construct the gradient of $v(x)$, $\nabla v(x)$, normal to each line (respectively, normal to each hyperplane or flat) determined by $v(x) = \|x\|_{W_F} = c$ (see, e.g., Figure 20). The following discussion is phrased in terms of two-dimensional systems. When $n > 2$, this discussion is modified in the obvious way. Also note that on each L_i, the normal vector $[\nabla v(\bullet)]_{L_i}$ is a constant vector for all points on L_i. Thus, for each L_i, $[\nabla v(\bullet)]_{L_i}$ needs to be computed only once. We now fix ℓ_i points, $x^{i1}, \ldots, x^{i\ell_i}$, at uniform intervals on each

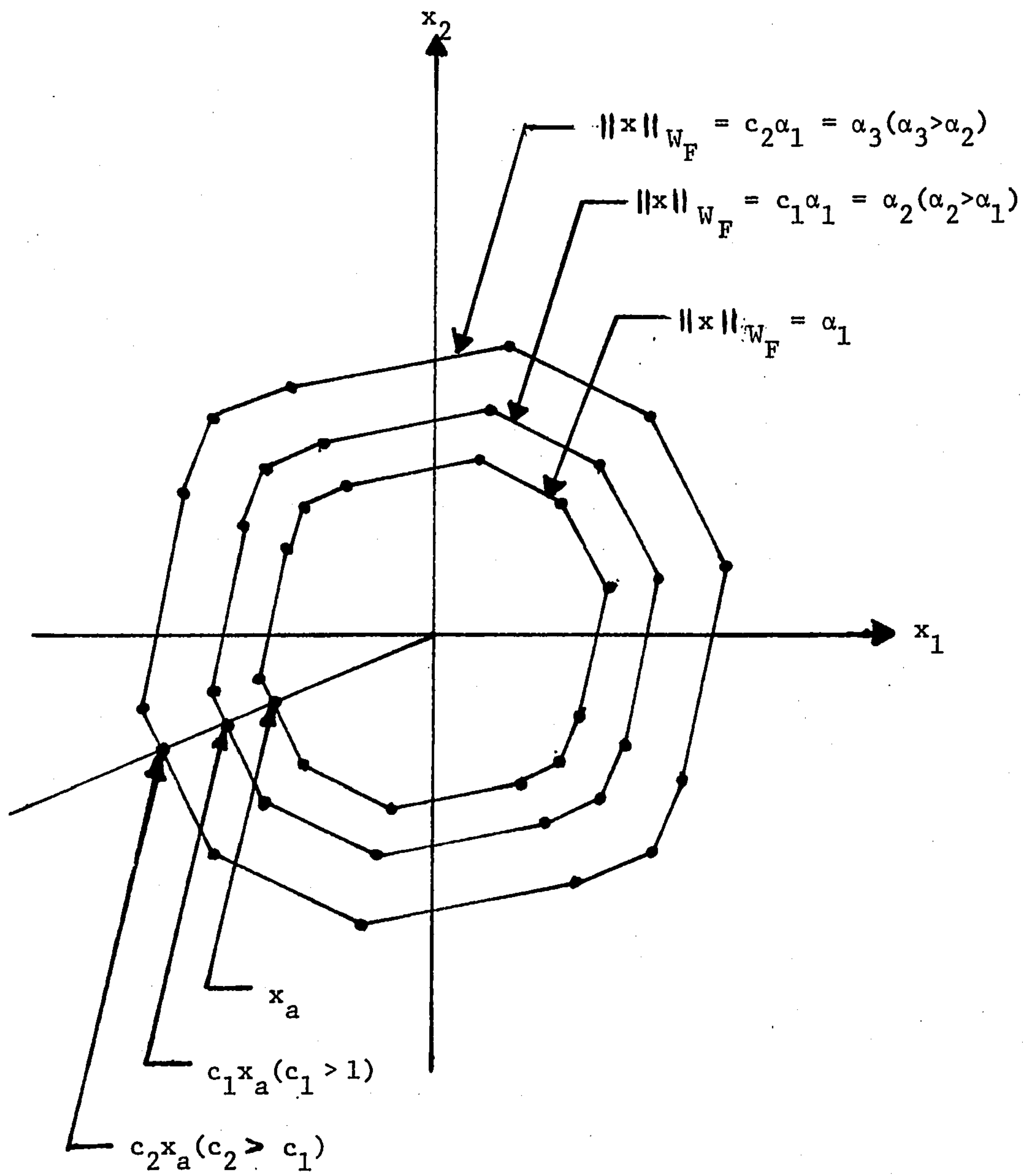


Figure 19. Example of a norm Lyapunov function

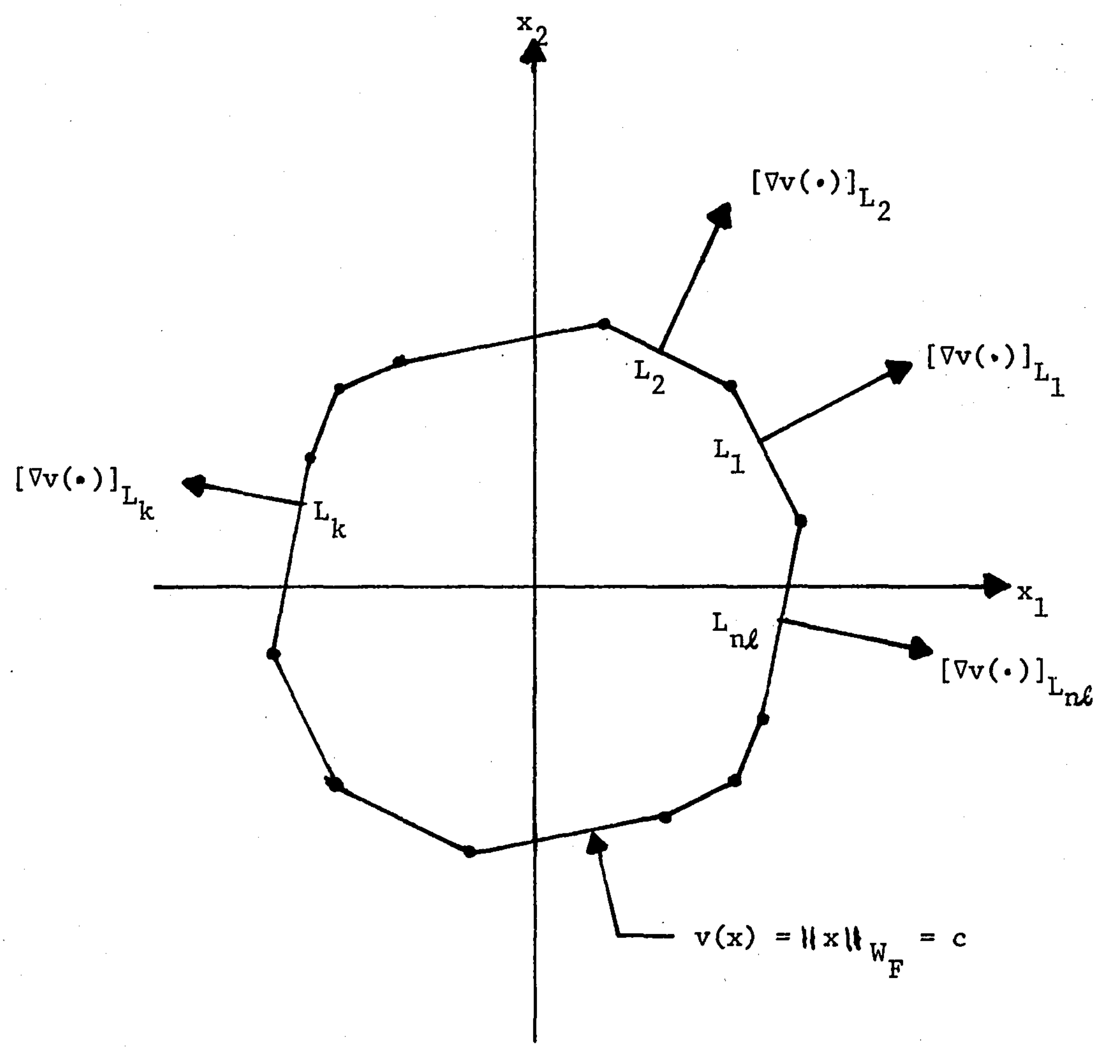


Figure 20. Example of a norm Lyapunov function with normals

line segment L_i forming the boundary of W_F, where ℓ_i is proportional to the length of L_i, $i = 1, \ldots, n_\ell$. Next, we perform a direct search to determine a constant, $c_{min} = \min_i \{c_i\}$ such that

$$Dv(x) \triangleq f(c_i x^{ik})^T \, [\nabla v(c_i x^{ik})]_{L_i} < 0, \quad \begin{array}{l} k = 1, \ldots, \ell_i, \\ i = 1, \ldots, n_\ell, \end{array} \tag{35}$$

where $[\nabla v(\bullet)]_{L_i}$ denotes the gradient vector evaluated on L_i (see Figure 20). If there exists such a constant c_{min}, $0 < c_{min} < \infty$, then $x = 0$ of (1) is asymptotically stable for all points in W_A, where

$$W_A = c_{min} \cdot W_F , \tag{36}$$

i.e., W_A is a subset of the domain of attraction of $x = 0$ for (1). In the next section, we summarize the preceding discussion in the form of an algorithm.

B. Summary of the Algorithm

<u>Step 1</u>: Find the eigenvalues of J in (32) and check if all the real parts are less than zero.

a) If this is not true, exit (not asymptotically stable).

b) If this is true, proceed to Step 2.

<u>Step 2</u>: Estimate a largest possible constant $h' > 0$ such that $|\lambda_i(I + h'J)| < 1$, $i = 1, \ldots, n$.

<u>Step 3</u>: a) Pick an arbitrary initial convex region W_0 containing the origin.

b) Set $k = 0$.

<u>Step 4</u>: Form the new vertex set $E(W_{k+1})$ using the constructive algorithm given in section III.B.

<u>Step 5</u>: Exit (not asymptotically stable) if $E(W_0) \cap E(W_{k+1}) = \emptyset$.

<u>Step 6</u>: a) Set $W_F = W_k$.

b) Go to Step 7 if $E(W_{k+1}) \subset \kappa[W_F]$.

c) Set $k = k + 1$ and go to Step 4.

<u>Step 7</u>: a) Fix ℓ_i points, $x^{i1}, \ldots, x^{i\ell_i}$, uniformly distributed over each of the n_ℓ line segments (respectively, flats) L_i which form the boundary of W_F.

b) Form n_ℓ normals, <u>one for each line</u> (flat) L_i which form the boundary of W_F.

<u>Step 8</u>: Find a constant $c_{min} = \min_i \{c_i\}$, $0 < c_i < \infty$, such that $Dv(x) = f(c_i x^{ik})^T [\nabla v(c_i x^{ik})]_{L_i} < 0$, $k = 1, \ldots, \ell_i$, $i = 1, \ldots, n_\ell$.

<u>Step 9</u>: a) If $c_{min} = 0$, no domain of attraction can be found with the particular W_0 chosen. Pick another W_0, set $k = 0$, and go to Step 4.

b) If $c_{min} \varepsilon (0,\infty)$, set $W_A = c_{min} \cdot W_F$. The estimate of the domain of attraction is given by $\{x \in E^n : x \in \kappa[W_A]\}$.

C. Discussion

The above algorithm was coded in Fortran WATFIV on the ITEL AS/6 system. The same 13 problems that were considered in section IV.D were treated by the present algorithm as well. The average amount of CPU time required for the completion of a single run for all these examples is 0.57 seconds. A comparison of the domains of attraction obtained for examples 1 through 13 (see, section IV.D) by the present algorithm and by the algorithm of Chapter IV is given in Figures 4 through 16. In terms of the estimates of the domains of attraction obtained, these examples do not suggest which algorithm is preferable since in some cases the present method yields a larger region for the domain of attraction than the method of Chapter IV, and vice versa. However, in terms of computational efficiency, the present algorithm seems to be considerably superior to the method of Chapter IV.

In the present algorithm, different initial convex regions W_0 may yield different final convex regions W_F, and thus, different estimates of the domain of attraction, W_A. In cases where W_F is identical to W_0, it may be advantageous to use several initial regions W_0 having different shapes to obtain different estimates of the domain of attraction. The domain of attraction obtained by the union of all such different estimates may be an improved estimate. For the majority of the problems considered, the domain of attraction was obtained with only one initial convex region and the estimate was found to be satisfactory.

As the positive constant h' decreases, the magnitudes of eigenvalues of $(I + h'J)$ will approach the unit circle, and thus it was observed that the computation time required to find W_F from a given W_0 increased as h' decreased. Examples 6 and 13, which have small values of h', consumed more than the average amount of computation time for convergence.

In contrast to other methods (such as the methods in[31] and in Chapter IV), the search procedure to determine the negative definiteness of $\dot{v}_{(1)}(x)$ needs to be used only <u>once</u> for every W_F in the present algorithm. This, along with the fact that for each flat L_i, only <u>one</u> normal vector $[\nabla v(\bullet)]_{L_i}$ has to be computed points to the simplicity of the present method and explains why the present method seems to be computationally more efficient than any of the existing methods which we examined in section III.C and Chapter IV.

VI. APPLICATION OF THE COMPARISON PRINCIPLE

The results of Brayton and Tong [4], [5] are not practical for high dimensional systems since they exceed the capabilities of most modern computers when the dimension of the system under consideration is approximately greater than eight. In principle, the algorithms presented in Chapters IV and V can be used for any dimension but their implementation may not be very efficient at high dimensions. The purpose of this chapter is to develop an efficient method for the analysis of large-scale systems (high dimensional systems). The subject of stability analysis of large-scale systems received great attention during the last decade and an extensive treatment on this subject can be found in Michel and Miller [2].

The general idea involved in the analysis of large-scale systems of the form (1) is to view such systems as interconnected systems (which is frequently true for systems of practical interest) of the form

$$\dot{z}_i = e_i(z_i) + g_i(z_1, \ldots, z_\ell), \qquad i = 1, \ldots, \ell, \tag{A}$$

where $z_i \in E^{n_i}, \sum_{j=1}^{\ell} n_j = n$, $x^T = [z_1^T, \ldots, z_\ell^T] \in E^n$, $g_i : E^n \to E^n$, and $e_i : E^{n_i} \to E^{n_i}$. As usual, we assume that (A) has an isolated equilibrium at $x = 0$. A system described by (A) may be viewed as a nonlinear interconnection of ℓ systems represented by equations of the form

$$\dot{z}_i = e_i(z_i). \tag{B}$$

We assume that for every $t_0 \in R^+$ and every $z_{i0} \in E^{n_i}$, equation (B) has a unique solution $z_i(t, z_0, t_0)$ for $t \geq t_0$ with $z_i(t_0, z_0, t_0) = z_{i0}$. We refer to (B) as the ith isolated subsystem or as the ith free subsystem or as the ith uncoupled subsystem. Frequently, these isolated subsystems are relatively simple and they are of low order. Thus, if the stability properties of the free subsystems are known a priori, we are often in a position to search for Lyapunov functions with certain general properties. Now for the analysis of interconnected system (A) (respectively (1)), generally two methods are used. One of these methods is based on scalar Lyapunov functions which consist of weighted sums of Lyapunov functions for the isolated subsystems, and the other method is based on vector Lyapunov functions whose components are Lyapunov functions for the isolated subsystems. In this chapter, we show how the comparison principle can sometimes be applied in an efficient analysis of large-scale systems using vector Lyapunov functions. Vector Lyapunov functions were originally introduced by Bellman [45]. Perhaps, Bailey [46], [47] was the first to apply the comparison principle to vector Lyapunov functions in analyzing interconnected systems. Subsequently, much work has been done in the area, and for further details refer to [2] and [30].

This chapter consists of three sections. In section A, some background material on the comparison principle is presented, and in section B we show how sometimes the comparison principle and stability preserving maps can be combined into methods such as those developed in Chapters IV and V (and in [4] and [5]) in the analysis of high dimensional systems. Also, in section B, we present two specific examples to demonstrate the applicability of the results developed in sections A and B of the present chapter. In section C, we discuss the results presented in the first two sections.

A. Comparison Principle

Henceforth, we concern ourselves with systems described by equations of the form

$$\dot{x} = g(x,t) \qquad \text{(I)}$$

where $x \in E^n$, $g: E^n \times R^+ \rightarrow E^n$ and $g(0,t) = 0$ for all $t \in R^+$. For (I) we consider a vector valued <u>comparison equation</u> given by

$$\dot{y} = h(y,t) \qquad \text{(C)}$$

where $y \in R^{\ell}$, $\ell \leq n$, $t \in R^+$, $h: R^{\ell} \times R^+ \rightarrow R^{\ell}$ and $h(0,t) = 0$ for all $t \in R^+$. We assume that h and g are continuous on their respective domains of definition. We also require that h be quasimonotone. Recall that a function $h(y,t) = [h_1(y,t), \ldots, h_{\ell}(y,t)]^T$ is said to be <u>quasimonotone</u> if for each component h_j, $j = 1, \ldots, \ell$, the inequality $h_j(y,t) \leq h_j(z,t)$ is true whenever $y_i \leq z_i$ for all $i \neq j$ and $y_i = z_i$. Finally, we will employ <u>vector Lyapunov functions</u> of the form

$$V(x,t) = [v_1(x,t), \ldots, v_{\ell}(x,t)]^T$$

where $v_i: E^n x R^+ \rightarrow R$, and where the v_i are assumed to be continuous on $E^n \times R^+$, and where the v_i are assumed to be locally Lipschitz in x, $i = 1, \ldots, \ell$.

We will require the following comparison result.

Theorem 4: Let g and h be continuous on their respective domains of definition and let h be quasimonotone. Let $V(x,t)$ be a continuous non-negative vector Lyapunov function (of dimension ℓ) such that $|V(x,t)|$ is positive definite and such that the vector differential inequality

$$DV(x,t) \leq h(V(x,t),t)$$

holds. (Here $DV(x,t) = \lim_{h \to 0} \sup[V(x(t+h),t+h) - V(x(t),t)]/h.$) Then the following statements are true.

(i) If $|V(x,t)|$ is decrescent and if the trivial solution of (C) is uniformly asymptotically stable, then the trivial solution of (I) is also uniformly asymptotically stable.

(ii) If in (i) $|V(x,t)|$ is radially unbounded and if the trivial solution of (C) is uniformly asymptotically stable in the large, then the trivial solution of (I) is also uniformly asymptotically stable in the large.

In view of the above theorem, we can under certain conditions deduce the stability properties of the equilibrium $x = 0$ of an n-dimensional system from the stability properties of the equilibrium $y = 0$ of an ℓ-dimensional system (C), $\ell \leq n$.

From the point of view of applications, the following special case of (C),

$$\dot{y} = Py + m(y,t) \tag{C'}$$

is particularly important. Here $P = [p_{ij}]$ is a real $\ell \times \ell$ matrix and the function $m: E^{\ell} \times R^{+} \rightarrow E^{\ell}$ is assumed to consist of second or higher order terms, so that

$$\lim_{|y|\to 0} |m(y,t)|/|y| = 0 \quad \text{uniformly in } t \geq 0.$$

Applying the principle of stability in the first approximation to (C') (see section III.A and [3]), the following result follows from Theorem 4.

Corollary 1: Let g be continuous and let $V(x,t)$ be a nonnegative continuous vector Lyapunov function (of dimension ℓ) such that $|V(x,t)|$ is positive definite and decrescent. Suppose there is an $\ell \times \ell$ matrix $P = [p_{ij}]$ and a function $m(V,t)$ such that

$$DV_{(I)}(x,t) \leq PV(x,t) + m(V(x,t),t) \tag{37}$$

and

$$p_{ij} \geq 0 \text{ if } i \neq j \tag{38}$$

and $m(V,t)$ is quasimonotone in V, and

$$\lim_{|y|\to 0} |m(y,t)|/|y| = 0, \text{ uniformly in } t \geq 0.$$

Under the above conditions if the matrix P has only eigenvalues with negative real parts, then the trivial solution of (I) is uniformly asymptotically stable.

Note that in Corollary 1, the quasimonoticity condition (38) means that $-P$ is an M-matrix. Recall that a real $\ell \times \ell$ matrix $D = [d_{ij}]$ is said to be an M-matrix if $d_{ij} \leq 0$, $i \neq j$, and if all principal minor determinants of D are positive. Therefore, the condition that all eigenvalues of P have negative real parts is equivalent to the computable conditions (see [2, p 62])

$$(-1)^k \begin{vmatrix} p_{11} & \cdots & p_{1k} \\ \vdots & & \vdots \\ p_{k1} & \cdots & p_{kk} \end{vmatrix} > 0, \quad k = 1, \ldots, \ell.$$

Since $-P$ is an M-matrix, note also that the condition $\mathrm{Re}[\lambda_k(P)] < 0$, $k = 1, \ldots, \ell$, are equivalent to the existence of a column vector $\alpha^T = [\alpha_1, \ldots, \alpha_\ell]^T > 0$ such that $\alpha^T P < 0$. So in this case we can define a <u>scalar Lyapunov function</u>

$$v(x,t) = \sum_{i=1}^{\ell} \alpha_i v_i(x,t) \tag{39}$$

where $V(x,t) = [v_1(x,t), \ldots, v_\ell(x,t)]^T$. Under the assumptions of Corollary 1, this function $v(x,t)$ is a positive definite and decrescent scalar Lyapunov function such that

$$\begin{aligned} Dv_{(I)}(x,t) &\leq \alpha^T[PV(x,t) + m(V(x,t),t)] \\ &= (\alpha^T P)\, V(x,t) + \alpha^T m(V(x,t),t). \end{aligned}$$

(Here $Dv_{(I)}(x,t) = \limsup_{h \to 0} [v(x(t+h),t+h) - v(x(t),t)]/h$.) Since $(\alpha^T P)V(x,t)$ is negative definite and since $\alpha^T m(V,t)$ consists of terms of second or higher order in $V(x,t)$, it follows that $Dv_{(I)}(x,t)$ is negative definite in a neighborhood of the origin $x = 0$. This argument shows that in the important special case of (37) the vector Lyapunov function $V(x,t)$ can be reduced to a scalar Lyapunov function $v(x,t)$ and because of the quasimonoticity requirement, the seemingly more general approach of applying the comparison principle to vector Lyapunov functions is really equivalent to an approach utilizing scalar Lyapunov functions (as discussed, e.g., in [2]). Note, however, that this equivalence will not necessarily hold for systems of inequalities more general than (37).

B. Estimation of the Domain of Attraction

Several existing papers address the problem of estimating the domain of attraction of $x = 0$ for (I) by invoking the comparison principle (i.e., Corollary 1). For a review of this literature, refer to the survey paper by Pai [48] and to the recent paper by Chen and Schinzinger [49]. Although the details vary somewhat in these papers, most of them deal with <u>interconnected systems</u> (see [2]) and they seek to satisfy the hypotheses of Corollary 1, making use of Lyapunov functions of the form (39). Specifically, if we let C_k denote the level curves

$$C_k = \Big\{x \in E^n : v(x,t) = \sum_{i=1}^{\ell} \alpha_i v_i(x,t) \le c_k,\ c_k > 0 \text{ and } \alpha_i > 0 \text{ for all } i = 1, \ldots, \ell \text{ and for all } t \in R^+; \text{ and } v \text{ is positive definite and decrescent; and } Dv_{(I)}(x,t) \text{ is negative definite}\Big\},$$

then $C_k \subset E^n$ will be contained in the domain of attraction of the equilibrium $x = 0$ for (I). In the above literature, given $\alpha^T = [\alpha_1, \ldots, \alpha_\ell] > 0$ one seeks to find the largest $c_{\hat{k}}$ to yield the largest estimate of $C_{\hat{k}}$. In addition, using linear programming techniques, Chen and Schinzinger [49] find the optimal $\hat{\alpha}$ to yield the largest estimate of $\hat{C}$ for the domain of attraction by this procedure.

Results obtained by the above approach (when Corollary 1 is applied to interconnected systems) are frequently overly pessimistic since its implementation involves several majorizations, since it requires a quasimonoticity condition, and since it makes use of a special form of Lyapunov functions. In the following, we discuss an approach which frequently may involve fewer majorizations, which does not employ the Lyapunov functions of the special form (39), and which is based on Theorem 4 rather than Corollary 1. (However, this approach still involves a monoticity condition, which is needed in Theorem 4.) This method uses a stability preserving map to establish an estimate for the domain of attraction of $x = 0$ for (I) from the domain of attraction of $y = 0$ for the comparison system (C), and vice versa. To determine an estimate for the domain of attraction of the lower dimensional comparison system (C), we can use the algorithms developed in Chapters IV and V, or any other existing algorithm (e.g., [31]).

We will find it useful to employ the notion of a stability preserving map, originally considered by Thomas [50] and Hahn [51], and later applied to interconnected systems by Michel, Miller and Tang [52]. Let $X = R^n$ and $Y = R^{\ell}$. For every $t \geq 0$, let $x^0(t) \in X$, where $x^0(t)$ denotes some <u>fixed reference motion</u>. Let $p: X \times R^+ \to Y$. Then p is said to be <u>stability preserving</u> if the image motion $y^0(t) = p(x^0(t),t)$ possesses the same stability properties as $x^0(t)$. Thomas [50] has shown that p is stability preserving if it is a homeomorphism. Subsequently, Hahn [51] showed that p preserves uniform stability, as well as uniform asymptotic stability if there exists a function ϕ of class K, and for any other motion $x(t)$ for which $y(t)$ is defined, we have

$$|y(t) - y^0(t)| < \phi(|x(t) - x^0(t)|) \text{ for all } t \geq t_0 \geq 0.$$

If in particular, $x^0(t) \equiv 0$, the above inequality assumes the form

$$|y(t)| = |p(x(t),t)| < \phi(|x(t)|) \text{ for all } t \geq t_0 \geq 0. \tag{40}$$

It can also be shown that if p is stability preserving with $\phi \in K$ and if there exist positive constants k, α, β such that

$$\phi(r) \leq kr^\alpha, \qquad 0 < r < \beta, \tag{41}$$

then p <u>preserves exponential stability</u> of the reference motion $x^0(t) \equiv 0$. If in addition, $\phi \in KR$, then p <u>preserves the global exponential stability</u> of the trivial motion $x^0(t) \equiv 0$. Although the above results constitute only sufficient conditions, we will agree to accept conditions (40) and (41) as definitions of these concepts.

Now let us consider system (I), the comparison system (C), and the vector Lyapunov function V(x,t) used in Theorem 4. Let us assume that hypothesis (i) of Theorem 4 is satisfied. Then x = 0 for (I) is uniformly asymptotically stable, and furthermore, $|V(x,t)|$ is by assumption positive definite and decrescent. This means that there exist Ψ_1, $\Psi_2 \in K$ such that

$$\Psi_1(|x|) \leq |V(x,t)| \leq \Psi_2(|x|) \tag{42}$$

and it also means that Ψ_1^{-1}, Ψ_2^{-1} exist and that

$$\Psi_2^{-1}(|V(x,t)|) \leq |x| \leq \Psi_1^{-1}(|V(x,t)|) \tag{43}$$

in some appropriate neighborhood of $x = 0$ and for all $t \geq 0$. This shows that when hypothesis (i) of Theorem 4 is satisfied, then there exists a stability preserving map from the X-space (on which (I) is defined) to the Y-space (on which (C) is defined). It also shows that under these conditions there exists a stability preserving map from the Y-space to the X-space.

If we assume that hypothesis (ii) of Theorem 4 is satisfied, then there exist positive constants $k_2 \geq k_1$ and c such that

$$k_1 |x|^c \leq |V(x,t)| \leq k_2 |x|^c$$

and

$$(1/k_2)^c \, |V(x,t)|^{1/c} \leq |x| \leq (1/k_1)^c \, |V(x,t)|^{1/c}$$

in some neighborhood of the origin $x = 0$ and for all $t \geq 0$. This shows that when hypothesis (ii) of Theorem 4 is satisfied, then there exists an exponential stability preserving map from the X-space to the Y-space, and vice versa.

In view of these observations, it is reasonable to ask if we can determine an estimate of the domain of attraction of $x = 0$ for (I) from the domain of attraction of $y = 0$ for (C). The answer to this question is affirmative, as can be seen from the following.

Assume that hypothesis (i) of Theorem 4 is satisfied with the range of V equal to all of $R^{\ell} \times R^{+}$, let $D_X(t_0)$ denote the domain of attraction of $x = 0$ for (I) at t_0, and let $D_Y(t_0)$ denote the domain of attraction of $y = 0$ for (C) at t_0. Now if $y_0 \in D_Y(t_0)$, then $|y(t,y_0,t_0)| \rightarrow 0$ as

$t \to \infty$. (Here, $y(t,y_0,t_0)$ denotes the solution of (C) with $y(t_0,y_0,t_0) = y_0$.) Let $V(x_0,t_0) = y_0$. Then in view of (43) we have

$$|x(t,x_0,t_0)| \leq \Psi_1^{-1}(|V(x(t,x_0,t_0),t)|) \leq \Psi_1^{-1}(|y(t,y_0,t_0)|) \to 0$$

as $t \to \infty$. (Here $x(t, x_0,t_0)$ denotes a solution of (I) with $x(t_0,x_0,t_0) = x_0$.) From this we conclude that $V^{-1}(D_Y(t_0)) \subset D_X(t_0)$, i.e., the inverse image of $D_Y(t_0)$ under V is contained in $D_X(t_0)$.

We summarize the above observations in the following.

Corollary 2: Assume that the hypotheses of Theorem 4 hold with the range of V equal to all of $R^{\ell} \times R^{+}$. Then the following statements are true:

a) There exists a stability preserving map (respectively, an exponential stability preserving map) from the X-space (on which (I) is defined) to the Y-space (on which (C) is defined) and a stability preserving map from the Y-space to the X-space.

b) The inverse image of the domain of attraction (at t_0) of $y = 0$ for (C) under the stability preserving map is contained in the domain of attraction (at t_0) of $x = 0$ for (I).

Thus, from part b of Corollary 2 we can deduce an estimate of the domain of attraction (at t_0) of $x = 0$ for (I) from the domain of attraction (at t_0) of $y = 0$ for (C). The algorithms developed in Chapters IV and V enable us to estimate the domain of attraction of $y = 0$ for (C).

We now consider two specific examples. In the first of these, we obtain an estimate for the domain of attraction of a four-dimensional system by using a two-dimensional comparison system. In the second example, we demonstrate how the comparison principle can be used to extend the applicability of the results in [4] and [5] to higher dimensional systems.

Example 14: Consider the fourth order system of equations given by

$$\left.\begin{aligned} \dot{x}_1 &= -x_1[1 - (x_1^2 + x_2^2)] + k_1 x_2 x_4^2 \\ \dot{x}_2 &= -x_2[1 - (x_1^2 + x_2^2)] \\ \dot{x}_3 &= -x_3[9 - (x_3^2 + x_4^2)] + k_3 x_1 \\ \dot{x}_4 &= -x_4[9 - (x_3^2 + x_4^2)] - k_3 x_2. \end{aligned}\right\} \qquad (44)$$

Note that this system has an isolated equilibrium at $x = (x_1, x_2, x_3, x_4)^T = 0$. Note also that system (44) may be viewed as an interconnection of two subsystems given by

$$\left.\begin{aligned} \dot{x}_1 &= -x_1[1 - (x_1^2 + x_2^2)] \\ \dot{x}_2 &= -x_2[1 - (x_1^2 + x_2^2)] \end{aligned}\right\} \qquad (45)$$

and

$$\left.\begin{aligned} \dot{x}_3 &= -x_3[9 - (x_3^2 + x_4^2)] \\ \dot{x}_4 &= -x_4[9 - (x_3^2 + x_4^2)] \;. \end{aligned}\right\} \qquad (46)$$

The domain of attraction of $(x_1, x_2)^T = 0$ for (45) is the unit circle, and the domain of attraction of $(x_3, x_4)^T = 0$ for (46) is the circle with radius equal to 3.

Now choose $v_1 = x_1^2 + x_2^2$ and $v_2 = x_3^2 + x_4^2$ and note that

$$\dot{v}_1 = -2v_1(1-v_1) + 2k_1x_1x_2x_4^2 \leq -2v_1(1-v_1) + 2k_1\left(\frac{x_1^2 + x_2^2}{2}\right) x_4^2$$

$$\leq -2v_1(1-v_1) + k_1v_1v_2$$

and

$$\dot{v}_2 = -2v_2(9-v_2) + 2k_3(x_1x_3 - x_2x_4)$$

$$\leq -2v_2(9-v_2) + k_3(v_1 + v_2).$$

To apply part b of Corollary 2, we need to find an estimate $\tilde{D}_Y$ of the domain of attraction D_Y of the second order system

$$\dot{v}_1 = -2v_1(1-v_1) + k_1v_1v_2$$

$$\dot{v}_2 = -2v_2(9-v_2) + k_3(v_1 + v_2).$$

This was accomplished in Example 7 for $k_1 = k_3 = .1$, where in the present context, only the values $v_1 \geq 0$, $v_2 \geq 0$ are applicable. The desired estimate $\tilde{D}_X \subset D_X$ is now given by

$$\tilde{D}_X = \left\{x \in E^4 : v_1 = x_1^2 + x_2^2,\ v_2 = x_3^2 + x_4^2,\ (v_1, v_2)^T \in \tilde{D}_Y\right\}.$$

<u>Example 15</u>: Consider the fourth order system of equations given by

$$\left.\begin{aligned} \dot{x}_1 &= -4x_1 + x_2 + x_1e^{-(x_1^2 + x_2^2)}\,\beta(x_3^2 + x_4^2) \\ \dot{x}_2 &= -x_1 - 4x_2 - \frac{x_2^3}{1 + x_2^4} + x_2e^{-(x_1^2 + x_2^2)}\beta(x_3^2 + x_4^2) \end{aligned}\right\} \quad (47)$$

$$\left.\begin{aligned}\dot{x}_3 &= -2x_3 + x_4 + x_3 e^{-(x_3^2 + x_4^2)}\,\gamma(x_1^2 + x_2^2)\\ \dot{x}_4 &= -x_3 - 2x_4 + x_4 e^{-(x_3^2 + x_4^2)}\,\gamma(x_1^2 + x_2^2)\end{aligned}\right\}$$

where $\beta(\theta) = \gamma(\theta) = \sin(\theta)$. Note that this system has an isolated equilibrium at $x = (x_1, x_2, x_3, x_4) = 0$. By picking a Lyapunov function $v(x) = x_1^2 + x_2^2 + x_3^2 + x_4^2$ we can prove that the equilibrium $x = 0$ for (47) is asymptotically stable. We also note that the interconnected system (47) has two isolated subsystems given by

$$\left.\begin{aligned}\dot{x}_1 &= -4x_1 + x_2\\ \dot{x}_2 &= -x_1 - 4x_2 - \frac{x_2^3}{1 + x_2^4}\end{aligned}\right\} \qquad (48)$$

$$\left.\begin{aligned}\dot{x}_3 &= -2x_3 + x_4\\ \dot{x}_4 &= -x_3 - 2x_4\,.\end{aligned}\right\} \qquad (49)$$

Now we choose $v_1 = x_1^2 + x_2^2$ and $v_2 = x_3^2 + x_4^2$ and note that

$$\begin{aligned}\dot{v}_1 &= -8(x_1^2 + x_2^2) + 2(x_1^2 + x_2^2)e^{-(x_1^2 + x_2^2)}\,\beta(x_3^2 + x_4^2) - \frac{x_2^4}{1+x_2^4}\\ &\leq -8v_1 + 2v_1 e^{-v_1}\,\beta(v_2)\\ &\leq -8v_1 + \frac{2}{e}\,\beta(v_2)\end{aligned}$$

and

$$\begin{aligned}\dot{v}_2 &= 2(x_3^2 + x_4^2)e^{-(x_3^2 + x_4^2)}\,\gamma(x_1^2 + x_2^2) - 4(x_3^2 + x_4^2)\\ &= 2v_2 e^{-v_2}\,\gamma(v_1) - 4v_2\\ &\leq \frac{2}{e}\,\gamma(v_1) - 4v_2\,.\end{aligned}$$

Now the comparison system for system (47) is given by

$$\dot{v}_1 = -8v_1 + \frac{2}{e}\beta(v_2)$$
$$\dot{v}_2 = \frac{2}{e}\gamma(v_1) - 4v_2. \tag{50}$$

Note that the system (50) has an isolated equilibrium point at $v = [v_1, v_2]^T = 0$. Also, system (50) is identical to system (6-1) used in Brayton and Tong's paper (see, [5, p. 1127]). Now, an analysis identical to the one used in [5] can be used to determine the stability property of the equilibrium $v = 0$. The equilibrium $v = 0$ of (50) was found to be globally asymptotically stable. Thus, by using part b of Corollary 2 we conclude that $x = 0$ of (47) is globally asymptotically stable.

C. Discussion

The results developed in Chapters IV and V and in references [4] and [5] were extended to higher dimensional systems by invoking the comparison principle. In doing so, some new results were established which relate the domain of attraction of the original system to the domain of attraction of a lower dimensional comparison system. Two specific examples were presented to demonstrate the applicability of the results developed in this chapter. These same specific examples demonstrate how sometimes the present results greatly simplify the analysis, with respect to computational times and complexity, of higher dimensional systems. Since the final estimate of the domain of attraction for (I) is determined from the final estimate of the domain of attraction for (C), any improvement that can be obtained by using the methods developed in Chapters IV and V in the estimate of the domain of attraction for (C) is of great value. The results obtained for Example 14 (refer to Figure 10) clearly demonstrate the usefulness of the methods developed in Chapters IV and V.

Most of the existing results make use of a scalar Lyapunov function of the form (39) to determine an estimate for the domain of attraction of $x = 0$ for (I). The final estimate that can be obtained through such results depends on the ingenuity and experience of the user in selecting proper α_i's. The only exception is the procedure due to Chen and Schinzinger [49] who make use of linear programming techniques to determine optimal α_i's. The applicability of the present results, when the dimension of system (I) is high, depends to some degree on the availability of an appropriate method which decomposes (I) into an interconnection of subsystems. Also, the estimates of the domains of attraction obtained for (I) through the present method depend to some extent on the majorizations of the interconnections.

VII. CONCLUSIONS

Now some comments on the results presented in this dissertation are in order. Some of the results presented in this dissertation are also presented in the papers [53], [54], and [55]. Two new algorithms were presented for obtaining estimates of the domain of attraction of an equilibrium.

The first algorithm makes use of quadratic Lyapunov functions and was motivated by the work of Davison and Kurak [31]. Analysis of the results obtained by the present algorithm for the problems considered in [31] indicates that the estimates of the domains of attraction obtained by the two methods are virtually identical. However, the present algorithm seems to be significantly more efficient, with respect to computational time, than the existing algorithms which include the algorithm given in [31]. Also, the present algorithm is much simpler to implement and requires significantly less computer memory space. In references [33] and [34], the algorithm due to Davison and Kurak was used with some difficulties. Difficulties along such lines were not encountered by the present algorithm.

The second algorithm, which is an entirely new method, makes use of norm Lyapunov functions and was motivated by the recent work of Brayton and Tong [4], [5]. In terms of the estimates of the domains of attraction obtained, these examples do not suggest which algorithm is preferable since, in some cases, this method yields a larger region for the domain of attraction than the method of Chapter IV, and vice versa. However, in terms of computational efficiency, this algorithm seems to be considerably superior to any of the existing algorithms. In addition, this algorithm extends the results of [4], [5] by making them applicable to systems with more than one equilibrium point and by yielding asymptotic stability results which need not be global.

The applicability of the above two algorithms and the results presented in [4], [5] for the analysis of high dimensional systems was accomplished by utilizing the concepts of comparison principle and stability preserving maps. In doing so, some new results were established which relate the domain of attraction of the original system to the domain of attraction of a lower dimensional comparison system.

Next, some recommendations for further research are in order. When using the algorithm based on quadratic Lyapunov functions, the initial choice of Lyapunov matrix may have an effect on the final estimate of the domain of attraction. Instead of setting $C = I$ in the Lyapunov matrix equation, it might be advantageous to treat the elements of matrix C as parameters. This enables us to use some iterative procedure to minimize the sum or product of eigenvalues of matrix B subject to the positive definite constraints of matrices B and C. Also, it might be possible to extend the domain of attraction obtained by the present method by making use of k-th degree and 2k-th degree Lyapunov functions considered in [32]. One method suggested for the improvement of the estimate of domain of attraction obtained by norm Lyapunov functions is to use several initial convex regions W_0's. Yet another method which might be successful in improving the estimate and which requires further study is to evaluate the Jacobian matrix at the extreme vertices of the estimated domain of attraction W_A and then determine the extreme matrices. These extreme matrices should be used with the same algorithm to determine a new estimate for the domain of attraction and such an estimate might be an improvement over W_A. When using the method presented in Chapter VI for the estimation of the domain of attraction of a high dimensional system (I), a method which decomposes (I) into an interconnected system is required. Hence, further work is needed to develop such a decomposition method.

VIII. REFERENCES

1. Lyapunov, A. M. "Probleme Général de la Stabilité du Mouvement." Reprinted in Annals of Mathematical Studies No. 17. Princeton, New Jersey: Princeton University Press, 1949.

2. Michel, A. N. and Miller, R. K. Qualitative Analysis of Large-Scale Dynamical Systems. New York: Academic Press, 1977.

3. Hahn, W. Stability of Motion (Transl. by A. P. Baartz). New York: Springer Verlag, 1967.

4. Brayton, R. K. and Tong, C. H. "Stability of Dynamical Systems: A Constructive Approach." IEEE Transactions on Circuits and Systems 26 (April 1979): 224-234.

5. Brayton, R. K. and Tong, C. H. "Constructive Stability and Asymptotic Stability of Dynamical Systems." IEEE Transactions on Circuits and Systems 27 (November 1980): 1121-1130.

6. Householder, A. S. The Theory of Matrices in Numerical Analysis. 1st ed. New York: Blaisdell Pub. Co., 1964.

7. Krasovskii, N. N. "On the Stability with Large Perturbations." Prikl. Mat. Mekh. 21 (1957): 309-319.

8. Krasovskii, N. N. Stability of Motion (Transl. by J. L. Brennan), Stanford, California: Stanford University Press, 1963.

9. Berger, A. J. and Lapidus, L. "Stability of High Dimensional Nonlinear Systems using Krasovskii's Theorem." AIChE Journal 15, No. 2 (1969): 171-177.

10. Luecke, R. H. and McGuire, M. L. "Chemical Reactor Stability by Liapunov's Direct Method." AIChE Journal 11 (1965): 749-750 and 762-763.

11. Perlmutter, D. D. Stability of Chemical Reactors. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1972.

12. Berger, A. J. and Perlmutter, D. D. "An Extended Region of Reactor Stability." Chem. Eng. Sci. 20 (1965): 147-156.

13. Luecke, R. H. and McGuire, M. L. "Stability Analysis by Liapunov's Direct Method." Ind. Eng. Chem. Fundamentals 6 (1967): 432-435.

14. Zubov, V. I. Mathematical Methods for the Study of Automatic Control Systems (Transl. by Y. Schorr-Kon). New York: Pergamon Press, 1963.

15. Margolis, S. G. and Vogt, W. G. "Control Engineering Applications of V. I. Zubov's Construction Procedure for Lyapunov Functions." IEEE Transactions on Automatic Control AC-8 (April 1963): 104-113.

16. Yu, Y. N. and Vongsurya, K. "Nonlinear Power Systems Stability Study by Liapunov Function and Zubov's Method." IEEE Transactions on Power Apparatus and Systems PAS-86, No. 12 (December 1967): 1480-1485.

17. DeSarkar, A. K. and Rao, N. Dharma "Zubov's Method and Transient-stability Problems of Power Systems." Proceedings of IEE, 118, No. 8 (August 1971): 1035-1040.

18. Szegö, G. P. "On a New Partial Differential Equation for the Stability Analysis of Time Invariant Control Systems." SIAM Journal of Control 1 (1962): 63-75.

19. Prabhakara, F. S.; El-Abiad, A. H.; and Koivo, A. J. "Application of Generalized Zubov's Method to Power System Stability." Int. J. of Control 20, No. 2 (1974): 203-212.

20. Burnard, G. and Sarlos, G. "Determination of Domain of Stability." Journal of Mathematical Analysis and Applications 23 (1968): 714-722.

21. Kormanik, J. and Li, C. C. "Decision Surface Estimate of Nonlinear System Stability Domain by Lie Series Method." IEEE Transactions on Automatic Control AC-17 (October 1972): 666-669.

22. Hewitt, J. R. and Storey, C. "Comparison of Numerical Methods in Stability Analysis." Int. J. of Control 10, No. 6 (1969): 687-701.

23. Aizerman, M. A. and Gantmacher, F. R. Absolute Stability of Regulator Systems. San Francisco: Holden Day, 1964.

24. Kalman, R. E. "Lyapunov Functions for the Problem of Luré in Automatic Control." Proc. Natn. Acad. Sci., U.S.A. 49 (1963): 201-205.

25. Brockett, R. W. and Willems, J. L. "Frequency-domain Stability Criteria." IEEE Transactions on Automatic Control AC-10 (1965): 255-261 and 407-413.

26. Walker, J. A. and McClamroch, N. H. "Finite Regions of Attraction for the Problem of Lur'e." Int. J. of Control 6, No. 4 (1967): 331-336.

27. Willems, J. L. "The Computation of Finite Stability Regions by Means of Open Lyapunov Surfaces." Int. J. of Control 10, No. 5 (1969): 537-544.

28. Wiessenberger, S. "Application of Results from the Absolute Stability Problem to the Computation of Finite Stability Domains." IEEE Transactions on Automatic Control AC-13 (February 1968): 124-125.

29. Fouad, A. A. "Stability Theory - Criteria for Transient Stability." ERDA Report, Engineering Research Institute, Iowa State University, Ames, June 1975.

30. Pai, M. A. "Stability of Large Scale Power Systems." In Handbook of System Engineering Applications (Eds. M. Singh and A. Titli) North Holland Publishing Co., December 1979.

31. Davison, E. J. and Kurak, E. M. "A Computational Method for Determining Quadratic Lyapunov Functions for Non-linear Systems." Automatica 7 (1971): 627-636.

32. Shields, D. N. and Storey, C. "The Behavior of Optimal Lyapunov Functions." International Journal of Control 21, No. 4 (1975): 561-573.

33. Bansal, R. K. "Estimation of Stability Domains for the Transient Stability Investigation of Power Systems." Ph.D. Dissertation, I.I.T., Kanpur, India, 1975.

34. Subramanian, R. and Bansal, R. K. "Estimation of Power System Stability Domains Using Quadratic Lyapunov Functions." Proceedings of the IEE 124, No. 7 (July 1977): 597-601.

35. Schwarz, H. R.; Rutishauser, H.; and Stiefel, E. Numerical Analysis of Symmetric Matrices (Transl. by P. Hertelendy), Englewood Cliffs, N.J.: Prentice-Hall, 1973.

36. Rosenbrock, H. H. and Storey, C. Computational Techniques for Chemical Engineers. London: Pergamon Press, 1966.

37. Rodden, J. J. "Numerical Application of Lyapunov Stability Theory." Proceedings of the JACC (1964): 261-268.

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

39. Kuester, J. L. and Mize, J. H. Optimization Techniques with FORTRAN. New York: McGraw-Hill Company, 1973.

40. Carnahan, B.; Luther, H. A.; and Wilkes, J. O. Applied Numerical Methods. New York: John Wiley & Sons, Inc., 1969.

41. Bingulac, S. P. "An Alternate Approach to Expanding PA + $A^TP=-Q$." IEEE Transactions on Automatic Control AC-15 (Feb 1970): 135-137.

42. LaSalle, J. and Lefschetz, S. Stability by Lyapúnov's Direct Method with Applications. New York: Academic Press, 1961.

43. Kalman, R. E. and Bertram, J. E. "Control System Analysis and Design Via the Second Method of Liapunov: I. Continuous-Time Systems." ASME J. Basic Engineering, Ser. D, 82 (1960): 371-393.

44. Ogata, K. State Space Analysis of Control Systems. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1967.

45. Bellman, R. "Vector Lyapunov Functions." SIAM J. Control 1 (1962): 32-34.

46. Bailey, F. N. "Stability of Interconnected Systems." Ph.D. Dissertation, University of Michigan, Ann Arbor, Michigan, 1964.

47. Bailey, F. N. "The Application of Lyapunov's Second Method to Interconnected Systems." SIAM J. Control 3 (1966): 443-462.

48. Pai, M. A. "Stability of Large Scale Interconnected Power Systems." Proceedings of the 1980 IEEE International Conference on Circuits and Computers. Port Chester, New York, October 1980.

49. Chen, Y. K. and Schinzinger, R. "Lyapunov Stability of Multi-Machine Power Systems Using Decomposition-Aggregation Method." Paper A-80-036-4, IEEE PES Winter Meeting, New York, 1980.

50. Thomas, J. "Über die Invarianz der Stabilität bei einem Phasenraum-Homöomorphismus." J. für Reine und Angewandte Mathematik 213 (1964): 147-150.

51. Hahn, W. "Über Stabilitätserhaltende Abbildungen and Ljapunovsche Funktionen." J. für die Reine und Angewandte Mathematik 228 (1967): 189-192.

52. Michel, A. N.; Miller, R. K.; and Tang, W. "Lyapunov Stability of Interconnected Systems: Decomposition into Strongly Connected Subsystems." IEEE Transactions on Circuits and Systems 28 (September 1978): 799-809.

53. Michel, A. N.; Sarabudla, N. R.; and Miller, R. K. "Some Constructive Approaches in Stability Analysis Proceedings of the 1981 IEEE International Symposium or Circuits and Systems, Chicago, Illinois, April 1981.

54. Michel, A. N.; Sarabudla, N. R.; and Miller, R. K. "Stability of Complex Dynamical Systems: Computational Methods." Proceedings of the 1981 Joint Automatic Control Conference, Charlottesville, Virginia, June 1981.

55. Michel, A. N.; Sarabudla, N. R. and Miller, R. K. "Stability of Complex Dynamical Systems: Computational Methods and Comparison Principle." Dept. of Electrical Eng'g, Iowa State Univ., unpublished paper.

IX. ACKNOWLEDGMENTS

I would like to express my heartfelt gratitude to Dr. Anthony N. Michel for the time and attention he has devoted to me and to this work. But for his sustained interest, encouragement, and support during the difficult moments of my life, this dissertation would not have been completed.

I would also like to thank Dr. Richard K. Miller for his valuable time, advice, and encouragement. Thanks are also due to Dr. Brown, Dr. Hale, Dr. Mahmoud, Dr. Triska, and all of my present and former teachers, who made my studies intellectually stimulating.

I would like to thank the Electrical Engineering Department and Engineering Research Institute at the Iowa State Univeristy and National Science Foundation for the financial support.

I would like to thank my family for their constant support and encouragement. I am greatly indebted to my mother and brother for constantly encouraging me to set and attain high goals. Last, but not least, I would like to thank my wife for her patience and invaluable companionship during the preparation of this dissertation.

X. APPENDIX A: STABILITY OF AN INFINITE SET OF MATRICES

The constructive algorithm developed in [4] and [5] is restricted to a finite set of matrices $\underline{A}$. The following theorem shows that it applies to infinite sets as well (see, [4, p. 230]).

Theorem: Let $\underline{A}$ be a set of matrices, and let $E(\underline{A})$ be the set of extreme matrices of $\underline{A}$. The $\kappa[\underline{A}]$ is stable if and only if $E(\underline{A})$ is stable.

Proof: Since $E(\underline{A}) \subset \underline{A}$, the assertion--if $\underline{A}$ is stable then $E(\underline{A})$ is stable--is obvious. Assume $\kappa[A]$ is unstable, and suppose $E(\underline{A})$ is stable. Then there exists $K > 0$ such that, for all $M \varepsilon E(\underline{A}')$, we have $\|Mx\| \leq K \|x\|$ for all $x \varepsilon E^n$. Since $\kappa[\underline{A}]$ is unstable, then by Lemma 3 (see [4]), there exists $L \varepsilon (\kappa[\underline{A}])'$ and $x \varepsilon E^n$ such that $\|Lx\| > K\|x\|$. Now L is a finite product of matrices in $\kappa[\underline{A}]$, say

$$L = M_1 \cdot M_2 \cdot \cdot \cdot M_r, \qquad M_i \varepsilon \kappa[\underline{A}].$$

Since $\kappa[\underline{A}]$ is convex and since $E(\underline{A})$ denotes the set of extreme matrices of $\kappa[\underline{A}]$ then each M_i can be represented as

$$M_i = \sum \lambda_{ij} S_j$$

where $S_j \varepsilon E(\underline{A})$, $\lambda_{ij} \geq 0$, $\sum_{j=1}^{m_i} \lambda_{ij} = 1$. Multiplying this out, we obtain

$$L = \sum_{i=1}^{p} \mu_i S_i'$$

where $\mu_i \geq 0, \sum \mu_i = 1$, and $S_i' \varepsilon E(\underline{A})'$. Now

$$\| Lx \| = \| \sum \mu_i S_i' x \| \leq \sum \| \mu_i S_i' x \| = \sum \| \mu_i Kx \| = \| Kx \|$$

since each $S_i' \in E(\underline{A})'$. Hence, $\| Lx \| \leq \| Kx \|$, a contradiction. Thus, $E(\underline{A})$ is unstable. ■

Since $\underline{A} \subseteq \mathbf{K}[\underline{A}]$, a set of matrices is stable if and only if its extreme points are stable. The convex hull of a set of matrices can be approximated to any arbitrary degree by a polytope containing it: since the set of extreme points of a polytope is finite, the stability of a set can be determined through a polyhedral approximation of that set and along with the use of the above theorem.

XI. APPENDIX B: ROSENBROCK'S OPTIMIZATION METHOD

In [38], Rosenbrock presented an "automatic" method for nonlinear function optimization subject to some constraints, if there are any. This is a sequential search technique which provides an acceleration in the direction as well as in the distance of search. Let $F = F(x_1, \ldots, x_p)$ be the objective function to be optimized. We now summarize his method in the form of an algorithm. For the purpose of simplicity, we present only the unconstrained version of the algorithm.

<u>Step 1</u>: Pick an initial search point $x_c = [x_1, \ldots, x_p]^T$ and refer to it as "<u>current point</u>." Also evaluate function F at this point. Store a set of orthogonal vectors

$$v_1^{(1)} = [1, 0, \ldots, 0]^T$$
$$v_2^{(1)} = [0, 1, \ldots, 0]^T$$
$$\vdots$$
$$v_p^{(1)} = [0, 0, \ldots, 1]^T$$

in the computer. Also, store the step length vector $e = [e_1, \ldots, e_p]^T$, whose elements can be either positive or negative, in the computer.

<u>Step 2</u>: A new search point, which we shall refer to as "<u>trial point</u>," $x_t = x_c + e_1 v_1^{(1)} = [x_1 + e_1, x_2, \ldots, x_p]^T$ is computed and the function F is evaluated at such a point. If $F(x_t)$ is an improvement over $F(x_c)$, then we set $x_c = x_t$, $F(x_c) = F(x_t)$ and

$e = \alpha \cdot e$ ($\alpha = 3$ was used by Rosenbrock). If $F(x_t)$ is not an improvement over $F(x_c)$, then we discard x_t and set $e = -\beta \cdot e$ ($\beta = 0.5$ was used by Rosenbrock). This procedure is repeated until one success and one failure are recorded in the direction of $v_i^{(1)}$, $i = 1, \ldots, p$. When this has happened, it will be interpreted as the completion of one stage of optimization.

Step 3: After completion of one stage of optimization, a new set of orthogonal unit vectors $v_1^{(2)}$, $i = 1, \ldots, p$ are computed through a method presented in [38] and [36]. This process is referred to as rotation of axes.

Step 4: This procedure of generating a new set of orthogonal direction vectors and of searching along these directions is repeated through a succession of stages until some appropriate convergence criterion is satisfied.

XII. APPENDIX C: EXPANSION OF LYAPUNOV MATRIX EQUATION

Bingulac [41] presented an efficient method for expanding the following Lyapunov matrix equation

$$BJ + J^T B = -C$$

(where J is an $n \times n$ matrix, and B and C are $n \times n$ positive definite symmetric matrices) into a system of linear algebraic equations of the form

$$VB_v = -C_v.$$

Here V is an $m \times m$ matrix, where $m = n(n+1)/2$. B_v and C_v are column vectors of order m. Elements of B_v correspond to the upper triangular part of unknown matrix B, i.e.,

$$B_v = [b_{11}, b_{12}, \ldots, b_{1n}, b_{22}, \ldots, b_{2n}, b_{33}, \ldots, b_{(n-1)(n-1)}, b_{(n-1)n}, b_{nn}]^T,$$

while the elements of C_v correspond to the upper triangular part of given matrix C by the following relation:

$$C_v = [c_{11}, c_{12}, \ldots, c_{1n}, c_{22}, \ldots, c_{2n}, c_{33}, \ldots, c_{(n-1)(n-1)}, c_{(n-1)n}, c_{nn}]^T.$$

We now summarize his method in the form of an algorithm.

Step 1: Read the elements of J and C matrices.

Step 2: Convert the upper triangular part of given matrix C into a column vector C_v.

Step 3: Use subroutine EXPAND, which will be presented later, to compute the elements of the $m \times m$ matrix V.

Step 4: Find the inverse of V and then compute $-V^{-1}C_v$ which provides us with the solution of the vector B_v.

Step 5: Form the $n \times n$ Lyapunov matrix B from the elements of column vector B_v.

We now present the subroutine EXPAND used in Step 3.

```
      SUBROUTINE EXPAND (J,V,N)
      DIMENSION L(5,5), J(5,5), V(15,15)
      M = N*(N+1)/2
      K = 0
      DO 10 I = 1,N
      DO 10 J = I,N
      K = K+1
      L(I,J) = K
   10 L(J,I) = K
      DO 11 I=1,M
      DO 11 J=1,M
   11 V(I,J) = 0
      DO 12 I=1,N
      DO 12 J=1,N
      DO 12 K=1,N
   12 V(L(I,K),L(J,K))=A(J,I)+V(L(I,K),L(J,K))
      DO 13 I=1,N
      DO 13 J=1,M
   13 V(L(I,I),J)=2.0*V(L(I,I),J)
      RETURN
      END
```

XIII. APPENDIX D: COMPUTER PROGRAMS

```
C*************************************************************
C
C    THIS PROGRAM DETERMINES AN ESTIMATE FOR THE DOMAIN OF
C    ATTRACTION BY QUADRATIC LYAPUNOV FUNCTION ALGORITHM
C    DEVELOPED IN CHAPTER IV.
C
C    MAIN PROGRAM WILL CALL THE FOLLOWING IMPORTANT
C    SUBROUTINES WHICH IN TURN MAY CALL OTHER SUBROUTINES.
C
C         1. LYAP  AND  LEQT2F:  TO SOLVE THE LYAPUNOV MATRIX
C            EQUATION.
C         2. DATSET: TO GENERATE A GRID ON THE BOUNDARY OF
C            V(X)=D.
C         3. TANPNT: TO FIND THE TANGENT POINTS BETWEEN
C            V(X)=D AND VDOT(X)=0.
C         4. ROSOPT: TO MAXIMIZE THE SIZE OF ESTIMATED
C            DOMAIN OF ATTRACTION.
C
C    ALL THE REQUIRED DATA WILL BE ENTERED IN THE MAIN
C    PROGRAM. TO USE THIS PROGRAM ENTER THE FOLLOWING DATA:
C
C         1. NDM = DIMENSION OF THE SYSTEM.
C         2.  JX = JACOBIAN MATRIX.
C         3.  QX = MATRIX ON THE RIGHT HAND SIDE OF LYAPUNOV
C                  MATRIX EQUATION.
C         4.  ES = STEP SIZES TO BE USED IN THE ROSENBROCKS
C                  OPTIMIZATION PROCEDURE.
C
C*************************************************************
C
C                         MAIN PROGRAM
C
C*************************************************************
      REAL JX(4,4),QX(4,4),L(4,4),JV(10,10),X(10),QZ(10,1)
      REAL A2(3,3),A3(6,6),Q2(3,1),Q3(6,1),W2(18),W3(54)
      REAL WKAREA(130),Y( 50,2),ES(10)
 545  FORMAT(I5)
 555  FORMAT(8F10.6)
 565  FORMAT(3F10.6)
 600  FORMAT(1H0,10F12.7)
 601  FORMAT(1H1)
 602  FORMAT(1H0,'ELEMENT VALUES OF JACOBIAN MATRIX')
 603  FORMAT(1H0,'DIMENSION OF THE SYSTEM=',I5/1H0,'THE NU',
     C'MBER OF INDEPENDENT VARIABLES IN P MATRIX=',I5/)
 604  FORMAT(1H0,'ELEMENT VALUES OF Q MATRIX IN,  JTP+PJ=Q')
 606  FORMAT(1H0,'THE  P  MATRIX( AFTER NORMALIZATION )')
      WRITE(6,601)
C*************************************************************
      READ(5,545)NDM
```

```
      READ(5,555)((JX(I,J),J=1,NDM),I=1,NDM)
      READ(5,555)((QX(I,J),J=1,NDM),I=1,NDM)
      MN=(NDM*(NDM+1))/2
      READ(5,565)(ES(I),I=1,MN)
C***********************************************************
      WRITE(6,603)NDM,MN
      WRITE(6,602)
      DO 11 I=1,NDM
      WRITE(6,600)(JX(I,J),J=1,NDM)
 11   CONTINUE
      WRITE(6,604)
      DO 12 I=1,NDM
      WRITE(6,600)(QX(I,J),J=1,NDM)
 12   CONTINUE
      NA=1
            CALL ASIGN1(QX,QZ,NDM,MN,NA)
            CALL LYAP(JX,JV,NDM,MN,L)
      IA=MN
      M=1
      IDGT=6
      NS=NDM-1
      GO TO(22,23,24),NS
 22   CONTINUE
      DO 30 I=1,MN
      Q2(I,1)=QZ(I,1)
      DO 30 J=1,MN
 30   A2(I,J)=JV(I,J)
      CALL LEQT2F(A2,M,MN,IA,Q2,IDGT,W2,IER)
      DO 31 I=1,MN
 31   QZ(I,1)=Q2(I,1)
      GO TO 25
 23   CONTINUE
      DO 32 I=1,MN
      Q3(I,1)=QZ(I,1)
      DO 32 J=1,MN
 32   A3(I,J)=JV(I,J)
      CALL LEQT2F(A3,M,MN,IA,Q3,IDGT,W3,IER)
      DO 33 I=1,MN
 33   QZ(I,1)=Q3(I,1)
      GO TO 25
 24   CONTINUE
      CALL LEQT2F(JV,M,MN,IA,QZ,IDGT,WKAREA,IER)
 25   CONTINUE
      XBIG=-99999.0
      DO 2 I=1,MN
      AQ=ABS(QZ(I,1))
      IF(AQ .GT. XBIG)XBIG=AQ
 2    X(I)=QZ(I,1)
      DO 3 I=1,MN
```

```
 3      X(I)=X(I)/XBIG
        NA=2
               CALL ASIGN1(QX, X,NDM,MN,NA)
        WRITE(6,606)
        DO 43 I=1,NDM
        WRITE(6,600)(QX(I,J),J=1,NDM)
 43     CONTINUE
        DO 51 I=1,NDM
        Y(1,I)=1.0
 51     Y(2,I)=-Y(1,I)
        ITEMP=2
        NCP=11
               CALL DATSET(Y,NDM,NCP,ITEMP)
        CALL TANPNT(Y,NDM,X(1),X(2),X(3),ITEMP,LIAPST,VSML)
        IF(LIAPST .EQ. 1)GO TO 1000
               CALL ROSOPT(X,MN,NDM,Y,ITEMP,VSML,ES)
 1000   CONTINUE
        STOP
        END
C
C
        SUBROUTINE ASIGN1(QX,QZ,NDM,MN,NA)
        REAL QX(4,4),QZ(10,1)
        K=0
        IF(NA .EQ. 2)GO TO 3
        DO 1 I=1,NDM
        DO 1 J=I,NDM
        K=K+1
 1      QZ(K,1)=QX(I,J)
        RETURN
 3      CONTINUE
        DO 2 I=1,NDM
        DO 2 J=I,NDM
        K=K+1
        QX(J,I)=QZ(K,1)
 2      QX(I,J)=QZ(K,1)
        RETURN
        END
C
C
        SUBROUTINE LYAP(JX,JV,NDM,MN,L)
        REAL JX(4,4),JV(10,10),L(4,4)
        K=0
        DO 1 I=1,NDM
        DO 1 J=I,NDM
        K=K+1
        L(I,J)=K
 1      L(J,I)=K
        DO 2 I=1,MN
```

```
      DO 2 J=1,MN
 2    JV(I,J)=0.0
      DO 3 I=1,NDM
      DO 3 J=1,NDM
      DO 3 K=1,NDM
      LI=L(I,K)
      LJ=L(J,K)
 3    JV(LI,LJ)=JX(J,I)+JV(LI,LJ)
      DO 4 I=1,NDM
      DO 4 J=1,MN
      LK=L(I,I)
 4    JV(LK,J)=2.0*JV(LK,J)
      RETURN
      END
C
C
      SUBROUTINE DATSET(X,NDM,NCP,ITEMP)
      DIMENSION IX(16,4),C(21),Z(4,4),X(   50,2),NZ(2),XX(4)
      DIMENSION YY(4)
 602  FORMAT(1X,I5,5X,4F10.6)
 606  FORMAT(1H0,'ITEMP=',I5)
      NT=2**NDM
      NTM=NT-1
      NDD=NDM-1
      NCM=NCP-1
      R2=NCM
      NST=0
      DO 41 I=1,NDM
 41   NST=NST+(NDM-I)
      DO 1 I=1,NTM
      IZ=I-1
      DO 2 J=1,NDM
      IX(I,J)=1
      IY=IZ-2**(NDM-J)
      IF(IY .GE. 0)IZ=IY
      IF(IY .LT. 0)IX(I,J)=-1
 2    CONTINUE
 1    CONTINUE
      DO 3 I=1,NCP
      R1=(NCP-I)
 3    C(I)=R1/R2
      DO 4 I=1,NDM
      ITEMP=ITEMP+1
      DO 4 J=1,NDM
      Z(I,J)=0.0
      IF(I .EQ. J)Z(I,J)=1.0
      IR=ITEMP+NDM
      X(IR,J)=-Z(I,J)
 4    X(ITEMP,J)=Z(I,J)
```

```
      ITEMP=IR
      IP=ITEMP+1
      DO 10 I=1,NDD
      I1=I+1
      DO 11 J=I1,NDM
      DO 12 K=2,NCM
      ITEMP=ITEMP+1
      DO 13 L=1,NDM
13    X(ITEMP,L)=C(K)*Z(I,L)+(1.0-C(K))*Z(J,L)
12    CONTINUE
11    CONTINUE
10    CONTINUE
      NRT=IR
      KT=IR+(NCP-2)
      IF(NDM .EQ. 2)GO TO 44
      NRT=IR+(NCP-2)*NST*2
      NRT=2*NDM+(NCP-2)*NST*2
      KP=ITEMP
      DO 31 I=IP,KP
      ITEMP=ITEMP+1
      DO 32 J=1,NDM
32    X(ITEMP,J)=-X(I,J)
31    CONTINUE
      DO 17 I=1,ITEMP
      WRITE(6,602)I,(X(I,J),J=1,NDM)
17    CONTINUE
      IF(NDM .EQ. 4)NDD=1
      DO 60 I=1,NDD
      I1=I+1
      DO 61 J=I1,NDM
      DO 62 K=2,NCM
      DO 63 L=1,NDM
63    XX(L)=C(K)*Z(I,L)+(1.0-C(K))*Z(J,L)
      IF(NDM .GT. 3)GO TO 22
      DO 14 M=1,NDM
      IF(M .EQ. I .OR. M .EQ. J)GO TO 14
      DO 15 N=2,NCM
      ITEMP=ITEMP+1
      DO 16 NA=1,NDM
16    X(ITEMP,NA)=C(N)*XX(NA)+(1.0-C(N))*Z(M,NA)
15    CONTINUE
14    CONTINUE
      GO TO 62
22    CONTINUE
      KR=0
      DO 23 NR=1,NDM
      IF(NR .EQ. I .OR. NR .EQ. J)GO TO 23
      KR=KR+1
      NZ(KR)=NR
```

```
23    CONTINUE
      DO 72 KA=2,NCM
      DO 73 KB=1,NDM
73    YY(KB)=C(KA)*Z(NZ(1),KB)+(1.0-C(KA))*Z(NZ(2),KB)
      DO 74 KC=2,NCM
      ITEMP=ITEMP+1
      DO 75 KD=1,NDM
75    X(ITEMP,KD)=C(KC)*XX(KD)+(1.0-C(KC))*YY(KD)
74    CONTINUE
72    CONTINUE
62    CONTINUE
61    CONTINUE
60    CONTINUE
      KT=ITEMP
44    CONTINUE
      NRP=NRT+1
      DO 6 I=1,NTM
      DO 7 J=NRP,KT
      ITEMP=ITEMP+1
      DO 8 K=1,NDM
8     X(ITEMP,K)=IX(I,K)*X(J,K)
7     CONTINUE
6     CONTINUE
      WRITE(6,606)ITEMP
      DO 38 I=1,ITEMP
      WRITE(6,602)I,(X(I,J),J=1,NDM)
38    CONTINUE
      RETURN
      END
C
C
      SUBROUTINE TANPNT(VTP,NDM,AA,BB,CC,ITEMP,LIAPST,VSML)
      DIMENSION VTP(50,2),YX(50,2)
110   FORMAT(1H0,'*****************************************',
     C'**************************************************'/)
107   FORMAT(1H0,'THE REGION OF ASYMPTOTIC STABILITY IS FO',
     C'R ALL  X  SUCH THAT V(X) IS < .OR. =',F15.8/)
600   FORMAT(1H0,'KTEMP=',I5/)
601   FORMAT(1H0,I5,2F15.10)
      KTEMP=0
            CALL ZZZZ0(AA,BB,CC,KTEMP,VTP,VSML,NDM)
      WRITE(6,600)KTEMP
      LTEMP=KTEMP
      IF(KTEMP .EQ. 0)GO TO 51
      DO 50 I=1,LTEMP
      KTEMP=KTEMP+1
      VTP(KTEMP,1)=-VTP(I,1)
      VTP(KTEMP,2)=-VTP(I,2)
50    CONTINUE
```

```
 51   CONTINUE
      KTEMP=ITEMP
      DO 10 I=1,KTEMP
      VTEMP=AA*(VTP(I,1)**2)+2.0*BB*VTP(I,1)*VTP(I,2)
      VTEMP=VTEMP+CC*(VTP(I,2)**2)
      CS=SQRT(VSML/VTEMP)
      YX(I,1)=VTP(I,1)*CS
      YX(I,2)=VTP(I,2)*CS
 10   CONTINUE
      NPOINT=KTEMP
      CSML=1.0
      LIAPST=0
      PCML=0.0
      CALL VDXCHK(YX,CSML,NPOINT,VSML,AA,BB,CC,LIAPST,PCML)
      IF(LIAPST .EQ. 1)VSML=0.0
      WRITE(6,600)KTEMP
      WRITE(6,601)(I, YX(I,1), YX(I,2),I=1,KTEMP)
      WRITE(6,110)
      WRITE(6,107)VSML
      WRITE(6,110)
      RETURN
      END
C
C
      SUBROUTINE ZZZZ0(AA,BB,CC,KTEMP,VTP,VSML,N)
      DIMENSION ST( 50,2),X(2),SS( 50,2),VTP( 50,2),VV( 50)
 100  FORMAT(1X,2F10.4)
 101  FORMAT(1H0,'GENERATED SEARCH POINTS FOR FINDING',
     C' TANGENT POINTS'//)
 102  FORMAT(1H0,'NUMBER OF COMPUTED TANGENT POINTS=',I5//)
 103  FORMAT(1H0,I5,2F15.10)
 108  FORMAT(1H0,'THE VALID TANGENT POINTS ARE'//)
 109  FORMAT(1H0,I5,2F15.10)
 600  FORMAT(1H0,' I=',I5,5X,'X(1)=',F15.7,5X,'X(2)=',F15.7)
      K=16
            CALL ZZZZ1(ST,K,N)
      WRITE(6,101)
      WRITE(6,100)(ST(I,1),ST(I,2),I=1,K)
      KK=K
      ITEMP=0
      DO 10 I=1,KK
      X(1)=ST(I,1)
      X(2)=ST(I,2)
            CALL ZZZZ2(ST,X,N,ITEMP,AA,BB,CC,SS)
      WRITE(6,600)I,X(1),X(2)
 10   CONTINUE
      KTEMP=0
      IF(ITEMP .EQ. 0)VSML=5.0
      IF(ITEMP .EQ. 0)RETURN
```

```
      WRITE(6,102)ITEMP
      WRITE(6,103)(I,SS(I,1),SS(I,2),I=1,ITEMP)
      VSML=70.0
      IVSML=1
      DO 30 LL=1,ITEMP
      X1=SS(LL,1)
      X2=SS(LL,2)
      VXT=AA*(X1**2)+2.0*BB*X1*X2+CC*(X2**2)
      VV(LL)=VXT
      IF(VXT .LT. VSML)IVSML=LL
      IF(VXT .LT. VSML)VSML=VXT
 30   CONTINUE
      KTEMP=0
            CALL ZZZZ3(SS,ITEMP,KTEMP,VSML,VV,VTP,IVSML)
      WRITE(6,108)
      WRITE(6,109)(I,VTP(I,1),VTP(I,2),I=1,KTEMP)
      RETURN
      END
C
C
      SUBROUTINE ZZZZ1(ST,K,N)
      INTEGER N,K,IP,M(2),IW(9),IER,J
      REAL A(2),B(2),S(2),ST( 50,2)
 101  FORMAT(1H0,'NUMBER OF POINTS TO BE GENERATED=',I5)
 103  FORMAT(1H0,'.....  IER   FROM ZSRCH =',I5)
      A(1)=-3.0
      B(1)=3.0
      A(2)=-3.0
      B(2)=3.0
      IP=0
      WRITE(6,101)K
      DO 5 J=1,K
            CALL ZSRCH(A,B,N,K,IP,S,M,IW,IER)
      ST(J,1)=S(1)
      ST(J,2)=S(2)
 5    CONTINUE
      WRITE(6,103)IER
      RETURN
      END
C
C
      SUBROUTINE ZZZZ2(ST,X,N,ITEMP,AA,BB,CC,SS)
      COMMON/APARMS/AAA,BBB,CCC
      INTEGER NSIG,N,ITMAX,IER
      REAL X(2),WA(8),PAR(1),EPS,AUX
      DIMENSION ST( 50,2),SS( 50,2)
      EXTERNAL AUX
      AAA=AA
      BBB=BB
```

```
      CCC=CC
      NSIG=4
      EPS=1.0E-4
      ITMAX=100
      IER=0
            CALL ZSYSTM(AUX,EPS,NSIG,N,X,ITMAX,WA,PAR,IER)
      IF(IER .EQ. 130 .OR. IER .EQ. 129)GO TO 10
      ITEMP=ITEMP+1
      SS(ITEMP,1)=X(1)
      SS(ITEMP,2)=X(2)
 10   CONTINUE
      RETURN
      END
C
C
      SUBROUTINE ZZZZ3(SS,ITEMP,KTEMP,VSML,VV,VTP,IVSML)
      DIMENSION SS( 50,2),VV( 50),VTP( 50,2)
      PREC=0.00001
      PREC1=0.001
      KTEMP=KTEMP+1
      VTP(KTEMP,1)=SS(IVSML,1)
      VTP(KTEMP,2)=SS(IVSML,2)
      DO 10 I=1,ITEMP
      VPCT=(VSML-VV(I))/VV(I)
      VPCT=ABS(VPCT)
      TEMP=ABS(ABS(SS(IVSML,1))-ABS(SS(I,1)))
      TEMP=ABS(ABS(SS(IVSML,2))-ABS(SS(I,2)))+TEMP
      IF(TEMP .LE. PREC1)GO TO 15
      IF(VPCT .GT. PREC)GO TO 15
      KTEMP=KTEMP+1
      VTP(KTEMP,1)=SS(I,1)
      VTP(KTEMP,2)=SS(I,2)
 15   CONTINUE
 10   CONTINUE
      RETURN
      END
C
C
      REAL FUNCTION AUX(X,K,PAR)
      COMMON/APARMS/AA,BB,CC
      INTEGER K
      REAL X(1),PAR(1)
      REAL H(4)
      X1=X(1)
      X2=X(2)
           CALL FN1(X1,X2,FX1,FX2,CF1,CF2,AA,BB,CC)
      IF(K .NE. 2)GO TO 3
           CALL FN2(X1,X2,DFX11,DFX12,DFX21,DFX22)
      DCF11=2.0*AA
```

```
      DCF12=2.0*BB
      DCF22=2.0*CC
      DVDX1=(DCF11*FX1+CF1*DFX11)+(DCF12*FX2+CF2*DFX21)
      DVDX2=(DCF12*FX1+CF1*DFX12)+(DCF22*FX2+CF2*DFX22)
 3    CONTINUE
      GO TO(1,2),K
 1    CONTINUE
      AUX=CF1*FX1+CF2*FX2
      RETURN
 2    CONTINUE
      DTEMP=SQRT(CF1**2+CF2**2)
      ETEMP=SQRT(DVDX1**2+DVDX2**2)
      DXD=DTEMP*ETEMP
      FTEMP=(DVDX1*CF1+DVDX2*CF2)/DXD
      H(1)=(CF1/DTEMP)-FTEMP*(DVDX1/ETEMP)
      H(2)=(CF2/DTEMP)-FTEMP*(DVDX2/ETEMP)
      AUX=ABS(H(1))+ABS(H(2))
      RETURN
      END
C
C
      SUBROUTINE VDXCHK(YX,CSML,N,VSML,AA,BB,CC,LIAPST,PCML)
      DIMENSION YX(50,2)
 600  FORMAT( 1X,'CSML=',F15.8,'PCSML=',F15.8,5X,
     C'PCML=',F15.8)
 601  FORMAT( 1X,'FROM SUBROUTINE VDXCHK, THE UPPER',
     C' BOUND ON VDOT(X) IS =',F15.8)
 602  FORMAT(1H0,' I=',I5,5X,'LIAPST=',I5,5X,'XP1=',F15.8)
      NPOINT=N
      DMAX=-999999.0
      PREC=-0.000001
      PCSML=CSML
      DO 1 I=1,NPOINT
      X1=YX(I,1)*PCSML
      X2=YX(I,2)*PCSML
            CALL FN1(X1,X2,FX1,FX2,CF1,CF2,AA,BB,CC)
      DVX=FX1*CF1+FX2*CF2
      IF(DVX .LT. 0.0)GO TO 11
      XXX=PCSML
      PXX=PCML
 25   CONTINUE
      XDIF=ABS(XXX-PXX)
      IF(XDIF .LE. 0.0000015 .AND. DVX .LT. 0.0)GO TO 70
      XP1=PXX+(XXX-PXX)*0.5
      X1=XP1*YX(I,1)
      X2=XP1*YX(I,2)
            CALL FN1(X1,X2,FX1,FX2,CF1,CF2,AA,BB,CC)
      DVX=FX1*CF1+FX2*CF2
      IF(DVX .GE. PREC .AND. DVX .LT. 0.0)GO TO 70
```

```
      IF(DVX .LT. PREC)GO TO 80
      IF(XP1 .LE. 0.001953125)LIAPST=1
      IF(LIAPST .NE. 1)GO TO 119
      WRITE(6,602)I,LIAPST,XP1
119   CONTINUE
      IF(LIAPST .EQ. 1)GO TO 120
      XXX=XP1
      GO TO 25
80    PXX=XP1
      GO TO 25
70    PCSML=XP1
11    CONTINUE
      IF(DVX .GT. DMAX)DMAX=DVX
1     CONTINUE
      WRITE(6,601)DMAX
      WRITE(6,600)CSML,PCSML,PCML
120   CONTINUE
      IF(CSML .EQ. PCSML)GO TO 3
      DO 2 I=1,NPOINT
      YX(I,1)=PCSML*YX(I,1)
2     YX(I,2)=PCSML*YX(I,2)
3     CONTINUE
      VSML=PCSML*PCSML*VSML
      RETURN
      END
C
C
      SUBROUTINE ROSOPT(X,P,NDM,YX,NPOINT,VSML,ES)
      DIMENSION X(10),E(10),V(10,10),SA(10),D(10),A(10,10)
      DIMENSION BX(10),VV(10,10),EINT(10),VM(10),PX(4,4)
      DIMENSION G(50),H(50),AL(50),PH(50),CG(50),CH(50)
      DIMENSION XST(10),ES(10),XQ(10,1),B(10,10),YX(50,2)
      DIMENSION YZ(50,2),CX(50)
      COMMON KOUNT
      INTEGER P,PR,R,C
      REAL LC,JX,NVMIN
601   FORMAT(1H1,10X,'ROSENBROCK HILLCLIMB PROCEDURE'/)
620   FORMAT(/1H0,'TOTAL NUMBER OF BOUNDARY POINTS=',I5/1H0,
     C'TOTAL NUMBER OF CONSTRAINTS=',I5/1H0,'LIMIT ON NUMB',
     C'ER OF FUNCTION EVALUATIONS=',I5/)
639   FORMAT( /1H0,'NUMBER OF FUNCTION EVALUATIONS=',I5)
666   FORMAT( 1X,'X(1)=',F15.8,5X ,'X(2)=',F15.8,5X,'X(3)=',
     CF15.8)
648   FORMAT(1H0,'*******  IFAIL=',I5,'******')
638   FORMAT(1H0,'*****************************************',
     C'*************************************************')
636   FORMAT( 1X,'FROM SUBROUTINE VDXCHK,   NVMIN=',F15.8)
641   FORMAT( 1X,'FROM SUBROUTINE VVV,      NVMIN=',F15.8)
640   FORMAT( 1X,'FROM SUBROUTINE VCHK, LIAPST=',I5)
```

```
665  FORMAT(/1H0,'SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS',
    C'SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS')
602  FORMAT(//,2X,'STAGE',8X,'FUNCTION',12X,'PROGRESS',9X,
    C'LATERAL PROGRESS')
603  FORMAT(2X,I5,3E20.12)
604  FORMAT(/,2X,'NUMBER OF FUNCTION EVALUATIONS =',I8)
605  FORMAT(/,2X,'VALUES OF X AT THIS STAGE')
606  FORMAT(/,2X,3(2HX(,I2,4H) = ,1PE14.6,4X))
607  FORMAT(///,2X,'THE STARTING POINT MUST NOT VIOLATE',
    C' THE CONSTRAINTS. IT APPEARS TO HAVE DONE SO.')
608  FORMAT(///,2X,'FINAL DIRECTION VECTOR MATRIX')
609  FORMAT(/,2X,3(2HV(,I2,1H,,I2,4H) = ,F16.8,4X))
610  FORMAT(//,2X,'FINAL STEP SIZES')
611  FORMAT(/,2X,3(2HS(,I2,4H) = ,1PE14.6,4X))
     DO 9998 J=1,P
     XQ(J,1)=X(J)
9998 E(J)=ES(J)
     NA=2
           CALL ASIGN1(PX,XQ,NDM,P,NA)
     PF0=X(1)*X(3)-X(2)*X(2)
     M=-1
     LOOPY=100
     PR=1
     NSTEP=1
     TDIFF=0.0000001
     NVMIN=VSML
     KTEMP=NPOINT
     L=NPOINT+NDM
     MKOUNT=P*50
     MFAIL=14*P
     DO 818 NZ=1,KTEMP
     YZ(NZ,1)=YX(NZ,1)
818  YZ(NZ,2)=YX(NZ,2)
10   CONTINUE
     WRITE(6,601)
     WRITE(6,620)NPOINT,L,MKOUNT
     INDEX=1
     KOUNT=0
9999 CONTINUE
     IFAIL=0
     LAP=PR-1
     LOOP=0
     ISW=0
     INIT=0
     TERM=0.0
     CKONST=1.0E-6
     F1=0.001
     F0=-9999.0
     MMF=1
```

```
      IF(F1 .LT. CKCNST)MMF=0
      CCONS1=1.0E-9
      CCONS2=1.0E-16
      CCONS3=1.0E-20
      DO 40 K=1,L
      IF(K .GT. NDM)GO TO 39
      CG(K)=0.00005
      CH(K)=10000.00005
      AL(K)=(CH(K)-CG(K))*5.0*CCONS1
      GO TO 38
39    CONTINUE
      CG(K)=CCONS2
      CH(K)=10000.0+CCONS2
      AL(K)=(CH(K)-CG(K))*CCONS3
38    CONTINUE
40    CONTINUE
      DO 60 I=1,P
      DO 60 J=1,P
      V(I,J)=0.0
      IF(I-J)60,61,60
61    V(I,J)=1.0
60    CONTINUE
      DO 65 KK=1,P
      EINT(KK)=E(KK)
65    CONTINUE
1000  CONTINUE
      DO 70 J=1,P
      IF(NSTEP .EQ. 0)E(J)=EINT(J)
      SA(J)=2.0
70    D(J)=0.0
      IF(MMF .EQ. 0)GO TO 450
80    I=1
      IF(INIT .EQ. 0)GO TO 120
90    CONTINUE
      DO 110 K=1,P
110   X(K)=X(K)+E(I)*V(I,K)
      DO 50 K=1,L
50    H(K)=F0
120   CONTINUE
      DO 8778 JNR=1,P
8778  XQ(JNR,1)=X(JNR)
      NA=2
            CALL ASIGN1(PX,XQ,NDM,P,NA)
      KNT=KOUNT+1
      IF(KNT .GT. MKOUNT)TERM=1.0
      IF(TERM .EQ. 1.0)GO TO 450
      CALL POSDEF(PX,CX,YX,NDM,NPOINT,INDEX,IVIOLT,F0,F9,M)
      IF(IVIOLT .EQ. 1)GO TO 420
      MMF=1
```

```
      IF(F9 .LT. CKONST)MMF=0
      IF(MMF .EQ. 0)GO TO 125
      F1=M*F9
      IF(ISW .EQ.0)F0=F1
      ISW=1
 125  CONTINUE
      J=1
 130  CONTINUE
      XC=CX(J)
      LC=CG(J)
      UC=CH(J)
      IF(XC .LE. LC)GO TO 420
      IF(XC .GE. UC)GO TO 420
      IF(XC .LT. LC+AL(J))GO TO 140
      IF(XC .GT. UC-AL(J))GO TO 140
      H(J)=F0
      GO TO 210
 140  CONTINUE
      BW=AL(J)
      IF(XC .LE.LC .OR. UC .LE. XC)GO TO 150
      IF(LC .LT. XC .AND. XC .LT. LC+BW)GO TO 160
      IF(UC-BW .LT. XC .AND. XC .LT. UC)GO TO 170
      PH(J)=1.0
      GO TO 210
 150  PH(J)=0.0
      GO TO 190
 160  PW=(LC+BW-XC)/BW
      GO TO 180
 170  PW=(XC-UC+BW)/BW
 180  PH(J)=1.0-3.0*PW+4.0*PW*PW-2.0*PW*PW*PW
 190  F1=H(J)+(F1-H(J))*PH(J)
210   CONTINUE
      IF(J .EQ. L)GO TO 220
      J=J+1
      GO TO 130
 220  INIT=1
      IF(F1 .LT. F0)GO TO 420
      IF(KOUNT .EQ. 1)GO TO 1515
            CALL VVV(YX,NPOINT,NVMIN,X(1),X(2),X(3))
      LIAPST=0
            CALL VCHK(YZ,KTEMP,LIAPST,X(1),X(2),X(3),NVMIN)
      IF(LIAPST .EQ. 1)GO TO 420
      WRITE(6,639)KNT
      WRITE(6,641)NVMIN
      WRITE(6,640)LIAPST
      NPOINT=KTEMP
      L=NPOINT+NDM
      PCML=1.0
      CSML=3.0
```

```
      NPO=NPOINT
      LIA=LIAPST
      CALL VDXCHK(YZ,CSML,NPO,NVMIN,X(1),X(2),X(3),LIA,PCML)
      LIAPST=LIA
      WRITE(6,636)NVMIN
      IFAIL=0
      DO 888 NZ=1,KTEMP
      YX(NZ,1)=YZ(NZ,1)
 888  YX(NZ,2)=YZ(NZ,2)
      F0=F1
      WRITE(6,666)X(1),X(2),X(3)
      WRITE(6,638)
 1515 CONTINUE
      DO 1514 ITI=1,P
 1514 XST(ITI)=X(ITI)
      IF(MMF .EQ. 0)TERM=1.0
      IF(TERM .EQ. 1.0)GO TO 450
      D(I)=D(I)+E(I)
      E(I)=3.0*E(I)
      F0=F1
      IF(SA(I) .GE. 1.5)SA(I)=1.0
 230  DO 240 JJ=1,P
      IF(SA(JJ) .GE. 0.5)GO TO 440
 240  CONTINUE
C     AXES ROTATION
      DO 250 R=1,P
      DO 250 C=1,P
 250  VV(C,R)=0.0
      DO 260 R=1,P
      KR=R
      DO 260 C=1,P
      DO 265 K=KR,P
 265  VV(R,C)=D(K)*V(K,C)+VV(R,C)
 260  B(R,C)=VV(R,C)
      BMAG=0.0
      DO 280 C=1,P
      BMAG=BMAG+B(1,C)*B(1,C)
 280  CONTINUE
      BMAG=SQRT(BMAG)
      BX(1)=BMAG
      DO 310 C=1,P
 310  V(1,C)=B(1,C)/BMAG
      DO 390 R=2,P
      IR=R-1
      DO 390 C=1,P
      SUMVM=0.0
      DO 320 KK=1,IR
      SUMAV=0.0
      DO 330 KJ=1,P
```

```
330   SUMAV=SUMAV+VV(R,KJ)*V(KK,KJ)
320   SUMVM=SUMAV*V(KK,C)+SUMVM
390   B(R,C)=VV(R,C)-SUMVM
      DO 340 R=2,P
      BBMAG=0.0
      DO 350 K=1,P
350   BBMAG=BBMAG+B(R,K)*B(R,K)
      BBMAG=SQRT(BBMAG)
      DO 340 C=1,P
340   V(R,C)=B(R,C)/BBMAG
      LOOP=LOOP+1
      LAP=LAP+1
      IF(LAP .EQ.PR)GO TO 450
      GO TO 1000
420   IF(INIT .EQ. 0)GO TO 450
      IFAIL=IFAIL+1
      IF(IFAIL .LT. MFAIL)GO TO 421
      WRITE(6,648)IFAIL
      TERM=1.0
      GO TO 450
421   CONTINUE
      DO 430 IX=1,P
430   X(IX)=X(IX)-E(I)*V(I,IX)
      E(I)=-0.5*E(I)
      IF(SA(I) .LT. 1.5)SA(I)=0.0
      GO TO 230
440   CONTINUE
      IF(I .EQ. P)GO TO 80
      I=I+1
      GO TO 90
450   CONTINUE
      WRITE(6,665)
      WRITE(6,602)
      WRITE(6,603)LOOP,F0,BMAG,BBMAG
      WRITE(6,604)KOUNT
      WRITE(6,605)
      WRITE(6,606)(JM,X(JM),JM=1,P)
      LAP=0
      IF(INIT .EQ. 0)GO TO 470
      IF(TERM .EQ. 1.0)GO TO 480
      IF(KOUNT .GE. MKOUNT)GO TO 480
      IF(LOOP .EQ. 2 .AND. INDEX .EQ. 1)GO TO 480
      IF(LOOP .GE. LOOPY)GO TO 480
      WRITE(6,665)
      GO TO 1000
470   WRITE(6,607)
480   CONTINUE
490   WRITE(6,608)
      DO 696 J=1,P
```

```
      WRITE(6,609)(J,I,V(J,I),I=1,P)
 696  CONTINUE
      WRITE(6,610)
      WRITE(6,611)(J,E(J),J=1,P)
      IF(INDEX .NE. 1)GO TO 8888
      IF(IFAIL .GE. MFAIL)GO TO 8888
      INDEX=INDEX+1
      DO 9997 JNR=1,P
 9997 E(JNR)=ES(JNR)
      DO 2 J=1,P
 2    X(J)=XST(J)
      GO TO 9999
 8888 CONTINUE
      RETURN
      END
C
C
      SUBROUTINE POSDEF(P,CX,YX,NDM,NP,INDEX,IVIOLT,F0,F9,M)
      DIMENSION YX( 50,2),CX( 50),P(4,4),Q(4,4)
      COMMON KOUNT
      NPOINT=NP
      KOUNT=KOUNT+1
      IVIOLT=0
      TRACE=0.0
      LSN=0
      DO 100 I=1,NDM
      TRACE=TRACE+P(I,I)
            CALL DETMNT(P,I,DET)
      LSN=LSN+1
      CX(LSN)=DET
      IF(DET .LT. 0.0)IVIOLT=1
 100  CONTINUE
      IF(TRACE .LT. 0.0)IVIOLT=1
      IF(IVIOLT .NE. 1)GO TO 110
      F9=9999.0
      GO TO 120
 110  CONTINUE
      F9=TRACE
      IF(INDEX .EQ. 2)F9=CX(NDM)
      F7=M*F9
      IF(F7 .LT. F0)IVIOLT=1
      IF(IVIOLT .EQ. 1)GO TO 120
      DO 101 I=1,NPOINT
      LSN=LSN+1
      P11=P(1,1)
      P12=P(1,2)
      P22=P(2,2)
      CALL FN1(YX(I,1),YX(I,2),FX1,FX2,CF1,CF2,P11,P12,P22)
      DET=FX1*CF1+FX2*CF2
```

```
      DET=-DET
      CX(LSN)=DET
      IF(DET .GT. 0.0)GO TO 101
      IVIOLT=1
      GO TO 120
 101  CONTINUE
 120  CONTINUE
      RETURN
      END
C
C
      SUBROUTINE DETMNT(A,KC,DET)
      DIMENSION A(4,4),B(4,4)
      IREV=0
      DO 1 I=1,KC
      DO 1 J=1,KC
 1    B(I,J)=A(I,J)
      DO 20 I=1,KC
      K=I
 9    IF(B(K,I))10,11,10
 11   K=K+1
      IF(K-KC)9,9,51
 10   IF(I-K)12,14,51
 12   CONTINUE
      DO 13 M=1,KC
      TEMP=B(I,M)
      B(I,M)=B(K,M)
 13   B(K,M)=TEMP
      IREV=IREV+1
 14   II=I+1
      IF(II .GT. KC)GO TO 20
      DO 17 M=II,KC
 18   IF(B(M,I))19,17,19
 19   TEMP=B(M,I)/B(I,I)
      DO 16 N=I,KC
 16   B(M,N)=B(M,N)-B(I,N)*TEMP
 17   CONTINUE
 20   CONTINUE
      DET=1.0
      DO 2 I=1,KC
 2    DET=DET*B(I,I)
      DET=(-1.0)**IREV*DET
      RETURN
 51   DET=0.0
      RETURN
      END
C
C
      SUBROUTINE VVV(YX,NPOINT,NVMIN,AA,BB,CC)
```

```
      DIMENSION YX( 50,2)
      REAL NVMIN
      NVMIN=0.0
      DO 1 I=1,NPOINT
      V=AA*(YX(I,1)**2)+2.0*BB*YX(I,1)*YX(I,2)+
     CCC*(YX(I,2)**2)
      IF(NVMIN .LT.V)NVMIN=V
 1    CONTINUE
      RETURN
      END
C
C
      SUBROUTINE VCHK(YZ,KTEMP,LIAPST,AA,BB,CC,NVMIN)
      DIMENSION YZ( 50,2),YY( 50,2)
      REAL NVMIN
      DO 1 I=1,KTEMP
      V=AA*(YZ(I,1)**2)+2.0*BB*YZ(I,1)*YZ(I,2)+
     CCC*(YZ(I,2)**2)
      CS=SQRT(NVMIN/V)
      YY(I,1)=CS*YZ(I,1)
      YY(I,2)=CS*YZ(I,2)
      CALL FN1(YY(I,1),YY(I,2),FX1,FX2,CF1,CF2,AA,BB,CC)
      DVX=FX1*CF1+FX2*CF2
      IF(DVX .GE. 0.0)LIAPST=1
      IF(LIAPST .EQ. 1)GO TO 2
 1    CONTINUE
 2    CONTINUE
      IF(LIAPST .NE. 0)GO TO 4
      DO 5 I=1,KTEMP
      YZ(I,1)=YY(I,1)
 5    YZ(I,2)=YY(I,2)
 4    CONTINUE
      RETURN
      END
C
C
      SUBROUTINE FN1(X1,X2,FX1,FX2,CF1,CF2,AA,BB,CC)
C     PROVIDE FX1=X1DOT AND FX2=X2DOT.
      CF1=2.0*(AA*X1+BB*X2)
      CF2=2.0*(BB*X1+CC*X2)
      RETURN
      END
C
C
      SUBROUTINE FN2(X1,X2,DFX11,DFX12,DFX21,DFX22)
C     PROVIDE JACOBIAN MATRIX.
      RETURN
      END
```

```
C*************************************************************
C
C    THIS PROGRAM DETERMINES AN ESTIMATE FOR THE DOMAIN OF
C    ATTRACTION BY NORM LYAPUNOV FUNCTION ALGORITHM DEVELOPED
C    IN CHAPTER V.
C
C    MAIN PROGRAM WILL CALL THE FOLLOWING IMPORTANT
C    SUBROUTINES WHICH IN TURN MAY CALL OTHER SUBROUTINES.
C
C         1. LINES: TO CONSTRUCT LINES.
C         2. MATMLT: TO MULTIPLY TWO NXN MATRICES.
C         3. COORDT: TO COMPUTE THE POINTS.
C         4. CHECK, COMPAR AND ELIMINT: TO ELIMINATE THE
C            POINTS WHICH ARE INSIDE THE CONVEX REGION.
C         5. REORDR: TO STORE THE GENERATED POINTS AT A
C            PROPER POSITION.
C         6. RNORML: TO CONSTRUCT NORMALS FOR THE LINES.
C         7. COEF: TO CHECK THE NEGATIVE DEFINITENESS OF
C            VDOT(X).
C
C    ALL THE REQUIRED DATA WILL BE ENTERED IN THE MAIN
C    PROGRAM. TO USE THIS PROGRAM ENTER THE FOLLOWING DATA:
C
C         1. NRUNS = TOTAL NUMBER OF RUNS TO BE MADE.
C         2.    NM = TOTAL NUMBER OF EXTREME MATRICES.
C         3.   IRR = TOTAL NUMBER OF EXTREME VERTICES OF
C                    INITIAL CONVEX REGION.
C         4.     X = EXTREME VERTICES OF INITIAL CONVEX
C                    REGION.
C
C*************************************************************
C
C                         MAIN PROGRAM
C
C*************************************************************
      DIMENSION X(512,2),PX(512,2),PY(512,2),G(512,2),H(512)
      DIMENSION RN(512,2),E(2,2),ISEG(512),CF(512)
      REAL WK(2),WABS(2),MI(2,2),MP(2,2)
      INTEGER ND,IAN,IJOB,IER,IZZ
      COMPLEX W(2),Z(2,2),ZN
      COMMON CF,LIAPST,ISTEMP
 1    FORMAT(2I5)
 2    FORMAT(4F10.4)
 3    FORMAT(I5)
 4    FORMAT(8F10.6)
 5    FORMAT(1X,I5,4F12.6)
 600  FORMAT(1H0,'******  NUMBER OF RUNS TO BE MADE=',I5/)
 606  FORMAT(1H1,'******   RUN NUMBER =',I5//)
 89   FORMAT(1H0,'THE INITIAL EXTREME VERTICES OF THE GIVE',
```

```
     C'N CONVEX HULL ARE   *****')
 82  FORMAT(5X,I5,5X,2F10.6)
 818 FORMAT(///1H0,'GENERATION OF FINAL CONVEX REGION FRO',
     C'M A GIVEN INITIAL CONVEX REGION BEGINS  *****'//1H0,
     C'TOTAL NUMBER OF EXTREME MATRICES =',I5//)
 819 FORMAT(1H0,'NUMBER OF THE MATRIX IS =',I5)
 81  FORMAT(1H0,2F10.4)
 202 FORMAT(1H0,'....  II =',I5,'....  ICOMP =',I5,'   **',
     C'**  FINAL CONVEX REGION HAS BEEN FOUND  ****'/)
C***********************************************************
      READ(5,1)NRUNS,NM
C***********************************************************
      WRITE(6,600)NRUNS
      DO 666 KPK=1,NRUNS
      ND=2
C***********************************************************
      READ(5,3)IRR
      READ(5,4)((X(I,J),J=1,ND),I=1,IRR)
C***********************************************************
      NDD=IRR/2
      WRITE(6,606)KPK
      IST=512
      LIAPST=0
      IJOB=0
      IAN=ND
      IZZ=ND
      II=NDD
      IR=II+1
      IRP=IRR+1
      WRITE(6,89)
      WRITE(6,82)(MA,(X(MA,MB),MB=1,ND),MA=1,II)
            CALL LINES(X,IRR,G,H)
      DO 48 I=IRP,IST
      X(I,1)=0.0
 48   X(I,2)=0.0
      IIP=II
      IW=IIP
      IRR=II+II
      WRITE(6,818)NM
      JI=24
      DO 50 IA=1,NM
C***********************************************************
      READ(5,2)((MI(I,J),J=1,ND),I=1,ND)
C***********************************************************
      WRITE(6,819)IA
      WRITE(6,81)((MI(I,J),J=1,ND),I=1,ND)
      DO 55 I=1,ND
      DO 55 J=1,ND
 55   MP(I,J)=MI(I,J)
```

```
      DO 61 J=1,JI
      IF(J .EQ. 1)GO TO 62
            CALL MATMLT(MI,MP)
62    CONTINUE
      ICOMP=II
      IIP=II
      DO 60 I=1,II
            CALL COORDT(MI,X(I,1),X(I,2),Z1,Z2)
      IREMOV=1
            CALL CHECK(Z1,Z2,G,H,IRR,IREMOV)
      IF(IREMOV .EQ. 0)GO TO 66
            CALL COMPAR(X,Z1,Z2,II,IIP)
66    CONTINUE
60    CONTINUE
      II=IIP
            CALL REORDR(X,II)
      WRITE(6,82)(MA,(X(MA,MB),MB=1,ND),MA=1,II)
      IF(II .EQ. ICOMP)GO TO 50
            CALL ELIMNT(X,II)
      WRITE(6,82)(MA,(X(MA,MB),MB=1,ND),MA=1,II)
      IIP=II
      IW=II
      IRR=II+II
      DO 40 IG=1,IW
      IT=IW+IG
      X(IT,1)=-X(IG,1)
40    X(IT,2)=-X(IG,2)
            CALL LINES(X,IRR,G,H)
61    CONTINUE
50    CONTINUE
      WRITE(6,202)II,ICOMP
      WRITE(6,82)(I,(X(I,J),J=1,ND),I=1,II)
      DO 41 IG=1,IW
      IT=IW+IG
      X(IT,1)=-X(IG,1)
41    X(IT,2)=-X(IG,2)
      IN=IT
            CALL RNORML(IN,X,RN)
            CALL COEF(X,PX,IN,INTOTL,CSMALL,RN,ISEG)
      IF(LIAPST .EQ. 1)GO TO 51
      DO 45 M=1,IN
      PY(M,1)=CSMALL*X(M,1)
      PY(M,2)=CSMALL*X(M,2)
      WRITE(6,5)M,X(M,1),X(M,2),PY(M,1),PY(M,2)
      X(M,1)=CSMALL*X(M,1)
      X(M,2)=CSMALL*X(M,2)
45    CONTINUE
      IF(CSMALL .LT. 2.0)GO TO 909
            CALL COEF(X,PX,IN,INTOTL,CSMALL,RN,ISEG)
```

```
      DO 454 M=1,IN
      PY(M,1)=CSMALL*X(M,1)
      PY(M,2)=CSMALL*X(M,2)
      WRITE(6,5)M,X(M,1),X(M,2),PY(M,1),PY(M,2)
 454  CONTINUE
 909  CONTINUE
 51   CONTINUE
 666  CONTINUE
      STOP
      END
C
C
      SUBROUTINE LINES(X,IRR,G,H)
      DIMENSION X(512,2),G(512,2),H(512)
      ID=IRR/2
      S=(X(1,1)-X(IRR,1))/(X(1,2)-X(IRR,2))
      G(ID,1)=1.0
      G(ID,2)=-S
      H(ID)=X(IRR,1)-S*X(IRR,2)
      IDP=ID+1
      S=(X(ID,1)-X(IDP,1))/(X(ID,2)-X(IDP,2))
      G(IDP,1)=1.0
      G(IDP,2)=-S
      H(IDP)=X(IDP,1)-S*X(IDP,2)
      DO 10 I=2,ID
      IM=I-1
      DEN=(X(I,2)-X(IM,2))
      IF(DEN .EQ. 0.0) GO TO 30
      S=(X(I,1)-X(IM,1))/(X(I,2)-X(IM,2))
      G(IM,1)=1.0
      G(IM,2)=-S
      H(IM)=X(IM,1)-S*X(IM,2)
      GO TO 10
 30   S=(X(I,2)-X(IM,2))/(X(I,1)-X(IM,1))
      G(IM,1)=-S
      G(IM,2)= 1.0
      H(IM)=-S*X(IM,1)+X(IM,2)
 10   CONTINUE
      ID2=ID+2
      IX=0
      DO 20 J=ID2,IRR
      IX=IX+1
      G(J,1)=G(IX,1)
      G(J,2)=G(IX,2)
      H(J)=-H(IX)
 20   CONTINUE
      RETURN
      END
C
```

```
C
      SUBROUTINE COORDT(MI,A,B,Z1,Z2)
      REAL MI(2,2)
      Z1=MI(1,1)*A+MI(1,2)*B
      Z2=MI(2,1)*A+MI(2,2)*B
      RETURN
      END
C
C
      SUBROUTINE MATMLT(MI,MP)
      DIMENSION MI(2,2),MP(2,2),D(2,2)
      REAL MI,MP
      ND=2
      DO 25 M=1,ND
      DO 25 N=1,ND
   25 D(M,N)=0.0
      DO 30 M=1,ND
      DO 30 N=1,ND
      DO 30 L=1,ND
   30 D(M,N)=MI(M,L)*MP(L,N)+D(M,N)
      DO 35 M=1,ND
      DO 35 N=1,ND
   35 MI(M,N)=D(M,N)
      RETURN
      END
C
C
      SUBROUTINE CHECK(Z1,Z2,G,H,IRR,IREMOV)
      DIMENSION G(512,2),H(512)
      ERR=0.001
      INDEX=0
      DO 10 I=1,IRR
      XZ=G(I,1)*Z1+G(I,2)*Z2
      IF(H(I) .LT. 0.0)GO TO 15
      XX=H(I)+ERR
      IF(XZ .LE. XX)INDEX=INDEX+1
      GO TO 10
   15 XX=H(I)-ERR
      IF(XZ .GE. XX)INDEX=INDEX+1
   10 CONTINUE
      IF(INDEX .EQ. IRR)IREMOV=0
      RETURN
      END
C
C
      SUBROUTINE COMPAR(X,Z1,Z2,II,IIP)
      DIMENSION X(512,2)
      IEQ=0
      ERR=0.0001
```

```
      IF(Z2 .GE. 0.0)GO TO 1
      Z1=-Z1
      Z2=-Z2
 1    CONTINUE
      DO 10 I=1,II
      Z1A=ABS(X(I,1)-Z1)
      Z2A=ABS(X(I,2)-Z2)
      IF(Z1A .LE. ERR .AND. Z2A .LE. ERR)IEQ=1
 10   CONTINUE
      IF(IEQ .EQ. 1)GO TO 11
      IIP=IIP+1
      X(IIP,1)=Z1
      X(IIP,2)=Z2
 11   CONTINUE
      RETURN
      END
C
C
      SUBROUTINE ELIMNT(X,II)
      DIMENSION X(512,2)
      ISTEP=1
      ERR=0.001
 1    ICOMP=II
      IT=II-1
      IF(ISTEP .NE. 1)GO TO 81
      I=1
      XX=X(I,1)
      XY=X(I,2)
      IM=2
      IP=2*II+1
      X(IP,1)=-X(II,1)
      X(IP,2)=-X(II,2)
      GO TO 82
 2    CONTINUE
      IM=II-1
      I=II
      XX=X(I,1)
      XY=X(I,2)
      IP=2*II+1
      X(IP,1)=-X(1,1)
      X(IP,2)=-X(1,2)
      GO TO 82
 81   CONTINUE
      I=2
 10   CONTINUE
      IM=I-1
      IP=I+1
      XX=X(I,1)
      XY=X(I,2)
```

```
      IF(XY .LE. X(IM,2) .AND. XY .LE. X(IP,2))GO TO 20
 82   CONTINUE
      DEN=(X(IM,2)-X(IP,2))
      IF(DEN .EQ. 0.0)GO TO 25
      XM=(X(IM,1)-X(IP,1))/(X(IM,2)-X(IP,2))
      XTEST=XX-XM*XY
      XR=X(IP,1)-XM*X(IP,2)
      GO TO 26
 25   XM=(X(IM,2)-X(IP,2))/(X(IM,1)-X(IP,1))
      XTEST=-XM*XX+XY
      XR=-XM*X(IP,1)+X(IP,2)
 26   CONTINUE
      IF(XR .LT. 0.0)GO TO 27
      XR=XR+ERR
      IF(XTEST .LE. XR)GO TO 20
      GO TO 32
 27   XR=XR-ERR
      IF(XTEST .GE. XR)GO TO 20
 32   CONTINUE
      ISTEP=ISTEP+1
      IF(ISTEP .EQ. 2)GO TO 2
      IF(ISTEP .EQ. 3)I=2
 28   CONTINUE
      I=I+1
      IF(I .LE. IT)GO TO 10
      GO TO 15
 20   ITEMP=I
      II=II-1
      IT=II-1
      DO 30 J=ITEMP,II
      JP=J+1
      X(J,1)=X(JP,1)
      X(J,2)=X(JP,2)
 30   CONTINUE
      X(JP,1)=0.0
      X(JP,2)=0.0
 31   CONTINUE
      IF(II .EQ. 2)GO TO 16
      ISTEP=ISTEP+1
      IF(ISTEP .EQ. 2)GO TO 2
      IF(ISTEP .EQ. 3)I=2
      IF(I .LE. IT)GO TO 10
 15   CONTINUE
      IF(ICOMP .NE. II)GO TO 1
 16   CONTINUE
      RETURN
      END
C
C
```

```
      SUBROUTINE REORDR(X,II)
      DIMENSION X(512,2),DEG(512)
      DO 40 K=1,II
      IF(X(K,1) .EQ.0.0 .AND. X(K,2) .GT. 0.0)GOTO 41
      DEG(K)=ATAN2(X(K,2),X(K,1))
      DEG(K)=(DEG(K)*180.0)/3.14159265
      GOTO 43
   41 DEG(K)=90.0
   43 CONTINUE
   40 CONTINUE
      NA=II-1
      DO 10 J=1,NA
      M=J
      MA=J+1
      DO 20 I=MA,II
      DEGDIF=ABS(DEG(I)-DEG(M))
      IF(DEGDIF .LE. 0.01)GOTO 21
      IF(DEG(I) .LT. DEG(M))M=I
      GO TO 20
   21 XNORMI=SQRT(X(I,1)**2+X(I,2)**2)
      XNORMM=SQRT(X(M,1)**2+X(M,2)**2)
      IF(XNORMM .LT. XNORMI)M=I
   20 CONTINUE
      TEMP1=X(J,1)
      TEMP2=X(J,2)
      TEMP3=DEG(J)
      X(J,1)=X(M,1)
      X(J,2)=X(M,2)
      DEG(J)=DEG(M)
      X(M,1)=TEMP1
      X(M,2)=TEMP2
      DEG(M)=TEMP3
   10 CONTINUE
      RETURN
      END
C
C
      SUBROUTINE RNORML(IN,X,RN)
      DIMENSION X(512,2),RN(512,2)
   20 FORMAT(1H1,'CONSTRUCTION OF NORMALS BEGINS'//)
   22 FORMAT(1X,I5,2F10.4)
      WRITE(6,20)
      XTEST=1.0
      DO 10 I=1,IN
      IP=I+1
      IF(I .EQ. IN)IP=1
      X21=X(I,1)-X(IP,1)
      Y21=X(I,2)-X(IP,2)
      AX21=ABS(X21)
```

```
      AY21=ABS(Y21)
      RN(I,1)=0.0
      RN(I,2)=0.0
      IF(AX21 .LE. 0.0001)GO TO 15
      IF(AY21 .GT. 0.0001)GO TO 16
      RN(I,2)=SIGN(XTEST,X(IP,2))
      WRITE(6,22)I,RN(I,1),RN(I,2)
      GO TO 17
 15   CONTINUE
      RN(I,1)=SIGN(XTEST,X(IP,1))
      WRITE(6,22)I,RN(I,1),RN(I,2)
      GO TO 17
 16   CONTINUE
      XDIF=X(IP,2)-((X(IP,1)*Y21)/X21)
      RN(I,2)=SIGN(XTEST,XDIF)
      RN(I,1)=-(RN(I,2)*Y21)/X21
      WRITE(6,22)I,RN(I,1),RN(I,2)
 17   CONTINUE
 10   CONTINUE
      RETURN
      END
C
C
      SUBROUTINE COEF(X,PX,IN,INTOTL,CSMALL,RN,ISEG)
      DIMENSION X(512,2),PX(512,2),CF(512),PY(512,2),C(200)
      DIMENSION ISEG(512),RN(512,2),NPNTS(512)
      DIMENSION CPN(3,200)
      COMMON CF,LIAPST,ISTEMP
 404  FORMAT(1H1,'.... THE CONSTANTS USED(FOR THREE AND SI',
     C'X POINT GRID) IN  (1-C)*X1+C*X2  ARE')
 4    FORMAT(1H0,2I5,F10.4)
 484  FORMAT(//)
 8    FORMAT(1X,I5,2F10.6,I5)
 7    FORMAT(//1H0,'TOTAL NUMBER OF POINTS CREATED ON THE ',
     C'BOUNDARY OF FINAL CONVEX REGIOIS=',I5//)
 405  FORMAT(1H1,'NEXT, WE USE DIRECT SEARCH PROCEDURE TO ',
     C'DETERMINE AN ESTIMATE FOR THE DOMAIN OF ATTRACTION'/)
 696  FORMAT(/1X,'........  CSMALL=',F15.8/)
 666  FORMAT(1H0,3F15.10)
 81   FORMAT(1X,I5,5F15.10)
 82   FORMAT(1X,I5,6F15.10)
 120  FORMAT(1H0,'.....  THE DOMAIN OF ATTRACTION COULD NO',
     C'T BE FOUND WITH THIS LIAPUNOV FUNCTION .....'//)
      ITM=11
      CMULT=1.0
      LIAPST=0
      PREC=-0.000001
      CSMALL=5.5
      WRITE(6,404)
```

```
      DO 3113 LRS=1,2
      RRRR=0.5
      IF(LRS .EQ. 2)RRRR=0.2
      IPTS=LRS*3
      IPTS=IPTS-1
      DO 3 LA=1,IPTS
      CPN(LRS,LA)=(LA-1)*RRRR
      WRITE(6,4)LRS,LA,CPN(LRS,LA)
3     CONTINUE
      WRITE(6,484)
3113  CONTINUE
      M=0
      DO 5 L=1,IN
      J=L+1
      IF(L .EQ. IN)J=1
      XDD1=X(J,1)-X(L,1)
      XDD2=X(J,2)-X(L,2)
      DIST=SQRT(XDD1**2+XDD2**2)
      DIST2=CMULT*DIST
      IPOINT=1
      IF(DIST2 .GT. 0.3)IPOINT=2
      IF(IPOINT .NE. 2)GO TO 1002
      IPTS=DIST2/0.2
      RRRR=1.0/IPTS
      IJI=IPTS+1
      DO 2001 NAR=1,IJI
      CPN(IPOINT,NAR)=(NAR-1)*RRRR
      IF(NAR .EQ. IJI)CPN(IPOINT,NAR)=1.0
2001  CONTINUE
1002  CONTINUE
      IF(IPOINT .EQ. 1)IPTS=1
      DO 6 K=1,IPTS
      M=M+1
      ISEG(M)=L
      C(K)=CPN(IPOINT,K)
      PX(M,1)=(1.0-C(K))*X(L,1)+C(K)*X(J,1)
      PX(M,2)=(1.0-C(K))*X(L,2)+C(K)*X(J,2)
      WRITE(6,8)M,PX(M,1),PX(M,2),ISEG(M)
6     CONTINUE
5     CONTINUE
      INTOTL=M
      WRITE(6,7)INTOTL
      WRITE(6,405)
      DO 10 I=1,INTOTL
      WRITE(6,696)CSMALL
      IF(I .EQ. 1)GO TO 686
      X1=CSMALL*PX(I,1)
      X2=CSMALL*PX(I,2)
            CALL DELVX(X1,X2,DVX,RN,I,ISEG)
```

```
      WRITE(6,666)X1,X2,DVX
      IF(DVX .LT. 0.0)GO TO 10
      XXXP=CSMALL
      PXXN=0.0
      GO TO 26
686   CONTINUE
      PXX=0.0
      XXX=0.0
      J=1
151   CONTINUE
      CF(I)=XXX
      IF(XXX .GE. CSMALL)GO TO 27
      XXX=XXX+0.5
      X1=XXX*PX(I,1)
      X2=XXX*PX(I,2)
            CALL DELVX(X1,X2,DVX,RN,I,ISEG)
      WRITE(6,81)I,PXX,XXX,X1,X2,DVX
      IF(DVX .GE.PREC .AND. DVX .LT.0.0)GO TO 16
      IF(DVX .LT. PREC)GO TO 13
      XXXP=XXX
      PXXN=PXX
26    CONTINUE
      AXDIF=ABS(XXXP-PXXN)
      IF(AXDIF .LE.0.0000015 .AND. DVX .LT.0.0)GO TO 17
      XP1=PXXN+(XXXP-PXXN)*0.5
      X1=XP1*PX(I,1)
      X2=XP1*PX(I,2)
            CALL DELVX(X1,X2,DVX,RN,I,ISEG)
      WRITE(6,82)I,PXXN,XXXP,XP1,X1,X2,DVX
      IF(DVX .GE.PREC .AND. DVX .LT.0.0)GO TO 17
      IF(DVX .LT. PREC)GO TO 18
      IF(XP1 .LE. 0.001953125)LIAPST=1
      IF(LIAPST .EQ. 1)GO TO 121
      XXXP=XP1
      GO TO 26
18    PXXN=XP1
      IF(XP1 .GE. CSMALL)GO TO 27
      GO TO 26
17    CF(I)=XP1
      IF(CF(I) .LT. CSMALL)ISTEMP=I
      IF(CF(I) .LT. CSMALL)CSMALL=CF(I)
      GO TO 27
13    PXX=XXX
      J=J+1
      IF(J .LE. ITM)GO TO 151
16    CF(I)=XXX
      IF(CF(I) .LT. CSMALL)ISTEMP=I
      IF(CF(I) .LT. CSMALL)CSMALL=CF(I)
27    CONTINUE
```

```
 10    CONTINUE
       GO TO 122
 121   CONTINUE
       WRITE(6,120)
 122   CONTINUE
       RETURN
       END
C
C
       SUBROUTINE DELVX(X1,X2,DVX,RN,I,ISEG)
       DIMENSION ISEG(512),RN(512,2)
             CALL FCNXY(X1,X2,FX1,FX2)
       DVX=FX1*RN(ISEG(I),1)+ FX2*RN(ISEG(I),2)
       RETURN
       END
C
C
       SUBROUTINE FCNXY(X1,X2,FX1,FX2)
C      PROVIDE FX1=X1DOT AND FX2=X2DOT.
       RETURN
       END
```