McMASTER UNIVERSITY BOOKSTORE
CUSTOM COURSEWARE
SERVICE

COMPUTER ENGINEERING 3KB3

Optimization, Vol. 1

John W. Bandler

Winter 1994

Computer Engineering 3KB3

Optimization, Vol. 1

John W. Bandler

December 1993

ACKNOWLEDGEMENTS

The author would like to acknowledge a number of individuals who have helped to define or develop some of the directions taken in this volume. They include H.L. Abdel-Malek, R.M. Biernacki, Q. Cai, C. Charalambous, S.H. Chen, S. Daijavad, M.A. El-Kady, P.C. Liu, K. Madsen, M.R.M. Rizk, A.E. Salama, R.E. Seviora, T.V. Srinivasan, H. Tromp and Q.J. Zhang.

The material presented has been collected over a period of more than two decades of teaching and research. Numerous students have been exposed to early versions of these chapters. They have also helped shape its form.

Chapter 1

INTRODUCTION

This volume deals with formulations and methods for mathematically based engineering design optimization which need to be implemented in ever increasing situations. The recommended approach is to use efficient, iterative automatic optimization methods ("optimizers") to achieve a design that meets or exceeds certain requirements. Often these techniques are implemented by user-friendly software systems in a manner which is transparent to the user. In spite of decades of development, however, optimization techniques are limited: classical analytical approaches usually need to be used to great advantage in estimating initial feasible designs for optimization. Thus, analytical and specialized expertise in the subject surrounding the problem to be solved is often required for the most effective use of optimization.

In order to make the mathematics tractable, classical methods are usually restricted to ideal or simplified commensurate systems. When we have to take into account active devices, a narrow range of element values, parameter uncertainties and tolerances, parasitic effects, high-speed or high-frequency operation, nonlinearities, frequency-dependent elements, noncommensurate elements, e.g., mixed lumped/distributed elements, devices characterized by measurement data, practical response constraints, and so on, classical methods of design provide, at best, only approximate answers. In some cases these answers adequately approximate the solution to the actual design problem, but in many cases they do not.

Generally speaking, iterative methods of optimization require more computation time than more specialized methods which do not require iteration, if they are available. Computing time is not, however, the only criterion an engineer has to consider. For example, in deciding whether or not to devote his own time to deriving or researching an analytical approach, as distinct from a numerical approach, he also has to ask himself how often either approach would be used, how well it would represent real situations, how widely applicable it would be, and last but not least, how accurate the numerical results would be. After all, as engineers, we are ultimately working toward producing meaningful solutions to realistic design problems.

Methods which have withstood the test of time in their application to optimal design processes are emphasized. *Ad hoc* cut-and-try techniques using a general purpose simulator are discouraged, particularly for design problems with anything other than the simplest of design specifications and a very small number of variable parameters. The pitfalls are the same as with automated methods, the strategy for dealing with them is inevitably less sophisticated, and in the long run it will almost certainly cost more. It is desirable that the multidimensional decision making process should, as far as possible, be left to the computer.

Poor or unacceptable results in computer-aided circuit optimization (or with any design process) are felt to be most likely due to bad preparation of the problem, a lack of understanding of the hazards that can be encountered and the wrong choice of algorithm. This volume, therefore, attempts to show how general design problems may be formulated effectively as optimization problems, to explain the differences between these formulations, to indicate an appropriate choice of optimization method, and to indicate how the results might be interpreted. Details of optimization algorithms, proofs of convergence,

etc., are beyond the scope of this work. Adequate references to the original papers and relevant text books will permit the student to investigate these for himself.

In order to set the stage for concepts and techniques to be considered later, some typical problems in the optimal computer-aided design of electrical networks and systems will be discussed briefly. Only the essence of the problems is emphasized. As a first step in solving them, appropriate objective functions to be optimized are suggested and the variable parameters are identified. Some of the implications of the objective function formulations are also discussed. It is hoped that the electrical engineering student will be sufficiently motivated by this introduction to pursue the somewhat more mathematical material which will follow.

1.1 The Unconstrained Optimization Problem

The unconstrained mathematical optimization problem may be abstractly stated as: minimize U with respect to (w.r.t.) ϕ, where

$$U \triangleq U(\phi) \tag{1.1}$$

and where

$$\phi \triangleq \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_k \end{bmatrix} \tag{1.2}$$

In this type of problem, U is a scalar *objective function* of k independent variables or designable parameters $\phi_1, \phi_2, \ldots, \phi_k$. Put another way, the objective is to adjust the variables to obtain an optimal set $\check{\phi}$ such that

$$U(\check{\phi}) < U(\phi) \tag{1.3}$$

for all permissible sets of ϕ in a neighbourhood of $\check{\phi}$. The point $\check{\phi}$ defines a *local minimum* of U in the k-dimensional parameter space.

1.2 Approximation of a Time Response

A frequently occurring problem in system design is the approximation of a desired response in the time domain, as shown in Fig. 1.1.

Let $f(t) = v_2(t)$ be the response to the input $v_1(t)$. It is desired to approximate the specified function $S(t)$ in a least squares sense. We can set up the objective function

$$U = \int_0^{t_f} [f(\phi, t) - S(t)]^2 \, dt \tag{1.4}$$

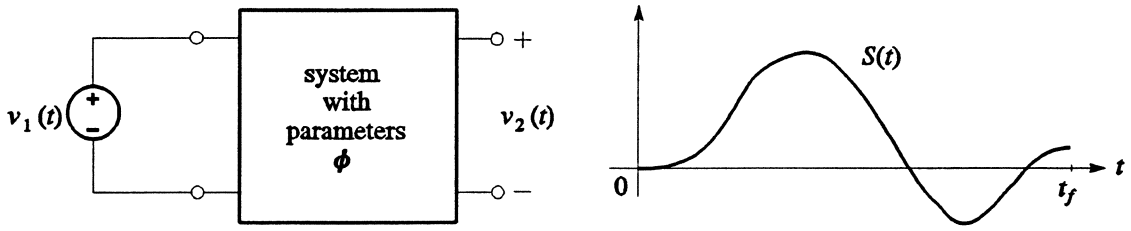

Fig. 1.1 A system in the time domain and its response specification $S(t)$.

to be minimized w.r.t. ϕ.

1.3 Active Filter Design

Consider the design of an active (inductorless) filter as shown in Fig. 1.2. Suppose the problem is to obtain values for the R and C components which result in a gain $G(\omega)$, where

$$G(\omega) \triangleq 20 \log_{10} \left| \frac{V_2(j\omega)}{V_1(j\omega)} \right| \tag{1.5}$$

as close, in some sense, as possible to the desired low pass gain characteristic $S(\omega)$ shown.

Let us form the objective function

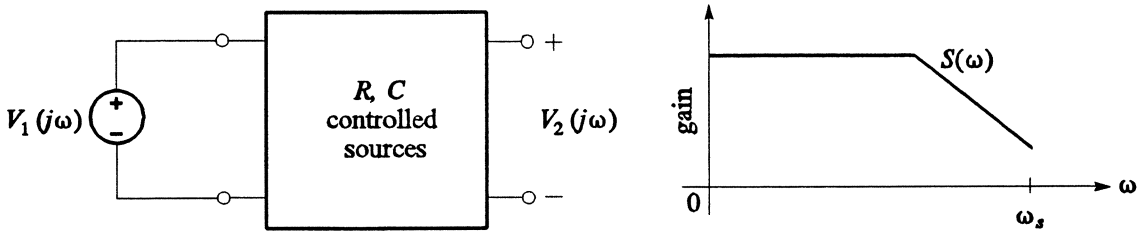

Fig. 1.2 Two-port representing an active filter and its gain characteristic.

$$U = \sum_{\omega_d \in \Omega_d} | G(\phi, \omega_d) - S(\omega_d) |^p \tag{1.6}$$

where Ω_d is a discrete set of frequencies selected from the interval $[0, \omega_s]$ and $p \geq 1$. Here,

$$\phi = \begin{bmatrix} R_1 \\ C_1 \\ R_2 \\ C_2 \\ \vdots \end{bmatrix} \tag{1.7}$$

where R_1, C_1, R_2, C_2, etc., are designable.

Since the gain G is usually a nonlinear function of ϕ we can expect, for particular fixed values of p, several minima of U and hence several candidates for a "best" response. The question of which value of p to choose must also be settled.

1.4 Nonlinear Network DC Analysis

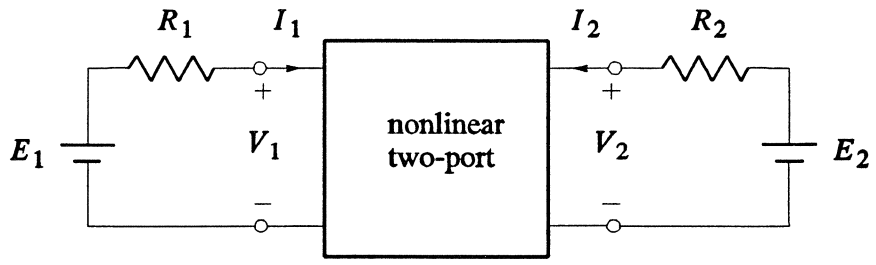

Fig. 1.3 A nonlinear two-port.

Consider the nonlinear resistive two-port shown connected to linear resistors and batteries. The problem is to determine the values of I_1, V_1, I_2 and V_2 which define the operating or equilibrium point.

Two mesh equations give

$$V_1 = E_1 - I_1R_1 \tag{1.8}$$
$$V_2 = E_2 - I_2R_2 \tag{1.9}$$

where E_1, E_2, R_1 and R_2 are constants. Let the network be current controlled. Then

$$V_1 = r_1(I_1, I_2) \tag{1.10}$$
$$V_2 = r_2(I_1, I_2) \tag{1.11}$$

where r_1 and r_2 are specified functions. We have four equations, two of them nonlinear, to solve in four unknowns.

Our first thought might be to let

$$\phi = \begin{bmatrix} I_1 \\ V_1 \\ I_2 \\ V_2 \end{bmatrix} \tag{1.12}$$

But V_1 and V_2 are dependent variables. So, we should let

$$\phi = \begin{bmatrix} I_1 \\ I_2 \end{bmatrix} \tag{1.13}$$

and solve

$$f_1(\phi) = E_1 - I_1R_1 - r_1(I_1, I_2) = 0 \tag{1.14}$$
$$f_2(\phi) = E_2 - I_2R_2 - r_2(I_1, I_2) = 0 \tag{1.15}$$

Consider the least squares objective function

$$U = f_1^2 + f_2^2 \tag{1.16}$$

Obviously, if $U = 0$ then $f_1^2 = 0$ and $f_2^2 = 0$ so that $f_1(\phi) = f_2(\phi) = 0$.

The "solution of equations" problem has been reformulated as an optimization problem. In general, there may be many local minima. If min $U = 0$ we have a solution to the equations; if min $U \neq 0$ we have no solution. Conditions for the existence and uniqueness of the solution depend on the properties of the nonlinear functions, which in turn affect the form of the objective function U.

1.5 Minimization of Overshoot

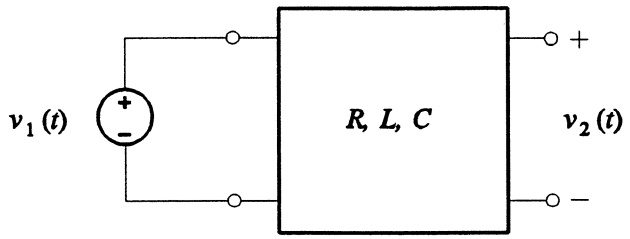

Fig. 1.4 A linear, time-invariant RLC network.

Suppose it is desired to minimize the overshoot in the step response of a linear, time-invariant RLC network. Let $v_1(t) = u(t)$, the unit step. See Fig. 1.4.

The problem is to adjust the values of the R, L and C components, often within specific upper and lower bounds, so as to minimize

$$U = \max_{t \in [0,\, t_f]} v_2(\phi,\, t) \tag{1.17}$$

where t_f is some final value of time. In this case

$$\phi = \begin{bmatrix} R_1 \\ L_1 \\ C_1 \\ \vdots \end{bmatrix} \tag{1.18}$$

1.6 Approximation by a Rational Function

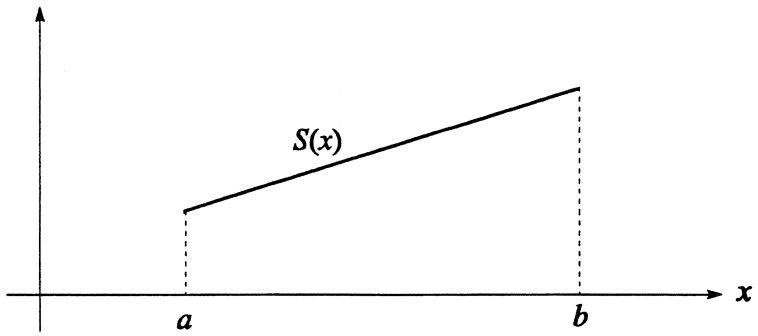

Fig. 1.5 A specified continuous function $S(x)$.

Consider the problem of approximating a specified continuous function $S(x)$, as shown in Fig. 1.5, by the rational approximating function

$$F(x) = \frac{P(x)}{Q(x)} = \frac{\sum_{i=0}^{n} a_i x^i}{1 + \sum_{i=1}^{m} b_i x^i} \tag{1.19}$$

To obtain an approximation in the Chebyshev or equal-ripple sense one needs to minimize

$$\max_{x \in [a,\, b]} \left| F(\phi,\, x) - S(x) \right| \tag{1.20}$$

where

$$\phi = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \\ b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix} \tag{1.21}$$

This problem falls into a class called *minimax approximation*, since we attempt to minimize the maximum deviation between the approximating function and the desired function.

The transfer function of a linear system (see Fig. 1.6)

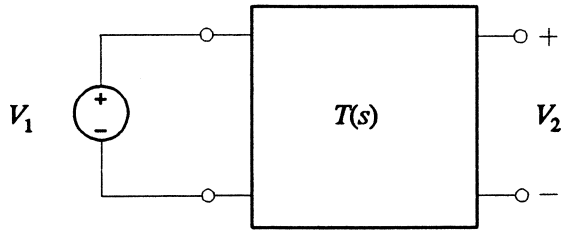

Fig. 1.6 A linear system with transfer function $T(s)$.

$$T(s) = \frac{V_2(s)}{V_1(s)} \tag{1.22}$$

can be optimized by such a formulation.

1.7 Finding Response Maxima

A common problem is the determination of the peak in a response or the maximum error between a response and a specification. Consider, for example, the problem of finding the peak gain of an amplifier for a specified set of parameter values (see Fig. 1.7).

We are required to maximize

$$U = G(\phi) \tag{1.23}$$

w.r.t. ϕ, where $\phi = \omega$, for $\omega_1 \leq \phi \leq \omega_2$.

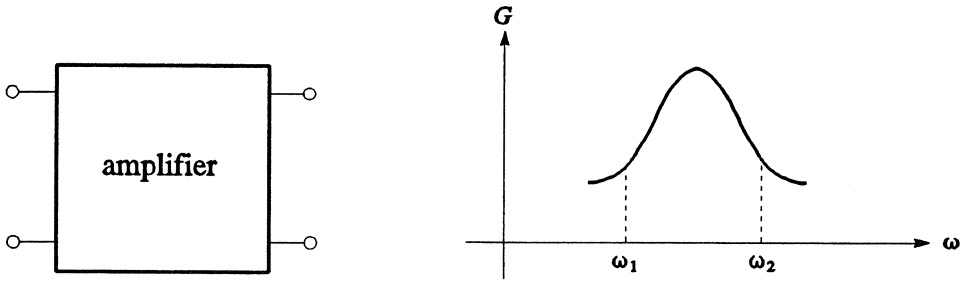

Fig. 1.7 An amplifier and its gain response.

1.8 Matching Coefficients of Rational Functions

Suppose we have a rational function

$$S(s) = \frac{\sum_{i=0}^{n} a_i s^i}{\sum_{i=0}^{m} b_i s^i} \tag{1.24}$$

with known (presumably, optimal) coefficients. Suppose we also have a network transfer function

$$T(\phi, s) = \frac{\sum_{i=0}^{n} a_i'(\phi) s^i}{\sum_{i=0}^{m} b_i'(\phi) s^i} \tag{1.25}$$

describing a network of the proper configuration. The coefficients are, in general, nonlinear functions of the network elements, as indicated. It is desired to adjust the element values so that $T(s)$ can be fitted as closely as possible to $S(s)$.

In this case, we have to solve the nonlinear equations

$$f \triangleq \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{m+n+2} \end{bmatrix} = \begin{bmatrix} a_0'(\phi) \\ a_1'(\phi) \\ \vdots \\ a_n'(\phi) \\ b_0'(\phi) \\ b_1'(\phi) \\ \vdots \\ b_m'(\phi) \end{bmatrix} - \phi_{k+1} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \\ b_0 \\ b_1 \\ \vdots \\ b_m \end{bmatrix} = \mathbf{0} \tag{1.26}$$

where f is a vector of functions, $\mathbf{0}$ is a null vector and ϕ_{k+1} is an unknown multiplicative constant.

One possible objective function to be minimized w.r.t. the $k+1$ variables could be

$$U = \sum_{i=1}^{m+n+2} f_i^2(\phi, \phi_{k+1}) \tag{1.27}$$

where $U = 0$ is the desired solution. Note that, in general,

$$m + n + 2 \neq k + 1 \tag{1.28}$$

1.9 Sensitivity Analysis

A most important practical problem is the investigation of the effect on a nominally optimal response of the circuit or system parameters. A complete study of the subject would be rather involved. Let us illustrate the idea by means of the following example of optimizing the input reflection coefficient of a one-port (Fig. 1.8).

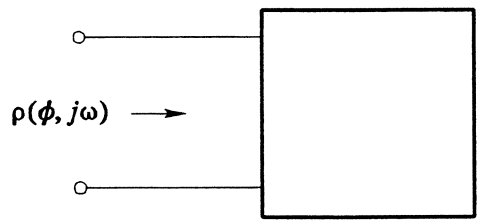

Fig. 1.8 A one-port network with reflection coefficient ρ.

We have an optimal set of parameters, say $\check{\phi}$ for the solution

$$\check{U} = \min_{\phi \in R} \max_{\omega \in \Omega} | \rho(\phi, j\omega) | \tag{1.29}$$

that is to say, we have found the value of ϕ within a *feasible region* R_a which minimizes the maximum magnitude of the input reflection coefficient of a circuit over a region of frequencies Ω. See Fig. 1.9.

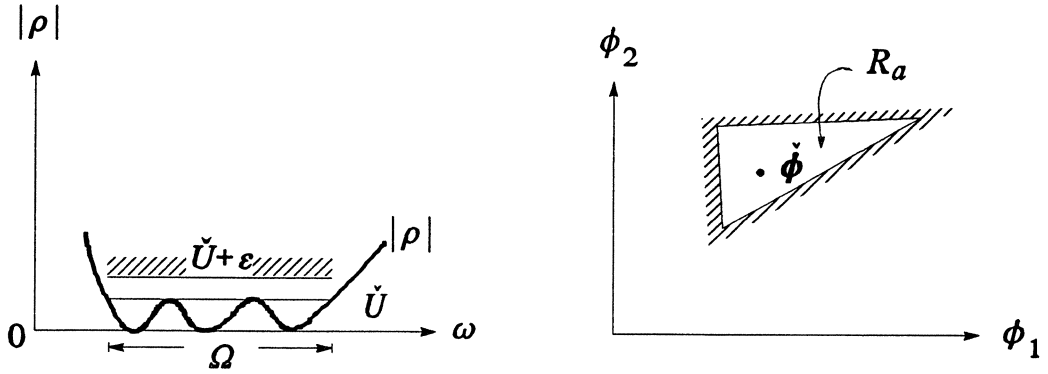

Fig. 1.9 Characteristic of $|\rho|$ and acceptable region R_a in the ϕ space.

One problem might be to find

$$\max_{\phi \in T,\ \omega \in \Omega} | \rho(\phi, j\omega) | \tag{1.30}$$

where T defines a region of tolerances on the components of ϕ around $\check{\phi}$ as shown in Fig. 1.10. In this case we would be trying to find the worst value of $|\rho|$ in the frequency band of interest.

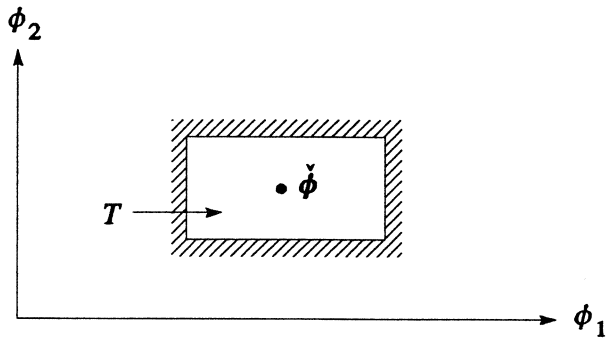

Fig. 1.10 A tolerance region in the ϕ space.

Suppose we can tolerate a deviation ε from the optimum $\check{U}$. The problem of finding a region R_a of acceptable parameters, i.e., where

$$R_a \triangleq \{\phi \,|\, |\rho(\phi, j\omega)| - \check{U} \leq \varepsilon,\ \omega \in \Omega\} \tag{1.31}$$

is by no means a trivial one. The region would be described by its boundary, on which $|\rho(\phi, j\omega)| - \check{U} = \varepsilon$.

A reason for wanting the region of acceptability is that one might be able to immediately predict the acceptability of any given design. Another that the nominal design may be more suitably located inside the region. Worst-case tolerances can be evaluated.

These considerations hold equally for large-change sensitivities as for first-order sensitivities. For first-order sensitivities, one could simplify the problem. Suppose we are considering a function $f(\phi)$. We wish to obtain

$$\max_{\phi \in T} \quad f(\phi) \tag{1.32}$$

using a linear approximation of $f(\phi)$ at a reference or nominal point ϕ^0. The problem then becomes that of finding

$$\max_{\phi \in T} \quad \{f(\phi^0) + \nabla^T f(\phi^0)\ \Delta\phi\} \tag{1.33}$$

where the expression in { } is the linear (Taylor series) approximation to $f(\phi)$ and

$$\Delta\phi = \phi - \phi^0 \tag{1.34}$$

Note that $f(\phi^0)$ and the gradient vector $\nabla f(\phi^0)$ are constant, as they are evaluated at the nominal point ϕ^0. Suppose, as is usually the case,

$$T \triangleq \{\phi \mid |\phi_i - \phi_i^0| \leq \varepsilon_i,\ i = 1, 2, \ldots, k\} \tag{1.35}$$

where ε_i is a (positive) prescribed tolerance limit on the parameter ϕ_i. Such a region is shown in Fig. 1.10. The worst-case solution is given by

$$\sum_{i=1}^{k} \varepsilon_i \left| \frac{\partial f}{\partial \phi_i} \right| \tag{1.36}$$

A region of acceptability with respect to $f(\phi)$ is given by

$$\{\phi \mid \nabla^T f(\phi^0)(\phi - \phi^0) \leq \varepsilon^0\} \tag{1.37}$$

If, as in the problem stated at the beginning of this section, we are dealing with $|\rho(\phi, j\omega)|$ then

$$R_a \triangleq \{\phi \mid |\rho(\check{\phi}, j\omega)| + \nabla^T |\rho(\check{\phi}, j\omega)|\,(\phi - \check{\phi}) - \check{U} \leq \varepsilon,\ \omega \in \Omega\} \tag{1.38}$$

Thus, any set of parameters ϕ which satisfies the above linear inequality at all values of ω in the band of interest lies in a region of acceptability defined by first-order sensitivities.

Chapter 2

VECTORS, MATRICES AND SENSITIVITIES

In this chapter we formally define notation used throughout this book, including vectors and matrices. We also present some elementary manipulations of vectors and matrices, making significant use of transposition and association. Such implementations as decomposition into respective rows or elements and assembly of elements into rows and columns are presented. We make use of various types of products in the process. Later we define derivative operators and Jacobians. We apply the manipulation of vectors and matrices to an abstract exposition of derivatives of matrices and inverse matrices, and also to elements of the solution vector of a system of linear equations.

2.1 Definitions

2.1.1 Vectors

Primal Vector $\boldsymbol{a}$

Let

$$\boldsymbol{a} \triangleq \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} \tag{2.1}$$

This n-dimensional vector has n rows and 1 column. It consists of n entries $a_1, a_2, \ldots, a_n$ organized in vertical format.

Transpose of $\boldsymbol{a}$

The operation of transposition is designated by the superscript T. It results in

$$\boldsymbol{a}^T \triangleq [a_1 \; a_2 \; \ldots \; a_n] \tag{2.2}$$

This n-dimensional vector has 1 row and n columns. It consists of n entries $a_1, a_2, \ldots, a_n$ organized in horizontal format.

Dual Vector $\boldsymbol{a}'$

A certain dual vector is distinguished from the original vector by the superscript $'$. In this case,

let

$$a' \triangleq \begin{bmatrix} a_1' \\ a_2' \\ \vdots \\ a_m' \end{bmatrix} \tag{2.3}$$

This m-dimensional vector has m rows and 1 column.

Transpose of a'

$$a'^T \triangleq [a_1' \; a_2' \; \ldots \; a_m'] \tag{2.4}$$

This m-dimensional vector has 1 row and m columns.

Unit Vector u_i

The unit vector is extremely important. In our notation we express it as

$$u_i \triangleq \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} \leftarrow i\text{th row} \tag{2.5}$$

This vector has m elements, all of which are 0 except the ith, which is 1.

Dual Unit Vector u_j

A dual unit vector is correspondingly considered as

$$u_j \triangleq \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} \leftarrow j\text{th row} \tag{2.6}$$

This vector has n elements, all of which are 0 except the jth, which is 1.

2.1.2 Matrices

Matrix A

As is traditional we let

$$A \triangleq \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \tag{2.7}$$

This $m \times n$ matrix has m rows and n columns of elements of the form a_{ij}.

Transpose of A

The operation of transposition of A results in

$$A^T \triangleq \begin{bmatrix} a_{11} & a_{21} & \cdots & a_{m1} \\ a_{12} & a_{22} & \cdots & a_{m2} \\ \vdots & \vdots & & \vdots \\ a_{1n} & a_{2n} & \cdots & a_{mn} \end{bmatrix} \tag{2.8}$$

This $n \times m$ matrix has n rows and m columns of elements of the form a_{ji}.

Symmetric Matrix A

A square matrix A is said to be symmetric if

$$A^T = A \tag{2.9}$$

Forms of A and A^T

Partitioned forms of A and A^T can be written as

$$A^T = [a_1 \; a_2 \; \cdots \; a_m] = \begin{bmatrix} a_1'^T \\ a_2'^T \\ \vdots \\ a_n'^T \end{bmatrix} = A' \tag{2.10}$$

and

$$A = \begin{bmatrix} a_1^T \\ a_2^T \\ \vdots \\ a_m^T \end{bmatrix} = [a_1' \; a_2' \; \ldots \; a_n'] = A'^T \tag{2.11}$$

Here, $a_1, a_2, \ldots, a_m$ are n-dimensional, whereas the dual vectors $a_1', a_2', \ldots, a_n'$ are m-dimensional.

Rows of A

$$A = \begin{bmatrix} a_1^T \\ a_2^T \\ \vdots \\ a_m^T \end{bmatrix} \tag{2.12}$$

This $m \times n$ matrix has rows which are the transposes of the column vectors

$$a_1 \triangleq \begin{bmatrix} a_{11} \\ a_{12} \\ \vdots \\ a_{1n} \end{bmatrix}, \quad a_2 \triangleq \begin{bmatrix} a_{21} \\ a_{22} \\ \vdots \\ a_{2n} \end{bmatrix}, \ldots, a_m \triangleq \begin{bmatrix} a_{m1} \\ a_{m2} \\ \vdots \\ a_{mn} \end{bmatrix} \tag{2.13}$$

Columns of A

$$A = [a_1' \; a_2' \; \ldots \; a_n'] \tag{2.14}$$

This $m \times n$ matrix has columns

$$a_1' \triangleq \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{bmatrix}, \quad a_2' \triangleq \begin{bmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{m2} \end{bmatrix}, \ldots, \quad a_n' \triangleq \begin{bmatrix} a_{1n} \\ a_{2n} \\ \vdots \\ a_{mn} \end{bmatrix} \tag{2.15}$$

The Identity Matrix

Let

$$u_1 \triangleq \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad u_2 \triangleq \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}, \ldots, u_n \triangleq \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} \tag{2.16}$$

Then

$$\mathbf{1} = [u_1 \; u_2 \; \ldots \; u_n] \tag{2.17}$$

is called the identity matrix.

2.2 Manipulations with Vectors and Matrices

2.2.1 Scalar Product

Let

$$a \triangleq \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}, \quad b \triangleq \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \tag{2.18}$$

Then

$$a^T b \triangleq a_1 b_1 + a_2 b_2 + \ldots + a_n b_n \tag{2.19}$$

The result is a scalar.

2.2.2 Selection

Element Selection from Vectors

$$u_j^T a \equiv a^T u_j = a_j \tag{2.20}$$

The result is the scalar element a_j.

$$u_i^T a' \equiv a'^T u_i = a_i' \tag{2.21}$$

The result is the scalar element a_i'.

Row Selection from Matrices

$$u_i^T A = u_i^T \begin{bmatrix} a_1^T \\ a_2^T \\ \vdots \\ a_m^T \end{bmatrix} = a_i^T \tag{2.22}$$

This result is the row vector corresponding to the ith row of A. In transposed form, we have

$$A^T u_i = a_i \tag{2.23}$$

Column Selection from Matrices

$$Au_j = [a_1' \quad a_2' \quad \ldots \quad a_n'] \, u_j = a_j' \tag{2.24}$$

This result is the column vector corresponding to the jth column of A. In transposed form, we have

$$u_j^T A^T = a_j'^T \tag{2.25}$$

Element Selection from Matrices

$$u_i^T A u_j = \begin{cases} u_i^T (A u_j) = u_i^T a_j' = a_{ij} \\ (u_i^T A) u_j = a_i^T u_j = a_{ij} \end{cases} \tag{2.26}$$

This result is the scalar element corresponding to the coefficient of A obtained from the intersection of row i and column j.

Element Representation

$$A = \begin{bmatrix} u_1^T a_1 & u_2^T a_1 & \ldots & u_n^T a_1 \\ u_1^T a_2 & u_2^T a_2 & \ldots & u_n^T a_2 \\ \vdots & \vdots & & \vdots \\ u_1^T a_m & u_2^T a_m & \ldots & u_n^T a_m \end{bmatrix} = A'^T \tag{2.27}$$

where u_j, $j = 1, \ldots, n$, is n-dimensional.

$$A'^T = \begin{bmatrix} u_1^T a_1' & u_1^T a_2' & \cdots & u_1^T a_n' \\ u_2^T a_1' & u_2^T a_2' & \cdots & u_2^T a_n' \\ \vdots & \vdots & & \vdots \\ u_m^T a_1' & u_m^T a_2' & \cdots & u_m^T a_n' \end{bmatrix} = A \tag{2.28}$$

where u_i, $i = 1, \ldots, m$ is m-dimensional.

2.2.3 Assembly

Assembly of an Element into a Column Vector

To place the parameter ϕ into the ith row of a column vector we write

$$u_i \phi = \begin{bmatrix} 0 \\ \vdots \\ \phi \\ \vdots \\ 0 \end{bmatrix} \leftarrow i\text{th row} \tag{2.29}$$

This results in a column vector containing zeros everywhere except in the ith row, which contains ϕ.

Assembly of an Element into a Row Vector

To place the parameter ϕ into the jth column of a row vector we write

$$\phi u_j^T = [0 \ \ldots \ \phi \ \ldots \ 0] \tag{2.30}$$

$$\uparrow$$

$$j\text{th col}$$

This results in a row vector containing zeros everywhere except in the jth column, which contains ϕ.

Assembly of a Vector into a Row of a Matrix

To place the n-dimensional vector a into the ith row of a matrix we write

$$\boldsymbol{u}_i\,\boldsymbol{a}^T = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 \\ a_1 & a_2 & \cdots & a_n \\ 0 & 0 & & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} \leftarrow i\text{th row} \tag{2.31}$$

This results in a matrix containing zeros everywhere except in the ith row, which contains $\boldsymbol{a}^T$.

Assembly of a Vector into a Column of a Matrix

To place the m-dimensional vector $\boldsymbol{a}'$ into the jth column of a matrix we write

$$\boldsymbol{a}'\,\boldsymbol{u}_j^T = \begin{bmatrix} 0 & \cdots & 0 & a_1' & 0 & \cdots & 0 \\ 0 & \cdots & 0 & a_2' & 0 & \cdots & 0 \\ \vdots & & \vdots & \vdots & \vdots & & \vdots \\ 0 & \cdots & 0 & a_m' & 0 & \cdots & 0 \end{bmatrix} \tag{2.32}$$

jth col

This results in a matrix containing zeros everywhere except in the column, which contains $\boldsymbol{a}'$.

Assembly of an Element into a Matrix

To place the parameter ϕ into the intersection of row i and column j we write

$$\boldsymbol{u}_i\,\phi\,\boldsymbol{u}_j^T = \begin{bmatrix} & \vdots & \\ \cdots & \phi & \cdots \\ & \vdots & \end{bmatrix} i\text{th row} \tag{2.33}$$

jth col

This results in a matrix containing zeros everywhere except in the i, j location, which contains ϕ.

Two other forms are convenient to represent. The first one is

$$\boldsymbol{u}_i\,(\phi\ \boldsymbol{u}_j^T) = \begin{bmatrix} \boldsymbol{0}^T \\ \vdots \\ \phi\,\boldsymbol{u}_j^T \\ \vdots \\ \boldsymbol{0}^T \end{bmatrix} \leftarrow i\text{th row} \tag{2.34}$$

The other form is

$$(\boldsymbol{u}_i\ \phi)\ \boldsymbol{u}_j^T = [0 \quad 0 \quad \ldots \underset{\substack{\uparrow \\ j\text{th col}}}{\boldsymbol{u}_i\ \phi} \quad \ldots \quad 0] \tag{2.35}$$

Assembly of a Matrix

To assemble $\boldsymbol{A}$ we may write

$$\boldsymbol{A} = \sum_{i=1}^{m}\sum_{j=1}^{n} \boldsymbol{u}_i\ a_{ij}\ \boldsymbol{u}_j^T \tag{2.36}$$

Assembly of the ith Row of a Matrix

To assemble the ith row of $\boldsymbol{A}$ we write

$$\boldsymbol{a}_i^T = \sum_{j=1}^{n} a_{ij}\ \boldsymbol{u}_j^T \tag{2.37}$$

Assembly of the jth Column of a Matrix

To assemble the jth column of $\boldsymbol{A}$ we write

$$\boldsymbol{a}_j' = \sum_{i=1}^{m} \boldsymbol{u}_i\ a_{ij} \tag{2.38}$$

Dyadic Product of $\boldsymbol{a}$ *and* $\boldsymbol{b}$

Let

$$\boldsymbol{a} \triangleq \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}, \quad \boldsymbol{b} \triangleq \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix} \tag{2.39}$$

Then

$$\boldsymbol{a}\,\boldsymbol{b}^T \triangleq \begin{bmatrix} a_1b_1 & a_1b_2 & \cdots & a_1b_m \\ a_2b_1 & a_2b_2 & \cdots & a_2b_m \\ \vdots & \vdots & & \vdots \\ a_nb_1 & a_nb_2 & \cdots & a_nb_m \end{bmatrix} \tag{2.40}$$

This result is the n by m matrix containing all possible products a_ib_j, $i = 1, 2, \ldots, n; j = 1, 2, \ldots, m$.

Placing Elements Arbitrarily

Let

$$\boldsymbol{u}_0 \triangleq \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \boldsymbol{u}_1 \triangleq \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \boldsymbol{u}_2 \triangleq \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}, \ldots, \boldsymbol{u}_n \triangleq \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} \tag{2.41}$$

and denote

$$\begin{aligned} \boldsymbol{u}_{ij} &\triangleq \boldsymbol{u}_i - \boldsymbol{u}_j \\ \boldsymbol{u}_{rs,kl} &\triangleq \boldsymbol{u}_r + \boldsymbol{u}_s - \boldsymbol{u}_k - \boldsymbol{u}_l \end{aligned} \tag{2.42}$$

hence

$$\boldsymbol{u}_{i_1 i_2 \cdots i_p, j_1 j_2 \cdots j_q} \triangleq \sum_{k=1}^{p} \boldsymbol{u}_{i_k} - \sum_{l=1}^{q} \boldsymbol{u}_{j_l} \tag{2.43}$$

For example,

$$\boldsymbol{u}_{345,16} = \begin{bmatrix} -1 \\ 0 \\ 1 \\ 1 \\ 1 \\ -1 \end{bmatrix} \tag{2.44}$$

Then if

$$\boldsymbol{u} \equiv \boldsymbol{u}_{345,16} \tag{2.45}$$

we have

$$\boldsymbol{u}^T\boldsymbol{u} = \text{card } \{3,4,5,1,6\} = 5 \tag{2.46}$$

and

$$u\,u^T = \begin{bmatrix} 1 & 0 & -1 & -1 & -1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 1 & 1 & -1 \\ -1 & 0 & 1 & 1 & 1 & -1 \\ -1 & 0 & 1 & 1 & 1 & -1 \\ 1 & 0 & -1 & -1 & -1 & 1 \end{bmatrix} \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{matrix} \qquad (2.47)$$

$$1\ \ 2\ \ 3\ \ 4\ \ 5\ \ 6$$

$$= (u_3 + u_4 + u_5 - u_1 - u_6)(u_3 + u_4 + u_5 - u_1 - u_6)^T$$

2.2.4 Some Important Properties of Square Matrices

Positive Definite

An $n \times n$ matrix A is positive definite, if

$$x^T A x > 0, \text{ for all } x \neq 0$$
$$\text{and } x^T A x = 0 \text{ implies that } x = 0 \qquad (2.48)$$

where x is an n-dimensional vector.

The diagonal elements are positive in a positive definite matrix. The element of largest modulus lies on the diagonal.

Positive Semidefinite

An $n \times n$ matrix A is positive semidefinite, if

$$x^T A x \geq 0 \qquad (2.49)$$

where x is an n-dimensional vector.

Diagonally Dominant

An $n \times n$ matrix A is diagonally dominant, if

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|, \text{ for all } i \qquad (2.50)$$

2.3 Sensitivities

2.3.1 Basic Derivative Operations

Derivative Operators

$$\frac{\partial}{\partial a} \triangleq \begin{bmatrix} \frac{\partial}{\partial a_1} \\ \frac{\partial}{\partial a_2} \\ \vdots \\ \frac{\partial}{\partial a_n} \end{bmatrix} \tag{2.51}$$

$$\frac{\partial}{\partial A} \triangleq \begin{bmatrix} \frac{\partial}{\partial a_{11}} & \frac{\partial}{\partial a_{12}} & \cdots & \frac{\partial}{\partial a_{1n}} \\ \frac{\partial}{\partial a_{21}} & \frac{\partial}{\partial a_{22}} & \cdots & \frac{\partial}{\partial a_{2n}} \\ \vdots & \vdots & & \vdots \\ \frac{\partial}{\partial a_{m1}} & \frac{\partial}{\partial a_{m2}} & \cdots & \frac{\partial}{\partial a_{mn}} \end{bmatrix} \tag{2.52}$$

Derivatives of Scalar Products

$$\frac{\partial (a^T b)}{\partial a} = \left[\frac{\partial a^T}{\partial a} \right] b = b \tag{2.53}$$

Also,

$$\frac{\partial (b^T a)}{\partial a} = \frac{\partial (a^T b)}{\partial a} = b \tag{2.54}$$

2.3.2 Jacobian Matrix

Jacobian Matrix [$b \neq a$]

Let

$$\boldsymbol{a} \triangleq \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}, \quad \boldsymbol{b} \triangleq \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix} \tag{2.55}$$

Then

$$\frac{\partial \boldsymbol{b}^T}{\partial \boldsymbol{a}} \triangleq \left[\frac{\partial}{\partial \boldsymbol{a}}\right] \boldsymbol{b}^T = \begin{bmatrix} \frac{\partial b_1}{\partial a_1} & \frac{\partial b_1}{\partial a_2} & \cdots & \frac{\partial b_1}{\partial a_n} \\ \frac{\partial b_2}{\partial a_1} & \frac{\partial b_2}{\partial a_2} & \cdots & \frac{\partial b_2}{\partial a_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial b_m}{\partial a_1} & \frac{\partial b_m}{\partial a_2} & \cdots & \frac{\partial b_m}{\partial a_n} \end{bmatrix}^T = \begin{bmatrix} \left[\frac{\partial b_1}{\partial \boldsymbol{a}}\right]^T \\ \left[\frac{\partial b_2}{\partial \boldsymbol{a}}\right]^T \\ \vdots \\ \left[\frac{\partial b_m}{\partial \boldsymbol{a}}\right]^T \end{bmatrix}^T$$

$$= \begin{bmatrix} \frac{\partial b_1}{\partial \boldsymbol{a}} & \frac{\partial b_2}{\partial \boldsymbol{a}} & \cdots & \frac{\partial b_m}{\partial \boldsymbol{a}} \end{bmatrix} = \begin{bmatrix} \frac{\partial \boldsymbol{b}}{\partial a_1} & \frac{\partial \boldsymbol{b}}{\partial a_2} & \cdots & \frac{\partial \boldsymbol{b}}{\partial a_n} \end{bmatrix}^T \tag{2.56}$$

Jacobian Matrix [$\boldsymbol{b} \equiv \boldsymbol{a}$]

$$\frac{\partial \boldsymbol{a}^T}{\partial \boldsymbol{a}} = \begin{bmatrix} \frac{\partial a_1}{\partial \boldsymbol{a}} & \frac{\partial a_2}{\partial \boldsymbol{a}} & \cdots & \frac{\partial a_n}{\partial \boldsymbol{a}} \end{bmatrix} = \begin{bmatrix} \frac{\partial \boldsymbol{a}}{\partial a_1} & \frac{\partial \boldsymbol{a}}{\partial a_2} & \cdots & \frac{\partial \boldsymbol{a}}{\partial a_n} \end{bmatrix}^T = \begin{bmatrix} \boldsymbol{u}_1 & \boldsymbol{u}_2 & \cdots & \boldsymbol{u}_n \end{bmatrix}^T = \mathbf{1} \tag{2.57}$$

where,

$$\boldsymbol{a} = \sum_{i=1}^{n} a_i \, \boldsymbol{u}_i \tag{2.58}$$

and

$$\frac{\partial}{\partial \boldsymbol{a}} = \sum_{i=1}^{n} \frac{\partial}{\partial a_i} \, \boldsymbol{u}_i \tag{2.59}$$

Hence,

$$\frac{\partial \boldsymbol{a}}{\partial a_i} \equiv \frac{\partial a_i}{\partial \boldsymbol{a}} \equiv \boldsymbol{u}_i \tag{2.60}$$

Jacobian Matrix [$b = A\ a$]

$$\frac{\partial b^T}{\partial a} = \frac{\partial (a^T A^T)}{\partial a} = A^T \tag{2.61}$$

Jacobian Matrix [$b = A^T a$]

$$\frac{\partial b^T}{\partial a} = \frac{\partial (a^T A)}{\partial a} = A \tag{2.62}$$

Jacobian Matrix [$b = c^T A\ a$]

$$\frac{\partial b}{\partial a} = A^T c \tag{2.63}$$

Jacobian Matrix [$b = a^T A\ c$]

$$\frac{\partial b}{\partial a} = A\ c \tag{2.64}$$

Jacobian Matrix [$b = a^T A\ a$]

$$\frac{\partial b}{\partial a} = A\ a + A^T a = (A + A^T)\ a \tag{2.65}$$

2.3.3 Matrix/Element Derivatives

Derivative of an Element of a Matrix w.r.t. A

$$\frac{\partial a_{ij}}{\partial A} = \begin{bmatrix} & \vdots & \\ \cdots & 1 & \cdots \\ & \vdots & \end{bmatrix} \quad i\text{th row} \tag{2.66}$$

$$j\text{th col}$$

$$= u_i\ u_j^T$$

The result is a matrix with zeros everywhere at the intersection of row i and column j, which has a 1.

Derivative of A w.r.t. an element a_{ij}

$$\frac{\partial A}{\partial a_{ij}} = u_i\ u_j^T \equiv \frac{\partial a_{ij}}{\partial A} \tag{2.67}$$

2.3.4 Derivatives of Matrix Inverses

Derivative of A^{-1} where $m = n$

We have, by definition

$$A^{-1} A = 1 \tag{2.68}$$

from which

$$\frac{\partial A^{-1}}{\partial \phi} = -A^{-1} \frac{\partial A}{\partial \phi} A^{-1} \tag{2.69}$$

Derivative of A^{-1} w.r.t. a_{ij}

$$\begin{aligned} \frac{\partial A^{-1}}{\partial a_{ij}} &= -A^{-1} u_i u_j^T A^{-1} \\ &= -p_i q_j^T \end{aligned} \tag{2.70}$$

where p_i and q_j are solutions to

$$\begin{aligned} A p_i &= u_i \\ A^T q_j &= u_j \end{aligned} \tag{2.71}$$

2.3.5 Derivatives of Solutions to Equations

Derivatives of the Solution of $Ax = b$

Let A be $n \times n$, x and b be n-dimensional. Then,

$$\begin{aligned} \frac{\partial x}{\partial a_{ij}} &= \frac{\partial (A^{-1} b)}{\partial a_{ij}} \\ &= -A^{-1} u_i u_j^T A^{-1} b = -p_i x_j \end{aligned} \tag{2.72}$$

Derivative of an Element of the Solution of $A x = b$

Let

$$\begin{aligned} \frac{\partial x_k}{\partial a_{ij}} = u_k^T \frac{\partial x}{\partial a_{ij}} &= -u_k^T A^{-1} u_i u_j^T A^{-1} b \\ &= -\hat{x}_k^T u_i u_j^T x \\ &= -\hat{x}_{ki} x_j \end{aligned} \tag{2.73}$$

Hence,

$$\frac{\partial x_k}{\partial A} = -\hat{x}_k \, x^T \tag{2.74}$$

where $\hat{x}_k$ is the solution of

$$A^T \hat{x}_k = u_k \tag{2.75}$$

Derivative of a Linear Combination of the Elements of the Solution of $A\,x = b$

Let

$$\bar{x} = u^T x \tag{2.76}$$

Then, using (2.70),

$$\begin{aligned} \frac{\partial \bar{x}}{\partial a_{ij}} &= \frac{\partial (u^T x)}{\partial a_{ij}} = \frac{\partial (u^T A^{-1} b)}{\partial a_{ij}} = -u^T p_i \, q_j^T \, b \\ &= -(u^T p_i)\,(b^T q_j) \end{aligned} \tag{2.77}$$

As seen, the result comes from the appropriate linear combination of p_i and q_j.

Alternatively, also from (2.70),

$$\begin{aligned} \frac{\partial \bar{x}}{\partial a_{ij}} &= -u^T A^{-1} u_i u_j^T A^{-1} b \\ &= -q^T u_i u_j^T x \\ &= -q_i x_j \end{aligned} \tag{2.78}$$

where q is the solution of

$$A^T q = u \tag{2.79}$$

Hence

$$\frac{\partial \bar{x}}{\partial A} = -q \, x^T \tag{2.80}$$

Also,

$$\frac{\partial \bar{x}}{\partial a_{ij}} = u_i^T \frac{\partial \bar{x}}{\partial A} u_j \tag{2.81}$$

Hence

$$\frac{\partial \bar{x}}{\partial \phi} = \sum_{i=1}^{n}\sum_{j=1}^{n} \frac{\partial \bar{x}}{\partial a_{ij}} \frac{\partial a_{ij}}{\partial \phi} = \sum_{i=1}^{n}\sum_{j=1}^{n} u_i^T \frac{\partial \bar{x}}{\partial A} u_j \frac{\partial a_{ij}}{\partial \phi} \tag{2.82}$$

Special Case 1

Let

$$\frac{\partial A}{\partial \phi} \equiv u\, u_i^T - u\, u_j^T \tag{2.83}$$

Then, from first principles

$$\frac{\partial \bar{x}}{\partial \phi} = -u^T \frac{\partial x}{\partial \phi} = -u^T A^{-1} \frac{\partial A}{\partial \phi} A^{-1} b \tag{2.84}$$

and substituting (2.83),

$$\begin{aligned} \frac{\partial \bar{x}}{\partial \phi} &= -u^T A^{-1} u\, (u_i - u_j)^T A^{-1} b \\ &= -u^T p\, (x_i - x_j) \end{aligned} \tag{2.85}$$

where p is the solution to

$$Ap = u \tag{2.86}$$

Let

$$u = u_i - u_j \tag{2.87}$$

Then

$$\frac{\partial \bar{x}}{\partial \phi} = -(p_i - p_j)(x_i - x_j) \tag{2.88}$$

and

$$\frac{\partial A}{\partial \phi} = (u_i - u_j)(u_i - u_j)^T = \begin{bmatrix} & \vdots & & \vdots & \\ \cdots & 1 & \cdots & -1 & \cdots \\ & \vdots & & \vdots & \\ \cdots & -1 & \cdots & 1 & \cdots \\ & \vdots & & \vdots & \end{bmatrix} \begin{matrix} \\ \leftarrow i\text{th row} \\ \\ \leftarrow j\text{th row} \\ \\ \end{matrix} \qquad (2.89)$$

$$\begin{matrix} \uparrow & \uparrow \\ i\text{th col} & j\text{th col} \end{matrix}$$

Special Case 2

Let

$$\frac{\partial A}{\partial \phi} \equiv (u_k - u_\ell)(u_k - u_\ell)^T \qquad (2.90)$$

Then

$$\begin{aligned} \frac{\partial \bar{x}}{\partial \phi} &= -u^T A^{-1} (u_k - u_\ell)(u_k - u_\ell)^T A^{-1} b \\ &= -(q_k - q_\ell)(x_k - x_\ell) \end{aligned} \qquad (2.91)$$

where q is the solution to

$$A^T q = u \qquad (2.92)$$

and u is the vector leading to $\bar{x} = u^T x$.

Comparison of Case 1 with Case 2

For Case 2 the indices of the linear combination u are independent of indices of A affected by ϕ, whereas in Case 1 the indices of the linear combination are, coincidentally, the same as those in A affected by ϕ. Note that in Case 1 we solve the original system for two different right hand sides, whereas in Case 2 we solve the original system once and the adjoint system once.

Chapter 3

SOLVING EQUATIONS

In this chapter we consider various aspects of solving linear, nonlinear and differential equations. Following some definitions we turn to the classical Gauss elimination procedure for solving linear equations. Multiple right hand side vectors, matrix inversion, pivoting and ill-conditioning are discussed. Next we introduce LU factorization. This section is followed by some numerical examples. Iterative (relaxation) methods are introduced. We then turn to the problem of solving one nonlinear equation in one unknown. The state equations are introduced. Taylor based and other algorithms for solving the state equations are outlined.

3.1 A Linear System of Equations

The System

We consider n linear simultaneous equations in n unknowns, namely,

$$\begin{aligned} a_{11}\,x_1 + a_{12}\,x_2 + \ldots + a_{1n}\,x_n &= b_1 \qquad (1) \\ a_{21}\,x_1 + a_{22}\,x_2 + \ldots + a_{2n}\,x_n &= b_2 \qquad (2) \\ \vdots \qquad\qquad\qquad\qquad\qquad &\quad\ \vdots \\ a_{n1}\,x_1 + a_{n2}\,x_2 + \ldots + a_{nn}\,x_n &= b_n \qquad (n) \end{aligned} \tag{3.1}$$

Notation

In matrix notation we write

$$\boldsymbol{A}\,\boldsymbol{x} = \boldsymbol{b} \tag{3.2}$$

where

$$\boldsymbol{A} \triangleq \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \ldots & a_{nn} \end{bmatrix}, \quad \boldsymbol{x} \triangleq \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad \boldsymbol{b} \triangleq \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \tag{3.3}$$

Note that

a_{ij}, $j = 1, 2, \ldots, n$ generates the ith row,
a_{ij}, $i = 1, 2, \ldots, n$ generates the jth column,
a_{ij}, $i, j = 1, 2, \ldots, n$ generates the matrix A.

3.2 Gauss Elimination

The well-known Gauss elimination algorithm is based on the following concepts.

3.2.1 Forward Reduction

Eliminating x_1

Divide (1) in (3.1) by a_{11} to give

$$x_1 + \frac{a_{12}}{a_{11}} x_2 + \ldots + \frac{a_{1n}}{a_{11}} x_n = \frac{b_1}{a_{11}} \tag{3.4}$$

Multiply this equation by a_{21} to give

$$a_{21} x_1 + a_{21} \frac{a_{12}}{a_{11}} x_2 + \ldots + a_{21} \frac{a_{1n}}{a_{11}} x_n = a_{21} \frac{b_1}{a_{11}} \tag{3.5}$$

Subtract this equation from (2) in (3.1) to give

$$0\, x_1 + \left[a_{22} - a_{21} \frac{a_{12}}{a_{11}}\right] x_2 + \ldots + \left[a_{2n} - a_{21} \frac{a_{1n}}{a_{11}}\right] x_n = b_2 - a_{21} \frac{b_1}{a_{11}} \tag{3.6}$$

If we had multiplied the first equation by a_{i1} and subtracted the result from the ith equation we would have had

$$0\, x_1 + \left[a_{i2} - a_{i1} \frac{a_{12}}{a_{11}}\right] x_2 + \ldots + \left[a_{in} - a_{i1} \frac{a_{1n}}{a_{11}}\right] x_n = b_i - a_{i1} \frac{b_1}{a_{11}} \tag{3.7}$$

Writing out the new system we have

$$\begin{bmatrix} 1 & \frac{a_{12}}{a_{11}} & \cdots & \frac{a_{1n}}{a_{11}} \\ 0 & a_{22} - a_{21}\frac{a_{12}}{a_{11}} & \cdots & a_{2n} - a_{21}\frac{a_{1n}}{a_{11}} \\ \vdots & \vdots & & \vdots \\ 0 & a_{n2} - a_{n1}\frac{a_{12}}{a_{11}} & \cdots & a_{nn} - a_{n1}\frac{a_{1n}}{a_{11}} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} \frac{b_1}{a_{11}} \\ b_2 - a_{21}\frac{b_1}{a_{11}} \\ \vdots \\ b_n - a_{n1}\frac{b_1}{a_{11}} \end{bmatrix} \tag{3.8}$$

The lower right partition involves $n-1$ equations in $n-1$ unknowns. Let the whole system be represented as originally but with superscript i to distinguish the results of different iterations. We now have

$$\begin{aligned} a_{ij}^2 &= a_{ij}^1 - a_{i1}^1 \frac{a_{1j}^1}{a_{11}^1}, \quad i = 2, 3, \ldots, n \\ & \qquad\qquad\quad j = 1, 2, \ldots, n \\ b_i^2 &= b_i^1 - a_{i1}^1 \frac{b_1^1}{a_{11}^1}, \quad i = 2, 3, \ldots, n \end{aligned} \tag{3.9}$$

Check: for $j = 1$, we have $a_{i1}^2 = a_{i1}^1 - a_{i1}^1 = 0$. Here, we have eliminated x_1.

Eliminating x_2

The same approach is repeated until x_2 is eliminated with $n-2$ equations remaining. In this case we have

$$\begin{aligned} a_{ij}^3 &= a_{ij}^2 - a_{i2}^2 \frac{a_{2j}^2}{a_{22}^2}, \quad i = 3, 4, \ldots, n \\ & \qquad\qquad\quad j = 2, 3, \ldots, n \\ b_i^3 &= b_i^2 - a_{i2}^2 \frac{b_2^2}{a_{22}^2}, \quad i = 3, 4, \ldots, n \end{aligned} \tag{3.10}$$

Eliminating x_k

When the kth variable is eliminated we have

$$
\begin{aligned}
a_{ij}^{k+1} &= a_{ij}^{k} - a_{ik}^{k} \frac{a_{kj}^{k}}{a_{kk}^{k}}, \quad i = k+1,\ k+2,\ \ldots,\ n \\
&\qquad\qquad\qquad\qquad\quad j = k,\ k+1,\ \ldots,\ n \\
b_{i}^{k+1} &= b_{i}^{k} - a_{ik}^{k} \frac{b_{k}^{k}}{a_{kk}^{k}}, \quad i = k+1,\ k+2,\ \ldots,\ n
\end{aligned}
\tag{3.11}
$$

The Final Result

The final result of Gauss elimination is

$$
\begin{bmatrix}
1 & \frac{a_{12}^{1}}{a_{11}^{1}} & \frac{a_{13}^{1}}{a_{11}^{1}} & \cdots & \frac{a_{1n}^{1}}{a_{11}^{1}} \\
0 & 1 & \frac{a_{23}^{2}}{a_{22}^{2}} & \cdots & \frac{a_{2n}^{2}}{a_{22}^{2}} \\
0 & 0 & 1 & \cdots & \frac{a_{3n}^{3}}{a_{33}^{3}} \\
\vdots & \vdots & \vdots & & \vdots \\
0 & 0 & 0 & \cdots & 1
\end{bmatrix}
\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix}
=
\begin{bmatrix}
\frac{b_{1}^{1}}{a_{11}^{1}} \\
\frac{b_{2}^{2}}{a_{22}^{2}} \\
\frac{b_{3}^{3}}{a_{33}^{3}} \\
\vdots \\
\frac{b_{n}^{n}}{a_{nn}^{n}}
\end{bmatrix}
\tag{3.12}
$$

Thus, the original matrix has been transformed to *upper-triangular* form.

3.2.2 Backward Substitution

The System

We have obtained a system of the form

$$
\begin{bmatrix}
1 & u_{12} & u_{13} & \cdots & u_{1n} \\
0 & 1 & u_{23} & \cdots & u_{2n} \\
0 & 0 & 1 & \cdots & u_{3n} \\
\vdots & \vdots & \vdots & & \vdots \\
0 & 0 & 0 & \cdots & 1
\end{bmatrix}
\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix}
=
\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{bmatrix}
\tag{3.13}
$$

Substitution

Now,

$$\begin{aligned} x_n &= b_n \\ x_{n-1} &= b_{n-1} - u_{n-1,n}\, x_n \\ x_{n-2} &= b_{n-2} - (u_{n-2,n-1}\, x_{n-1} + u_{n-2,n}\, x_n) \\ &\vdots \end{aligned} \tag{3.14}$$

and, in general,

$$x_i = b_i - \sum_{j=i+1}^{n} u_{ij}\, x_j \tag{3.15}$$

3.2.3 Multiple Right Hand Side Vectors

The Multipliers

If $\boldsymbol{A}$ stays the same for different problems, the upper triangular form obtained by forward reduction is unchanged. We need, however, the terms (multipliers)

$$m_{ik} = \frac{a_{ik}^{k}}{a_{kk}^{k}}, \quad i = k+1,\ k+2,\ \ldots,\ n \tag{3.16}$$

i.e.,

$$\begin{aligned} m_{i1} &= \frac{a_{i1}^{1}}{a_{11}^{1}}, \quad i = 2,\ 3,\ \ldots,\ n \text{ (hence: } m_{21},\ m_{31},\ \ldots,\ m_{n1}) \\ m_{i2} &= \frac{a_{i2}^{2}}{a_{22}^{2}}, \quad i = 3,\ 4,\ \ldots,\ n \text{ (hence: } m_{32},\ m_{42},\ \ldots,\ m_{n2}) \\ &\vdots \end{aligned} \tag{3.17}$$

Saving Storage

To save storage we may reuse the available matrix locations, namely,

$$\begin{bmatrix} a_{11}^{1} & a_{12}^{1} & a_{13}^{1} & \cdots & a_{1n}^{1} \\ m_{21} & a_{22}^{2} & a_{23}^{2} & \cdots & a_{2n}^{2} \\ m_{31} & m_{32} & a_{33}^{3} & \cdots & a_{3n}^{3} \\ \vdots & \vdots & \vdots & & \vdots \\ m_{n1} & m_{n2} & m_{n3} & \cdots & a_{nn}^{n} \end{bmatrix}, \quad \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix}, \quad \begin{bmatrix} b_1^{1} \\ b_2^{2} \\ b_3^{3} \\ \vdots \\ b_n^{n} \end{bmatrix} \tag{3.18}$$

since the number of multipliers is the same as the number of zero elements. Then we need only the calculations for b_i^{k+1}. Note that in practice only *one* variable m is needed at any time.

3.2.4 Matrix Inversion

The Identity Matrix

By definition,

$$A\,A^{-1} = \mathbf{1} \tag{3.19}$$

where

$$\mathbf{1} \triangleq \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \tag{3.20}$$

Simultaneous Elimination for Inversion

Suppose we want A^{-1}. Let us create the n by n matrix X as

$$X \triangleq [x_1\ x_2\ \cdots\ x_n] \tag{3.21}$$

Let us define the n right hand sides

$$b_1 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix},\quad b_2 = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix},\quad \ldots,\quad b_n = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} \tag{3.22}$$

from which it follows that

$$\mathbf{1} = [b_1\ b_2\ \cdots\ b_n]$$

We can solve the n systems of equations

$$\begin{aligned} A\,x_1 &= b_1 \\ A\,x_2 &= b_2 \\ &\vdots \\ A\,x_n &= b_n \end{aligned} \tag{3.23}$$

Then, because

$$A\,[x_1\ x_2\ \ldots\ x_n] = [b_1\ b_2\ \ldots\ b_n] \tag{3.24}$$

we recognize the X matrix (see (3.21)) as

$$X = A^{-1} \tag{3.25}$$

In a numerical example the Gauss elimination procedure may be applied *simultaneously* to the n problems defined above.

Evaluation of Determinants

It is not recommended to use the direct method. It requires $n!$ multiplications. After Gauss elimination take

$$\det A = a_{11}^{1}\ \ a_{22}^{2}\ \ \ldots\ a_{nn}^{n} \tag{3.26}$$

3.2.5 Pivoting

To avoid error accumulation which may destroy the significance of the numerical result we need the smallest possible multipliers, obtained as follows.

$$m_{ik} = \frac{a_{ik}^{k}}{a_{kk}^{k}},\quad i = k+1,\ k+2,\ \ldots,\ n \tag{3.27}$$

This generates the kth column. Arrange or rearrange the equations so that

$$|a_{kk}^{k}| = \max_{k \le i \le n} |a_{ik}^{k}| \tag{3.28}$$

In this case $|m_{ik}| \le 1$. In practice, search down the column and interchange appropriate rows. This will, of course, handle the situation when $a_{kk}^{k} = 0$, and is essential for ill-conditioned systems. Note that interchanging rows does not affect the order of the variables. *Complete pivoting* (rarely worthwhile) involves searching the whole remaining submatrix for the largest element and interchanging rows and columns.

3.2.6 Ill-Conditioning

Small changes in the coefficients leads to large changes in solution, for example,

$$\begin{aligned} x_1 - x_2 &= 1 & \qquad x_1 - x_2 &= 1 \\ x_1 - 1.00001\ x_2 &= 0 & \qquad x_1 - .99999\ x_2 &= 0 \end{aligned} \tag{3.29}$$

yield, respectively,

$$x = \begin{bmatrix} 100{,}001 \\ 100{,}000 \end{bmatrix} \qquad x = \begin{bmatrix} -99{,}999 \\ -100{,}000 \end{bmatrix} \tag{3.30}$$

Physically, this type of ill-conditioning can be thought of as being due to the intersection of nearly parallel lines.

Gauss elimination on the two sets of equations gives, respectively,

$$\begin{aligned} x_1 - x_2 &= 1 & x_1 - x_2 &= 1 \\ -.00001x_2 &= -1 & +.00001x_2 &= -1 \end{aligned} \tag{3.31}$$

3.2.7 The Computational Effort

It can be shown that the number of multiplications and divisions is as follows.

Forward reduction

$$\frac{n^3}{3} + \frac{n^2}{2} + \frac{n}{6} \tag{3.32}$$

Back substitution

$$\frac{n(n-1)}{2} \tag{3.33}$$

Total

$$\frac{n^3}{3} + n^2 - \frac{n}{3} \tag{3.34}$$

3.3 LU Factorization (Crout, Choleski, triangularization)

3.3.1 Basics of LU Factorization

Factored Forms

Assume that the coefficient matrix A can be decomposed into the factored form

$$A = L\,U \tag{3.35}$$

where L is lower triangular and U is upper triangular.

Example

Consider, for example,

$$\begin{bmatrix} \ell_{11} & 0 & 0 \\ \ell_{21} & \ell_{22} & 0 \\ \ell_{31} & \ell_{32} & \ell_{33} \end{bmatrix} \begin{bmatrix} 1 & u_{12} & u_{13} \\ 0 & 1 & u_{23} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \tag{3.36}$$

We can equate the coefficients as follows.

column 1 $\ell_{11} = a_{11}, \quad \ell_{21} = a_{21}, \quad \ell_{31} = a_{31}$

column 2 $\ell_{11}\, u_{12} = a_{12}, \quad \ell_{21}\, u_{12} + \ell_{22} = a_{22}, \quad \ell_{31}\, u_{12} + \ell_{32} = a_{32}$

column 3
$$\ell_{11}\, u_{13} = a_{13}$$
$$\ell_{21}\, u_{13} + \ell_{22}\, u_{23} = a_{23}$$
$$\ell_{31}\, u_{13} + \ell_{32}\, u_{23} + \ell_{33} = a_{33}$$

Substitution Phases

The system of linear equations is now given by

$$\boldsymbol{L\, U\, x = b} \tag{3.37}$$

where $\boldsymbol{A}$ has been replaced by the LU factors. Let

$$\boldsymbol{U\, x = y} \tag{3.38}$$

Then solve

$$\boldsymbol{L\, y = b} \tag{3.39}$$

for $\boldsymbol{y}$ after *forward* substitution followed by

$$\boldsymbol{U\, x = y} \tag{3.40}$$

for $\boldsymbol{x}$ after *backward* substitution. Note that factorization need be done only once for multiple $\boldsymbol{b}$.

Example

Specifically, for our 3×3 example, we take the following.

column 1

$$\ell_{i1} = a_{i1}$$

column 2

Solving for u_{12} from column 2

$$u_{12} = a_{12}/\ell_{11}$$
$$\ell_{22} = a_{22} - \ell_{21}\, u_{12}$$
$$\ell_{32} = a_{32} - \ell_{31}\, u_{12}$$

column 3

Solving for u_{13} from column 3

$$u_{13} = a_{13}/\ell_{11}$$

Then,

$$u_{23} = (a_{23} - \ell_{21}\, u_{13})/\ell_{22}$$
$$\ell_{33} = a_{33} - \ell_{31}\, u_{13} - \ell_{32}\, u_{23}$$

General Formulas

In general,

$$\begin{aligned} u_{ij} &= (a_{ij} - \sum_{k=1}^{i-1} \ell_{ik}\, u_{kj})/\ell_{ii}, \quad i < j \\ \ell_{ij} &= a_{ij} - \sum_{k=1}^{j-1} \ell_{ik}\, u_{uk}, \qquad i \geq j \end{aligned} \tag{3.41}$$

After conversion to u_{ij} and ℓ_{ij} simply replace a_{ij} appropriately since this is not needed again.

Example

Step 1

$$\begin{bmatrix} \ell_{11} & u_{12} & u_{13} \\ \ell_{21} & a_{22} & a_{23} \\ \ell_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12}/\ell_{11} & a_{13}/\ell_{11} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \tag{3.42}$$

Step 2

$$\begin{bmatrix} \ell_{22} & u_{23} \\ \ell_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} a_{22}-\ell_{21}u_{12} & (a_{23}-\ell_{21}u_{13})/\ell_{22} \\ a_{32}-\ell_{31}u_{12} & a_{33} \end{bmatrix} \tag{3.43}$$

Step 3

$$[\ell_{33}] = [a_{33} - \ell_{31}\, u_{13} - \ell_{32}\, u_{23}] \tag{3.44}$$

Hence,

$$\begin{bmatrix} \ell_{11} & u_{12} & u_{13} \\ \ell_{21} & \ell_{22} & u_{23} \\ \ell_{31} & \ell_{32} & \ell_{33} \end{bmatrix} \tag{3.45}$$

Choleski factorization can be described as follows. If $\boldsymbol{A}$ is symmetric, $\boldsymbol{A}^T = \boldsymbol{A}$. Make the diagonal elements of $\boldsymbol{L}$ and $\boldsymbol{U}$ equal and then $\boldsymbol{U} = \boldsymbol{L}^T$. Then we need only to evaluate $\boldsymbol{L}$.

3.3.2 The Computational Effort

Calculation of u_{ij}, ℓ_{ij}

$$\frac{n^3}{3} - \frac{n}{3} \tag{3.46}$$

Forward and backward substitution

$$n^2 \tag{3.47}$$

Total

$$\frac{n^3}{3} + n^2 - \frac{n}{3} \tag{3.48}$$

3.3.3 Matrix Inversion and Superposition

Resistive Network Formulas

Consider, for a resistor network,

$$G\,v = i \tag{3.49}$$

where G is the conductance matrix of the network. Hence, the solution vector v can be decomposed into a linear combination of vectors $v_1, v_2, \ldots, v_n$ such that

$$G\,[v_1\ v_2\ \ldots\ v_n] = [i_1\ i_2\ \ldots\ i_n] \tag{3.50}$$

where we place, on the right hand side, appropriate unit current excitations $i_1, i_2, \ldots, i_n$.

We can make up any desired vector i as a linear combination of i_i, i.e.,

$$i = a_1\,i_1 + a_2\,i_2 + \ldots a_n\,i_n \tag{3.51}$$

Note that this demonstrates superposition, since

$$\begin{aligned} v = G^{-1}\,i &= G^{-1}\,(a_1\,i_1 + a_2\,i_2 + \ldots + a_n\,i_n) \\ &= a_1\,v_1 + a_2\,v_2 + \ldots + a_n\,v_n \end{aligned} \tag{3.52}$$

where the a_i are specific constants. Note that n^2 multiplications are involved. This is the same as in the substitution steps in the LU factorization of the coefficient matrix G.

Circuit Example

For the 4-node circuit shown in Fig. 3.1

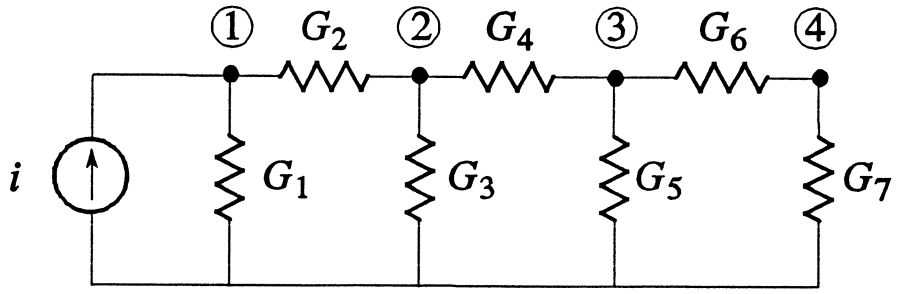

Fig. 3.1 A resistive circuit.

$$G = \begin{bmatrix} G_1+G_2 & -G_2 & 0 & 0 \\ -G_2 & G_2+G_3+G_4 & -G_4 & 0 \\ 0 & -G_4 & G_4+G_5+G_6 & -G_6 \\ 0 & 0 & -G_6 & G_6+G_7 \end{bmatrix} \tag{3.53}$$

This is a tridiagonal matrix. The inverse of G is of the form

$$G^{-1} = \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix} \tag{3.54}$$

whereas the LU decomposition of G is of the form

$$LU = \begin{bmatrix} \times & & & \\ \times & \times & & \\ & \times & \times & \\ & & \times & \times \end{bmatrix} \begin{bmatrix} \times & \times & & \\ & \times & \times & \\ & & \times & \times \\ & & & \times \end{bmatrix} \tag{3.55}$$

Here, two substitution steps require only $3n-2$ multiplications and divisions.

3.4 Examples

3.4.1 Numerical Inversion of a Matrix

We start with a pair of matrices

$$\begin{bmatrix} 3 & -2 & 0 \\ -2 & 5 & -2 \\ 0 & -2 & 3 \end{bmatrix}, \quad \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.56}$$

The first is the coefficient matrix, the second comprises three unit right hand side vectors assembled as unit matrix. Divide row 1 by 3:

$$\left[\begin{array}{ccc|ccc} 1 & -2/3 & 0 & 1/3 & 0 & 0 \\ -2 & 5 & -2 & 0 & 1 & 0 \\ 0 & -2 & 3 & 0 & 0 & 1 \end{array}\right] \tag{3.57}$$

Multiply row 1 by -2 and subtract from row 2 to update row 2:

$$\left[\begin{array}{ccc|ccc} 1 & -2/3 & 0 & 1/3 & 0 & 0 \\ 0 & 11/3 & -2 & 2/3 & 1 & 0 \\ 0 & -2 & 3 & 0 & 0 & 1 \end{array}\right] \tag{3.58}$$

Divide row 2 by 11/3:

$$\left[\begin{array}{ccc|ccc} 1 & -2/3 & 0 & 1/3 & 0 & 0 \\ 0 & 1 & -6/11 & 2/11 & 3/11 & 0 \\ 0 & -2 & 3 & 0 & 0 & 1 \end{array}\right] \tag{3.59}$$

Multiply row 2 by -2 and subtract from row 3 to update row 3:

$$\left[\begin{array}{ccc|ccc} 1 & -2/3 & 0 & 1/3 & 0 & 0 \\ 0 & 1 & -6/11 & 2/11 & 3/11 & 0 \\ 0 & 0 & 21/11 & 4/11 & 6/11 & 1 \end{array}\right] \tag{3.60}$$

Divide row 3 by 21/11:

$$\left[\begin{array}{ccc|ccc} 1 & -2/3 & 0 & 1/3 & 0 & 0 \\ 0 & 1 & -6/11 & 2/11 & 3/11 & 0 \\ 0 & 0 & 1 & 4/21 & 6/21 & 11/21 \end{array}\right] \tag{3.61}$$

Multiply row 3 by $-6/11$ and subtract from row 2 to update row 2:

$$\left[\begin{array}{ccc|ccc} 1 & -2/3 & 0 & 1/3 & 0 & 0 \\ 0 & 1 & 0 & 6/21 & 9/21 & 6/21 \\ 0 & 0 & 1 & 4/21 & 6/21 & 11/21 \end{array}\right] \tag{3.62}$$

Multiply row 2 by $-2/3$ and subtract from row 1 to update row 1:

$$\left[\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right] \quad \left[\begin{array}{ccc} 11/21 & 6/21 & 4/21 \\ 6/21 & 9/21 & 6/21 \\ 4/21 & 6/21 & 11/21 \end{array}\right] \tag{3.63}$$

We have reduced the coefficient matrix to a unit matrix. The right hand sides have been transformed to solutions collectively interpreted as the inverse.

3.4.2 Example of Matrix Inversion

Consider

$$G\,G^{-1} = \mathbf{1} \tag{3.64}$$

in the form

$$\left[\begin{array}{ccc} 3 & -2 & 0 \\ -2 & 5 & -2 \\ 0 & -2 & 3 \end{array}\right] \left[\begin{array}{ccc} a & d & e \\ d & b & f \\ e & f & c \end{array}\right] = \left[\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right] \tag{3.65}$$

We can write for column 1

$$\begin{aligned} 3a - 2d &= 1 \quad (1) \\ -2a + 5d - 2e &= 0 \quad (2) \\ -2d + 3e &= 0 \quad (3) \end{aligned} \tag{3.66}$$

from which, using (1) of (3.66),

$$a = \frac{1 + 2d}{3} \tag{3.67}$$

using (2) of (3.66) and substituting for this value of a

$$-\frac{2}{3}(1 + 2d) + 5d - 2e = 0 \tag{3.68}$$

or

$$-2 - 4d + 15d - 6e = 0 \tag{3.69}$$

or

$$11d - 6e = 2 \tag{3.70}$$

From (3) of (3.66)

$$-4d + 6e = 0 \tag{3.71}$$

which, added to (3.70), gives

$$7d = 2 \tag{3.72}$$

from which

$$d = 2/7 \tag{3.73}$$

Substituting for this value of d in (3.67)

$$a = \frac{1 + 4/7}{3} = \frac{11}{21} \tag{3.74}$$

and in (3) of (3.66)

$$e = \frac{2\ (2/7)}{3} = \frac{4}{21} \tag{3.75}$$

We can now write for column 2, substituting for d,

$$3\left[\frac{2}{7}\right] - 2b = 0 \tag{3.76}$$

from which

$$b = \frac{3}{7} \tag{3.77}$$

and, substituting for b,

$$-2\left[\frac{3}{7}\right] + 3f = 0 \tag{3.78}$$

from which

$$f = \frac{2}{7} \tag{3.79}$$

and for column 3, substituting for f,

$$-2\left[\frac{2}{7}\right] + 3c = 1 \tag{3.80}$$

from which

$$c = \frac{1 + 4/7}{3} = \frac{11}{21} \tag{3.81}$$

In conclusion, the desired inverse is given by

$$G^{-1} = \frac{1}{21}\begin{bmatrix} 11 & 6 & 4 \\ 6 & 9 & 6 \\ 4 & 6 & 11 \end{bmatrix} \tag{3.82}$$

3.5 Iterative Methods

3.5.1 Jacobi's Method

Consider again the n equations in n unknowns of (3.1). Rewrite this system in the form

$$\begin{aligned} x_1 &= \frac{1}{a_{11}}(b_1 - a_{12}x_2 - \ldots - a_{1n}x_n) \\ x_2 &= \frac{1}{a_{22}}(b_2 - a_{21}x_1 - \ldots - a_{2n}x_n) \\ &\vdots \\ x_n &= \frac{1}{a_{nn}}(b_n - a_{n1}x_1 - \ldots - a_{n,\,n-1}x_{n-1}) \end{aligned} \tag{3.83}$$

Guess at $x_1, x_2, \ldots, x_n$ on the right hand side to find the left hand side. Substitute on the right hand side and repeat. Convergence of this process is not guaranteed.

3.5.2 The Gauss-Seidel Method

The Gauss-Seidel method is similar to the Jacobi method except that new values are substituted immediately after they are obtained.

Consider the system of $Ax = b$ of (3.2). Partition the coefficient matrix A so that

$$A = L + 1 + U \tag{3.84}$$

(not to be confused with LU factorization) where we assume that each equation has already been divided by the corresponding diagonal element. The basic iteration is contrasted with the Jacobi iteration as follows.

Jacobi

$$x^{j+1} = -(L + U)\, x^j + b \tag{3.85}$$

Gauss-Seidel

$$x^{j+1} = -L\, x^{j+1} - U\, x^j + b \tag{3.86}$$

or

$$(1 + L)\, x^{j+1} = b - U\, x^j \tag{3.87}$$

The term relaxation method describes any method in which a new approximation is obtained from the previous approximation and residuals, i.e., for Gauss-Seidel,

$$x^{j+1} = x^j + [b - L\, x^{j+1} - x^j - U\, x^j] \tag{3.88}$$

where the expression in brackets is the residual (known).

3.5.3 Over-Relaxation

Over-relaxation is an approach generally used for the linear equations arising out of the solution of elliptic partial differential equations.

Example

Consider, for example,

$$x^{j+1} = a\, x^j + b \tag{3.89}$$

where $a = 1/2$ and $b = 1$. Let $x^0 = 0$. Then the iterative process produces the following results.

$$
\begin{aligned}
x^0 &= 0 \\
x^1 &= 1 \\
x^2 &= 1.5 \\
x^3 &= 1.75 \\
x^4 &= 1.875 \\
&\vdots \\
x^\infty &= 2
\end{aligned}
\tag{3.90}
$$

These results are plotted in Fig. 3.2. Analytically,

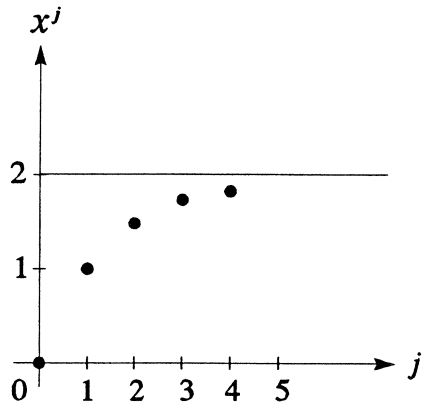

Fig. 3.2 Illustration diagram for over-relaxation.

$$
\begin{aligned}
x^1 &= b \\
x^2 &= ab + b \\
x^3 &= a^2b + ab + b \\
x^4 &= a^3b + a^2b + ab + b \\
&\vdots \\
x^n &= b(1 + a + a^2 + a^3 + \ldots + a^{n-1} + \ldots)
\end{aligned}
\tag{3.91}
$$

The term in brackets is a geometric series. As $n \to \infty$

$$
x^\infty = \frac{b}{1 - a} \quad \text{for } |a| < 1 \tag{3.92}
$$

For $a = 1/2$, $b = 1$, we have $x^\infty = 2$. To speed up convergence consider

$$x^{j+1} = x^j + \omega\,(\bar{x}^{j+1} - x^j), \quad \omega > 0 \tag{3.93}$$

where

$$\bar{x}^{j+1} = a\,x^j + b \tag{3.94}$$

Then

$$\begin{aligned} x^{j+1} &= x^j + \omega\,(a - 1)\,x^j + \omega b \\ &= (\omega a + 1 - \omega)\,x^j + \omega b \end{aligned} \tag{3.95}$$

Now when $\omega = 1$ we recover the basic relaxation method. If $\omega < 1$ we have under-relaxation. If $\omega > 1$ we have over-relaxation. Usually we take $1 < \omega < 2$.

Let us consider $\omega = 3/2$. Then we obtain

$$\begin{aligned} x^0 &= 0 \\ x^1 &= ((1.5)\,(0.5) + 1 - 1.5)0 + 1.5 = 1.5 \\ x^2 &= 1.875 \\ x^3 &= 1.96875 \\ &\vdots \end{aligned} \tag{3.96}$$

The rate of convergence, it should be noted, is twice what it was before.

General Formulas

Let us formally consider over-relaxation, as follows. We take

$$x^{j+1} = x^j + \omega(\bar{x}^{j+1} - x^j) \tag{3.97}$$

where x^{j+1} is calculated by Gauss-Seidel iteration. Therefore,

$$\begin{aligned} x^{j+1} &= x^j + \omega(b - L\,x^{j+1} - x^j - U\,x^j) \\ &= (-\omega\,U + (1 - \omega)\,1)\,x^j - \omega\,L\,x^{j+1} + \omega b \end{aligned} \tag{3.98}$$

It is hard to predict ω in advance. It is usually recommended to choose a high value in the range $1 < \omega < 2$.

3.5.4 Convergence

Iterative Formula

Let us consider the iterative formula

$$x^{j+1} = A\,x^j + B \tag{3.99}$$

where A and B are square matrices.

Solution

The solution is

$$x^\infty = A\,x^\infty + B \tag{3.100}$$

Example

In the previous example we have

$$2 = (0.5)2 + 1 \tag{3.101}$$

Error Vector

Then we may express the error vector as

$$e^j \triangleq x^\infty - x^j \tag{3.102}$$

so that

$$x^\infty - x^{j+1} = A\,(x^\infty - x^j) \tag{3.103}$$

or

$$e^{j+1} = A\,e^j = A^{j+1}\,e^0 \tag{3.104}$$

since

$$\begin{aligned} e^1 &= A\,e^0 \\ e^2 &= A\,e^1 = A^2\,e^0 \\ e^3 &= A\,e^2 = A^3\,e^0 \\ &\vdots \end{aligned} \tag{3.105}$$

where superscripts of A denote exponentiation.

Forms of A

Thus A must have the effect of reducing e^{j+1} for convergence in the Jacobi method. A takes the form

$$- L - U \tag{3.106}$$

whereas in the Gauss-Seidel method A takes the form

$$- (1 + L)^{-1} U \tag{3.107}$$

Diagonal Dominance

Diagonal dominance is sufficient for convergence, i.e., if

$$\frac{\sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|}{|a_{ii}|} < 1 \quad \text{for} \quad j = 1, 2, \ldots, n \tag{3.108}$$

Positive Definiteness

Positive definiteness also ensures convergence, i.e., for all x

$$x^T A x \geq 0 \tag{3.109}$$

and

$$x^T A x = 0 \tag{3.110}$$

implies that

$$x = 0 \tag{3.111}$$

The diagonal elements are positive in a positive-definite matrix. The element of largest modulus lies on the diagonal.

3.6 Nonlinear Equations

Consider the single nonlinear equation in one unknown x

$$h(x) = 0 \tag{3.112}$$

whose solution or solutions are desired. This is one-dimensional, i.e., we have to evaluate the single variable x. In other words, we want to evaluate the root(s) of h. Traditional situations in the solution of $h = 0$ are shown in Fig. 3.3. Fig. 3.4 shows that there are multiple solutions for $h = 0$. Fig. 3.5 shows the situations of no solutions for $h = 0$.

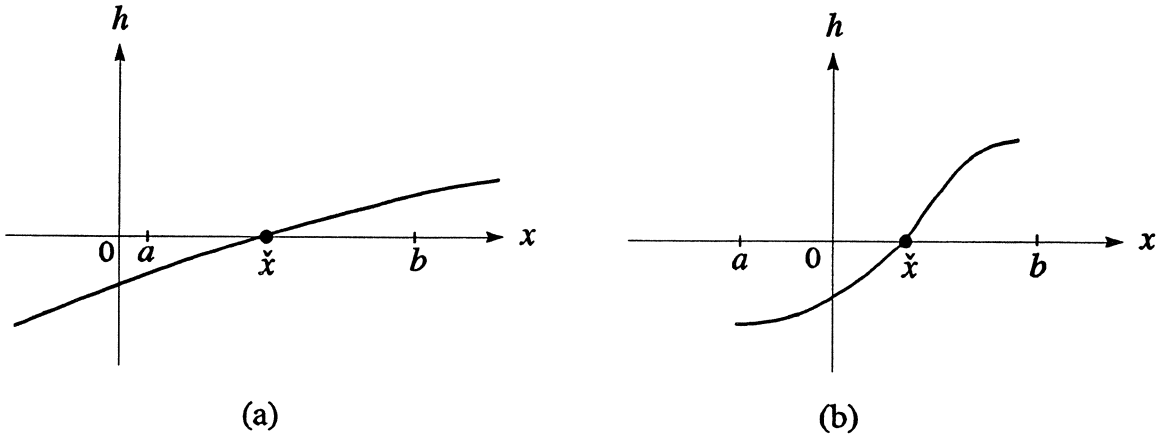

Fig. 3.3 Traditional situations in the solution of $h = 0$. (a) ideal, (b) convergence sensitive to the starting point.

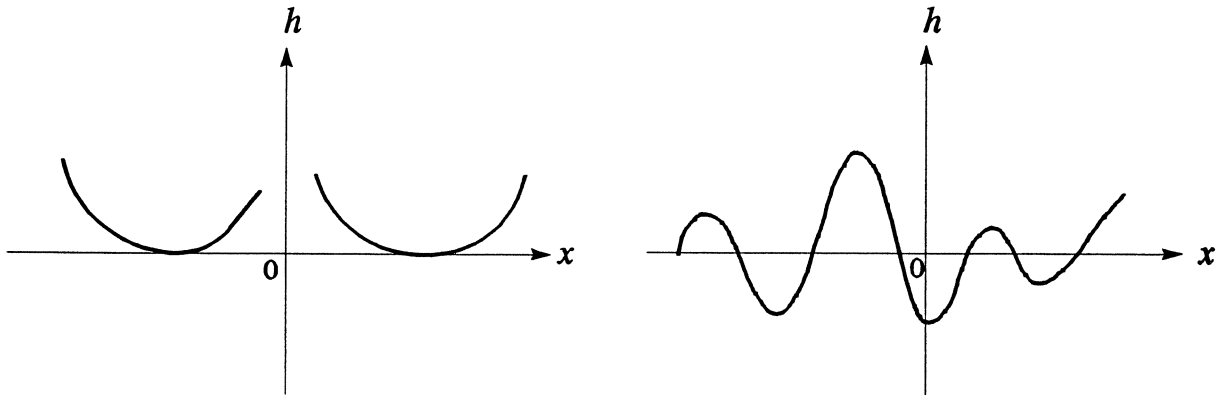

Fig. 3.4 Situations of multiple solutions for $h = 0$.

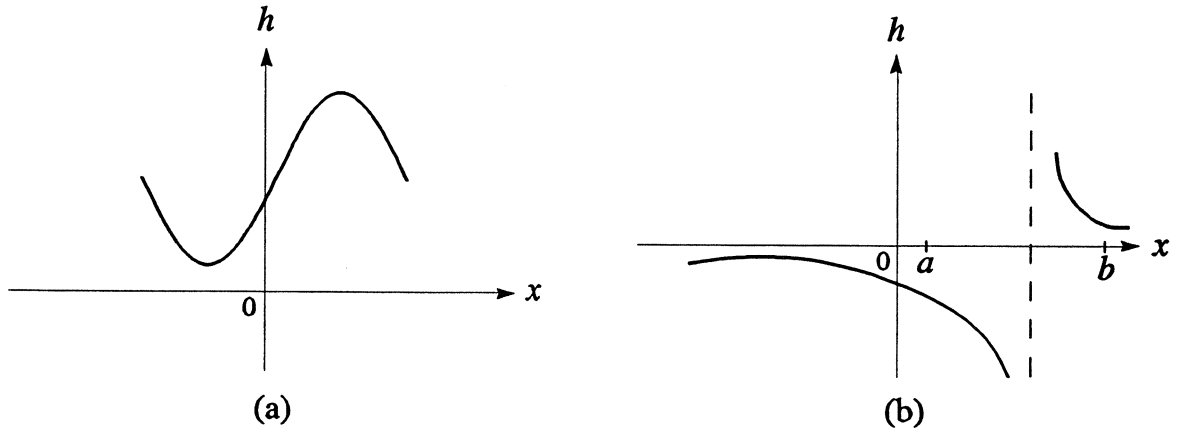

Fig. 3.5 Situations of no solutions for $h = 0$. In (a) there is no solution, in (b) sign $h(a) \neq$ sign $h(b)$ but there is no root.

3.6.1 Bracketing a Solution

If a sign $h(a) \neq$ sign $h(b)$ then we normally expect the root of h to be bracketed by a and b. In Fig. 3.6 we illustrate a simple method of obtaining brackets on a solution.

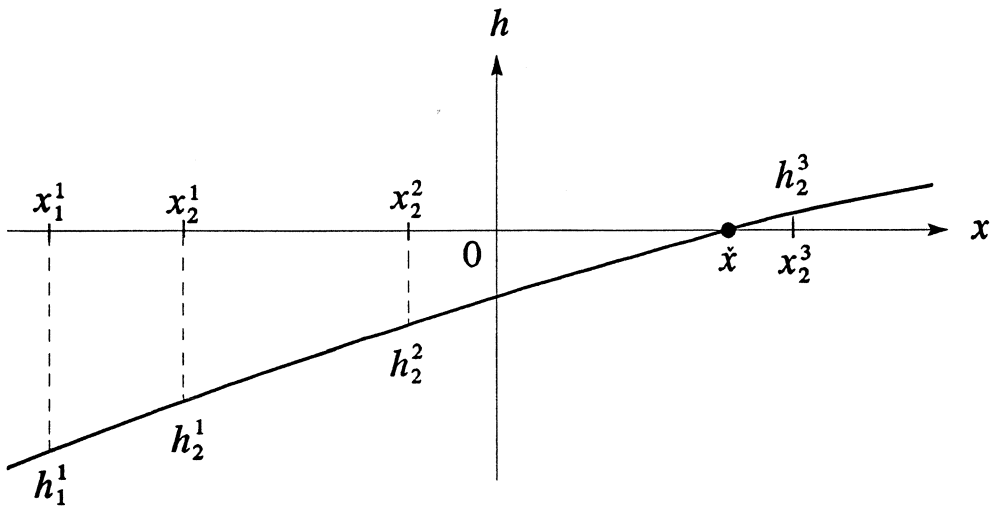

Fig. 3.6 Illustration of algorithm for bracketing a solution.

For the notation shown in Fig. 3.6 we have the following iterative process.

Iteration 1

$$\left.\begin{array}{l} h_1^1 \times h_2^1 \lessgtr 0 \\ |h_2^1| < |h_1^1| \end{array}\right\} \Rightarrow x_2^2 = x_2^1 + \text{factor} \times (x_2^1 - x_1^1) \tag{3.113}$$

$$\{x_2^1,\ h_2^1\} \Rightarrow \{x_1^2,\ h_1^2\}$$

Note that superscripts identify quantities associated with the iteration number, whereas subscripts denote quantities used within each iteration.

Iteration 2

$$\left.\begin{array}{l} h_1^2 \times h_2^2 \neq 0 \\ |h_2^2| < |h_1^2| \end{array}\right\} \Rightarrow x_2^3 = x_2^2 + \text{factor} \times (x_2^2 - x_1^2) \tag{3.114}$$

$$\{x_2^2, h_2^2\} \Rightarrow \{x_1^3, h_1^3\}$$

$$h_1^3 \times h_2^3 < 0 \Rightarrow a = x_1^3, \quad b = x_2^3$$

3.6.2 Bisection Method

Consider the illustration of Fig. 3.7. We already have the solution bracketed by x_1^1 and x_2^1

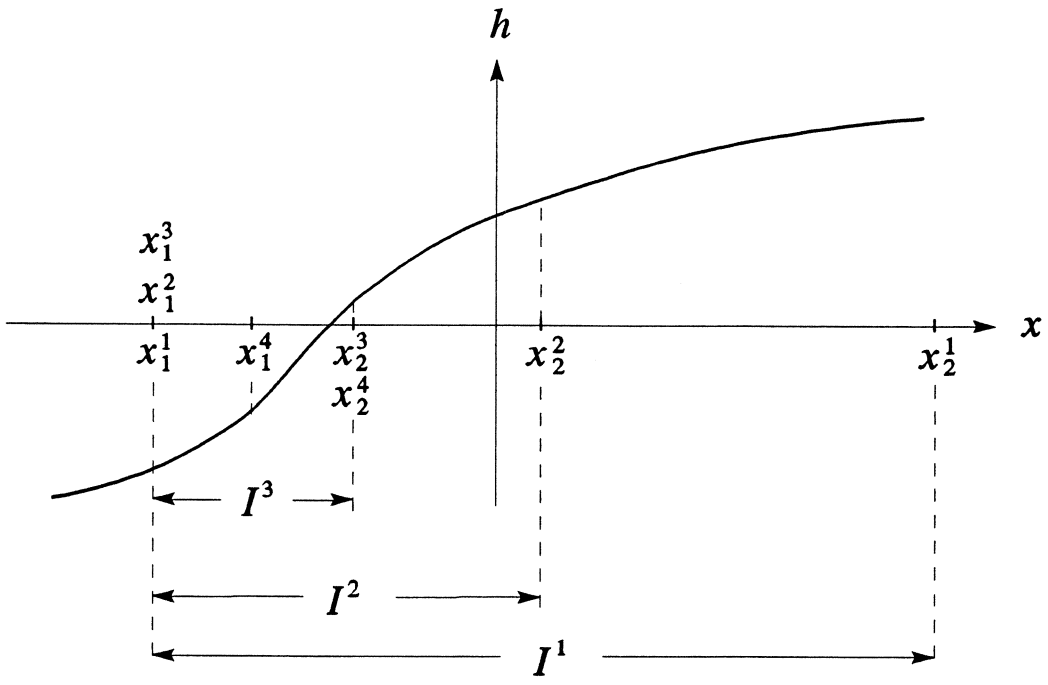

Fig. 3.7 Illustration of the bisection method.

Iteration 1

$$I^1 = x_2^1 - x_1^1$$

We accumulate 2 + 0 function evaluations.

Iteration 2

$$\frac{I^1}{2} = I^2 = x_2^2 - x_1^2$$

We accumulate 2 + 1 function evaluations.

Iteration 3

$$\frac{I^1}{2^2} = I^3 = x_2^3 - x_1^3$$

We accumulate 2 + 2 function evaluations.

Iteration 4

$$\frac{I^1}{2^3} = I^4 = x_2^4 - x_1^4$$

We accumulate 2 + 3 function evaluations.

After n Iterations

The reduction ratio after n iterations (n + 2 function evaluations) is

$$\frac{I^1}{I^{n+1}} = 2^n \tag{3.115}$$

e.g., after 3 iterations we have $I^1 = 2^3 I^4$.

Comment

We need one additional function evaluation per iteration. Iteration 1 starts with 2 function evaluations, ends with 3, etc.

Accuracy

For a desired interval σ we need n iterations such that

$$2^{n-1} < \frac{x_2^1 - x_1^1}{\sigma} \leq 2^n \tag{3.116}$$

Note on Convergence

$$I^{j+1} = \frac{1}{2} I^j = \text{constant} \times (I^j)^0 \tag{3.117}$$

where I^j is the interval at start of the jth iteration, I^{j+1} is the interval at end of the jth iteration and power 0 indicates the convergence rate, in this case linear. If

$$I^{j+1} = \text{constant} \times (I^j)^m, \quad m > 1 \tag{3.118}$$

convergence is said to be *superlinear.*

3.6.3 Newton-Raphson Method

Taylor Series

Expand h as a Taylor series, i.e.,

$$h(x + \delta) = h(x) + h'(x)\delta + \frac{h''(x)}{2}\delta^2 + \ldots \tag{3.119}$$

where

$$\begin{aligned} h' &\triangleq \frac{dh}{dx} \\ h'' &\triangleq \frac{d^2h}{dx^2} \end{aligned} \tag{3.120}$$

Linearization

If terms beyond linear are considered unimportant (linearization of h)

$$h^{j+1} = h^j + (h')^j \delta^j + \ldots \tag{3.121}$$

describes the jth iteration, where

$$\delta^j \triangleq x^{j+1} - x^j \tag{3.122}$$

and δ^j is the solution to

$$(h')^j \delta^j = -h^j \tag{3.123}$$

which results from setting $h^{j+1} = 0$ in (3.121) (the desired solution point).

The Newton Iteration

Then

$$\delta^j = -[(h')^j]^{-1} h^j = -\frac{h^j}{(h')^j} = \left[\left(\frac{dh}{dx}\right)^j\right]^{-1} h^j \tag{3.124}$$

or

$$x^{j+1} = x^j - \left[\left(\frac{dh}{dx}\right)^j\right]^{-1} h^j \tag{3.125}$$

Example

Illustrations of the Newton-Raphson method are given in Fig. 3.8.

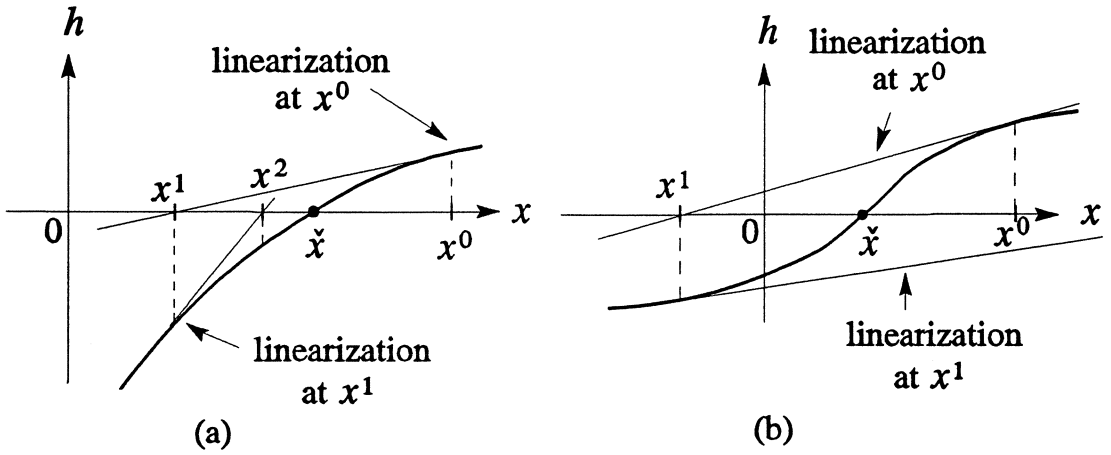

Fig. 3.8 Illustrations of the Newton-Raphson method, (a) process with convergence, and (b) process with no convergence.

$$\begin{aligned} x^1 &= x^0 - \frac{h^0}{(h')^0} \\ x^2 &= x^1 - \frac{h^1}{(h')^1} \\ x^3 &= \ldots \end{aligned} \tag{3.126}$$

Convergence of the Newton Method

Consider the solution $x = \check{x}$. Then, for small ϵ we expand h as

$$h(\check{x} + \epsilon) = h(\check{x}) + \epsilon h'(\check{x}) + \epsilon^2 \frac{h''(\check{x})}{2} + \ldots \tag{3.127}$$

and h' as

$$h'(\check{x} + \epsilon) = h'(\check{x}) + \epsilon h''(\check{x}) + \ldots \tag{3.128}$$

Now, from (3.125),

$$x^{j+1} - \hat{x} = x^j - \hat{x} - \frac{h^j}{(h')^j} \tag{3.129}$$

so that

$$\epsilon^{j+1} = \epsilon^j - \frac{h^j}{(h')^j} \tag{3.130}$$

Also

$$h(\hat{x}) = 0 \tag{3.131}$$

so that, from (3.127),

$$h^j = 0 + \epsilon^j (h')^j + \frac{(\epsilon^j)^2 (h'')^j}{2} + \ldots \tag{3.132}$$

giving

$$\frac{h^j}{(h')^j} = \epsilon^j + \frac{(\epsilon^j)^2 (h'')^j}{2(h')^j} + \ldots \tag{3.133}$$

or

$$\epsilon^j - \frac{h^j}{(h')^j} = \frac{(\epsilon^j)^2 (h'')^j}{2(h')^j} + \ldots \tag{3.134}$$

Consequently, using (3.130),

$$\epsilon^{j+1} = \frac{(\epsilon^j)^2 (h'')^j}{2(h')^j} + \ldots \tag{3.135}$$

Note that the distance to the solution on the $(j+1)$th iteration = constant $\times$ the square of the distance at the jth iteration. This implies quadratic convergence, i.e., the number of significant digits of accuracy doubles on each iteration near the solution. In the foregoing discussion we assume that

$$\frac{(h'')^j}{(h')^j} \rightarrow \frac{h''(\hat{x})}{h'(\hat{x})} \tag{3.136}$$

3.6.4 Newton Method with Approximate Derivative

Derivative Approximation

The exact Newton method requires both h and h' at x. Consider the derivative approximation

$$h'(x) \approx \frac{h(x + \delta x) - h(x)}{\delta x} \tag{3.137}$$

The following are drawbacks when using approximate derivatives. Two function evaluations are required per iteration: convergence is superlinear of order $\sqrt{2}$. Too small a value of δx is subject to roundoff, too large δx makes convergence linear.

Modified Newton Method

To counteract the aforementioned drawbacks we can devise the modified approach

$$x^{j+1} = x^j - [(h')^0]^{-1} h^j \tag{3.138}$$

where $(h')^0$ is kept constant for several iterations. An illustration of the modified Newton method is given in Fig. 3.9.

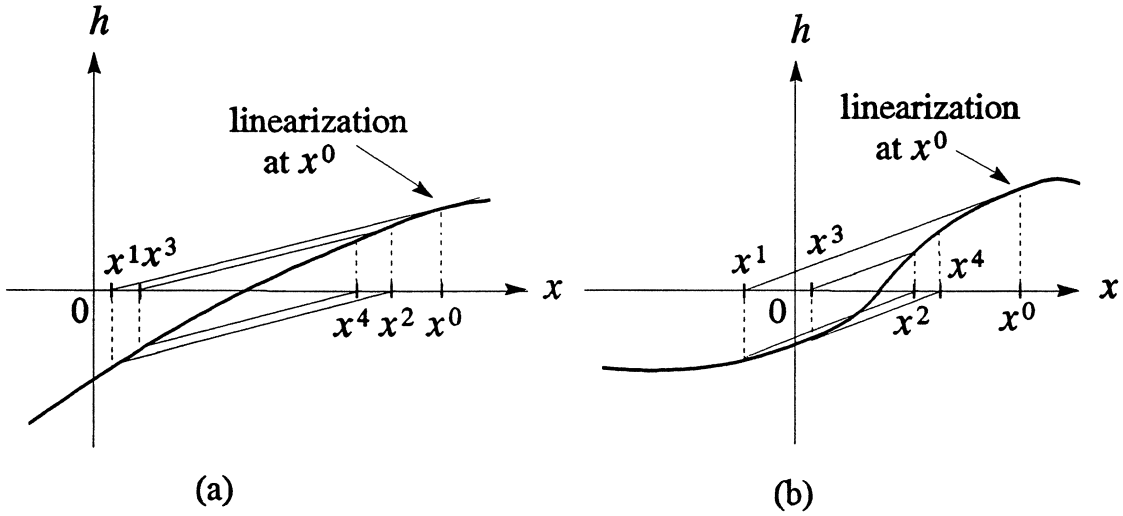

Fig. 3.9 Illustration of the modified Newton method with all linear approximations using $(h')^0$. (a) convergence, and (b) forced convergence (see Fig. 3.3(b)).

3.7 The State Equations [1]

3.7.1 Basic Concepts [1]

The Problem

We are given the equations of motion for a nonlinear dynamic system in the normal form

$$\dot{x} = f(x, t) \tag{3.139}$$

where

$$x \triangleq \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad \dot{x} \triangleq \frac{d}{dt} x \tag{3.140}$$

This system of first-order differential equations is to be solved numerically using difference methods, given the initial state vector

$$x(t_0) = x_0 \tag{3.141}$$

for $t \geq t_0$ over the time interval T.

Definitions

The function $\bar{x}(t)$ is a solution to this initial value problem if

$$\begin{aligned} \bar{x}(t_0) &= x_0 \\ \dot{\bar{x}}(t) &= f(\bar{x}(t), t), \text{ for all } t \in \{t_0, t_0 + T\} \end{aligned} \tag{3.142}$$

The solution time T is divided for computational purposes into increments

$$h_i \triangleq (\Delta t)_i \tag{3.143}$$

called the step size. We require

$$\bar{x}(t) \text{ at } t_k \triangleq t_0 + \sum_{i=1}^{k} h_i, \; k = 1, 2, \ldots, N \tag{3.144}$$

where

$$t_N = t_0 + T \tag{3.145}$$

If $\bar{x}(t_k)$ is the *exact* value and $\bar{x}_k$ the *computed* value, then

$$\|\bar{x}(t_k) - \bar{x}_k\| \tag{3.146}$$

is the total error at $t = t_k$. This will depend on *truncation* error and *roundoff* error. The truncation error depends on the algorithm used and the roundoff error on the machine.

Local Error

The local error after one time step assuming x is exact at the previous time step is

$$\|\bar{x}(t_1) - \bar{x}_1\| \tag{3.147}$$

Local Truncation Error

The local error as above but due to the algorithm only is

$$\|\bar{x}(t_1) - x_1\| \tag{3.148}$$

Note that $\bar{x}_k$ includes roundoff and truncation errors, whereas x_k has no roundoff error.

Numerical Stability

An algorithm whose local roundoff error decays with an increasing number of time steps is called *numerically stable*.

Taylor Expansion Approach

Such algorithms are usually called Runge-Kutta algorithms.

Polynomial Approximation Approach

Such algorithms are usually called numerical integration algorithms.

Uniform Step Size

We will assume the uniform step

$$h = h_i \tag{3.149}$$

hence

$$t_n = t_0 + nh, \; n = 1, 2, \ldots, N \tag{3.150}$$

3.7.2 Taylor Algorithms

Since all formulas are easily extended to many equations in many state variables, we consider only one equation in one state variable.

Let $\bar{x}(t)$ be the exact solution. Then,

$$\bar{x}(t_{n+1}) = \bar{x}(t_n) + \frac{\bar{x}^{(1)}(t_n)}{1!}(t_{n+1} - t_n) + \frac{\bar{x}^{(2)}(t_n)}{2!}(t_{n+1} - t_n)^2 + \ldots + \frac{\bar{x}^{(p)}(t_n)}{p!}(t_{n+1} - t_n)^p + \ldots \tag{3.151}$$

where

$$\bar{x}^{(j)} \triangleq \frac{d^j \bar{x}}{dt^j} \tag{3.152}$$

Let

$$t_{n+1} - t_n = h \tag{3.153}$$

then

$$\bar{x}(t_{n+1}) = \bar{x}(t_n) + \frac{h}{1!}\bar{x}^{(1)}(t_n) + \frac{h^2}{2!}\bar{x}^{(2)}(t_n) + \frac{h^3}{3!}\bar{x}^{(3)}(t_n) + \ldots + \frac{h^p}{p!}\bar{x}^{(p)}(t_n) + \ldots \tag{3.154}$$

Hence

$$\bar{x}(t_{n+1}) \approx \bar{x}(t_n) + \frac{h}{1!}f(\bar{x}(t_n), t_n) + \frac{h^2}{2!}f^{(1)}(\bar{x}(t_n), t_n) + \ldots + \frac{h^p}{p!}f^{(p-1)}(\bar{x}(t_n), t_n) \tag{3.155}$$

To develop the algorithm, write

$$x_{n+1} = x_n + h\,T_p \tag{3.156}$$

where

$$T_p \triangleq T_p(x_n, t_n; h) \triangleq f(x_n, t_n) + \frac{h}{2!}f^{(1)}(x_n, t_n) + \ldots + \frac{h^{p-1}}{p!}f^{(p-1)}(x_n, t_n) \tag{3.157}$$

Taylor: Order 1: Forward Euler

$p = 1$, hence the Taylor algorithm reduces to

$$x_{n+1} = x_n + h\, f(x_n,\ t_n) \tag{3.158}$$

The exact solution is $\bar{x}(t_{n+1})$ based on the initial condition x_n at t_n. The approximate solution is

$$x_{n+1} = x_n + h \left.\frac{d\bar{x}}{dt}\right|_{t\,=\,t_n} \tag{3.159}$$

The local truncation error is $x_{n+1} - \bar{x}(t_{n+1})$. This method is seldom used, as the truncation error is too large.

Taylor: Order 2

$p = 2$, hence

$$\begin{aligned} x_{n+1} &= x_n + h\left[f(x_n,\ t_n) + \frac{h}{2} f^{(1)}\ (x_n,\ t_n)\right] \\ &= x_n + h\left[f(x_n,\ t_n) + \frac{h}{2}\left[\frac{\partial f}{\partial x}\frac{dx}{dt} + \frac{\partial f}{\partial t}\right]_{(x_n,\ t_n)}\right] \\ &= x_n + h\left[f + \frac{h}{2}\left[\frac{\partial f}{\partial x} f + \frac{\partial f}{\partial t}\right]\right]_{(x_n,\ t_n)} \end{aligned} \tag{3.160}$$

Taylor: Order 3

$p = 3$, hence

$$\begin{aligned} x_{n+1} &= x_n + h\left[f + \frac{h}{2}\left[\frac{\partial f}{\partial x} f + \frac{\partial f}{\partial t}\right] + \frac{h^2}{3!}\left[\frac{d}{dt}\left(\frac{\partial f}{\partial x} f + \frac{\partial f}{\partial t}\right)\right]\right] \\ &= x_n + \ldots \end{aligned} \tag{3.161}$$

This formula is inconvenient, error-prone and requires explicit second-order partial derivatives.

3.7.3 Other Algorithms

Runge-Kutta Algorithm

The term

$$T_p \triangleq T_p\ (x_n,\ t_n;\ \ h) \tag{3.162}$$

which requires partial derivative evaluation, is replaced by

$$K_p \triangleq K_p\ (x_n,\ t_n;\ \ h) \tag{3.163}$$

such that

$$|K_p - T_p| \leq R\, h^p \tag{3.164}$$

where R is a constant. The truncation error in the Taylor algorithm is of order h^p, hence the Runge-Kutta algorithm has the same order of magnitude of truncation error.

Runge-Kutta: Order 2

$$x_{n+1} = x_n + h\, K_2 \tag{3.165}$$

where, for $\alpha_2 \neq 0$

$$K_2 = (1 - \alpha_2)\, f(x_n, t_n) + \alpha_2 f\left[x_n + \frac{h}{2\alpha_2} f(x_n, t_n),\ t_n + \frac{h}{2\alpha_2}\right] \tag{3.166}$$

Heun's Algorithm (Modified Trapezoidal)

For $\alpha_2 = 0.5$

$$x_{n+1} = x_n + \frac{h}{2}\left[f(x_n, t_n) + f\left[x_n + h\, f(x_n, t_n),\ t_n + h\right]\right] \tag{3.167}$$

Modified Euler-Cauchy Algorithm

For $\alpha_2 = 1$

$$x_{n+1} = x_n + h\, f\left[x_n + \frac{h}{2} f(x_n, t_n),\ t_n + \frac{h}{2}\right] \tag{3.168}$$

Fourth-Order Runge-Kutta Algorithm

This algorithm is widely used for its accuracy and for its larger step size requirement h. Here

$$x_{n+1} = x_n + h\, K_4 \tag{3.169}$$

where

$$K_4 = \frac{1}{6}\left[k_1 + 2k_2 + 2k_3 + k_4\right]$$
$$k_1 \triangleq f(x_n, t_n) \tag{3.170}$$

$$
\begin{aligned}
k_2 &\triangleq f\left[x_n + \frac{h}{2}\, k_1,\ t_n + \frac{h}{2}\right] \\
k_3 &\triangleq f\left[x_n + \frac{h}{2}\, k_2,\ t_n + \frac{h}{2}\right] \\
k_4 &\triangleq f\left[x_n + h\, k_3,\ t_n + h\right]
\end{aligned}
\tag{3.171}
$$

The procedure takes the weighted average of four separately calculated slopes k_1, k_2, k_3 and k_4. This information is not required again, hence relatively a great amount of effort is required per iteration.

If f is independent of x, then we have the familiar Simpson's rule for integration.

3.8 References

[1] L.O. Chua and P.M. Lin, *Computer-Aided Analysis of Electronic Circuits: Algorithms and Computational Techniques*. Englewood Cliffs, NJ: Prentice-Hall, 1975.

Chapter 4

FUNDAMENTALS OF OPTIMIZATION

Following a section on basic concepts which are essential for an understanding of optimization theory, a formulation and description of typical objectives and objective functions is presented. Constraints and some methods of dealing with them are discussed in fair detail in the next section, including the conditions for a constrained minimum. Minimax approximation, including conditions for a minimax optimum, is then dealt with.

4.1 Basic Concepts

4.1.1 Problem Statement

The problem of optimization may be stated as follows. Minimize the scalar *objective function*, also called *cost function, performance index,* or *error function*, U, where

$$U \triangleq U(\boldsymbol{\phi}) \tag{4.1}$$

subject to the *inequality constraints*

$$\boldsymbol{g}(\boldsymbol{\phi}) \geq \mathbf{0} \tag{4.2}$$

and *equality constraints*

$$\boldsymbol{h}(\boldsymbol{\phi}) = \mathbf{0} \tag{4.3}$$

In (4.1) to (4.3) $\boldsymbol{\phi}$ is a vector of k independent variables or parameters. They may be element values, residues, critical frequencies, geometrical dimensions, etc. Thus

$$\boldsymbol{\phi} \triangleq \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_k \end{bmatrix} \tag{4.4}$$

defining a k-dimensional space. In general, we might have m inequality constraints and s equality constraints so that

$$g(\phi) \triangleq \begin{bmatrix} g_1(\phi) \\ g_2(\phi) \\ \vdots \\ g_m(\phi) \end{bmatrix} \tag{4.5}$$

and

$$h(\phi) \triangleq \begin{bmatrix} h_1(\phi) \\ h_2(\phi) \\ \vdots \\ h_s(\phi) \end{bmatrix} \tag{4.6}$$

The *feasible region* R is defined by all vectors ϕ satisfying (4.2) and (4.3). This may be written

$$R \triangleq \{\phi \,|\, g(\phi) \geq 0,\ h(\phi) = 0\} \tag{4.7}$$

R is said to be *closed* if, as in (4.2), equalities are allowed. If no equalities are allowed it is said to be *open*. A *proper* minimum of U located by a vector $\check{\phi}$ on the *response hypersurface* generated by $U(\phi)$ is such that

$$\check{U} \triangleq U(\check{\phi}) < U(\phi) \tag{4.8}$$

for any feasible ϕ close but not equal to $\check{\phi}$. $U(\check{\phi}) \leq U(\phi)$ can also define a minimum, but $\check{\phi}$ may then be nonunique. Since we cannot generally guarantee to find a *global minimum*, we usually have to resign ourselves to a consideration of *local minima*. Our objective then is to find a feasible $\check{\phi}$, if it indeed exists, such that

$$U(\check{\phi}) = \min_{\phi \in R} U(\phi) \tag{4.9}$$

Fig. 4.1 is an illustration of the problem in two dimensions, and it contains a number of features usually encountered in optimization problems. Note that only inequality constraints, i.e., constraints of the form of (4.2), are indicated.

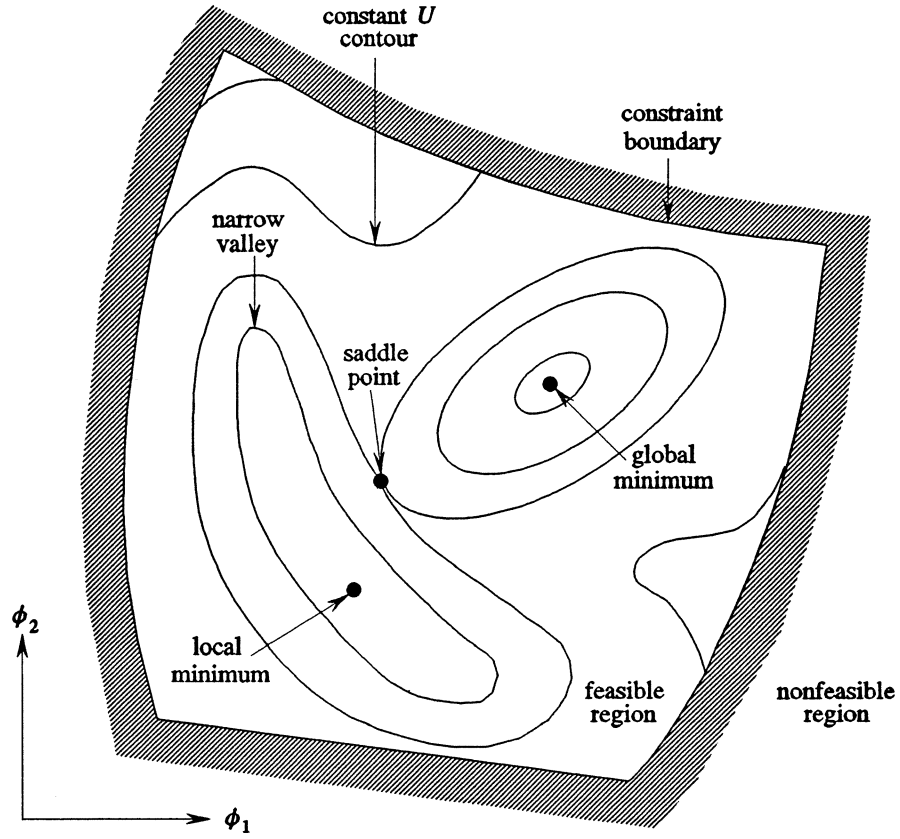

Fig. 4.1 Some features encountered in optimization problems.

4.1.2 Convexity

Examples of *unimodal* and *multimodal* functions of one variable are shown in Fig. 4.2. *Strictly convex* and *strictly concave* functions of one variable are shown in Fig 4.3. A unimodal function for our purposes is one having a unique optimum in the feasible region. It may or may not be continuous with continuous derivatives.

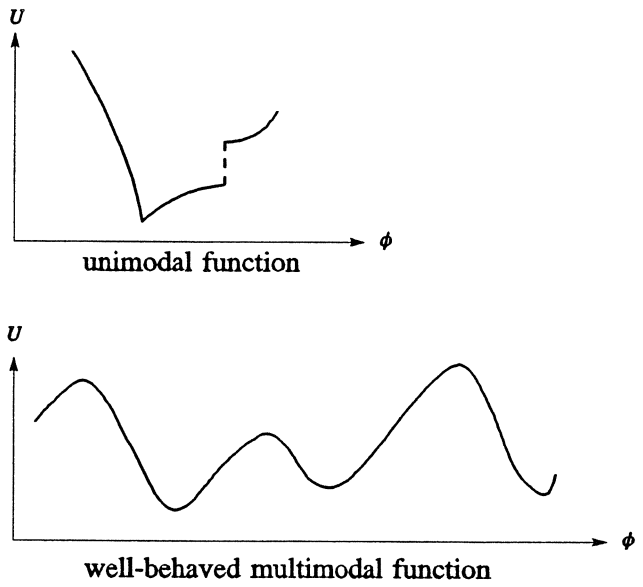

Fig. 4.2 Unimodal function and well-behaved multimodal function of one variable.

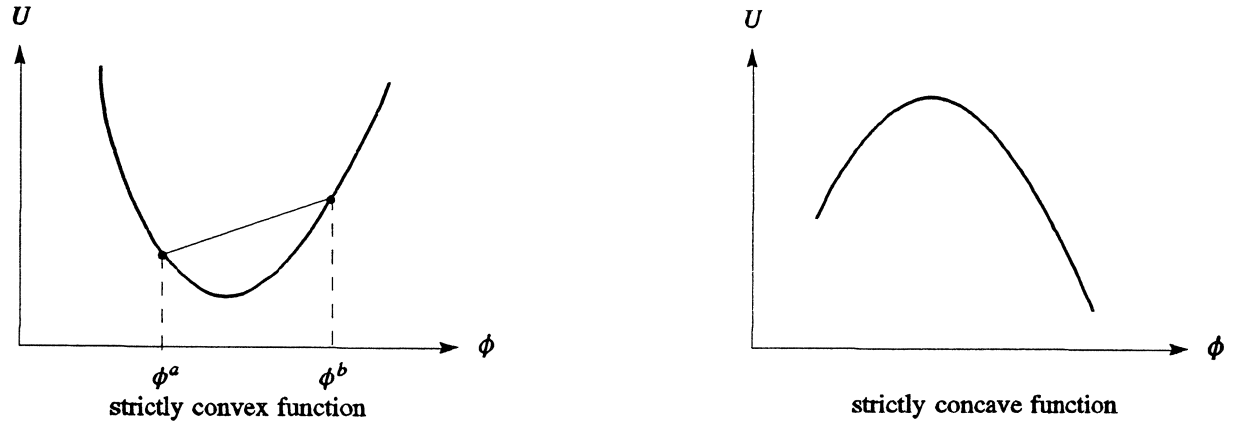

Fig. 4.3 Strictly convex and strictly concave functions of one variable.

A *strictly convex* function is one which can only be overestimated by a linear interpolation between two points on its surface. Thus, for $\phi^a \neq \phi^b$,

$$U(\phi^a + \lambda(\phi^b - \phi^a)) < U(\phi^a) + \lambda(U(\phi^b) - U(\phi^a)) \qquad 0 < \lambda < 1 \tag{4.10}$$

for a strictly convex function. See Fig. 4.4. A *strictly concave* function is one whose negative is strictly convex. Note that if we omit strictly, then we imply that equality of the function and a linear interpolation can occur, i.e., (4.10) would have to admit equalities.

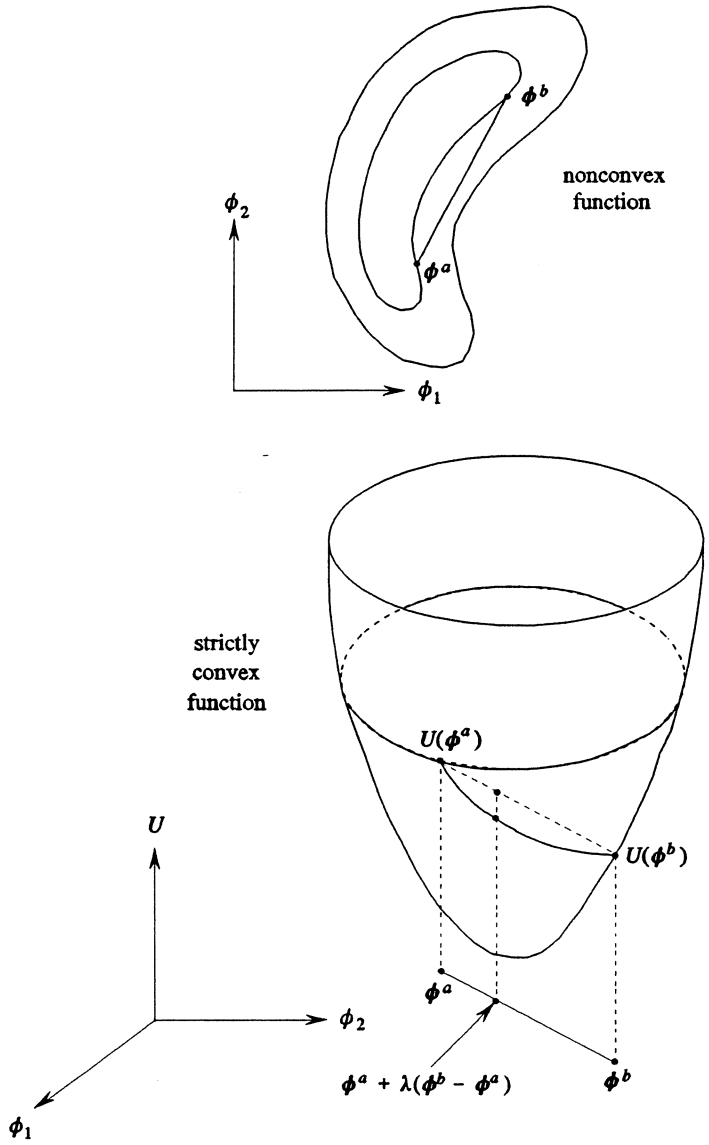

Fig. 4.4 Illustration of convexity.

A region R is convex if for all ϕ^a, $\phi^b \in R$ all points

$$\phi = \phi^a + \lambda(\phi^b - \phi^a)$$
$$0 \leq \lambda \leq 1 \tag{4.11}$$

lie in R. Illustrations of convex and nonconvex regions are given in Fig. 4.5.

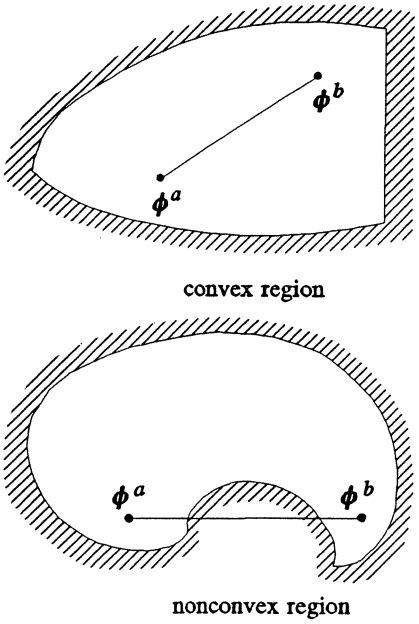

Fig. 4.5 Convex and nonconvex regions.

4.1.3 Some Gradient Concepts

Gradient Vector

$$\lim_{\lambda \to 0^+} \frac{U(\phi + \lambda s) - U(\phi)}{\lambda} = (\nabla U)^T s \tag{4.12}$$

where

$$\nabla U \triangleq \begin{bmatrix} \frac{\partial U}{\partial \phi_1} \\ \frac{\partial U}{\partial \phi_2} \\ \vdots \\ \frac{\partial U}{\partial \phi_k} \end{bmatrix} \tag{4.13}$$

is a vector containing the first partial derivatives of the objective function called the *gradient vector*.

Consequence

Suppose

$$(\nabla U)^T s > 0 \tag{4.14}$$

then there exists $\sigma > 0$ such that for all λ, $\sigma \geq \lambda > 0$

$$U(\phi + \lambda s) > U(\phi) \tag{4.15}$$

Mean Value Theorem

If U is differentiable in the open interval (a, b) and continuous at a and b, then there is a number c with $a < c < b$ such that

$$\frac{U(b) - U(a)}{b - a} = U'(c) \tag{4.16}$$

Therefore,

$$U(b) = U(a) + U'(c)(b - a) \tag{4.17}$$

This is often referred to as the Extended Mean Value Theorem (Taylor's Formula).

4.1.4 Taylor Series

The first three terms of a multidimensional Taylor series expansion of $U(\phi)$ are given by

$$U(\phi + \Delta\phi) = U(\phi) + \nabla U^T \Delta\phi + \frac{1}{2} \Delta\phi^T H \Delta\phi + \ldots \tag{4.18}$$

where the vector

$$\Delta\phi \triangleq \begin{bmatrix} \Delta\phi_1 \\ \Delta\phi_2 \\ \vdots \\ \Delta\phi_k \end{bmatrix} \tag{4.19}$$

contains k parameter increments, $\Delta\phi^T$ is the transposed (row) vector, ∇U is defined by (4.13) and

$$H \triangleq \begin{bmatrix} \dfrac{\partial^2 U}{\partial \phi_1^2} & \dfrac{\partial^2 U}{\partial \phi_1 \, \partial \phi_2} & \cdots & \dfrac{\partial^2 U}{\partial \phi_1 \, \partial \phi_k} \\ \dfrac{\partial^2 U}{\partial \phi_2 \, \partial \phi_1} & \dfrac{\partial^2 U}{\partial \phi_2^2} & \cdots & \dfrac{\partial^2 U}{\partial \phi_2 \, \partial \phi_k} \\ \vdots & \vdots & & \vdots \\ \dfrac{\partial^2 U}{\partial \phi_k \, \partial \phi_1} & \dfrac{\partial^2 U}{\partial \phi_k \, \partial \phi_2} & \cdots & \dfrac{\partial^2 U}{\partial \phi_k^2} \end{bmatrix} \tag{4.20}$$

is a symmetric $k \times k$ matrix containing the second partial derivatives and called the *Hessian* matrix. At a minimum of a continuous function with continuous first and second partial derivatives $\nabla U(\check{\phi}) = \mathbf{0}$ and $H(\check{\phi})$ is positive semidefinite. (It should be noted that H might not be positive definite, even in some cases when $U(\phi)$ is strictly convex.) Invoking these conditions in (4.18) but with H taken as positive definite, it may be shown that (4.8) is satisfied, implying that we have a proper minimum. $U(\phi)$ is strictly convex in a region where H is positive definite as may be seen by relating (4.10) with (4.18).

4.1.5 Remarks

The problem formulated in (4.1) to (4.3) is called a *mathematical programming* problem. If all the functions are linear, we have *linear programming*; if not, we have *nonlinear programming*. The term *convex programming* is often used to describe the problem defined by (4.1) and (4.2) when $U(\phi)$ is convex and $g(\phi)$ is concave. Under these conditions R is convex, and $\check{U}$ is the global minimum.

In practical situations, it is usually out of the question to determine whether a specific problem falls into the domain of convex programming. Nevertheless, it seems a fair generalization to make, that the most reliable and efficient methods of optimization for practical problems are invariably those which invoke some of the nice properties of convex programming in their proofs of convergence. The better methods usually have built-in safeguards for dealing with the hazards of more general nonlinear programming problems while substantially retaining their desirable convergence features. Note that essentially unconstrained problems are regarded as special cases in this discussion.

4.2 Some Objectives and Objective Functions

4.2.1 Optimization by Solving Nonlinear Equations

Classically, to find $\check{U}$ we must in general solve k nonlinear equations in k unknowns, namely

$$\nabla U = \mathbf{0} \tag{4.21}$$

Denoting this set of equations $f(\phi) = \mathbf{0}$ where

$$f(\phi) \triangleq \begin{bmatrix} f_1(\phi) \\ f_2(\phi) \\ \vdots \\ f_k(\phi) \end{bmatrix} \tag{4.22}$$

we could define a new objective function

$$U(\phi) = f^T f \tag{4.23}$$

to be minimized. A minimum of value zero would imply that the solution to $f(\phi) = 0$ had been found. Now, using a Taylor series expansion

$$f(\phi + \Delta\phi) = f(\phi) + J\,\Delta\phi + \ldots \tag{4.24}$$

where

$$J \triangleq \begin{bmatrix} \frac{\partial f_1}{\partial \phi_1} & \frac{\partial f_1}{\partial \phi_2} & \cdots & \frac{\partial f_1}{\partial \phi_k} \\ \frac{\partial f_2}{\partial \phi_1} & \frac{\partial f_2}{\partial \phi_2} & \cdots & \frac{\partial f_2}{\partial \phi_k} \\ \vdots & \vdots & & \vdots \\ \frac{\partial f_k}{\partial \phi_1} & \frac{\partial f_k}{\partial \phi_2} & \cdots & \frac{\partial f_k}{\partial \phi_k} \end{bmatrix} \tag{4.25}$$

is a $k \times k$ *Jacobian* matrix. The well-known *Newton-Raphson* method of solution is based on the hope that, if we evaluate f and J at ϕ, then the incremental change

$$\Delta\phi = -J^{-1} f(\phi) \tag{4.26}$$

brings one closer to the solution.

4.2.2 Quadratic Objective Function Model

Consider a quadratic objective function

$$U(\phi) = \frac{1}{2}\,\phi^T A \phi + b^T \phi + c \tag{4.27}$$

where A is a $k \times k$ constant symmetric matrix, b is a constant vector with k components and c is a constant. In this case, it is readily shown that

$$\nabla U = A\phi + b \tag{4.28}$$

$$H = A \tag{4.29}$$

A stationary point of $U(\phi)$ can be found by solving the linear equations

$$A\phi + b = 0 \tag{4.30}$$

If A is nonsingular, the point is unique and can be found in a finite number of operations. The term *quadratic convergence* (Fletcher [1] prefers the term "property Q") is used to describe the convergence properties of optimization methods, which guarantee to find the minimum of a quadratic function in a finite number of steps. Such methods can be expected to be very efficient in minimizing functions adequately representable by positive-definite quadratic forms in the vicinity of a minimum. Some of them are discussed in later sections.

4.2.3 Error Criteria

Most electrical network design problems can be formulated as approximation problems. Let us, therefore, introduce a weighted error or deviation between a specified function and an approximating function as

$$e(\phi, \psi) \triangleq w(\psi)[F(\phi, \psi) - S(\psi)] \tag{4.31}$$

where

$S(\psi)$ is the real or complex, specified function,
$F(\phi, \psi)$ is the real or complex approximating function,
$w(\psi)$ is a weighting function,
ψ is an independent variable,
ϕ represents the adjustable parameters.

Thus $F(\phi, \psi)$ may be a network response, $S(\psi)$ may be the desired response, and ψ may be frequency, time, temperature, etc. See Fig. 4.6.

We may define a norm

$$\|e\|_p \triangleq \left[\int_{\psi_l}^{\psi_u} |e(\phi, \psi)|^p \, d\psi\right]^{1/p}, \quad 1 \leq p \leq \infty \tag{4.32}$$

for the continuous case; and a norm

$$\|e\|_p \triangleq \left[\sum_{i \in I} |e_i(\phi)|^p\right]^{1/p}, \quad 1 \leq p \leq \infty \tag{4.33}$$

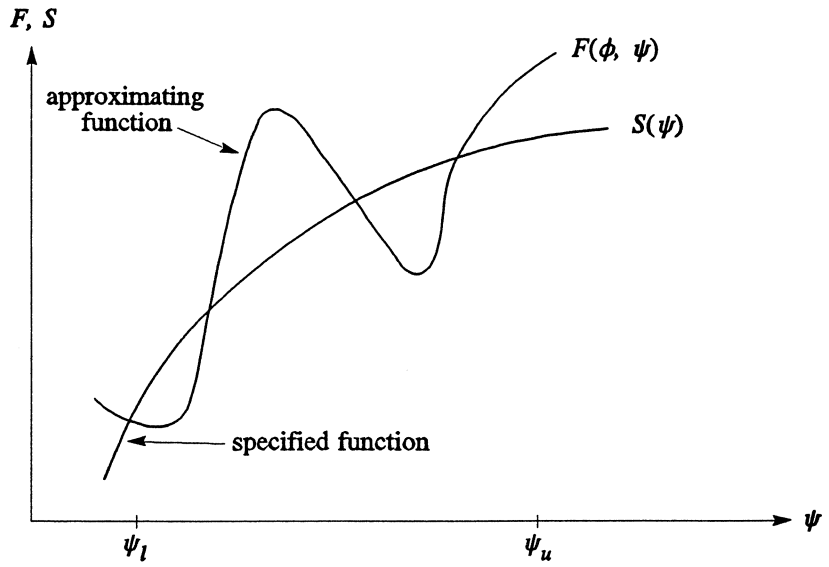

Fig. 4.6 An approximation problem.

for the discrete case, where

$$e(\phi) \triangleq \begin{bmatrix} e_1(\phi) \\ e_2(\phi) \\ \vdots \\ e_n(\phi) \end{bmatrix} \tag{4.34}$$

$$e_i(\phi) \triangleq e(\phi, \psi_i) = w(\psi_i)[F(\phi, \psi_i) - S(\psi_i)] \tag{4.35}$$

and

$$I \triangleq \{1, 2, \ldots, n\} \tag{4.36}$$

Thus, I is an index set relating to discrete values of ψ on an interval $[\psi_l, \psi_u]$, which is closed and finite.

Now for well-behaved functions

$$\max_{[\psi_l, \psi_u]} |e(\phi, \psi)| = \lim_{p \to \infty} \left[\frac{1}{\psi_u - \psi_l} \int_{\psi_l}^{\psi_u} |e(\phi, \psi)|^p \, d\psi \right]^{1/p} \tag{4.37}$$

when $|e(\phi, \psi)|$ is defined on $[\psi_l, \psi_u]$. If $|e(\phi, \psi)|$ is continuous on a finite interval $[\psi_l, \psi_u]$, then (4.37) is certainly valid. Similarly,

$$\max_i |e_i(\phi)| = \lim_{p\to\infty} \left[\sum_i |e_i(\phi)|^p\right]^{1/p}, \quad i\in I \tag{4.38}$$

Suppose we formulate an objective function as

$$U = \int_{\psi_l}^{\psi_u} |e(\phi, \psi)|^p \, d\psi \tag{4.39}$$

for the continuous case and

$$U = \sum_{i\in I} |e_i(\phi)|^p \tag{4.40}$$

for the discrete case. The minimization of the U of (4.39) or (4.40) is called *least pth approximation.* A minimum for the continuous case is called a best approximation with respect to $\|e\|_p$, defined in (4.32). A minimum for the discrete case is called a best approximation with respect to $\|e\|_p$, defined in (4.33). Now $\|e\|_\infty$ and $\|e\|_\infty$ are called *Chebyshev* or *uniform* norms. Because of the consequences of (4.37) and (4.38), minimization with respect to $\|e\|_\infty$ or $\|e\|_\infty$ is widely referred to as *minimax approximation.* Least pth approximation is supposed to tend to minimax approximation as $p \to \infty$.

A word of caution concerning the weighting function $w(\psi)$ and the index p is in order. Clearly their purpose is to emphasize or deemphasize the difference between $F(\phi, \psi)$ and $S(\psi)$. Thus, an optimum with respect to one weighting function or value of p may not be an optimum with respect to another. Large errors will be emphasized by large values of p. If one knew in advance where these large errors would be, $w(\psi)$ might also be used to emphasize them. The use of $w(\psi)$ to do this is a poor approach, however, and should be discouraged.

4.3 Constraints

It is rare to find an engineering design problem which is unconstrained. When physical considerations indicate that the optimum will lie in the interior of the feasible region, the designer is lucky and should take advantage of it. Often this will not be possible, and steps have to be taken to ensure that a realizable and practical design will be achieved. One of the great advantages of computer-aided circuit optimization is that, if the design problem has been properly formulated, a feasible design can always be achieved assuming the initial design is feasible.

Constraints in network design can take a variety of forms. They include upper and lower bounds on parameters; they can include nonnegativity requirements on network elements. The topology, overall size, the suppression of unwanted modes of operation, considerations for parasitic effects whether reactive or lossy, and the stability of active devices can all result in constraints on parameters. Response constraints such as constraints on the phase while the amplitude is optimized can also occur.

Designers often treat constraints as an afterthought, and then complain that the optimization process gave them nonfeasible results. The author would like to stress that a thorough consideration should be given to constraints *before* the selection of an optimization strategy.

In this section we will look at some methods of converting constrained problems into essentially unconstrained ones. For other methods of nonlinear programming, the reader should refer to appropriate texts.

4.3.1 Parameter Transformations

Various upper and lower bounds on the variable parameters are probably the most common kinds of constraints [2]. In Table 4.1 we show some simple parameter constraints falling into this class with appropriate transformations. It is useful to distinguish between constraints defining open and closed feasible regions. If the optimum is expected to lie away from the boundary or if it is desired to discourage the solution from getting too close, the former type might be chosen.

TABLE 4.1 SIMPLE PARAMETER CONSTRAINTS AND TRANSFORMATIONS

Constraint	Transformation
$\phi_i \geq 0$	$\phi_i = \phi_i'^2$
$\phi_i > 0$	$\phi_i = \exp \phi_i'$
$\phi_i \geq \phi_{li}$	$\phi_i = \phi_{li} + \phi_i'^2$
$\phi_i > \phi_{li}$	$\phi_i = \phi_{li} + \exp \phi_i'$
$-1 \leq \phi_i \leq 1$	$\phi_i = \sin \phi_i'$
$0 \leq \phi_i \leq 1$	$\phi_i = \sin^2 \phi_i'$
$0 < \phi_i < 1$	$\phi_i = \dfrac{\exp \phi_i'}{1 + \exp \phi_i'}$
$\phi_{li} \leq \phi_i \leq \phi_{ui}$	$\phi_i = \phi_{li} + (\phi_{ui} - \phi_{li})\sin^2 \phi_i'$ $\phi_i = \frac{1}{2}(\phi_{li} + \phi_{ui}) + \frac{1}{2}(\phi_{ui} - \phi_{li})\sin \phi_i'$
$\phi_{li} < \phi_i < \phi_{ui}$	$\phi_i = \phi_{li} + (\phi_{ui} - \phi_{li}) \dfrac{\exp \phi_i'}{1 + \exp \phi_i'}$ $\phi_i = \phi_{li} + \frac{1}{\pi}(\phi_{ui} - \phi_{li})\cot^{-1} \phi_i'$ for $0 < \cot^{-1} \phi_i' < \pi$

4.3.2 More General Considerations

Parameter constraints of the form

$$\phi_{li} \leq \phi_i \leq \phi_{ui} \tag{4.41}$$

can if necessary be written as

$$\begin{aligned} \phi_i - \phi_{li} &\geq 0 \\ \phi_{ui} - \phi_i &\geq 0 \end{aligned} \tag{4.42}$$

in order to fit them into the scheme of (4.2). Frequency- or time-dependent constraints may be put into the form

$$c_j(\phi, \psi) \geq 0 \tag{4.43}$$

where j denotes some jth function, or at discrete points on the ψ-axis into the form

$$c_j(\phi, \psi_i) \geq 0 \tag{4.44}$$

where i denotes an ith sample point. The form of (4.44) is preferable to that of (4.43), since it allows us to consider a finite rather than an infinite number of constraints.

Elimination of Constraints

We might eliminate a number of constraints on physical or logical grounds. Consider the following situations.

1. $U(\phi) \to \infty$ as $g_i(\phi) \to 0$. The attenuation of a filter becomes infinite, for example, if a zero valued element short circuits the structure.
2. Some $h_i(\phi) = 0$ can be explicitly written as $\phi_j = f(\phi_1, \phi_2, \ldots, \phi_{j-1}, \phi_{j+1}, \ldots, \phi_k)$. In this case we can optimize with $k-1$ parameters.
3. $g_i(\check{\phi})$ is known *a priori* to be positive.

Finding a Feasible Solution

Our design problem may be so complicated that we cannot easily find an initial design to serve as a feasible starting point in the optimization process. We could try to find one by unconstrained optimization by minimizing

$$-\sum_{i=1}^{m} w_i g_i(\phi) + \sum_{j=1}^{s} h_j^2(\phi) \qquad w_i \begin{cases} = 0 & g_i(\phi) \geq 0 \\ > 0 & g_i(\phi) < 0 \end{cases} \tag{4.45}$$

If the minimum is zero we have a feasible point. Failure to converge to zero does not necessarily mean that a feasible point does not exist.

Discouraging Undesirable Points

Having obtained a feasible starting point we might decide to simply reject nonfeasible points if they are obtained during optimization. Equivalently we might set $U(\phi)$ to a most unattractive value if any violation occurs. Alternatively, we could add the term

$$\sum_{i=1}^{m} w_i g_i^2(\phi) + \sum_{j=1}^{s} h_j^2(\phi) \qquad w_i \begin{cases} = 0 & g_i(\phi) \geq 0 \\ > 0 & g_i(\phi) < 0 \end{cases} \tag{4.46}$$

to the objective function. The objective function is not penalized as long as the constraints are satisfied. This procedure does not, unfortunately, always insure a strictly feasible solution.

The simple approaches just described have other disadvantages also. Discontinuities in the new function or its derivatives may be introduced. Steep walls or valleys may be formed at the boundary of the feasible region which can drastically slow down the optimization process. A method which simply rejects nonfeasible points can easily terminate at a false minimum [3].

4.3.3 Method of Lagrange Multipliers

A classical minimization problem is

$$\begin{aligned} &\text{minimize } U(\boldsymbol{\phi}) \\ &\text{subject to } \boldsymbol{h}(\boldsymbol{\phi}) = \mathbf{0} \end{aligned} \tag{4.47}$$

The method of *undetermined multipliers* or *Lagrange multipliers* is to find a stationary point for

$$\begin{aligned} L(\boldsymbol{\phi}, \lambda) &\triangleq U(\boldsymbol{\phi}) + \lambda^T \boldsymbol{h}(\boldsymbol{\phi}) \\ &= U(\boldsymbol{\phi}) + \sum_{i=1}^{s} \lambda_i h_i(\boldsymbol{\phi}) \end{aligned} \tag{4.48}$$

where

$$\lambda \triangleq \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_s \end{bmatrix} \tag{4.49}$$

Thus we must solve the system of $s+k$ equations

$$\begin{aligned} \nabla U + \sum_{i=1}^{s} \lambda_i \nabla h_i &= \mathbf{0} \\ \boldsymbol{h} &= \mathbf{0} \end{aligned} \tag{4.50}$$

in k unknowns $\boldsymbol{\phi}$ and s unknowns λ. The reason is immediately apparent if we write out these equations in expanded form, namely,

$$\begin{bmatrix} \frac{\partial L}{\partial \phi_1} \\ \vdots \\ \frac{\partial L}{\partial \phi_k} \\ \frac{\partial L}{\partial \lambda_1} \\ \vdots \\ \frac{\partial L}{\partial \lambda_s} \end{bmatrix} = \begin{bmatrix} \nabla U + \sum_{i=1}^{s} \lambda_i \nabla h_i \\ h \end{bmatrix} = 0 \tag{4.51}$$

Let (ϕ^0, λ^0) be a stationary point for the Lagrangian. In the neighbourhood of (ϕ^0, λ^0), we may write, for $h(\phi) = 0$,

$$U(\phi) = L(\phi, \lambda^0) \begin{matrix} \downarrow \\ \gtrless \\ \uparrow \end{matrix} \begin{matrix} \min U \\ \\ \max U \end{matrix} L(\phi^0, \lambda^0) = L(\phi^0, \lambda) = U(\phi^0). \tag{4.52}$$

Thus (ϕ^0, λ^0) is actually a degenerate saddle point. To handle inequality constraints we can generalize the Lagrange multiplier technique. But the computational effort is greatly increased.

4.3.4 Sequential Unconstrained Minimization Techniques

Barrier Function

One of the best known and most high developed of the sequential unconstrained minimization techniques (SUMT) will be briefly outlined here [4]. This method is also referred to as the Fiacco-McCormick method. Consider first the problem of minimization subject to inequality constraints defined in (4.1) and (4.2). Let us define the *barrier function*

$$B(\phi, r) \triangleq U(\phi) + rG(g) \tag{4.53}$$

where $G(g)$ is continuous for $g > 0$ and $G(g) \to \infty$ for any $g_i(\phi) \to 0$, and where $r > 0$. Two possible candidates for $G(g)$ immediately suggest themselves, namely

$$G(g) = \sum_{i=1}^{m} \frac{1}{g_i(\phi)} \tag{4.54}$$

and

$$G(g) = -\sum_{i=1}^{m} \log g_i(\phi) \tag{4.55}$$

Let us denote the interior of the region R of feasible points by R^0, where

$$R^0 \triangleq \{\phi \,|\, g(\phi) > 0\} \tag{4.56}$$

and

$$R \triangleq \{\phi \,|\, g(\phi) \geq 0\} \tag{4.57}$$

The procedure is to select a ϕ and a value of r, initially $\phi^0 \in R^0$ and $r_1 > 0$, respectively, and minimize the function B of (4.53). The form of this equation is such that one would expect the minimum, namely $\check{\phi}(r_1)$, to lie in R^0. Repeat the procedure for different values of r such that

$$r_1 > r_2 > \ldots r_j > 0 \quad \text{and} \quad \lim_{j\to\infty} r_j = 0 \tag{4.58}$$

each minimization being started at the previous minimum. The minimization of $B(\phi, r_2)$ would be started at $\check{\phi}(r_1)$, and so on.

The effect of the penalty is reduced every time the parameter r is reduced, so it is reasonable to expect that, under suitable conditions

$$\lim_{j\to\infty} \check{\phi}(r_j) = \check{\phi}$$

since by (4.58)

$$\lim_{j\to\infty} r_j = 0$$

so that

$$\lim_{j\to\infty} U[\check{\phi}(r_j)] = \check{U}$$

for the constrained minimum. A minimum of B should always be available in R^0, so any nonfeasible point that may be encountered can be rejected. This safeguard should not be overlooked, for obvious reasons.

Search for a Feasible Point

This procedure is termed an interior point unconstrained minimization technique, and it requires an initial $\phi^0 \in R^0$. If one is not available, the following approach may be adopted. Let

$$S \triangleq \{s \,|\, g_s(\phi) \leq 0, \quad s \in \{1, 2, \ldots, m\}\}$$
$$T \triangleq \{t \,|\, g_t(\phi) > 0, \quad t \in \{1, 2, \ldots, m\}\}$$

Now define a

$$B(\phi, r) = -\sum_{s \in S} g_s(\phi) + r \sum_{t \in T} G_t(g_t(\phi)) \tag{4.59}$$

to be minimized for a sequence of r values satisfying (4.58). The implications of (4.59) are that any satisfied constraints are prevented from becoming violated while an attempt to satisfy the rest is being made. As soon as any constraint is satisfied the corresponding index is transferred from S to T, and the procedure repeated. When S becomes empty we have obtained a $\phi^0 \in R^0$ and the solution process of the original problem can commence.

Convergence of the Barrier Function Problem

To prove convergence one must invoke the requirements for convex programming (See Section 4.1). In practice, however, the conditions may be difficult to verify even if they hold. Nevertheless, the method should work successfully on a wide variety of practical problems for which convergence is not readily proved. Bad initial choices of r and ϕ will slow down convergence. Too large a value of r_1 may render the first few minima of B to be relatively independent of U, whereas too small a value may render the penalty ineffective except near the boundary where elongated valleys with steep sides are produced. Because of this and the fact that a sequence of unconstrained problems has to be solved, efficient gradient methods are generally required.

A reduction factor of 10 for the values of r is probably as good as any once the process has started. The arbitrariness of this can be somewhat alleviated by using the SUMT method without the r parameters [4,5].

Equality Constraints

To include equality constraints the term

$$\frac{1}{r^{1/2}} \sum_{j=1}^{s} h_j^2(\phi) \tag{4.60}$$

can be added to the right hand side of (4.53). Clearly, as $r \to 0$, $h(\phi)$ must approach $\mathbf{0}$ or a minimum will not be reached.

The reader is referred to a lucid discussion found in Kowalik and Osborne [5].

4.3.5 Conditions for a Constrained Minimum

Necessary conditions which a stationary point ϕ^0 must satisfy in the problem of minimizing $U(\phi)$ subject to $g(\phi) \geq \mathbf{0}$ can be formulated. Assume $U(\phi)$ and $g(\phi)$ to be differentiable in the neighbourhood of a feasible stationary point ϕ^0, then

$$\nabla U(\phi^0) = \sum_{i=1}^{m} u_i \nabla g_i(\phi^0) \tag{4.61}$$

and

$$u^T g(\phi^0) = 0 \tag{4.62}$$

where

$$u \triangleq \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_m \end{bmatrix} \geq 0$$

These necessary conditions can be interpreted as follows: $\nabla U(\phi^0)$ is a nonnegative linear combination of the gradients $\nabla g_i(\phi^0)$ of those constraints which are active at ϕ^0 (a constraint $g_i(\phi) \geq 0$ is active at ϕ^0 if $g_i(\phi^0) = 0$).

Under the conditions of convex programming, i.e., if $U(\phi)$ is convex, $g(\phi)$ is concave, and R^0 is nonempty, the conditions become sufficient for ϕ^0 to be $\check{\phi}$, the constrained minimum. The relations (4.61) and (4.62) are called the Kuhn-Tucker relations [6]. An interpretation is sketched in Fig. 4.7. Note that if we have been using a reliable optimization method, and if the relations are satisfied, we can be reasonably sure that a local minimum has been attained even if the convexity requirements are not met.

For a detailed treatment of the Kuhn-Tucker relations, including their derivations and a discussion of the constraint qualification which must also hold, the reader is referred to an appropriate book.

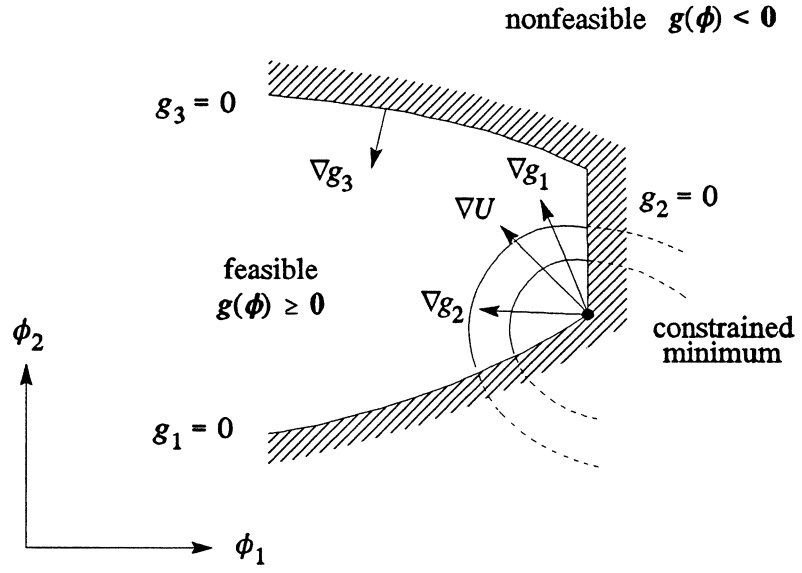

Fig. 4.7 Sufficient conditions for a constrained minimum, $u_1 > 0$, $u_2 > 0$, $u_3 = 0$.

4.4 Minimax Approximation

Classically, minimax approximation (See Section 4.2 for definitions) has implied the selection of the coefficients of a suitable polynomial or rational function so that it fits some desired specification (usually continuous on a closed interval) in an optimal equal-ripple manner. The Remez method and its generalizations are notable examples of iterative processes for obtaining best approximations using polynomials and rational functions. Applications are in filter design which can be solved by the classical

approach.

This section will emphasize less specialized methods applicable to a wider range of practical design problems.

4.4.1 Formulation in Terms of Inequality Constraints

Fig. 4.8 illustrates a typical filter design problem. We would like to find the (constrained) parameters of a suitable network so that certain passband and stopband specifications are met or exceeded. Assuming the approximating function and the specifications are real, let the error functions e_u and e_l be given by

$$\begin{aligned} e_u(\phi, \psi) &\triangleq w_u(\psi)[F(\phi, \psi) - S_u(\psi)] \\ e_l(\phi, \psi) &\triangleq w_l(\psi)[F(\phi, \psi) - S_l(\psi)] \end{aligned} \tag{4.63}$$

so that

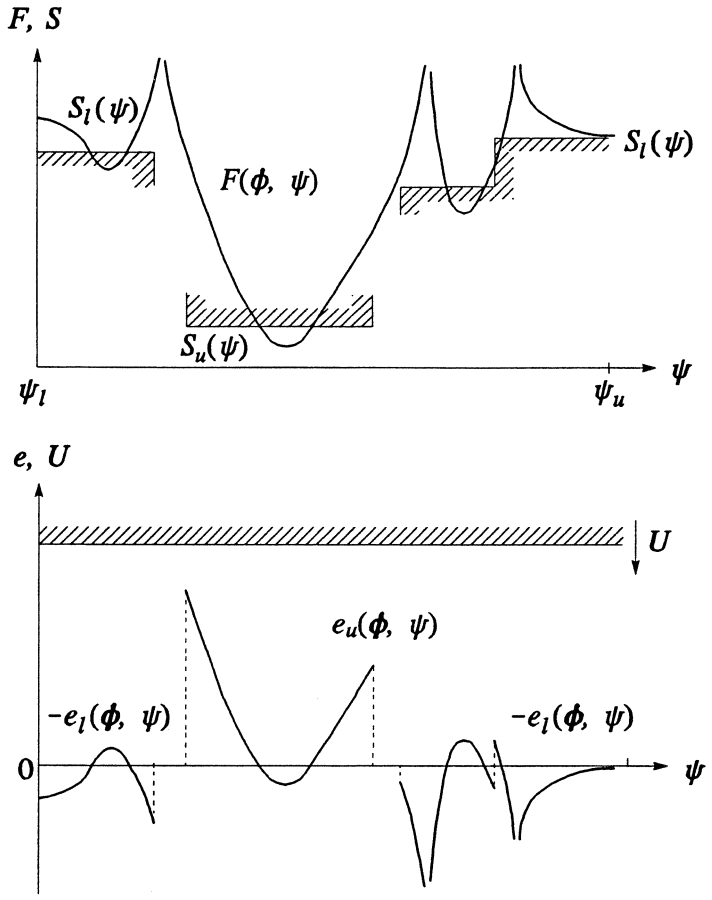

Fig. 4.8 Typical filter design problem (the specifications are violated).

$$\begin{aligned} e_{ui}(\phi) &\triangleq e_u(\phi, \psi_i), \quad i \in I_u \\ e_{li}(\phi) &\triangleq e_l(\phi, \psi_i), \quad i \in I_l \end{aligned} \tag{4.64}$$

This is simply a generalization of (4.31), (4.35), and (4.36), where the symbols have the same meaning. In the present case of (4.63) and (4.64), the subscript u refers to the upper or *passband* specification, the l to the lower or *stopband* specification.

Since approximation in the time and other domains can also be formulated in these terms, ψ is used rather than frequency. Furthermore, the index sets I_u and I_l are not necessarily disjoint.

The optimization problem can now be specified as: minimize the quantity

$$U = \phi_{k+1} \tag{4.65}$$

subject to

$$\begin{aligned} \phi_{k+1} &\geq e_{ui}(\phi), \quad i \in I_u \\ \phi_{k+1} &\geq -e_{li}(\phi), \quad i \in I_l \end{aligned} \tag{4.66}$$

and also to all other constraints, such as on ϕ. Observe that ϕ_{k+1} is an *additional* independent variable. As shown in Fig. 4.8 it may be visualized as a level or ceiling which is forced down on the deviations e_u and $-e_l$.

At a minimum at least one constraint in (4.66) must be an equality. Otherwise U can be lowered without violation. Further if

1. $\check{U} < 0$, the minimum amount by which the network response exceeds the specifications is maximized;
2. $\check{U} > 0$, the maximum amount by which the network response violates the specifications is minimized.

For loss or phase equalization, or time-domain approximation, for example, we might have only one specification, namely, $S(\psi)$. To treat these special cases we simply drop the subscripts u and l in (4.63), (4.64) and (4.66) and the objective is equivalent to minimizing

$$U = \max_{i \in I} |e_i(\phi)| \tag{4.67}$$

The weighting functions in (4.63) serve the following purpose. If one is much larger than the other, it emphasizes the deviation associated with it at the expense of the rest of the response if the specifications are violated. When the specifications are satisfied (we can now set the weighting function effectively to infinity if required), effort is switched to the rest of the response.

4.4.2 Methods for Minimax Approximation

Method of Waren, Lasdon and Suchman

An approach successfully implemented in optimal filter design [7] is to use sequential unconstrained minimization. We could, for example, define

$$B(\phi, \phi_{k+1}, r) = \phi_{k+1} + r \left\{ \sum_{i \in I_u} \frac{w_{ui}}{\phi_{k+1} - e_{ui}(\phi)} + \sum_{i \in I_l} \frac{w_{li}}{\phi_{k+1} + e_{li}(\phi)} + \text{other terms} \right\} \tag{4.68}$$

where the other terms might include parameter constraints. Note that in (4.68) the elements of $g(\phi)$ include

$$\begin{aligned} \frac{1}{w_{ui}} [\phi_{k+1} - e_{ui}(\phi)] &\geq 0, \quad i \in I_u \\ \frac{1}{w_{li}} [\phi_{k+1} + e_{li}(\phi)] &\geq 0, \quad i \in I_l \end{aligned} \tag{4.69}$$

Further, it should be remembered that ϕ_{k+1} is an additional independent variable. The appropriate formulations described in Section 4.3 are thus applicable to minimax approximation.

Method of Ishizaki and Watanabe

Ishizaki and Watanabe [8] have described a method which applies linear programming iteratively to achieve a best approximation in the minimax sense. Let us concern ourselves with the objective suggested by (4.67). this should not, however, be taken to imply that the method is less general than the one already outlined.

Linearizing $e_i(\phi)$, which is taken as real, at some point ϕ^j the problem becomes one of minimizing U of (4.65) subject to

$$\begin{aligned} \frac{1}{w_i} [\phi_{k+1} - e_i(\phi^j) - \nabla e_i^T(\phi^j)\, \Delta\phi^j] &\geq 0 \\ \frac{1}{w_i} [\phi_{k+1} + e_i(\phi^j) + \nabla e_i^T(\phi^j)\, \Delta\phi^j] &\geq 0 \end{aligned} \qquad i = 1, 2, \ldots, n > k \tag{4.70}$$

and other (linearized) constraints. Noting that the variables for linear programming should all be nonnegative, and imposing a rather practical constraint that the elements of ϕ should not change sign we have the linear program in $x \triangleq [x_1\ x_2\ \ldots\ x_{k+1}]^T$ such as to

$$\text{minimize } U = x_{k+1}$$

subject to

$$\pm\{e_i(\phi^j) + \nabla e_i^T(\phi^j) \begin{bmatrix} \phi_1^j\, x_1 - \phi_1^j \\ \phi_2^j\, x_2 - \phi_2^j \\ \vdots \\ \phi_k^j\, x_k - \phi_k^j \end{bmatrix}\} \leq x_{k+1},\ i = 1, 2, \ldots, n > k \tag{4.71}$$

$$x \geq 0$$

where

$$x_i \triangleq \frac{\Delta\phi_i^j}{\phi_i^j} + 1, \quad i = 1, 2, \ldots, k$$

The solution produces a direction given by $\Delta\phi^j$. Next we find α^j such that $\max_i |e_i(\phi^j + \alpha^j \Delta\phi^j)|$ is a minimum, set $\phi^{j+1} = \phi^j + \alpha^j \Delta\phi^j$ and repeat the process. Other linearized constraints can also be considered. Clearly such an approach is directly applicable to linear functions such as polynomials, for which $k + 1$ equal extrema results at the optimum.

Descent Minimax Method

Bandler, Srinivasan, and Charalambous [9] have described a descent type of algorithm for minimax approximation which also employs linear programming. Basically, the algorithm attempts to find a locally optimal downhill direction for the problem of minimizing M, where

$$M(\phi) = \max_{i \in I} f_i(\phi) \tag{4.72}$$

where the $f_i(\phi)$ are real nonlinear differentiable functions generally. Linearizing $f_i(\phi)$ and letting

$$J \triangleq \{i | f_i(\phi) = M(\phi), \quad i \in I\} \tag{4.73}$$

we can obtain, at some feasible point ϕ^j, the first-order changes

$$\Delta f_i(\phi^j) = \nabla f_i^T(\phi^j)\, \Delta\phi^j, \quad i \in J \tag{4.74}$$

In order for $\Delta\phi^j$ to define a descent direction for $M(\phi)$ we must have

$$\nabla f_i^T(\phi^j)\, \Delta\phi^j < 0, \; i \in J$$

Consider

$$\Delta\phi^j = -\sum_{i \in J} \alpha_i^j \nabla f_i(\phi^j) \tag{4.75}$$

$$\sum_{i \in J} a_i^j = 1 \tag{4.76}$$

$$a_i^j \geq 0, \quad i \in J \tag{4.77}$$

which suggests the linear program

$$\text{maximize } a_{r+1}^j, \quad i \in J \tag{4.78}$$

subject to

$$-\nabla f_i^T(\phi^j) \sum_{i \in J} \alpha_i^j \nabla f_i(\phi^j) \leq -\alpha_{r+1}^j, \quad i \in J \tag{4.79}$$

plus (4.76) and (4.77), where it is assumed that J has r elements.

Observe that J should be nonempty, and that if J has only one element, we obtain the steepest descent direction for the corresponding maximum of the $f_i(\phi)$. The solution to the linear program

provides $\Delta\phi^j$. We then find γ^j corresponding to the minimum value of $M(\phi^j + \gamma^j\Delta\phi^j)$. ϕ^{j+1} is set to $\phi^j + \gamma^j \Delta\phi^j$ and the procedure is repeated.

In practice, we will not have a set of $f_i(\phi)$ identically equal to the maximum value. An appropriate tolerance must, therefore, be introduced in (4.73) and a more suitable selection procedure for the elements of J formulated. For further details the original paper should be consulted [9]. It can be proved that the algorithm will, if correctly implemented, converge to the minimax solution.

4.4.3 Wavequide Transformer Example

Fig. 4.9 shows an example of minimax approximation [10]. The objective was to find

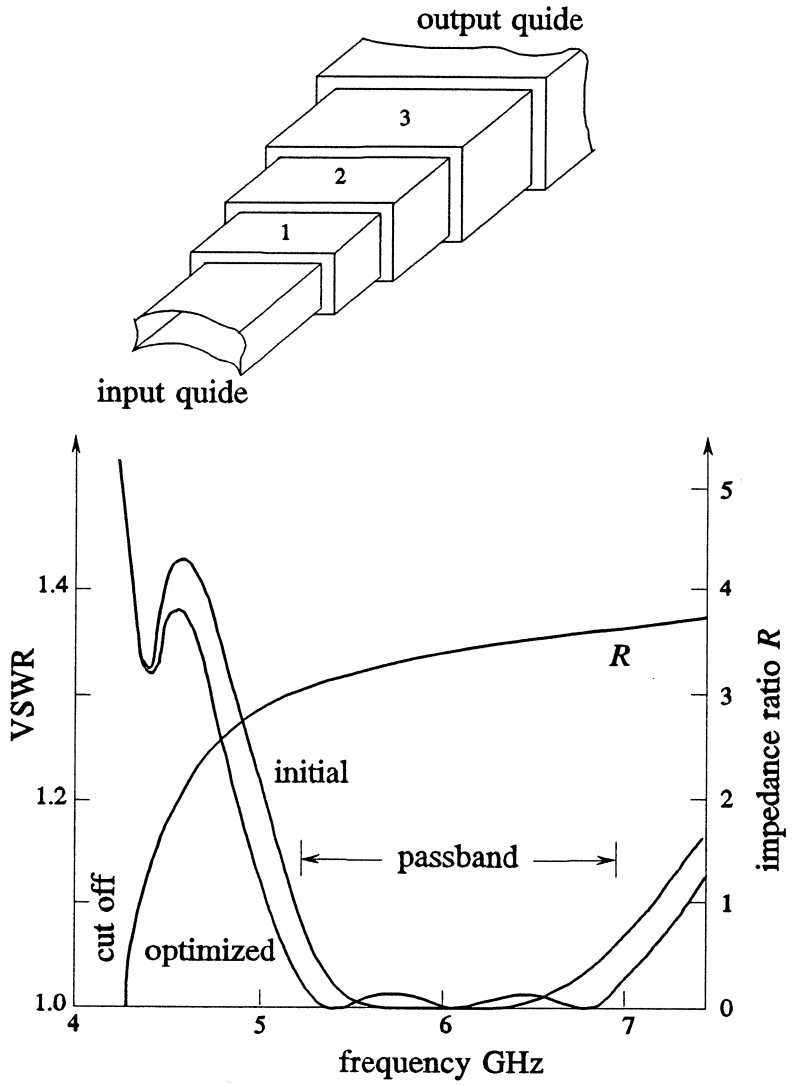

Fig. 4.9 Example of constrained minimax approximation (sketches).

$$\check{M} = \min_{\phi} \left\{ \max_{[f_l, f_u]} |\rho(\phi, f)| \right\} \tag{4.80}$$

for the 3-section inhomogeneous rectangular waveguide impedance transformer, where ρ is the reflection coefficient, and f is frequency in GHz. The parameters ϕ to be varied were the actual geometrical dimensions of the sections. The lower and upper band edges were $f_l = 5.4$ GHz and $f_u = 6.95$ GHz. It should be noted that (1) both input and output waveguides had different cut-off frequencies so that an exact synthesis was not possible, (2) severe constraints were placed on the parameters for a variety of physical reasons, (3) junction discontinuity susceptances could be taken directly into account and (4) the razor search method [11] (See Section 5.7) was employed. The reader is referred to Bandler [10] for further details of this type of problem and for some other numerical results.

4.4.4 Seven Section Filter Example

Let us consider in a little more detail, the optimization of a seven-section cascaded transmission-line filter of the type shown in Fig. 4.10. It is terminated at each end by

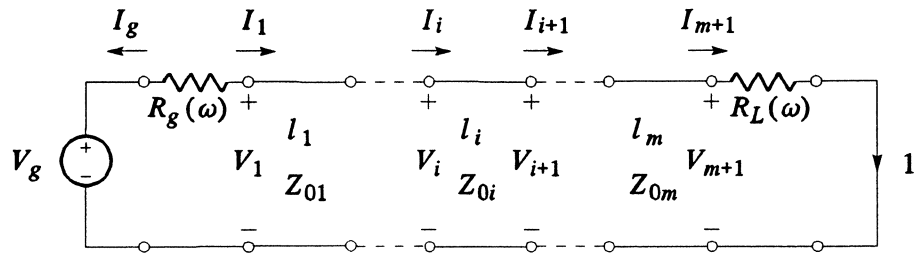

Fig. 4.10 Cascaded transmission-line filter between frequency-variable resistors.

$$R_g(\omega) = R_L(\omega) = \frac{377}{\sqrt{1 - (f_c/f)^2}}$$

where f is frequency in GHz and $f_c = 2.077$ GHz. The frequency variation of the terminations is thus like that of rectangular waveguides operating in the H_{10} mode with cut-off frequency 2.077 GHz. This interesting problem was previously considered by Carlin and Gupta [12]. All section lengths were kept fixed at 1.5 cm so that the maximum stopband insertion loss would occur at about 5 GHz. The passband 2.16 to 3 GHz was selected, for which a maximum of 0.4 dB loss was specified. The solution obtained by the method of Carlin and Gupta was used as the initial design as shown by Fig. 4.11.

As optimized by Bandler and Lee-Chan [13], the problem was to minimize M of (4.72), where

$$f_i(\phi) = \begin{cases} \frac{1}{2}[|\rho_i(\phi)|^2 - r^2] & \text{in the passband} \\ \frac{1}{2}[1 - |\rho_i(\phi)|^2] & \text{in the stopband} \end{cases} \tag{4.81}$$

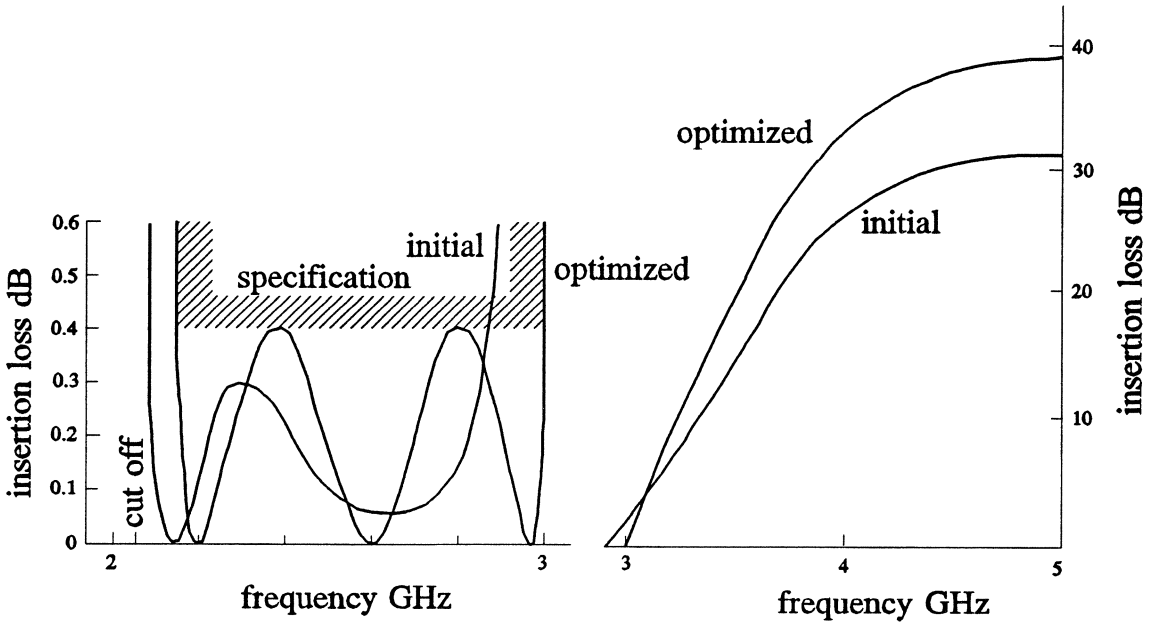

Fig. 4.11 Comparison between the initial and optimized responses of the filter of Fig. 4.9 (sketches).

and

$$\phi = \begin{bmatrix} Z_{01} \\ Z_{02} \\ \vdots \\ Z_{07} \end{bmatrix}$$

r is the reflection coefficient magnitude corresponding to an insertion loss of 0.4 dB, and $\rho_i(\phi)$ is the reflection coefficient of the filter at the ith frequency point. In particular, 22 uniformly spaced frequencies were selected from the passband and a single frequency, namely, 5 GHz for the stopband. The appropriately optimized response is shown in Fig. 4.11. These results have also been reproduced by the method of Bandler, Srinivasan, and Charalambous [9], using

$$\phi = \begin{bmatrix} Z_{01} \\ Z_{02} \\ Z_{03} \\ Z_{04} \end{bmatrix}$$

and letting

$$Z_{05} = Z_{03},\ Z_{06} = Z_{02},\ Z_{07} = Z_{01}$$

The method used to analyze the filter at each frequency is suggested in Fig. 4.10. A load current of 1 A was assumed and a simple *ABCD* matrix analysis was carried out to find all the other voltage and current variables shown (V_g will, of course, be generally complex and frequency dependent in this case). The appropriate partial derivatives were obtained from *one* such analysis per frequency point, using the adjoint network method (Chapter 7).

4.4.5 Conditions for a Minimax Optimum

To derive some insight into the necessary conditions which a stationary point $\boldsymbol{\phi}^0$ must satisfy in a minimax approximation problem [14], let us reduce it to the form

$$\text{minimize } U = \phi_{k+1} \tag{4.82}$$

subject to constraints of the form

$$\phi_{k+1} \geq f_i(\boldsymbol{\phi}), \quad i = 1, 2, \ldots, m \tag{4.83}$$

Rewriting the constraints as

$$g_i(\boldsymbol{\phi}) \triangleq \phi_{k+1} - f_i(\boldsymbol{\phi}) \geq 0, \quad i = 1, 2, \ldots, m \tag{4.84}$$

allows us to apply the Kuhn-Tucker relations (Section 4.3). Assuming U and the $f_i(\boldsymbol{\phi})$ to be differentiable in the neighbourhood of $\boldsymbol{\phi}^0$, we have at $\boldsymbol{\phi} = \boldsymbol{\phi}^0$

$$\begin{bmatrix} \nabla U \\ \dfrac{\partial U}{\partial \phi_{k+1}} \end{bmatrix} = \sum_{i=1}^{m} u_i \begin{bmatrix} \nabla \\ \dfrac{\partial}{\partial \phi_{k+1}} \end{bmatrix} (\phi_{k+1} - f_i(\boldsymbol{\phi})) \tag{4.85}$$

$$\boldsymbol{u}^T \boldsymbol{g} = 0$$

where $\boldsymbol{u}$ is defined by (4.62). But

$$\begin{aligned} \nabla U = \nabla \phi_{k+1} &= \mathbf{0} \\ \frac{\partial U}{\partial \phi_{k+1}} &= 1 \\ \frac{\partial f_i(\boldsymbol{\phi})}{\partial \phi_{k+1}} &= 0 \end{aligned} \tag{4.86}$$

everywhere. Furthermore, at least one constraint must be an equality. For convenience, assume the first m_0 constraints are equalities. Then

$$\begin{bmatrix} 0 \\ 1 \end{bmatrix} = \sum_{i=1}^{m_0} u_i \begin{bmatrix} -\nabla f_i(\phi^0) \\ 1 \end{bmatrix} \tag{4.87}$$

since

$$u_i = 0, \quad i = m_0 + 1, m_0 + 2, \ldots, m$$

Alternatively, the necessary conditions may be written as

$$\begin{aligned} \sum_{i=1}^{m_0} u_i \nabla f_i(\phi^0) &= 0 \\ \sum_{i=1}^{m_0} u_i &= 1 \\ u_i &\geq 0, \quad i = 1, 2, \ldots, m_0 \end{aligned} \tag{4.88}$$

An interpretation of these relations is sketched in Fig. 4.12. Under the conditions of convex programming, the $f_i(\phi)$ would have to be convex, and the conditions become sufficient for ϕ^0 to be $\check{\phi}$, the minimax optimum. Often m_0 will be equal to $k + 1$, but this is not a general requirement. The reader should observe the correspondence between (4.75) to (4.77) for $\Delta\phi^j = \mathbf{0}$ with (4.88). More insight into these relations, in particular as they relate to filter problems, should be gained by referring to Bandler [14].

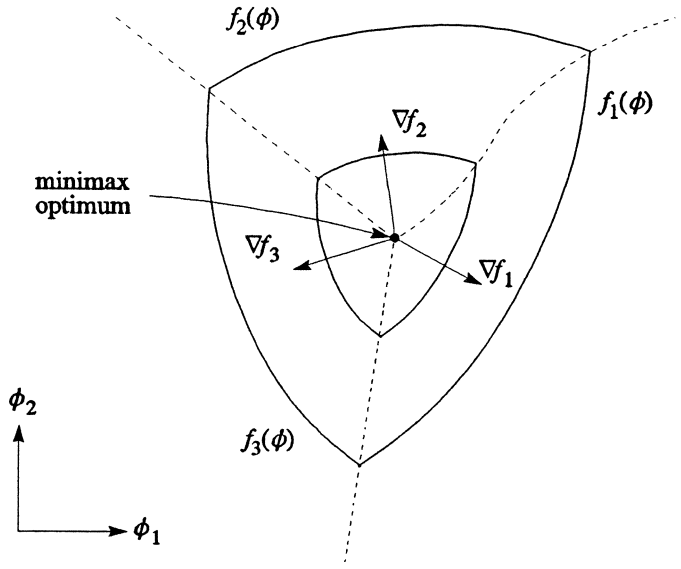

Fig. 4.12 Sufficient conditions for a minimax optimum, $u_1 > 0$, $u_2 > 0$, $u_3 > 0$.

4.5 References

[1] R. Fletcher, "A review of methods for unconstrained optimization," *Optimization*, R. Fletcher, Ed. New York: Academic Press, 1969.

[2] M.J. Box, "A comparison of several current optimization methods, and the use of transformations in constrained problems," *Computer J.*, vol. 9, 1966, pp. 67-77.

[3] J.W. Bandler, "Optimization methods for computer-aided design," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-17, 1969, pp. 533-552.

[4] A.V. Fiacco and G.P. McCormick, *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*. New York: Wiley, 1968.

[5] J. Kowalik and M.R. Osborne, *Methods for Unconstrained Optimization Problems*. New York: Elsevier, 1968.

[6] H.W. Kuhn and A.W. Tucker, "Non-linear programming," *Proc. 2nd Symp. on Math. Statistics and Probability*. Berkeley, CA: University of California Press, 1951, pp. 481-493.

[7] A.D. Waren, L.S. Lasdon and D.F. Suchman, "Optimization in engineering design," *Proc. IEEE*, vol. 55, 1967, pp. 1885-1897.

[8] Y. Ishizaki and H. Watanabe, "An iterative Chebyshev approximation method for network design," *IEEE Trans. Circuit Theory*, vol. CT-15, 1968, pp. 326-336.

[9] J.W. Bandler, T.V. Srinivasan, and C. Charalambous, "Minimax optimization of networks by grazor search," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-20, 1972, pp. 596-604.

[10] J.W. Bandler, "Computer optimization of inhomogeneous waveguide transformers," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-17, 1969, pp. 563-571.

[11] J.W. Bandler and P.A. MacDonald, "Optimization of microwave networks by razor search," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-17, 1969, pp. 552-562.

[12] H.J. Carlin and O.P. Gupta, "Computer design of filters with lumped-distributed elements or frequency variable terminations," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-17, 1969, pp. 598-604.

[13] J.W. Bandler and A.G. Lee-Chan, "Gradient razor search method for optimization," *IEEE International Microwave Symp. Dig.* (1971), pp. 118-119.

[14] J.W. Bandler, "Conditions for a minimax optimum," *IEEE Trans. Circuit Theory*, vol. CT-18, 1971, pp. 476-479.

Chapter 5

ALGORITHMS

This chapter deals with many algorithms for optimization, both one-dimensional and multidimensional, gradient and nongradient. They have distinguished themselves over the past few decades.

5.1 One-Dimensional Optimization

5.1.1 Introduction

There are several reasons for investigating the optima of functions for one variable. The most obvious reason is that this is the problem we are given. Another is that the multidimensional optimization strategy we are using may call for one-dimensional techniques for searching along some feasible direction to find the minimum in that direction. A third possibility is that we are dealing with an approximation problem for which we require the extrema of the error or deviation between the specified function and the approximating function.

5.1.2 Well-behaved Multimodal Function

Fig. 5.1 shows a relatively well-behaved function of one variable. The function is continuous

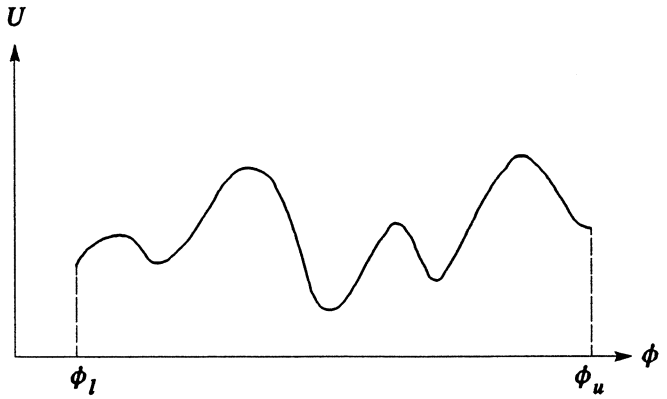

Fig 5.1 A well-behaved function of one variable.

and has continuous derivatives. The first derivative vanishes at the extrema (except at ϕ_l and ϕ_u). The turning points could be found by the *indirect* method of finding the zeros of $dU/d\phi = 0$, assuming that U is easily differentiated, and the resulting equation easily solved.

5.1.3 Badly-behaved Multimodal Function

Fig. 5.2 shows a badly-behaved function of one variable. The function is discontinuous, has discontinuous derivatives, and the first derivative vanishes at only one extremum. The extrema would have to be found by a *direct* method.

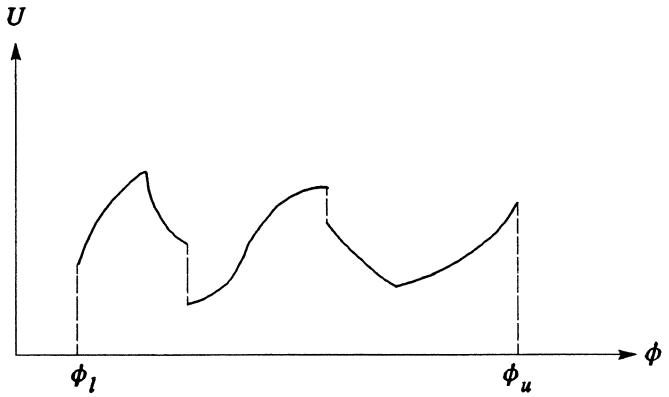

Fig 5.2 A badly-behaved function of one variable.

5.1.4 Note on Notation

Note that although we might, in practice, be dealing with a function of ψ, such as $e(\phi, \psi)$ for a particular value of ϕ, the discussion here will, without loss of generality, consider a function U of a single variable ϕ.

5.1.5 Methods for Unimodal Functions

Powerful methods are available for functions that are known to be unimodal on a particular interval. These will be discussed first. Then we will consider methods of finding such intervals.

The methods can be divided into two classes: 1) the minimax *direct elimination* methods – minimax, because they minimize the maximum interval which could contain the minimum, and 2) the *approximation* or *interpolation* methods. The latter are generally effective on smooth functions, but the former can be applied to arbitrary unimodal functions.

5.2 Elimination Methods

5.2.1 Interval of Uncertainty

Suppose that at the start of the jth iteration of a search method for a minimum we have a unimodal interval $[\phi_l^j, \phi_u^j]$. See Fig. 5.3. The *interval of uncertainty* I^j as to where the minimum is located is, therefore, given by

$$I^j \triangleq \phi_u^j - \phi_l^j \tag{5.1}$$

In order to reduce the interval of uncertainty using function values only, the function must be evaluated at *two* interior points, say ϕ_a^j and ϕ_b^j. (Evaluation of the function at only one interior point is not enough – the minimum could lie on either side.) We take

$$\phi_l^j < \phi_a^j < \phi_b^j < \phi_u^j \tag{5.2}$$

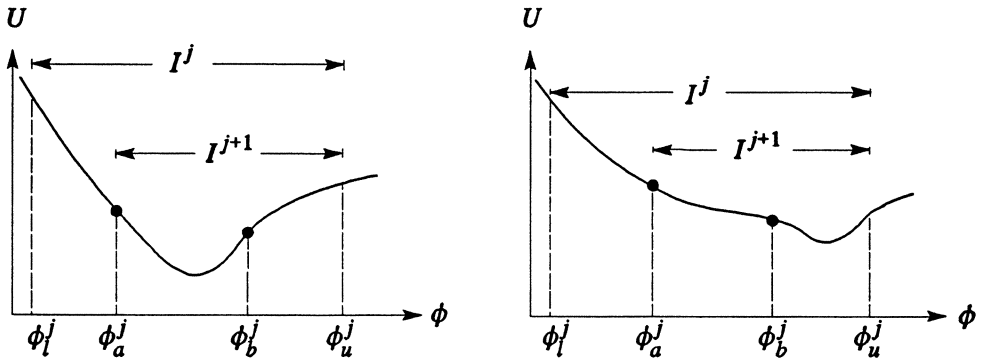

Fig 5.3 Uncertainty intervals and their reduction.

5.2.2 Reduction of the Interval

Let

$$U_a^j \triangleq U(\phi_a^j) \tag{5.3}$$

and

$$U_b^j \triangleq U(\phi_b^j) \tag{5.4}$$

If $U_a^j > U_b^j$ the minimum lies in $[\phi_a^j, \phi_u^j]$ and $I^{j+1} = \phi_u^j - \phi_a^j$ (5.5)

If $U_a^j < U_b^j$ the minimum lies in $[\phi_l^j, \phi_b^j]$ and $I^{j+1} = \phi_b^j - \phi_l^j$ (5.6)

The uncertainty intervals are shown in Fig. 5.3.

The difference between two well-known and efficient methods, the Fibonacci search and the Golden Section search, is in how these interior points are located. We next discuss the slightly less efficient but simpler Golden Section search method. The reader is referred elsewhere for detailed accounts of the various methods [1-3].

5.3 Golden Section Search

5.3.1 Theory

Symmetrical Placement of Points

In the absence of any relevant *a priori* knowledge, it would first of all be reasonable to select ϕ_a^j and ϕ_b^j such that, whatever the outcome of comparing U_a^j with U_b^j

$$I^{j+1} = \phi_u^j - \phi_a^j = \phi_b^j - \phi_l^j \tag{5.7}$$

which is achieved if ϕ_a^j and ϕ_b^j are located symmetrically in $[\phi_l^j, \phi_u^j]$. Secondly, in an effort to make I^{j+1} as small as possible we would arrange ϕ_a^j and ϕ_b^j so that

$$\phi_u^j - \phi_b^j = \phi_a^j - \phi_l^j \geq \phi_b^j - \phi_a^j \tag{5.8}$$

Reduction Process

Suppose $U_a^j > U_b^j$. Then we would set

$$\phi_l^{j+1} = \phi_a^j, \quad \phi_a^{j+1} = \phi_b^j, \quad \phi_u^{j+1} = \phi_u^j \tag{5.9}$$

Following this procedure we would find that

$$I^{j+2} = \phi_u^{j+1} - \phi_a^{j+1} = \phi_u^j - \phi_b^j \tag{5.10}$$

since ϕ_b^{j+1} is to be placed so that

$$\phi_l^{j+1} < \phi_a^{j+1} < \phi_b^{j+1} < \phi_u^{j+1} \tag{5.11}$$

But

$$\phi_u^j - \phi_b^j = \phi_a^j - \phi_l^j \tag{5.12}$$

so that

$$I^j = I^{j+1} + I^{j+2} \tag{5.13}$$

Our next idea would probably be to try to arrange that the interval of uncertainty containing the

minimum is reduced by a constant factor with each iteration. Thus,

$$\ldots = \frac{I^j}{I^{j+1}} = \frac{I^{j+1}}{I^{j+2}} = \ldots \triangleq \tau \tag{5.14}$$

where τ is a constant.

Golden Section

Solving (5.14) with (5.13) we obtain

$$\tau^2 = \tau + 1 \tag{5.15}$$

The solution of interest is

$$\tau = \frac{1}{2}(1 + \sqrt{5}) = 1.6180\ldots \tag{5.16}$$

The division of a line according to (5.14) and (5.15) is called the Golden Section of a line.

Note that earlier ((5.8)) we called for

$$I^{j+2} \geq I^{j+1} - I^{j+2} \tag{5.17}$$

i.e.,

$$\frac{I^{j+1}}{I^{j+2}} \leq 2 \tag{5.18}$$

which is satisfied since $\tau < 2$.

5.3.2 Algorithm

Point Selection Scheme

For a normalized interval of uncertainty (i.e., from 0 to 1), the two inner points are at $1/\tau^2$ and $1/\tau$. See the graph in Fig. 5.4.

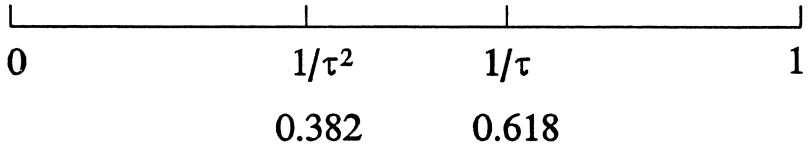

Fig 5.4 Placement of Golden Section points on a normalized interval.

The symmetry of the Golden Section Search method requires that

$$1/\tau - 0 = 1 - 1/\tau^2$$

or

$$1/\tau = 1 - 1/\tau^2$$

Multiplying both sides of this equation by τ^2, we have

$$\tau = \tau^2 - 1$$

which yields the Golden Section ratio consistent with (5.16).

Placement of Points

At the jth iteration of Golden Section search we have

$$\left.\begin{aligned} \phi_a^j &= \frac{1}{\tau^2} I^j + \phi_l^j \\ \phi_b^j &= \frac{1}{\tau} I^j + \phi_l^j \end{aligned}\right\} \quad j = 1, 2, 3, \ldots \tag{5.19}$$

Observe that each iteration except the first actually requires only one function evaluation due to symmetry.

$$\text{If } U_a^j > U_b^j \text{ then } \phi_l^{j+1} = \phi_a^j, \quad \phi_a^{j+1} = \phi_b^j, \quad \phi_u^{j+1} = \phi_u^j, \quad U_a^{j+1} = U_b^j \tag{5.20}$$

$$\text{If } U_a^j < U_b^j \text{ then } \phi_l^{j+1} = \phi_l^j, \quad \phi_b^{j+1} = \phi_a^j, \quad \phi_u^{j+1} = \phi_b^j, \quad U_b^{j+1} = U_a^j \tag{5.21}$$

In either case only one additional point is required with the function value at that point.

Reduction Ratio

After n function evaluations the reduction ratio of intervals of uncertainty is

$$\frac{I^1}{I^n} = \tau^{n-1} \tag{5.22}$$

See Table 5.1.

Table 5.1 Uncertainty Interval Reduction Ratio.

Interval Reduction in terms of τ^i	Interval Reduction in terms of I^1/I^{i+1}	Iteration i	Number of Function Evaluations
$\tau 1$	I^1/I^2	1	2
τ^2	I^1/I^3	2	3
τ^3	I^1/I^4	3	4
.	.	.	.
.	.	.	.
.	.	.	.
τ^{n-1}	I^1/I^n	n-1	n

After n function evaluations the uncertainty interval is reduced by τ^{n-1}.

Accuracy

For a desired accuracy of σ we should choose n so that

$$\tau^{n-2} < \frac{\phi_u^1 - \phi_l^1}{\sigma} \leq \tau^{n-1} \tag{5.23}$$

5.3.3 Example

An example involving four function evaluations is shown in Fig. 5.5. In the example the initial interval has been reduced by a factor of about 4.2. Eleven evaluations would have reduced the interval by about 121.

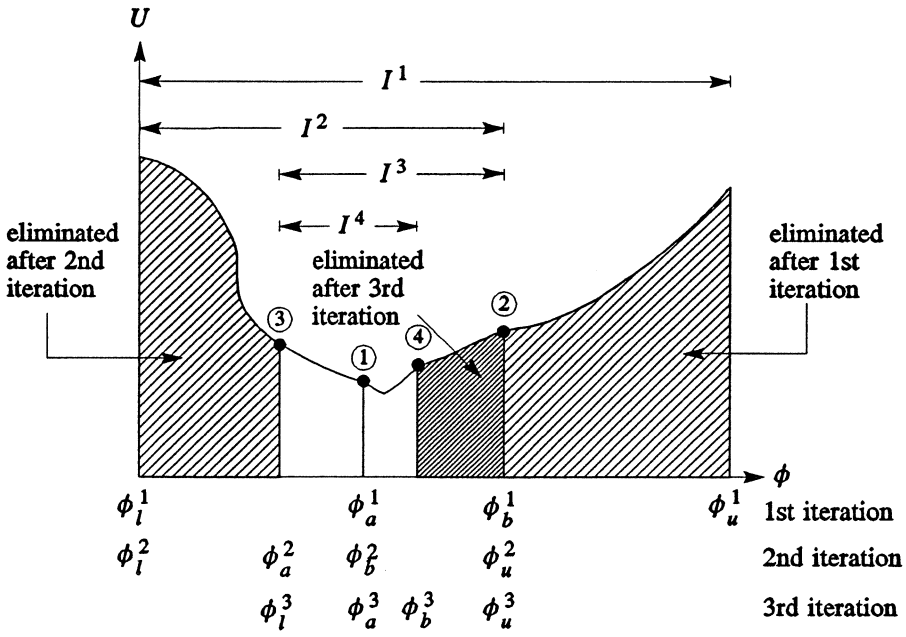

Fig 5.5 A Golden Section search scheme involving three iterations (four function evaluations) on a unimodal function of one variable.

5.4 Fibonacci Search

5.4.1 Theory

After $n-1$ Function Evaluations

Suppose we have already made $n-1$ function evaluations, the $(n-1)$th point being an interior point of an interval of length I^{n-1}. Obviously, I^n will be least if the $(n-1)$th point was in the middle of the interval so that the nth (and final) point could be placed as close as possible to it. Alternatively, we might, in accordance with our previous discussion prefer to think of the $(n-1)$th and nth points to be symmetrically placed with a separation ε and let $\varepsilon \to 0$. See Fig. 5.6(a). In practice, if only the minimum value obtained with its location is desired we could simply omit the nth function evaluation.

After $n-2$ Function Evaluations

Suppose we have already made $n-2$ function evaluations, the $(n-2)$th point being an interior point of an interval of length I^{n-2}. See Fig. 5.6(b). Assuming a particular value for I^{n-1} we could ask what the value is of I^{n-2} consistent with symmetrical placement of the $(n-1)$th point (for reasons already discussed) and optimal placement of the nth point. The answer is that

$$I^{n-2} = I^{n-1} + I^n \tag{5.24}$$

(assuming $\varepsilon = 0$).

After $n-3$ Function Evaluations

Continuing in this vein (Fig. 5.6(c)) we see that

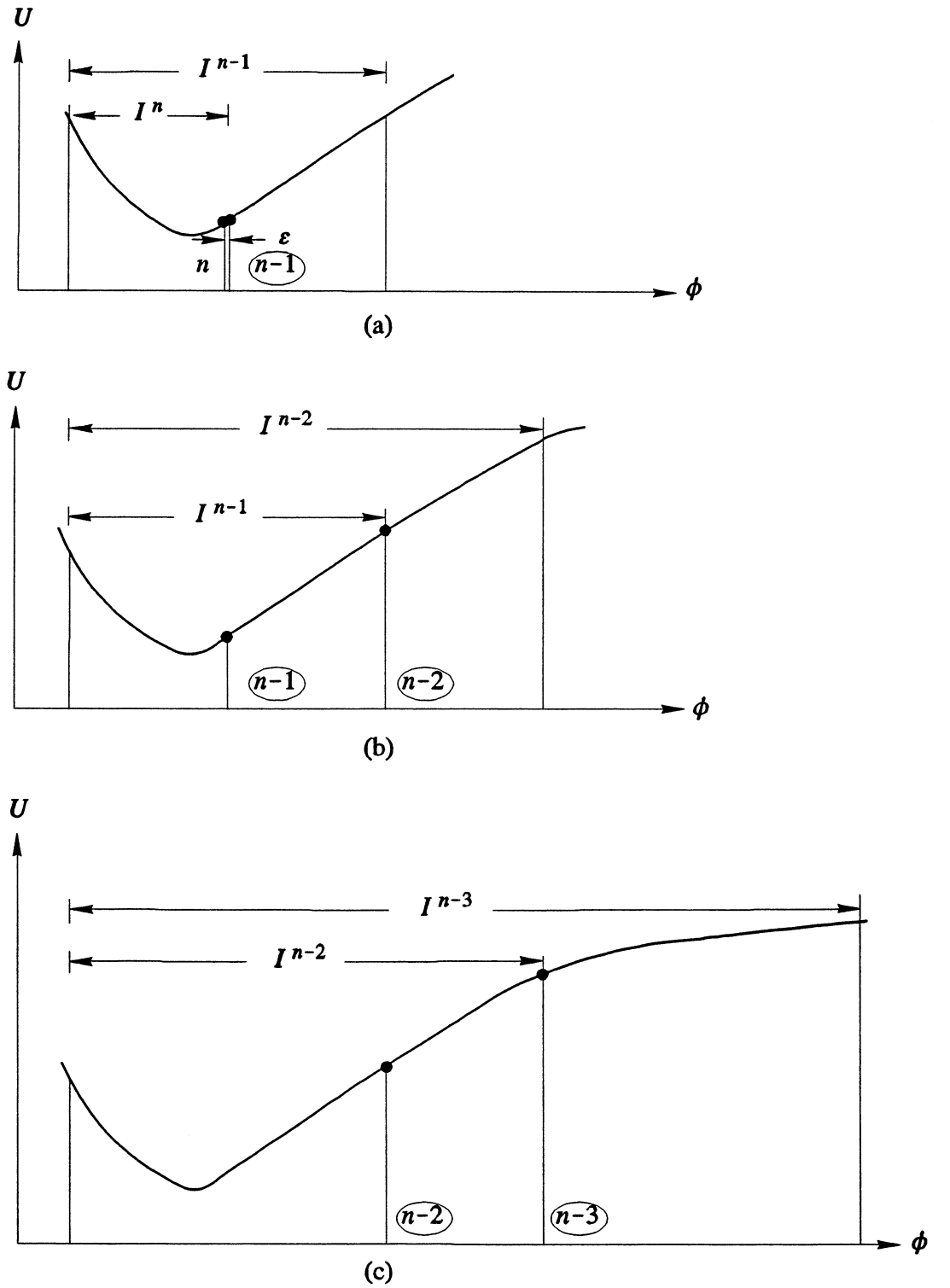

Fig 5.6 Intervals for Fibonacci search, (a) the final placement of points, (b) the previous set, and (c) the second previous set.

$$I^{n-3} = I^{n-2} + I^{n-1} \tag{5.25}$$

Formalization of the Method

As for Golden Section search

$$I^j = I^{j+1} + I^{j+2}, \quad j = 1, 2, \ldots, n-2, \quad n > 2 \tag{5.26}$$

Define the reduction ratio after n function evaluations as F_n, i.e.,

$$F_n \triangleq \frac{I^1}{I^n} \tag{5.27}$$

Let $I^n = 1$, for convenience. Then $I^{n-1} = 2$. Using the relationships (5.24), (5.25) ...

$$\begin{aligned} I^n &= 1 \\ I^{n-1} &= 2 \\ I^{n-2} &= 3 \\ I^{n-3} &= 5 \\ &\vdots \\ I^{j+1} &= F_{n-j} \\ \\ I^j &= F_{n+1-j} \\ &\vdots \\ I^2 &= F_{n-1} \\ I^1 &= F_n \end{aligned}$$

The Fibonacci Numbers

We recognize this sequence of numbers as being in the Fibonacci sequence of numbers defined by

$$\begin{aligned} F_0 &= F_1 = 1 \\ F_i &= F_{i-1} + F_{i-2}, \quad i = 2, 3, \ldots \end{aligned} \tag{5.28}$$

the first six terms, for example, being 1, 1, 2, 3, 5, 8. For example, $n = 4$ gives $I^1 = 5 = F_4$. Note that I^0, the interval of uncertainty after 0 evaluations, is equal to $I^1 = 1$, as expected.

The solution to the recurrence relationship can be shown to be

$$F_n = \frac{1}{\sqrt{5}} \left[\left(\frac{1 + \sqrt{5}}{2} \right)^{n+1} - \left(\frac{1 - \sqrt{5}}{2} \right)^{n+1} \right] \tag{5.29}$$

5.4.2 Algorithm

Placement of Points

At the jth iteration of Fibonacci search using n function evaluations ($n \geq 2$) we have

$$\left.\begin{aligned} \phi_a^j &= \frac{F_{n-1-j}}{F_{n+1-j}} I^j + \phi_l^j \\ \phi_b^j &= \frac{F_{n-j}}{F_{n+1-j}} I^j + \phi_l^j \end{aligned}\right\} \quad j = 1, 2, \ldots, n-1 \tag{5.30}$$

As with Golden Section search each iteration except the first actually requires only one function evaluation. It may easily be verified that the same relationship holds as for Golden Section search whether $U_a^j > U_b^j$ or $U_a^j < U_b^j$.

Reduction Ratio

The interval of uncertainty after j iterations is

$$I^{j+1} = \phi_u^j - \phi_a^j = \phi_b^j - \phi_l^j \tag{5.31}$$

reducing the interval I^j by a factor

$$\frac{I^j}{I^{j+1}} = \frac{F_{n+1-j}}{F_{n-j}} \tag{5.32}$$

As a check, we find that after $n-1$ iterations, assuming infinite resolution, the reduction ratio is

$$\frac{I^1}{I^n} = \frac{F_n}{F_{n-1}} \frac{F_{n-1}}{F_{n-2}} \cdots \frac{F_2}{F_1} = F_n \tag{5.33}$$

Accuracy

For a desired accuracy of σ we should choose n so that

$$F_{n-1} < \frac{\phi_u^1 - \phi_l^1}{\sigma} \leq F_n \tag{5.34}$$

5.4.3 Example

An example involving four function evaluations is shown in Fig. 5.7. In the example the initial interval has been reduced by a factor of 5 after 4 function evaluations. Eleven function evaluations would have reduced the interval by a factor of 144.

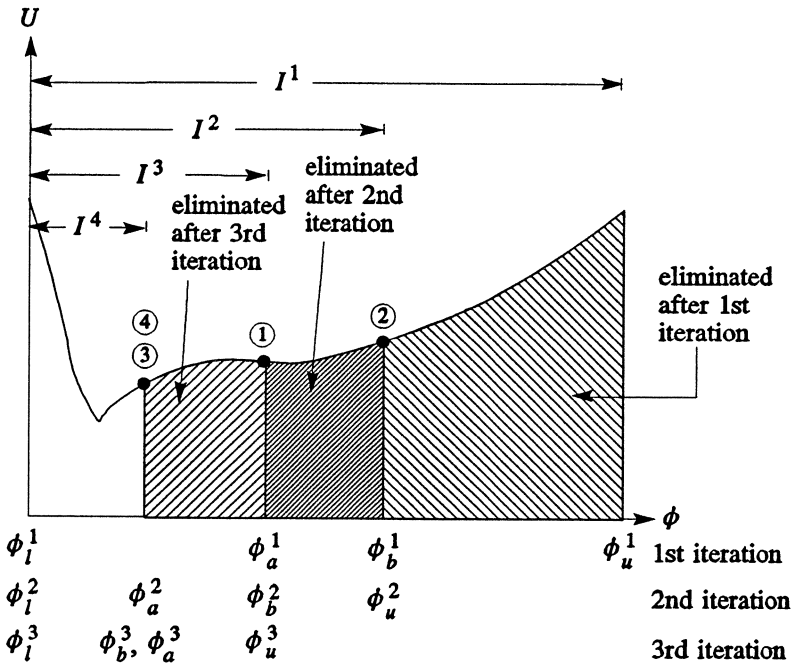

Fig 5.7 A Fibonacci search scheme involving three iterations (four function evaluations) on a unimodal function of one variable.

First Evaluation

$$\phi_a^1 = \frac{F_2}{F_4} I^1 + \phi_l^1 = \frac{2}{5} I^1 + \phi_l^1$$

Second Evaluation

$$\phi_b^1 = \frac{F_3}{F_4} I^1 + \phi_l^1 = \frac{3}{5} I^1 + \phi_l^1$$

Since $U_a^1 < U_b^1$ we take $\phi_l^2 = \phi_l^1$, $\phi_b^2 = \phi_a^1$, $\phi_u^2 = \phi_b^1$, $I^2 = \frac{3}{5} I^1$

Third Evaluation

$$\phi_a^2 = \frac{F_1}{F_3} I^2 + \phi_l^2 = \frac{1}{3}\frac{3}{5} I^1 + \phi_l^1 = \frac{1}{5} I^1 + \phi_l^1$$

$$\phi_b^2 = \frac{F_2}{F_3} I^2 + \phi_l^2 = \frac{2}{3}\frac{3}{5} I^1 + \phi_l^1 = \phi_a^1$$

Since $U_a^2 < U_b^2$ we take $\phi_l^3 = \phi_l^2$, $\phi_b^3 = \phi_a^2$, $\phi_u^3 = \phi_b^2$, $I^3 = \frac{2}{5} I^1$

Fourth Evaluation (may be omitted)

$$\begin{cases} \phi_a^3 = \dfrac{F_0}{F_2} I^3 + \phi_l^3 = \dfrac{1}{2}\dfrac{2}{5} I^1 + \phi_l^1 = \dfrac{1}{5} I^1 + \phi_l^1 = \phi_a^2 \\ \phi_b^3 = \dfrac{F_1}{F_2} I^3 + \phi_l^3 = \dfrac{1}{2}\dfrac{2}{5} I^1 + \phi_l^1 = \dfrac{1}{5} I^1 + \phi_l^1 = \phi_a^2 \end{cases}$$

5.5 Discussion of Fibonacci and Golden Section Methods

5.5.1 Ratio of Effectiveness

Limits for Large n

For Fibonacci search as $n \to \infty$

$$\frac{I^1}{I^n} = F_n \approx \frac{1}{\sqrt{5}} \left[\frac{1 + \sqrt{5}}{2} \right]^{n+1} = \frac{\tau^{n+1}}{\sqrt{5}} \tag{5.35}$$

and

$$\lim_{n \to \infty} \frac{F_n}{F_{n-1}} = \tau \tag{5.36}$$

The ratio of effectiveness of the Fibonacci search as compared with Golden Section is, therefore, given by

$$\lim_{n \to \infty} \frac{F_n}{\tau^{n-1}} = \frac{\tau^2}{\sqrt{5}} = 1.1708 \tag{5.37}$$

Consider the reduction ratio per iteration, namely,

$$\frac{I^j}{I^{j+1}} = \lim_{n \to \infty} \frac{F_{n+1-j}}{F_{n-j}} = \tau \tag{5.38}$$

For a very large number of function evaluations, the reduction ratio per iteration is practically the same for both methods. It is easily shown, for example, that ϕ_a^1 and ϕ_b^1 are practically the same for both methods with n large. Golden Section search ultimately provides an interval of uncertainty only some 17% greater than Fibonacci search.

Accuracy

Thus, Golden Section search is frequently preferred because the number of function evaluations need not be fixed in advance, it is simple to implement and at most one additional function evaluation is required to achieve a specified accuracy, since

$$\tau^n \geq F_n, \quad n = 0, 1, 2, \ldots \tag{5.39}$$

where τ^n is the reduction ratio for $n+1$ evaluations by Golden Section search and F_n is the reduction ratio for n evaluations by Fibonacci search.

5.5.2 Finding an Initial Interval

The Algorithm

Suppose an initial unimodal interval is not available. It can be found by the following procedure. Select a convenient increment in the direction of decreasing U and evaluate $U(\phi^i)$ for

$$\phi^i = \phi^0 + \sum_{j=1}^{i} \tau^{j-1}\Delta\phi, \quad i = 1, 2, \ldots \tag{5.40}$$

Stop when

$$U(\phi^i) \geq U(\phi^{i-1}) \tag{5.41}$$

for some value of i. We now can define a unimodal interval and proceed directly with Golden Section search.

$$\text{If } \Delta\phi > 0 \text{ then } \phi_l^1 = \phi^{i-2}, \ \phi_a^1 = \phi^{i-1}, \ \phi_u^1 = \phi^i \tag{5.42}$$

$$\text{If } \Delta\phi < 0 \text{ then } \phi_l^1 = \phi^i, \ \phi_b^1 = \phi^{i-1}, \ \phi_u^1 = \phi^{i-2} \tag{5.43}$$

Illustration

Consider the situation of Fig. 5.8. Here,

$$\begin{aligned} \phi_a^1 &= \phi_l^1 + \frac{\tau^3\Delta\phi}{\tau^3\Delta\phi + \tau^4\Delta\phi}(\phi_u^1 - \phi_l^1) \\ &= \phi_l^1 + \frac{1}{1+\tau}(\phi_u^1 - \phi_l^1) \\ &= \frac{1}{\tau^2} I^1 + \phi_l^1 \end{aligned} \tag{5.44}$$

using the Golden Section formula (5.15). Thus, we have the desired unimodal interval with an interior point and we are ready to continue with search by Golden Section.

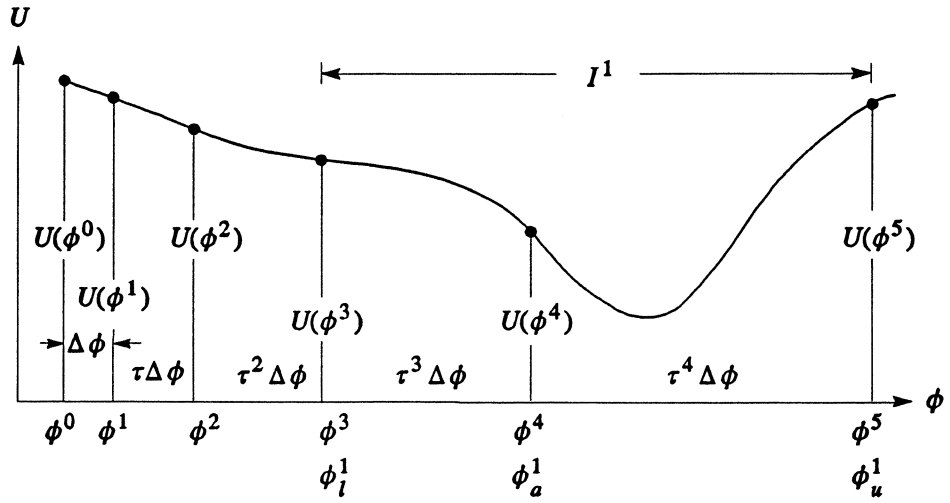

Fig 5.8 Illustration of finding an initial interval for the Golden Section method.

5.6 Interpolation Methods

There are several interpolation methods, including quadratic and cubic, which are available [2-5]. A rather straightforward method was suggested by Davies, Swann and Campey [2,3].

In this section we will discuss the method of Davies, Swann and Campey, quadratic and cubic interpolation methods.

5.6.1 The Method of Davies, Swann and Campey [2,3]

Bounding the Minimum

This method does not initially require bounds on the minimum. We select a convenient increment $\Delta\phi$ in the direction of decreasing U and evaluate $U(\phi^i)$ for

$$\phi^i = \phi^0 + \sum_{j=1}^{i} 2^{j-1}\Delta\phi, \quad i = 1, 2, \ldots \tag{5.45}$$

When

$$U(\phi^i) > U(\phi^{i-1}) \tag{5.46}$$

we evaluate U at

$$\phi^{i+1} = \phi^{i-1} + (\phi^{i-1} - \phi^{i-2}) \tag{5.47}$$

Selection of Points

We now have 4 points uniformly spaced along the ϕ axis, namely

$$\phi^{i-2}, \phi^{i-1}, \phi^{i+1}, \phi^i$$

$$\text{If}\begin{cases}\Delta\phi > 0 \text{ and } \begin{cases} U(\phi^{i+1}) < U(\phi^{i-1}) \text{ then } a = \phi^{i-1},\ b = \phi^{i+1},\ c = \phi^{i} \\ U(\phi^{i+1}) > U(\phi^{i-1}) \text{ then } a = \phi^{i-2},\ b = \phi^{i-1},\ c = \phi^{i+1} \end{cases} \\ \Delta\phi < 0 \text{ and } \begin{cases} U(\phi^{i+1}) < U(\phi^{i-1}) \text{ then } a = \phi^{i},\ b = \phi^{i+1},\ c = \phi^{i-1} \\ U(\phi^{i+1}) > U(\phi^{i-1}) \text{ then } a = \phi^{i+1},\ b = \phi^{i-1},\ c = \phi^{i-2} \end{cases}\end{cases} \tag{5.48}$$

The points obtained are shown in Fig. 5.9.

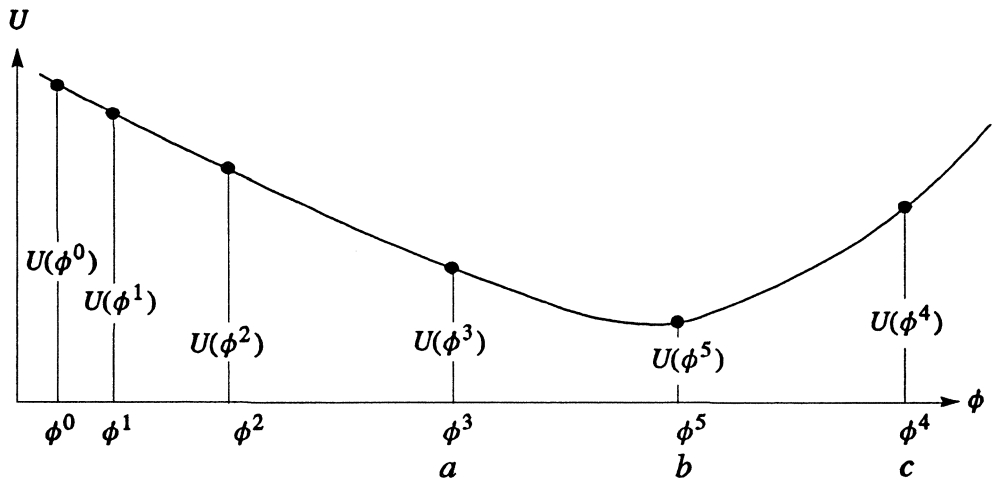

Fig 5.9 Illustration of the method of Davies, Swann and Campey.

Interpolation Formula

Now it is easily shown that the minimum of a quadratic fitted at a, b and c is at

$$d = b + \frac{(b - a)(U_a - U_c)}{2(U_a - 2U_b + U_c)} \tag{5.49}$$

Evaluation of U at d resulting in the estimate of the minimum and completes one stage of the method. A new stage with reduced $\Delta\phi$ can then be started at b or d whichever corresponds to the smaller value.

5.6.2 Quadratic Interpolation Method

Interpolation Formula

At the jth iteration of the typical method involving quadratic interpolation we have a unimodal function over $[\phi_l^j, \phi_u^j]$ with an interior point ϕ_m^j (see Fig. 5.10). Let

$$a = \phi_l^j,\ b = \phi_m^j,\ c = \phi_u^j \tag{5.50}$$

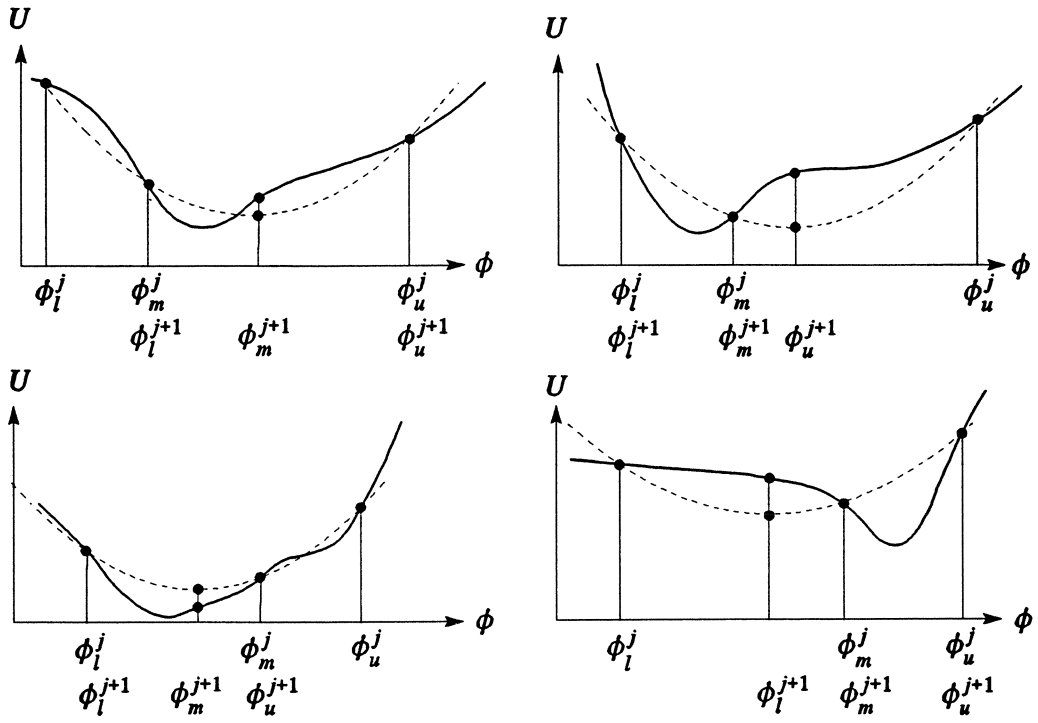

Fig 5.10 Illustration of the quadratic interpolation method (sketches).

Then the minimum of the quadratic through a, b and c is at

$$d = \frac{1}{2}\,\frac{(b^2 - c^2)U_a + (c^2 - a^2)U_b + (a^2 - b^2)U_c}{(b - c)U_a + (c - a)U_b + (a - b)U_c} \tag{5.51}$$

Selection of Points

Then ϕ_l^{j+1}, ϕ_m^{j+1} and ϕ_u^{j+1} are obtained as follows.

$$\text{If}\ \begin{cases} b > d \text{ and } \begin{cases} U_b > U_d \text{ then } \phi_l^{j+1} = a,\ \phi_m^{j+1} = d,\ \phi_u^{j+1} = b \\ U_b < U_d \text{ then } \phi_l^{j+1} = d,\ \phi_m^{j+1} = b,\ \phi_u^{j+1} = c \end{cases} \\ b < d \text{ and } \begin{cases} U_b > U_d \text{ then } \phi_l^{j+1} = b,\ \phi_m^{j+1} = d,\ \phi_u^{j+1} = c \\ U_b < U_d \text{ then } \phi_l^{j+1} = a,\ \phi_m^{j+1} = b,\ \phi_u^{j+1} = d \end{cases} \end{cases} \tag{5.52}$$

The procedure may be repeated for greater accuracy, convergence being guaranteed.

Bounding the Minimum

Powell's version of this algorithm does not initially require bounds on the minimum. Having selected three suitable initial points, an attempt to find the minimum through these points is made. For badly behaved functions or points far from the minimum the extrapolated point could be far from the minimum or could turn out to be an estimate of the maximum. Checks for such possibilities should be made. It is felt that a more reasonable scheme would be a combination of the method of Davies, Swann and Campey with quadratic interpolation such that quadratic interpolation is used only when the minimum has been bounded.

5.6.3 Cubic Interpolation Method

The method to be presented could be found useful in one-dimensional searches involved in the Davidon, Fletcher-Powell or Fletcher-Reeves methods (Section 5.8).

Bounding the Minimum

Consider the cubic interpolation of a function as shown in Fig. 5.11.

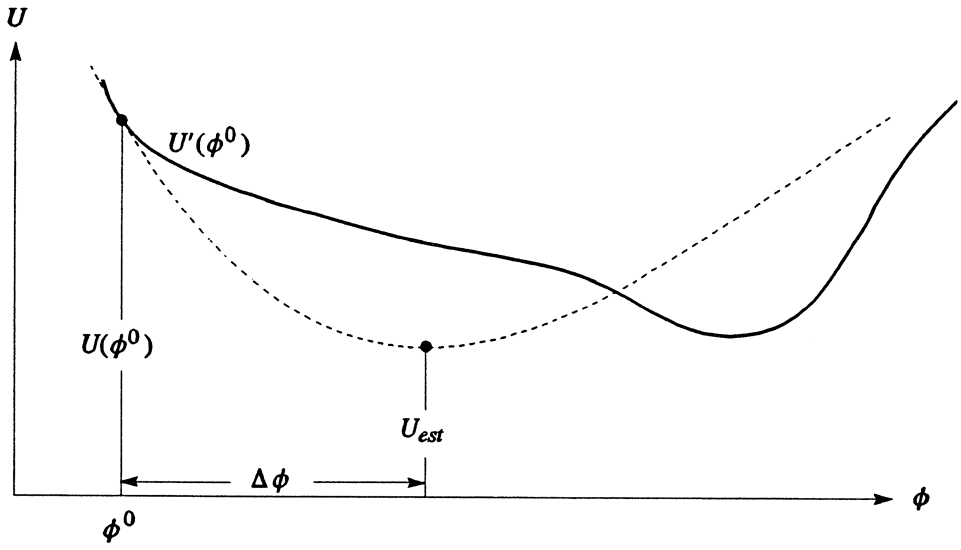

Fig 5.11 Cubic interpolation of a function (sketch).

First we assume that ϕ^0, $U(\phi^0)$ and $U'(\phi^0)$ are available. We also have an estimate of the minimum $U_{est} < U(\phi^0)$. Then

$$\Delta\phi = \frac{-2(U(\phi^0) - U_{est})}{U'(\phi^0)} \tag{5.53}$$

would give the increment in ϕ necessary to reach U_{est}. To limit the size of the step we could have

$$\text{if } |\Delta\phi| > \delta \text{ then } \Delta\phi = \delta s \tag{5.54}$$

where

$$s = -\frac{U'(\phi^0)}{|U'(\phi^0)|} \tag{5.55}$$

and $\delta > 0$. This feature may be necessary when ϕ^0 is already close to the minimum in which case $U'(\phi^0) \approx 0$.

This procedure may be repeated from $\phi^1 = \phi^0 + \Delta\phi$ and continued until for some i $U'(\phi^i)/U'(\phi^{i-1}) < 0$. In this case the minimum has been bounded.

Selection of Points

$$\text{If } U'(\phi^i) > 0 \text{ then } \phi_l^1 = \phi^{i-1},\ \phi_u^1 = \phi^i \tag{5.56}$$

$$\text{If } U'(\phi^i) < 0 \text{ then } \phi_l^1 = \phi^i,\ \phi_u^1 = \phi^{i-1} \tag{5.57}$$

Alternatively, we may prefer to evaluate $U(\phi^i)$ for

$$\phi^i = \phi^0 + 2\phi^{i-1}\Delta\phi,\ \ i = 1, 2, \ldots \tag{5.58}$$

until

$$U(\phi^i) \geq U(\phi^{i-1})$$

$$\text{If } \begin{cases} \Delta\phi > 0 \text{ and } \begin{cases} U'(\phi^{i-1}) > 0 \text{ then } \phi_l^1 = \phi^{i-2},\ \phi_u^1 = \phi^{i-1} \\ U'(\phi^{i-1}) < 0 \text{ then } \phi_l^1 = \phi^{i-1},\ \phi_u^1 = \phi^i \end{cases} \\ \Delta\phi < 0 \text{ and } \begin{cases} U'(\phi^{i-1}) > 0 \text{ then } \phi_l^1 = \phi^i,\ \phi_u^1 = \phi^{i-1} \\ U'(\phi^{i-1}) < 0 \text{ then } \phi_l^1 = \phi^{i-1},\ \phi_u^1 = \phi^{i-2} \end{cases} \end{cases} \tag{5.59}$$

Interpolation Formula

Letting $a = \phi_l^1$ and $b = \phi_u^1$ cubic interpolation between a and b predicts a minimum at

$$c = b - \frac{(b - a)(U'(b) + x - y)}{U'(b) - U'(a) + 2x} \tag{5.60}$$

where

$$y = U'(a) + U'(b) + 3\frac{U(a) - U(b)}{b - a} \tag{5.61}$$

and

$$x = (y^2 - U'(a)\ U'(b))^{1/2} \tag{5.62}$$

If $U(a) < U(c)$ or $U(b) < U(c)$ a further interpolation may be required over $[a, c]$ if $U'(c) > 0$ or over $[c, b]$ if $U'(c) < 0$. The minimum of a quadratic can be found in a single application of the interpolation formula.

5.7 Direct Search Methods

Methods which do not rely explicitly on evaluation or estimation of partial derivatives of the objective function at any point are usually called *direct search* methods. Broadly speaking, they rely on the sequential examination of trial solutions in which each solution is compared with the best obtained up to that time, with a strategy generally based on past experience for deciding where the next trial solution should be located.

Falling into the category of direct search are: random search, when points within a region are selected and investigated at random; one-at-a-time search, when one coordinate direction at a time is investigated; pattern search and similar methods which attempt to align a direction of search along a valley; some quadratically convergent methods; and simplex methods, not to be confused with linear programming. Multidimensional extensions of Fibonacci search have been reported. Interval elimination methods have not nearly been as widely used as some of the *climbing* methods to be discussed.

5.7.1 One-at-a-Time Search

In this method first one parameter is allowed to vary, generally until no further improvement is obtained, and then another one, and so on.

Thus, for a typical iteration

$$\phi^{j+1} = \phi^j + \alpha^j s^j \tag{5.63}$$

where

$$s^j = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ -\dfrac{\partial U^j}{\partial \phi_i} \Big/ \left[\dfrac{\partial U^j}{\partial \phi_i} \right] \\ \vdots \\ 0 \end{bmatrix} \tag{5.64}$$

is the direction of improvement (decreasing U) from ϕ^j along the ϕ_i coordinate, and α^j is a positive scale factor. s^j is usually obtained by trial and error and α^j obtained from a suitable one-dimensional minimization method to find the minimum in the s^j direction. In practice, simple methods using only function values are employed. Fig. 5.12 shows the progress of minimization by a one-at-a-time method. As Fig. 5.12 shows progress will be slow on narrow valleys which are not oriented in the direction of any coordinate axis.

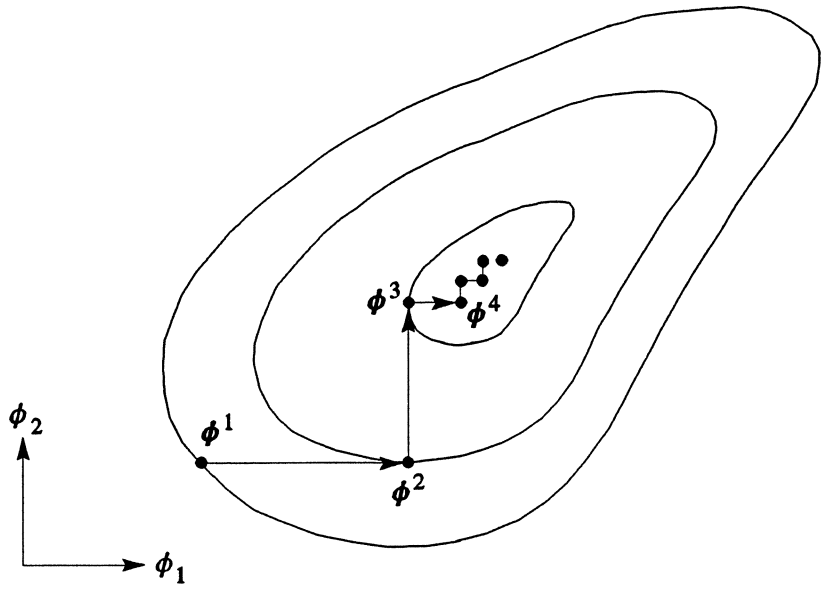

Fig 5.12 Minimization by a one-at-a-time method.

5.7.2 Pattern Search

Pattern Search

The pattern search strategy presented by Hooke and Jeeves [6] is able to follow along fairly narrow valleys because it attempts to align a search direction along the valley.

The tactics employed by pattern search will be explained by means of the example shown in Fig. 5.13.

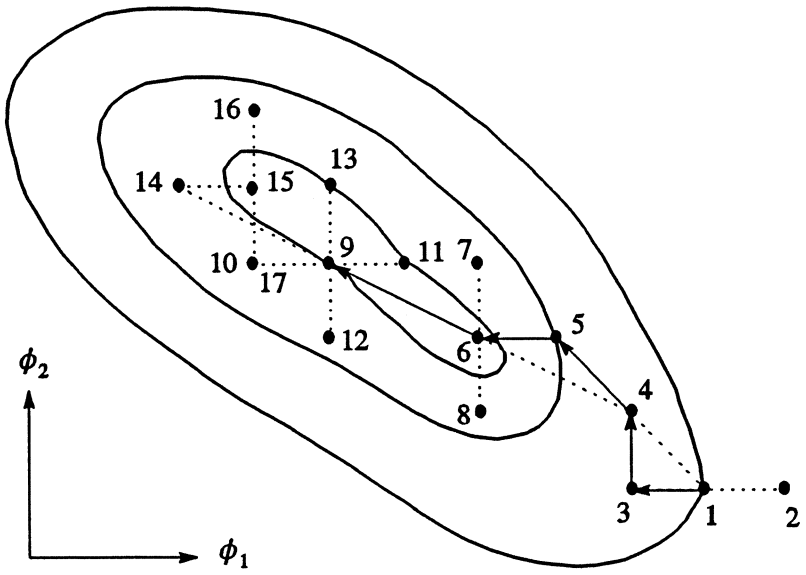

Fig 5.13 Following valleys by pattern search.

The first *base point* b^1 is taken as the starting point ϕ^1. A series of *exploratory moves* from ϕ^1 is initiated to find the second base point. In the example, ϕ_1 is incremented leading us to ϕ^2. Now $U^2 > U^1$ so ϕ^2 is rejected, and ϕ_1 is incremented in the opposite direction to ϕ^3. Exploration with ϕ_1 is over. $U^3 < U^1$ so ϕ^3 is retained and exploration with ϕ_2 begins. $U^4 < U^3$ so ϕ^4 is retained in place of ϕ^3. The first set of exploratory moves is complete, and so ϕ^4 becomes the second base point b^2. In the expectation that our success would be repeated we make a *pattern move* to $\phi^5 = 2b^2 - b^1$, which is in the direction $b^2 - b^1$. By another set of exploratory moves we try to find the most a promising point in the vicinity of ϕ^5. Here, this point is ϕ^6 which becomes the third base point b^3, since $U^6 < U^4$. The search continues with a pattern move in the direction $b^3 - b^2$ to ϕ^9.

The pattern direction is destroyed when a pattern move followed by exploration fails, as around ϕ^{14}. The strategy is to return to the previous base point. If the exploratory moves around the base point fail, as around ϕ^9, the parameter increments are reduced and the procedure is restarted at that point. The search may be terminated either when the parameter increments fall below prescribed levels or the number of function evaluations or running time have reached upper limits.

Razor Search

The *razor search* method of Bandler and Macdonald [7] is a development of pattern search suited to direct optimization in the minimax sense without using derivatives. The name was suggested by the fact that "razor sharp" valleys are, in general, generated by an attempt to minimize functions of the form of (4.67). Paths of discontinuous derivatives are found along the bottom of such valleys, as indicated in Figs. 4.12 and 5.13.

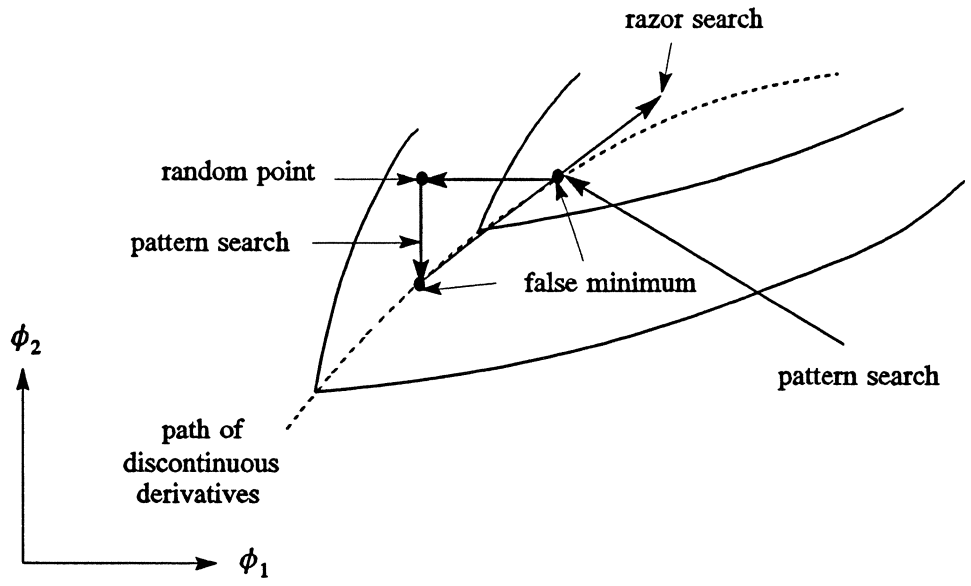

Fig 5.14 Essence of the razor search method.

An investigation of the behaviour of pattern search revealed that pattern search failed when a sharp valley whose contours lay entirely within an orthant of the coordinate axes was encountered. In that case no improvement was possible by searching parallel to these axes.

The razor search method makes a random move from a point where pattern search fails (assuming

a false minimum) and uses pattern search to return to the path of discontinuous derivatives. (See Fig. 5.14.) When pattern search fails again, an attempt is made to establish a pattern in the apparent downhill direction and resume with pattern search. The results shown in Fig. 4.9 were produced by the razor search method [8].

An observation worth making here is that manual optimization in the minimax sense, using an interactive system and employing, say, the one-at-a-time method, can easily terminate at a false minimum. A false minimum in the present context is a point representing a possibly equal-ripple response but which is not a local optimum in the minimax sense.

Pattern Search Details

The mathematical flow diagram for pattern search and for exploratory moves are sketched in Fig. 5.15 and Fig. 5.16, respectively. The variables are defined as follows.

i	subscript
k	dimensionality of space
U	objective function to be minimized
U_{ϕ}	value of U at ϕ
$U_{\phi 0}$	value of U at ϕ^0
α	acceleration factor for pattern move (positive)
β	reduction factor for exploratory increments (positive)
δ	vector containing exploratory increments
δ_i	ith component of δ
ε_i	ith component of vector containing minimum required exploratory increments
ϕ	current, projected or exploratory point
ϕ_i	ith component of ϕ
ϕ^0	current base point
ϕ_i^0	ith component of ϕ^0

5.7.3 Rotating Coordinates

Rosenbrock's Method [9]

Let $u_1^j, u_2^j, \ldots, u_k^j$ be the k mutually orthogonal directions (unit vectors) of search during the jth exploratory stage. For convenience, the given coordinate directions are chosen initially, i.e.,

$$u_1^1 = \begin{bmatrix} \phi_1/|\phi_1| \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad u_2^1 = \begin{bmatrix} 0 \\ \phi_2/|\phi_2| \\ \vdots \\ 0 \end{bmatrix}, \ldots, u_k^1 = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ \phi_k/|\phi_k| \end{bmatrix} \tag{5.65}$$

A step e_i is taken in the u_i^j direction. At the very beginning of the jth exploratory stage, for example, we go to $\phi^j + e_1 u_1^j$. If the move is successful (objective function does not become greater than the current best value) the *success* is noted, the point is retained and a new exploratory increment αe_i, where $\alpha > 1$, is defined for the u_i^j direction. If the move is unsuccessful, the *failure* is noted, we return to the previous point and a new exploratory increment $-\beta e_i$, where $0 < \beta < 1$, is defined for the u_i^j direction.

This process is carried out for all $i = 1, 2, \ldots, k$, and *repeated* if necessary until one success followed by one failure has occurred in each direction. Let the point arrived at finally be ϕ^{j+1}.

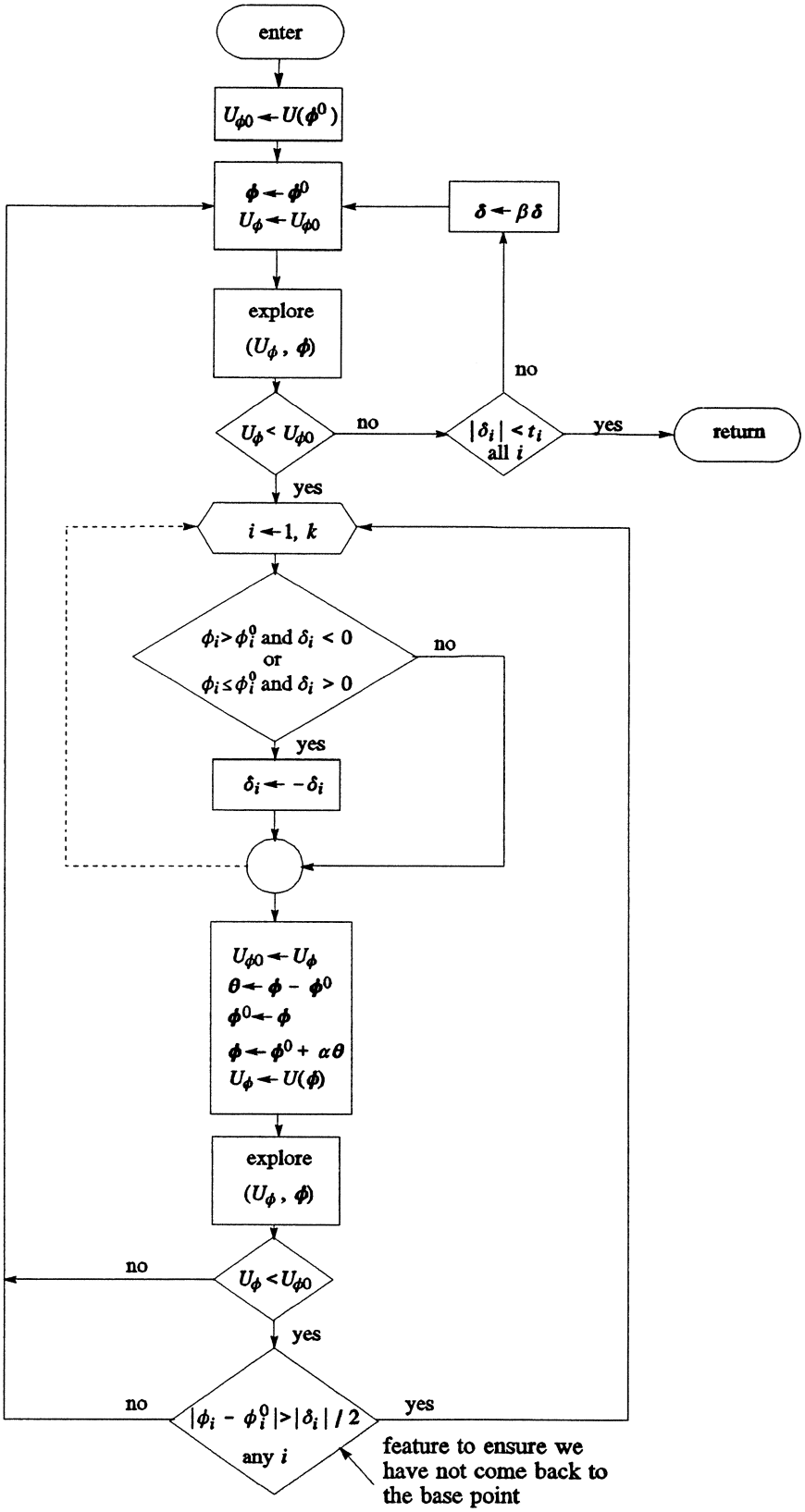

Fig 5.15 Mathematical flow diagram for the pattern search strategy.

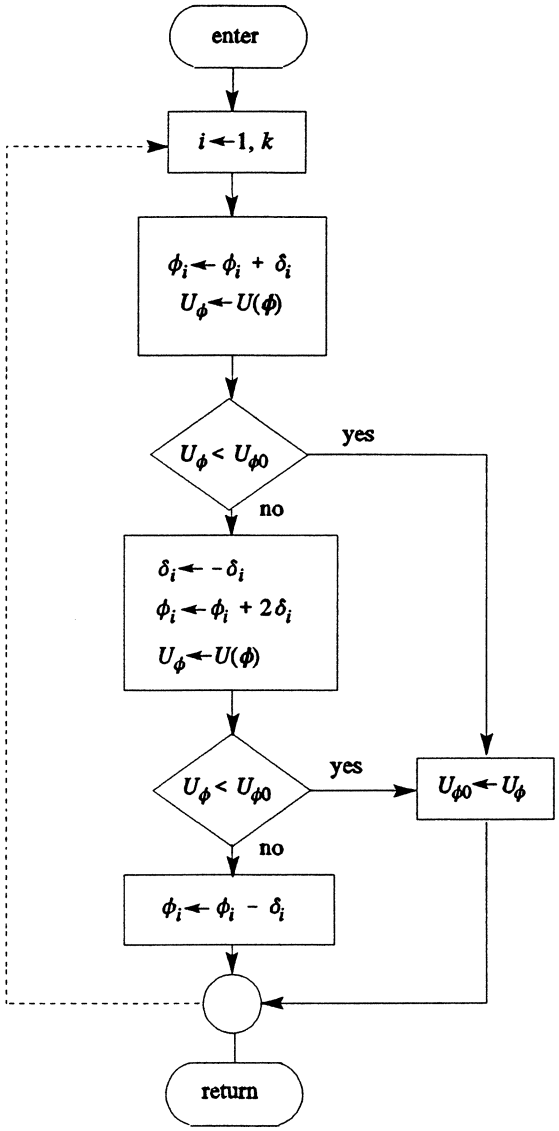

Fig 5.16 Mathematical flow diagram for exploratory moves in pattern search. This is represented as explore ($U_{\phi 0}$, ϕ) in Fig. 5.14.

When the jth exploratory stage is complete, the coordinates are rotated as follows. First we set

$$\begin{aligned} v_k &= d_k\, u_k^j \\ v_i &= d_i\, u_i^j + v_{i+1}, \quad i = k-1, \ldots, 1 \end{aligned} \tag{5.66}$$

where $d_1, d_2, \ldots, d_k$, are the distances moved in the respective directions since the previous rotation of the axes, i.e.,

$$\phi^{j+1} - \phi^j = d_1\, u_1^j + d_2 u_2^j + \ldots + d_k\, u_k^j \tag{5.67}$$

The new set of orthogonal unit vectors $u_1^{j+1}, u_2^{j+1}, \ldots, u_k^{j+1}$ are obtained by the Gram-Schmidt procedure as follows.

$$\left.\begin{aligned} w_1 &= v_1 \\ u_1^{j+1} &= \frac{w_1}{\|w_1\|} \\ w_i &= v_i - \sum_{p=1}^{i-1} (v_i^T\, u_p^{j+1})\, u_p^{j+1} \\ u_i^{j+1} &= \frac{w_i}{\|w_i\|} \end{aligned}\right\} \quad i = 2, 3, \ldots, k \tag{5.68}$$

Observe that the first of these directions always lies in the direction of total progress made during the jth stage since

$$u_1^{j+1} = \frac{\phi^{j+1} - \phi^j}{\|\phi^{j+1} - \phi^j\|} \tag{5.69}$$

Consider the case $k = 2$.

$$\begin{aligned} v_1 &= d_1 u_1^j + d_2 u_2^j \\ v_2 &= d_2 u_2^j \end{aligned}$$

Let

$$d_{n1} = \frac{d_1}{\sqrt{d_1^2 + d_2^2}}, \quad d_{n2} = \frac{d_2}{\sqrt{d_1^2 + d_2^2}}$$

Then

$$w_1 = v_1$$

and

$$u_1^{j+1} = d_{n1} u_1^j + d_{n2} u_2^j$$

Also,

$$\begin{aligned} w_2 &= v_2 - (v_2^T u_1^{j+1}) u_1^{j+1} \\ &= d_2 u_2^j - [d_2 u_2^{jT} (d_{n1} u_1^j + d_{n2} u_2^j)](d_{n1} u_1^j + d_{n2} u_2^j) \\ &= d_2 u_2^j - d_2 d_{n2} (d_{n1} u_1^j + d_{n2} u_2^j) \\ &= \frac{d_1^2 d_2 u_2^j + d_2^3 u_2^j - d_2^2 d_1 u_1^j - d_2^3 u_2^j}{d_1^2 + d_2^2} \\ &= \frac{d_1 d_2}{d_1^2 + d_2^2} (d_1 u_2^j - d_2 u_1^j) \end{aligned}$$

leading to

$$u_2^{j+1} = d_{n1} u_2^j - d_{n2} u_1^j$$

u_1^{j+1} and u_2^{j+1} are clearly orthogonal unit vectors since

$$\| u_1^{j+1} \| = \| u_2^{j+1} \| = 1$$

and

$$u_1^{j+1\,T} u_2^{j+1} = 0$$

A new exploratory stage, the $(j+1)$th, is started at ϕ^{j+1}. The search may be terminated after a predetermined number of function evaluations or when the total progress made during each of several successive exploratory stages becomes smaller than a predetermined value.

If any of the d_i are zero the orthogonalization procedure may break down. Rosenbrock's method avoids this however by ensuring that some success is always achieved in every direction.

Experimentally, Rosenbrock [9] found that $\alpha = 3$, $\beta = 1/2$, gives reasonable efficiency. Fig. 5.17 shows the contours of a famous standard test problem. It is called Rosenbrock's function and it is given by

$$U = 100(\phi_2 - \phi_1^2)^2 + (1 - \phi_1)^2 \tag{5.70}$$

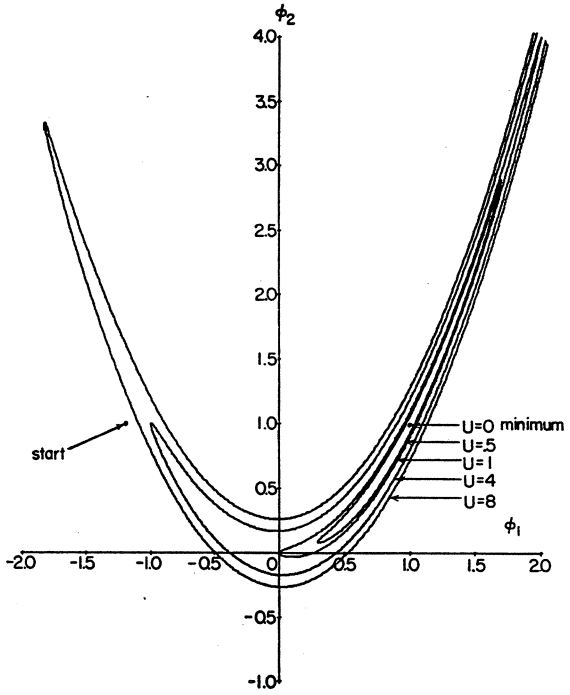

Fig 5.17 Contours of a famous standard test problem: Rosenbrock's function.

The Method of Davies, Swann and Campey [3]

This method was reported by Swann to be more efficient than the methods of Hooke and Jeeves or Rosenbrock. It is essentially an improvement of Rosenbrock's method employing linear minimizations once along each of k mutually orthogonal directions in turn, after which the coordinates are rotated. The progress of the method of Davies, Swann and Campey is shown in Fig. 5.18.

Let ϕ_i^j be the starting point of the ith minimization during the jth stage. Then

$$\phi_{i+1}^j = \phi_i^j + d_i \, u_i^j \tag{5.71}$$

where d_i is selected by the one-dimensional method involving quadratic interpolation suggested by Davies, Swann and Campey (Section 5.6.1). It is recommended that just one quadratic interpolation along each direction may be enough. The process is carried out for all $i = 1, 2, \ldots, k$, with ϕ_1^{j+1} being set equal to ϕ_{k+1}^j. When the jth stage is complete the coordinates can be rotated as already described.

Suppose that after one of the stages one or more of the d's is zero. Consider specifically that $d_r = d_s = 0$. The following procedure can then be adopted.

Reorder the directions

$$u_1^j, u_2^j, \ldots, u_{r-1}^j, u_r^j, u_{r+1}^j, \ldots, u_{s-1}^j, u_s^j, u_{s+1}^j, \ldots, u_k^j$$

to

$$u_1^j, u_2^j, \ldots, u_{r-1}^j, u_{r+1}^j, \ldots, u_{s-1}^j, u_{s+1}^j, \ldots, u_k^j, u_r^j, u_s^j$$

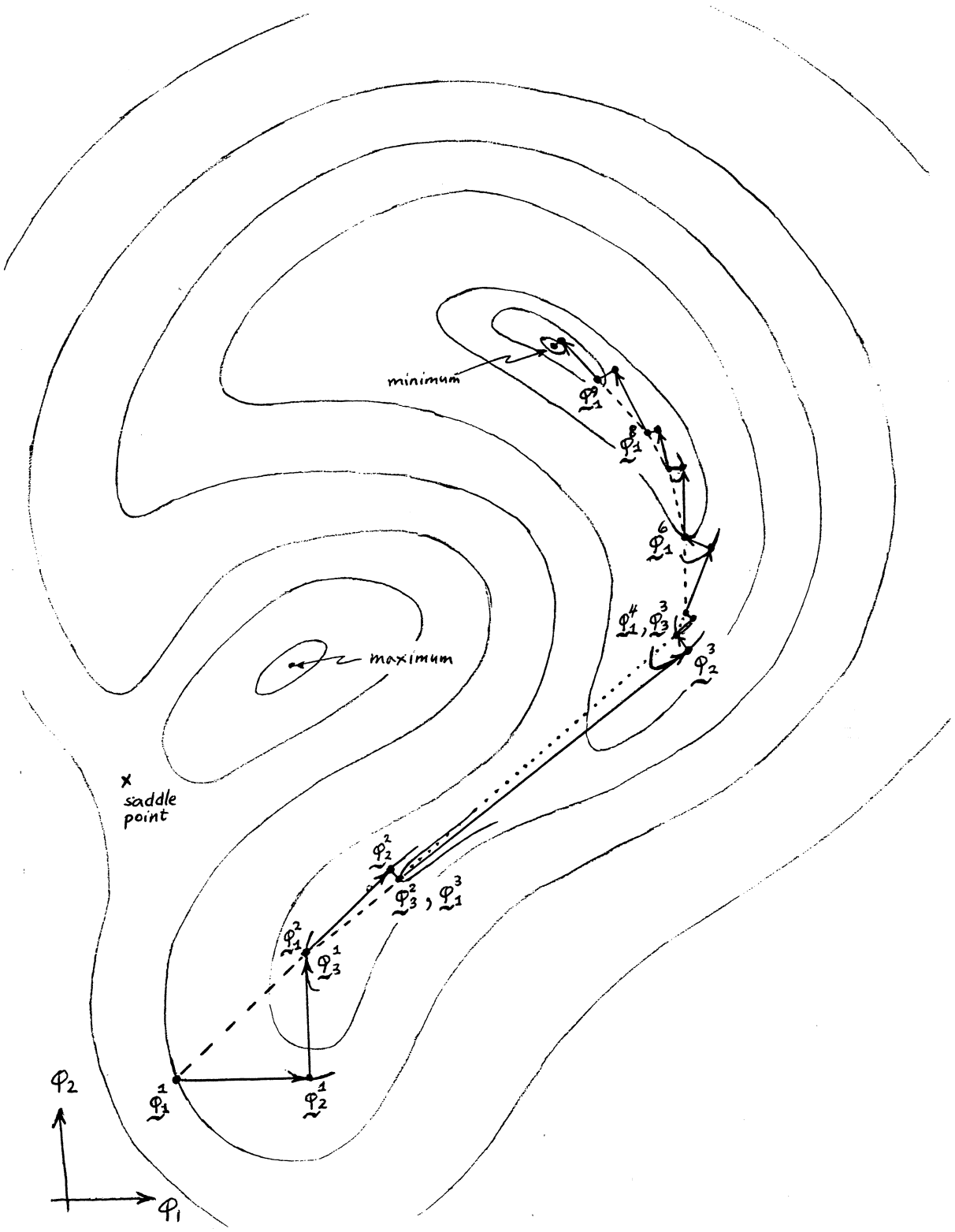

Fig 5.18 Following a valley by the method of Davies, Swann and Campey.

Carry out the orthogonalization on the first $k-2$ directions so that finally we have $k-2$ new orthogonal directions plus two previous directions so that $u_{k-1}^{j+1} = u_r^j$ and $u_k^{j+1} = u_s^j$. Because the first $k-2$ vectors had no components in the directions u_r^j and u_s^j all the directions $u_1^{j+1}, u_2^{j+1}, \ldots, u_k^{j+1}$ are mutually orthogonal.

A suitable way of deciding whether the increment for the one-dimensional search is to be reduced is to compare it with the total distance moved during the previous stage. If the distance moved is felt to be too small the increment is reduced and the one-dimensional search is repeated along the previous k directions. Convergence is assumed when the increment has fallen below a prescribed level.

5.7.4 Simplex Methods

In simplex methods of nonlinear optimization, the objective function is evaluated at the $k+1$ vertices of a *simplex* in a k-dimensional space. In two dimensions, for example, we would have a triangle, for three dimensions a tetrahedron. An attempt is then made to replace the point with the greatest objective function value by another point. The simplex is called *regular* if the points are equidistant.

The objective function is evaluated at each vertex and an attempt to form a new simplex by replacing the vector with the greatest value of the objective function by another point is made. A simplex method was first introduced by Spendley, Hext and Himsworth. A further development of the idea has been presented by Nelder and Mead [10].

The Method of Nelder and Mead [10]

The Nelder and Mead method has very desirable valley-following properties. The basic move is to *reflect* the point having the greatest function value in the centroid of the simplex formed by the remaining points. If the reflected point results in a function value lower than the current lowest, an *expansion* is attempted. Otherwise the point is retained if it results in a function value lower than the second highest. *Contraction* is attempted if reflection fails. Finally, *shrinking* of the simplex about the vertex corresponding to the lowest function value occurs following an unsuccessful attempt at contraction. Some of these moves are illustrated in Fig. 5.19.

Illustration and Definitions

Here we use two illustrations to describe the method of Nelder and Mead. The first is drawn in Fig. 5.20.

Let

$$U_h = U(\phi_h) = \max_i U(\phi_i), \quad i = 1, 2, \ldots, k+1 \tag{5.72}$$

where $\phi_1, \phi_2, \ldots, \phi_{k+1}$ are the vertices of the simplex.

Let

$$U_s = U(\phi_s) = \max_i U(\phi_i), \quad i \neq h \tag{5.73}$$

and

$$U_l = U(\phi_l) = \min_i U(\phi_i) \tag{5.74}$$

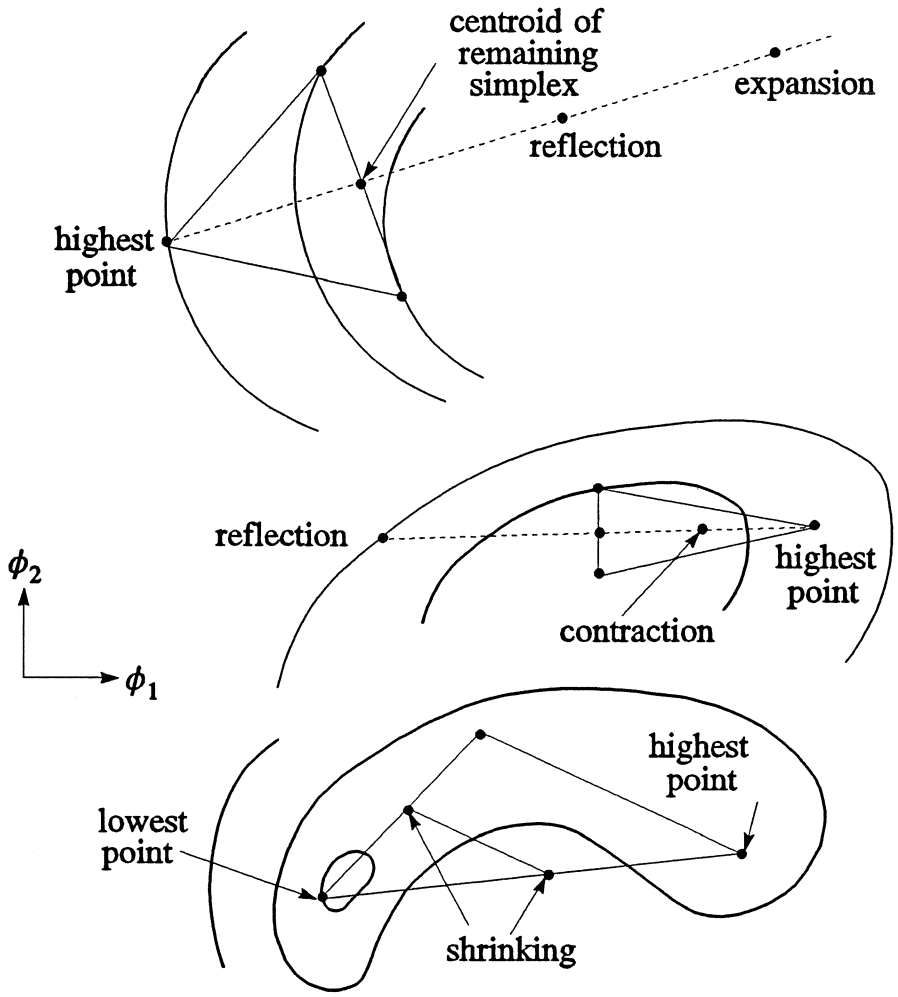

Fig 5.19 Examples of moves made by the simplex method of Nelder and Mead.

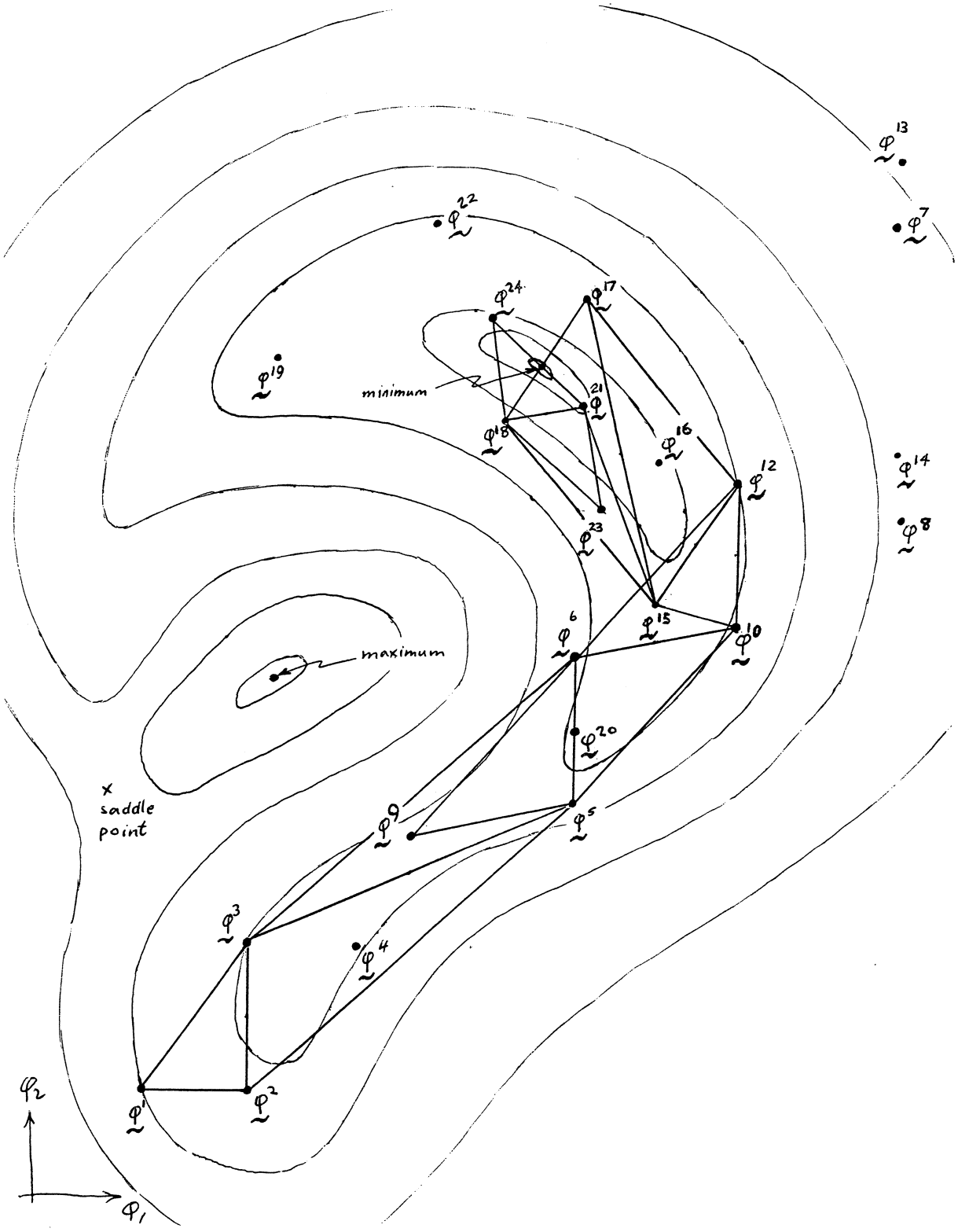

Fig 5.20 Following a valley by the simplex method of Nelder and Mead.

Thus, U_h corresponds to the highest function value, U_s to the second highest and U_l to the lowest function value during the current stage.

The centroid of all points excluding ϕ_h is

$$\bar{\phi} = \frac{1}{k} \sum_{\substack{i=1 \\ i \neq h}}^{k+1} \phi_i \tag{5.75}$$

Consider the example of Fig. 5.20. Clearly

$$\begin{aligned} \phi_h &= \phi^1 \\ \phi_s &= \phi^2 \\ \phi_l &= \phi^3 \\ \bar{\phi} &= 0.5(\phi^2 + \phi^3) \end{aligned} \tag{5.76}$$

An attempt to replace ϕ_h is carried out by defining a *reflection* in $\bar{\phi}$ as

$$\phi_r = \bar{\phi} + \alpha(\bar{\phi} - \phi_h) \tag{5.77}$$

where $\alpha > 0$ is called the *reflection coefficient*. In Fig. 5.20 $\phi_r = \phi^4$. Let $U_r = U(\phi_r)$. If $U_r < U_l$, i.e., if the point just obtained is better than the current best one an *expansion* is attempted, namely,

$$\phi_e = \bar{\phi} + \gamma(\phi_r - \bar{\phi}) \tag{5.78}$$

where $\gamma > 1$ is called the *expansion coefficient*. In Fig. 5.20 $U^4 < U^3$ so we try $\phi_e = \phi^5$. Let $U_e = U(\phi_e)$. If $U_e < U_l$ we replace ϕ_h by ϕ_e and restart the process be redefining the ϕ_h, ϕ_s and ϕ_l. Since $U^5 < U^3$ points ϕ^2, ϕ^3 and ϕ^5 form the new simplex.

The point ϕ^6 is obtained by reflection of ϕ^2 in $0.5(\phi^3 + \phi^5)$. Since $U^6 < U^5$ we expand to ϕ^7. However $U^7 \not< U^5$, so we return to ϕ^6. Thus, if expansion has failed we replace ϕ_h by ϕ_r and restart the process as before. So our third simplex is formed by ϕ^3, ϕ^5 and ϕ^6. Next we try ϕ^8, which is unacceptable. Thus, if $U_r > U_h$, we attempt a *contraction* by defining

$$\phi_c = \bar{\phi} + \beta(\phi_h - \bar{\phi}) \tag{5.79}$$

where $0 < \beta < 1$ is called the *contraction coefficient*. Contraction is deemed successful if $U_c < U_h$, where $U_c = U(\phi_c)$. Note that if $U_h > U_r > U_s$ then we first replace ϕ_h by ϕ_r and redefine ϕ_h accordingly before contracting. In the example ϕ^9 is the result of successful contraction.

If contraction fails, i.e., $U_c \geq U_h$ the following *shrinking* tactic is employed. Set

$$\phi_i \leftarrow 0.5(\phi_i + \phi_l), \quad \text{all } i \neq l \tag{5.80}$$

Following successful contraction or shrinking the process is restarted with the new simplex.

If $U_s \geq U_r \geq U_l$ then ϕ_h is replaced by ϕ_r. This occurs with the simplex formed by ϕ^{18}, ϕ^{21} and ϕ^{23}. Observe that $U^{18} > U^{24} > U^{21}$.

A possible terminating criterion could be

$$\frac{1}{k}\sum_{i=1}^{k+1}(U(\phi_i) - U(\bar{\phi}))^2 < \varepsilon^2 \tag{5.81}$$

where ε is prescribed.

In Fig. 5.20 $\alpha = 1$, $\beta = 1/2$ and $\gamma = 2$.

Some reports state that this method is remarkably efficient for up to four parameters, progress on problems having more dimensions being rather slow. Yet other reports appear much more optimistic. This technique has been incorporated into contemporary commercial software.

Second Illustration

The second example of the simplex strategy is shown in Fig. 5.21. Observe that ϕ^4, ϕ^6, ϕ^8, ϕ^9, and ϕ^{10} have resulted from reflection; ϕ^5 from expansion; and ϕ^7 and ϕ^{11} from contraction. The reader should follow the strategy through carefully to ensure his understanding of it. Its desirable valley-following properties result from its ability to align elongated simplexes in the direction of the valleys. In particular, repeated success, for example, if a long straight valley is being followed, tends to increase the size of the moves, whereas repeated failure, for example, if a bend in the valley is encountered, tends to cause a decrease in the size of the moves.

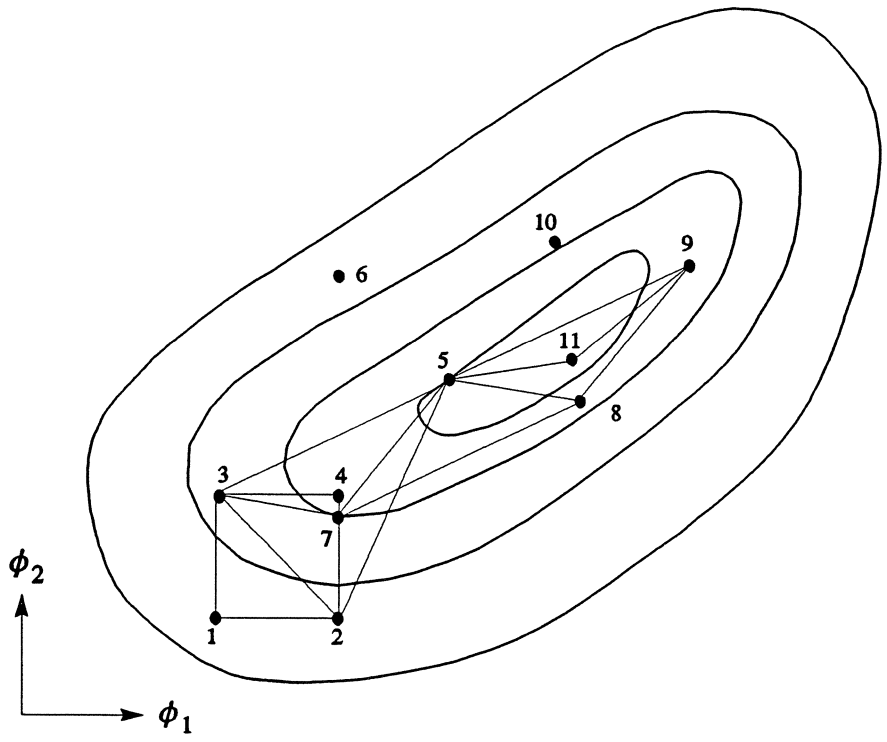

Fig 5.21 Optimization by the simplex method.

It has been claimed to the author on a number of occasions that, unlike some other direct search methods, the simplex method can be successfully employed for minimax approximation. In the author's experience the simplex method is no less infallible than pattern search, for example. The principal fallacy in the argument is the assumption that, if the method requires no derivative information, it can necessarily

handle problems with discontinuous derivatives.

5.8 Gradient Methods of Minimization

We turn our attention now to a class of minimization methods which require derivatives. By and large the most efficient algorithms currently available rely on evaluation of the gradient vector [1-4, 11-13].

5.8.1 Steepest Descent

The jth Iteration

At the jth iteration of most gradient methods, we proceed to

$$\phi^{j+1} = \phi^j + \alpha^j s^j \tag{5.82}$$

where s^j is (hopefully) a downhill direction of search and $\alpha^j > 0$ is a scale factor chosen to minimize $U(\phi^j + \alpha^j s^j)$ in the direction of search. One-dimensional minimization methods suitable for this purpose are discussed in Section 5.7.

Steepest Descent Direction

The most obvious choice for s^j is the *steepest descent* direction at ϕ^j, defined as follows. Referring back to (4.18), we note that a first-order change in the objective function is given by

$$\Delta U = \nabla U^T \Delta\phi \tag{5.83}$$

If $\Delta\phi = \alpha s$, where $\alpha > 0$ is fixed and $\|s\| = 1$, then it is easy to show that the s minimizing ΔU is

$$s = -\frac{\nabla U}{\|\nabla U\|} \tag{5.84}$$

The s in (5.84) is the negative of the *normalized gradient* vector. Although $-\nabla U/\|\nabla U\|$ provides the greatest local change, success of the steepest descent method is highly dependent on scaling. As Fig. 5.22 shows, the first few iterations may give good reduction in U, but subsequently the method usually deteriorates rapidly into oscillations, and progress becomes very slow.

5.8.2 The Newton Method

The Newton Step

This method was already mentioned in Section 4.2 in the context of solution of nonlinear equations. Differentiating the Taylor series (4.18)

$$\nabla U(\phi + \Delta\phi) = \nabla U(\phi) + H\,\Delta\phi + \ldots \tag{5.85}$$

For $\phi + \Delta\phi$ to be the minimizing point $\check{\phi}$,

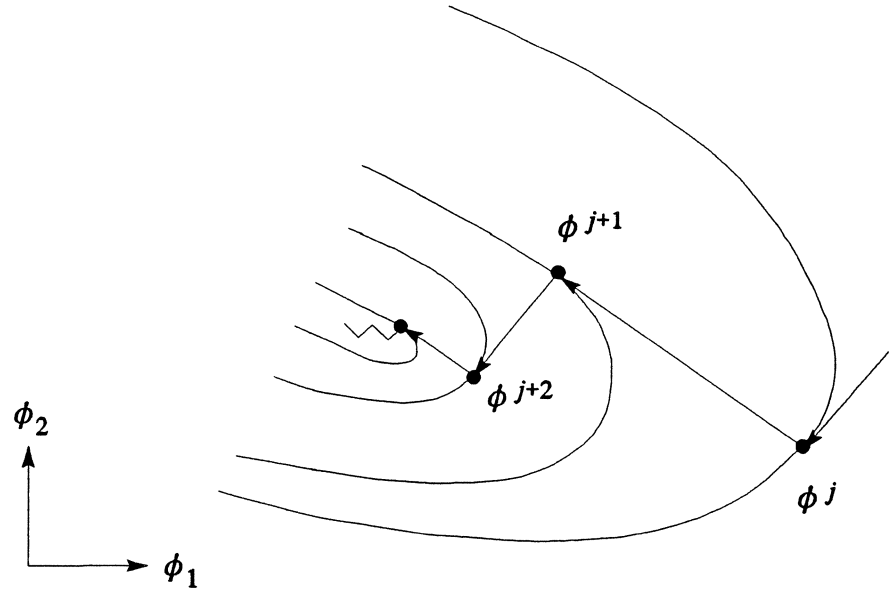

Fig 5.22 A steepest-descent strategy.

$$\nabla U(\phi + \Delta\phi) = \mathbf{0} \tag{5.86}$$

so that, neglecting higher-order terms,

$$\Delta\phi = -H^{-1} \nabla U \tag{5.87}$$

This incremental change takes us to the minimum in only one iteration if we are dealing with a positive definite quadratic function (Section 4.2). It is instructive to compare (5.87) and (4.26).

Downhill Property

To show that the Newton step is downhill consider (5.83). Using $\Delta\phi$ from (5.87) we have

$$\Delta U = -\nabla U^T H^{-1} \nabla U = -\delta \tag{5.88}$$

If $\boldsymbol{H}$ is positive definite so is $\boldsymbol{H}^{-1}$, which makes $\delta > 0$ and $\Delta U < 0$. The change in U being negative designates the direction (5.87) as downhill.

Iterative Schemes

When U is not quadratic, we could try the iterative scheme

$$\phi^{j+1} = \phi^j - H^{-1} \nabla U^j \tag{5.89}$$

where $\boldsymbol{H}^{-1}$ is the inverse of the Hessian matrix at the jth iteration. This scheme has, however, several disadvantages. $\boldsymbol{H}$ must be positive definite otherwise divergence could occur. In particular, $-\boldsymbol{H}^{-1}\nabla U^j$ might not point downhill. To counteract these possibilities, the modification

$$\phi^{j+1} = \phi^j - \alpha^j H^{-1} \nabla U^j \tag{5.90}$$

can be employed where α^j is chosen to minimize U^{j+1} in the direction $-H^{-1}\nabla U^j$. This might also be ineffective: α^j may have to be negative; H may be locally singular. Finally, the computation of H and its inverse are time consuming.

5.8.3 Conjugate Directions

Certain gradient methods which exploit the properties of *conjugate directions* associated with quadratic functions and do not explicitly evaluate Hessian H or its inverse are highly effective. Before discussing them let us define conjugate directions.

The directions u_i and u_j are said to be conjugate with respect to a positive definite matrix A if

$$u_i^T A u_j = 0, \quad i \neq j \tag{5.91}$$

In Fig. 5.23, a two-dimensional interpretation of conjugate directions is given. Methods which generate such directions will minimize a quadratic function in a finite number of iterations. It is evident that one linear minimization along each direction in turn locates the minimum.

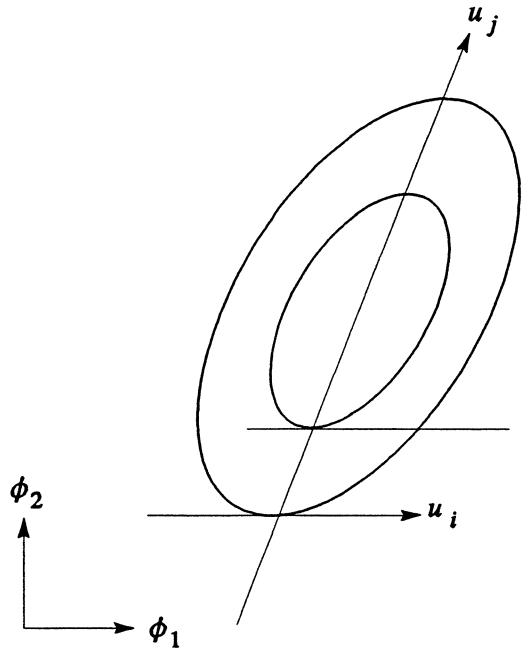

Fig 5.23 An illustration of a pair of conjugate directions.

Three well-known methods which use conjugate directions are the *conjugate gradient* method described by Fletcher and Reeves [13], the Fletcher-Powell-Davidon method [4], and the Powell-Zangwill method [5,14] which does not require derivatives (See also references [2,11]).

5.8.4 The Conjugate Gradient Method

We start the conjugate gradient method at ϕ^0. The direction of search s^j is given by [13]

$$s^0 = -\nabla U^0 \tag{5.92}$$

$$s^j = -\nabla U^j + \beta^j s^{j-1}, \quad j > 0 \tag{5.93}$$

where

$$\beta^j = \frac{(\nabla U^j)^T \nabla U^j}{(\nabla U^{j-1})^T \nabla U^{j-1}} \tag{5.94}$$

Thus the first iteration is in the direction of steepest descent. Apart from round-off errors, the procedure will terminate at the minimum of a quadratic in at most k iterations. In general, however, it is recommended that $k + 1$ iterations be completed before restarting the procedure.

5.8.5 The Fletcher-Powell-Davidon Method

The Updating Formula

Redefining H^j is any (jth) positive definite matrix, we have the direction of search [4]

$$s^j = -H^j \nabla U^j \tag{5.95}$$

Note that H^j is the jth approximation to the *inverse* of the Hessian matrix. We start the process at ϕ^0. Initially,

$$H^0 = 1 \tag{5.96}$$

and again we have the steepest descent direction.

H^j is continually updated using first derivative information such that

$$\phi^{j+1} - \phi^j = H^{j+1} g^j \tag{5.97}$$

is preserved, where

$$g^j \triangleq \nabla U^{j+1} - \nabla U^j \tag{5.98}$$

The following updating procedure is used.

$$H^{j+1} = H^j + \frac{\Delta\phi^j \, \Delta\phi^{jT}}{\Delta\phi^{jT} g^j} - \frac{H^j g^j g^{jT} H^j}{g^{jT} H^j g^j} \tag{5.99}$$

where

$$\Delta\phi^j = \alpha^j s^j \tag{5.100}$$

and α^j is found by a one-dimensional search (Section 5.7).

Positive Definiteness of H^j

Fletcher and Powell prove by induction that if H^j is positive definite then H^{j+1} is also positive definite. H^0, being the unit matrix, is clearly positive definite.

Quadratic Functions

On a quadratic function it is further proved that H^k is the inverse of the Hessian matrix, i.e.,

$$H^k = H^{-1} \tag{5.101}$$

and

$$\nabla U^k = 0 \tag{5.102}$$

apart from round-off errors.

Comments

Both the proof of convergence and success in practice depend on accurate location of the minimum in the linear searches. If necessary, H^j may be reset to the unit matrix.

This method is generally acknowledged as the basis for the best general purpose gradient optimizers.

5.8.6 Least *p*th Gradient

For objective functions in the form of (4.39) and (4.40) we can write

$$\nabla U = \int_{\psi_l}^{\psi_u} Re\{p\,|e(\phi,\psi)|^{p-2} e^*(\phi,\psi)\,\nabla e(\phi,\psi)\}\,d\psi \tag{5.103}$$

for the continuous case and

$$\nabla U = \sum_{i \in I} Re\{p\,|e_i(\phi)|^{p-2} e_i^*(\phi) \nabla e_i(\phi)\} \tag{5.104}$$

for the discrete case. If the appropriate derivatives, namely ∇e, are available, we could proceed to optimize with a suitable gradient method of the kind already described.

In more complicated situations we can envisage a linear combination of functions in the form (4.39) and (4.40), for example,

$$U = \alpha_1 U_1 + \alpha_2 U_2 + \ldots \tag{5.105}$$

Simultaneous approximation of more than one response specification might be posed in this way (See

Chapter 7). The factors α_1, α_2, etc., would be given values commensurate with the importance of U_1, U_2, etc.

5.8.7 Gauss-Newton Least *p*th Method

The Objective Function

Temes and Zai [15] have extended the well-known least squares method of Gauss [1,2,16] to a *least pth method.* Since the former method falls out as a special case, the latter method will be briefly described. For definiteness, assume the objective function is of the form (with real $e_i(\phi)$)

$$U = \sum_{i=1}^{n} [e_i(\phi)]^p \tag{5.106}$$

where $n > k$ and p is any positive even integer.

The Gradient Vector

Following (5.104),

$$\nabla U = \sum_{i=1}^{n} p e_i^{p-1} \nabla e_i \tag{5.107}$$

The Hessian Matrix

Applying the ∇ operator to (5.107) we have

$$\boldsymbol{H} = \nabla(\nabla U)^T = \sum_{i=1}^{n} [p e_i^{p-1} \nabla(\nabla e_i)^T + p(p-1) e_i^{p-2} \nabla e_i (\nabla e_i)^T] \tag{5.108}$$

Linearization

Now assume that the first term may be neglected in comparison with the second. this really corresponds to a linearization of $e_i(\phi)$. Then

$$\boldsymbol{H} \approx \sum_{i=1}^{n} p(p-1) e_i^{p-2} \nabla e_i (\nabla e_i)^T \tag{5.109}$$

This can be rewritten as

$$\boldsymbol{H} \approx p(p-1) \boldsymbol{A}^T \boldsymbol{B} \boldsymbol{A} \tag{5.110}$$

where

$$\boldsymbol{A} \triangleq [\nabla e_1 \ \nabla e_2 \ \ldots \ \nabla e_n]^T$$

and

$$B \triangleq \begin{bmatrix} e_1^{p-2} & 0 & \cdots & 0 \\ 0 & e_2^{p-2} & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & e_n^{p-2} \end{bmatrix}$$

The Newton Step

Letting

$$\varepsilon \triangleq [e_1^{p-1} \quad e_2^{p-2} \quad \cdots \quad e_n^{p-1}]^T$$

(5.107) becomes

$$\nabla U = pA^T \varepsilon \tag{5.111}$$

Using the step given by the Newton method (5.87),

$$\Delta\phi = -H^{-1}\nabla U = -(p-1)^{-1} (A^T BA)^{-1} A^T \varepsilon \tag{5.112}$$

If $p = 2$ we have the standard least squares formulation. In this case $B = 1$ and $H = 2A^TA$. This is exact if the e_i are linear and $p = 2$.

Under suitable conditions, it can be shown that $\Delta\phi$ points in the downhill direction. The modified Newton procedure

$$\phi^{j+1} = \phi^j - \alpha^j (p-1)^{-1} (A^T BA)^{-1} A^T \varepsilon \tag{5.113}$$

is recommended where α^j is chosen to minimize U^{j+1}.

Damping

Damping techniques similar to those used in the Gauss method are applicable [2,16]. Define, for example,

$$D(\phi, \lambda) = U(\phi) + \lambda \Delta\phi^T \Delta\phi \tag{5.114}$$

Then the Hessian of D is

$$\bar{H} \approx H + 2\lambda 1_k \tag{5.115}$$

and

$$\Delta\phi = -p[p(p-1)A^T BA + 2\lambda 1_k]^{-1} A^T \varepsilon \tag{5.116}$$

It may be shown that the convergence and downhill properties are preserved and that for $\lambda > 0$ the step is no larger than the undamped step. As $\lambda \to 0$ the process is undamped, while for $\lambda \to \infty$ the step is in the steepest descent direction. The introduction of α^j to permit a linear search as in (5.113) is also possible.

5.8.8 Example

An example of least *p*th approximation [17] compared with minimax approximation is depicted in Fig. 5.24. The structure is the seven-section cascade of transmission lines acting as a filter discussed in Section 4.4.4. The problem here was to see how small the passband insertion loss could be made under the constraints of the problem (if R_g and R_L were frequency independent, or the lengths were allowed to vary, the answer would be trivial).

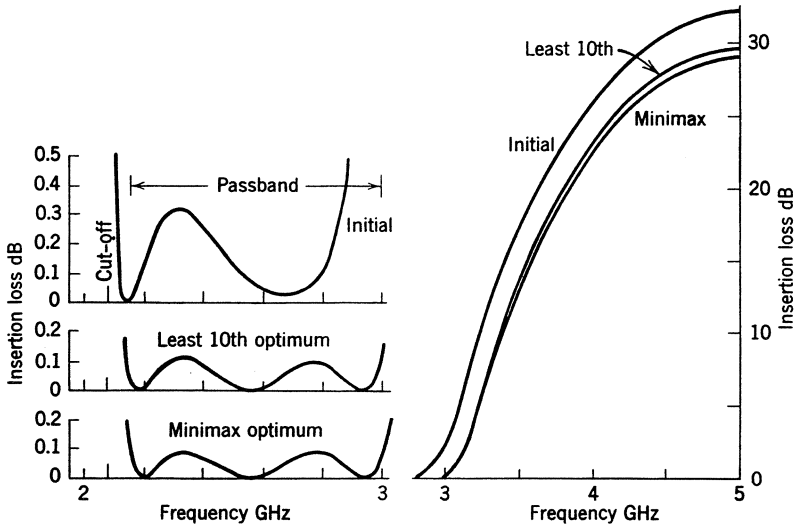

Fig 5.24 Example of least 10th approximation compared with minimax approximation in optimizing the passband of the filter of Fig. 4.10.

A least *p*th objective function was set up with $p = 10$, using 51 uniformly spaced points in the passband. The objective function was of the form

$$U = \sum_{i=1}^{n} \frac{1}{p} \, |\rho_i(\phi)|^p \tag{5.117}$$

The Fletcher-Powell-Davidon method (Section 5.8.5) was used, the required first derivatives being obtained from *one* network analysis using the adjoint network method (Chapter 7). Compare the almost equal-ripple passband response obtained with a maximum insertion loss of about 0.1 dB with the equal-ripple response (maximum insertion loss 0.086 dB) produced by minimax approximation. The latter solution was obtained by Bandler and Lee-Chan [18] using a gradient algorithm with quadratic interpolation used to locate the ripple extrema.

The main conclusion to be reached from this example is that acceptable results can be achieved

with relatively moderate values of p. Unless special precautions are taken to avoid ill-conditioning, the use of values of p much greater than 10 is discouraged.

5.9 References

[1] J.W. Bandler, "Optimization methods for computer-aided design," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-17, 1969, pp. 533-552.

[2] J. Kowalik and M.R. Osborne, *Methods for Unconstrained Optimization Problems*. New York: Elsevier, 1968.

[3] M.J. Box, D. Davies and W.H. Swann, *Non-linear Optimization Techniques*. Edinburgh: Oliver and Boyd, 1969.

[4] R. Fletcher and M.J.D. Powell, "A rapidly convergent descent method for minimization," *Computer J.*, vol. 6, 1963, pp. 163-168.

[5] M.J.D. Powell, "An efficient method for finding the minimum of a function of several variables without calculating derivatives," *Computer J.*, vol. 7, 1964, pp. 155-162.

[6] R. Hooke and T.A. Jeeves, "'Direct search' solution of numerical and statistical problems," *J. ACM*, vol. 8, 1961, pp. 212-229.

[7] J.W. Bandler and P.A. Macdonald, "Optimization of microwave networks by razor search," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-17, 1969, pp. 552-562.

[8] J.W. Bandler, "Computer optimization of inhomogeneous waveguide transformers," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-17, 1969, pp. 563-571.

[9] H.H. Rosenbroek, "An automatic method for function minimization," *Computer J.*, vol. 3, 1960, pp. 175-184.

[10] J.A. Nelder and R. Mead, "A simplex method for function minimization," *Computer J.*, vol. 7, 1965, pp. 308-313.

[11] R. Fletcher, "A review of methods for unconstrained optimization," in *Optimization*, R. Fletcher Ed. New York: Academic Press, 1969.

[12] M.J. Box, "A comparison of several current optimization methods, and the use of transformations in constrained problems," *Computer J.*, vol. 9, 1966, pp. 67-77.

[13] R. Fletcher and C.M. Reeves, "Function minimization by conjugate gradients," *Computer J.*, vol. 7, 1964. pp. 149-154.

[14] W.I. Zangwill, "Minimizing a function without calculating derivatives," *Computer J.*, vol. 10, 1967, pp. 293-296.

[15] G.C. Temes and D.Y.F. Zai, "Least pth approximation," *IEEE Trans. Circuit Theory*, vol. CT-16, 1969, pp. 235-237.

[16] G.C. Temes and D.A. Calahan, "Computer-aided network optimization the state of the art," *Proc. IEEE*, vol. 55, 1967, pp. 1832-1863.

[17] J.W. Bandler and R.E. Seviora, "Current trends in network optimization," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-18, 1970, pp. 1159-1170.

[18] J.W. Bandler and A.G. Lee-Chan, "Gradient razor search method for optimization," *IEEE Int. Microwave Symp. Dig.* (1971), pp. 118-119.

Chapter 6

SIMULATION

In this chapter we turn our attention to some fundamental topics in the solution of linear and nonlinear systems. From the outset we take into account the requirement of sensitivity analysis. In particular, we focus upon LU decomposition and related matters: solution of the adjoint system, matrix inversion, the use of superposition, symmetry and the implication of tridiagonal systems. We consider a relationship between the first-order and large-change sensitivities w.r.t. an element of a matrix. Iterative improvement of a solution to a set of linear equations is discussed. We provide a detailed interpretation of derivative evaluation in terms of linear circuits. Finally we consider the solution of linear equations through singular value decomposition.

6.1 Nonlinear Simulation

6.1.1 Basic Problem

Consider the function

$$f(y(x),\, x) \tag{6.1}$$

where

$$x \triangleq \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_k \end{bmatrix}, \qquad y \triangleq \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \tag{6.2}$$

subject to

$$h(x,y) = 0 \tag{6.3}$$

where

$$\boldsymbol{h} \triangleq \begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_n \end{bmatrix} \tag{6.4}$$

given $\boldsymbol{x}$. Here, f is some response function of the nonlinear system. We wish to calculate

$$\frac{\partial f}{\partial \boldsymbol{x}} \text{ s.t. } \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{0} \tag{6.5}$$

where

$$\frac{\partial f}{\partial \boldsymbol{x}} \triangleq \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_k} \end{bmatrix} \tag{6.6}$$

That is, we want to obtain the gradient vector of f w.r.t. $\boldsymbol{x}$ subject to the equality constraints defining the nonlinear system, which is also to be solved.

6.1.2 Solution of Nonlinear Equations

Solution of the nonlinear equations can be carried out by linearization (Newton's method) as follows. At the jth iteration,

$$\boldsymbol{h}^{j+1} = \boldsymbol{h}^j + \left[\frac{\partial \boldsymbol{h}^T}{\partial \boldsymbol{y}}\right]^T \Bigg|_j (\boldsymbol{y}^{j+1} - \boldsymbol{y}^j) + \ldots \tag{6.7}$$

where

$$\boldsymbol{h}^j \triangleq \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{y}^j) \tag{6.8}$$

and

$$\frac{\partial \boldsymbol{h}^T}{\partial \boldsymbol{y}} \triangleq \left[\frac{\partial h_1}{\partial \boldsymbol{y}} \quad \frac{\partial h_2}{\partial \boldsymbol{y}} \quad \cdots \quad \frac{\partial h_n}{\partial \boldsymbol{y}}\right] = \begin{bmatrix} \frac{\partial h_1}{\partial y_1} & \frac{\partial h_2}{\partial y_1} & \cdots & \frac{\partial h_n}{\partial y_1} \\ \frac{\partial h_1}{\partial y_2} & \frac{\partial h_2}{\partial y_2} & \cdots & \frac{\partial h_n}{\partial y_2} \\ \vdots & \vdots & & \vdots \\ \frac{\partial h_1}{\partial y_n} & \frac{\partial h_2}{\partial y_n} & \cdots & \frac{\partial h_n}{\partial y_n} \end{bmatrix} \tag{6.9}$$

Set $\boldsymbol{h}^{j+1} = \boldsymbol{0}$ to obtain the linear system

$$\left[\frac{\partial \boldsymbol{h}^T}{\partial \boldsymbol{y}}\right]^T \Bigg|_j \boldsymbol{y}^{j+1} = \left[\frac{\partial \boldsymbol{h}^T}{\partial \boldsymbol{y}}\right]^T \Bigg|_j \boldsymbol{y}^j - \boldsymbol{h}^j = \boldsymbol{e}^j \tag{6.10}$$

to be solved for $\boldsymbol{y}^{j+1}$. A byproduct of this process is the evaluation of the Jacobian

$$\boldsymbol{J}^T \triangleq \frac{\partial \boldsymbol{h}^T}{\partial \boldsymbol{y}} \tag{6.11}$$

in addition to the solution, which will be denoted $\boldsymbol{y}$.

6.1.3 The Adjoint System

Now

$$\frac{\partial f}{\partial \boldsymbol{x}}\Bigg|_{\boldsymbol{h}=\boldsymbol{0}} = \frac{\partial \boldsymbol{y}^T}{\partial \boldsymbol{x}} \frac{\partial f}{\partial \boldsymbol{y}} + \frac{\partial f}{\partial \boldsymbol{x}} \tag{6.12}$$

where $\partial \mathbf{y}^T/\partial \mathbf{x}$ is defined by analogy with $\partial \boldsymbol{h}^T/\partial \boldsymbol{y}$. Furthermore,

$$\frac{\partial \boldsymbol{h}^T}{\partial \boldsymbol{x}}\Bigg|_{\boldsymbol{h}=\boldsymbol{0}} = \frac{\partial \boldsymbol{y}^T}{\partial \boldsymbol{x}} \frac{\partial \boldsymbol{h}^T}{\partial \boldsymbol{y}} + \frac{\partial \boldsymbol{h}^T}{\partial \boldsymbol{x}} = \boldsymbol{0} \tag{6.13}$$

Substituting for $\partial \boldsymbol{y}^T/\partial \boldsymbol{x}$ we have

$$\frac{\partial f}{\partial \boldsymbol{x}}\Bigg|_{\boldsymbol{h}=\boldsymbol{0}} = -\frac{\partial \boldsymbol{h}^T}{\partial \boldsymbol{x}} \left[\frac{\partial \boldsymbol{h}^T}{\partial \boldsymbol{y}}\right]^{-1} \frac{\partial f}{\partial \boldsymbol{y}} + \frac{\partial f}{\partial \boldsymbol{x}} = -\frac{\partial \boldsymbol{h}^T}{\partial \boldsymbol{x}} \hat{\boldsymbol{y}} + \frac{\partial f}{\partial \boldsymbol{x}} \tag{6.14}$$

where $\hat{\boldsymbol{y}}$ is the solution to the adjoint linear system

$$\left[\frac{\partial h^T}{\partial y}\right] \hat{y} = \frac{\partial f}{\partial y} \tag{6.15}$$

6.1.4 Conclusions

The Newton method is summarized by

$$J^j y^{j+1} = J^j y^j - h^j \tag{6.16}$$

and the adjoint system is described by

$$J^T \hat{y} = \frac{\partial f}{\partial y} \tag{6.17}$$

where we assume that

$$y^j \to y \quad \text{and} \quad J^j \to J \quad \text{as} \quad h^j \to 0 \tag{6.18}$$

The matrix of coefficients of the adjoint system is the transpose of the matrix used in the solution of the nonlinear system by linearization. Hence, LU factors can be reused for sensitivity evaluation.

6.2 Nonlinear Network Simulation

6.2.1 Companion Network Model

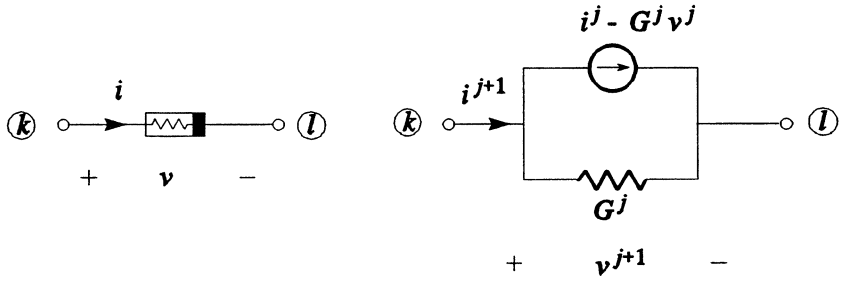

Fig. 6.1 A nonlinear conductor and its companion network.

Linearizing the nonlinear characteristic

$$i = i(v) \tag{6.19}$$

at the jth iteration gives

$$i^{j+1} = i^j + \left[\frac{di}{dv}\right]^j (\delta v)^j \tag{6.20}$$

where the incremental change $(\delta v)^j$ is given by

$$(\delta v)^j = v^{j+1} - v^j \tag{6.21}$$

Let

$$G^j = \left[\frac{di}{dv}\right]^j \tag{6.22}$$

and rewrite (6.20) as

$$i^{j+1} = G^j v^{j+1} + (i^j - G^j v^j) \tag{6.23}$$

Then we can treat G^j as any conductance at the jth iteration and $(i^j - G^j v^j)$ as any independent current source at the jth iteration. The nodal equations at the jth iteration are of the form

$$\begin{bmatrix} \text{assemble linear} \\ \text{(constant) and non-} \\ \text{linear conductances} \\ \text{in the usual way} \end{bmatrix}^j \begin{bmatrix} v_1^{j+1} \\ v_2^{j+1} \\ v_3^{j+1} \\ \vdots \end{bmatrix} = \begin{bmatrix} \text{assemble} \\ \text{independent} \\ \text{current} \\ \text{sources} \end{bmatrix}^j \tag{6.24}$$

Solve these linearized node equations at the jth iteration to yield node voltages at the $(j+1)$th iteration. That is, at the jth iteration

$$[i^j, v^j] \Rightarrow \begin{bmatrix} \text{linearize all elements,} \\ \text{assemble nodal} \\ \text{matrix and solve} \\ \text{nodal equation at} \\ \text{the } j\text{th iteration} \end{bmatrix} \Rightarrow [i^{j+1}, v^{j+1}] \tag{6.25}$$

For a nonlinear element connected between nodes k and l, place

$$(u_k - u_l)G^j(u_k - u_l)^T \tag{6.26}$$

into the matrix and place

$$-(u_k - u_l)(i^j - G^j v^j) \tag{6.27}$$

or

$$(i^j - G^j v^j)u_l - (i^j - G^j v^j)u_k \tag{6.28}$$

on the right hand side, where u_k and u_l are unit vectors.

6.3 Linear Simulation

6.3.1 Basic Problem

Consider the function

$$f(y(x),\ x) \tag{6.29}$$

subject to

$$A(x)\ y\ =\ b(x) \tag{6.30}$$

given x. Here, f is some response function of the linear system. We wish to calculate

$$\frac{\partial f}{\partial x} \text{ s.t. } A(x)\ y\ =\ b(x) \tag{6.31}$$

That is, we want to simulate the linear system of (6.30) followed by calculation of the partial derivatives of f w.r.t. x.

Association with Nonlinear Equation Formulation

Here,

$$h\ \triangleq\ A(x)\ y\ -\ b(x) \tag{6.32}$$

hence, for

$$A\ \triangleq \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \tag{6.33}$$

we have

$$\begin{aligned}
\frac{\partial h_1}{\partial y} &= [a_{11} \;\; a_{12} \;\; \ldots \;\; a_{1n}]^T \\
\frac{\partial h_2}{\partial y} &= [a_{21} \;\; a_{22} \;\; \ldots \;\; a_{2n}]^T \\
&\vdots \\
\frac{\partial h_n}{\partial y} &= [a_{n1} \;\; a_{n2} \;\; \ldots \;\; a_{nn}]^T
\end{aligned} \tag{6.34}$$

Note that all coefficients may be functions of $\boldsymbol{x}$.

Summarizing,

$$\frac{\partial \boldsymbol{h}^T}{\partial y} = \left[\frac{\partial h_1}{\partial y} \;\; \frac{\partial h_2}{\partial y} \;\; \ldots \;\; \frac{\partial h_n}{\partial y}\right] = \boldsymbol{A}^T \tag{6.35}$$

Adjoint System (Network)

We have the linear system in $\hat{\boldsymbol{y}}$

$$\boldsymbol{A}^T \hat{\boldsymbol{y}} = \frac{\partial f}{\partial y} \tag{6.36}$$

where $\partial f/\partial y$ must be readily available.

6.3.2 Sensitivity Evaluation

Consider

$$\left.\frac{\partial f}{\partial \boldsymbol{x}}\right|_{\boldsymbol{A}y=\boldsymbol{b}} = -\frac{\partial \boldsymbol{h}^T}{\partial \boldsymbol{x}} \hat{\boldsymbol{y}} + \frac{\partial f}{\partial \boldsymbol{x}} \tag{6.37}$$

where $\partial \boldsymbol{h}^T/\partial \boldsymbol{x}$ depends upon the structure of the matrix $\boldsymbol{A}$ and the functional dependence of each coefficient on $\boldsymbol{x}$ and $\partial f/\partial \boldsymbol{x}$ must be readily known, and is often $\boldsymbol{0}$.

Sensitivity Evaluation for $f \equiv y_i$

Since

$$\boldsymbol{h} = \boldsymbol{A}\,\boldsymbol{y} - \boldsymbol{b} \tag{6.38}$$

it follows that the jth row of $\partial \boldsymbol{h}^T/\partial \boldsymbol{x}$ is given by

$$\frac{\partial \boldsymbol{h}^T}{\partial x_j} = \boldsymbol{y}^T \frac{\partial \boldsymbol{A}^T}{\partial x_j} - \frac{\partial \boldsymbol{b}^T}{\partial x_j} \tag{6.39}$$

Assembling these rows, we now have

$$\left.\frac{\partial f}{\partial x}\right|_{Ay=b} = \left.\frac{\partial y_i}{\partial x}\right|_{Ay=b} = -\begin{bmatrix} y^T \dfrac{\partial A^T}{\partial x_1}\hat{y}_i \\ y^T \dfrac{\partial A^T}{\partial x_2}\hat{y}_i \\ \vdots \\ y^T \dfrac{\partial A^T}{\partial x_k}\hat{y}_i \end{bmatrix} + \begin{bmatrix} \dfrac{\partial b^T}{\partial x_1}\hat{y}_i \\ \dfrac{\partial b^T}{\partial x_2}\hat{y}_i \\ \vdots \\ \dfrac{\partial b^T}{\partial x_k}\hat{y}_i \end{bmatrix} \tag{6.40}$$

where we have let $\hat{y}_i$ be defined as the solution of the adjoint system associated with the sensitivity of y_i, namely,

$$A^T \hat{y}_i = \frac{\partial f}{\partial y} = \frac{\partial y_i}{\partial y} = u_i \tag{6.41}$$

where

$$u_i \triangleq \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \\ 0 \end{bmatrix} \leftarrow i\text{th row} \tag{6.42}$$

and where

$$\frac{\partial f}{\partial x} = \frac{\partial y_i}{\partial x} = \mathbf{0} \tag{6.43}$$

The principal assumption in using this approach to sensitivity evaluation is that few y_i are being considered and that

$$\frac{\partial A}{\partial x_j}, \qquad j = 1, 2, \ldots, k \tag{6.44}$$

are easily formulated and evaluated.

6.3.3 Linear Network Sensitivity Evaluation

Consider

$$Y V = I \tag{6.45}$$

where

$$Y \triangleq \Lambda Y_b \Lambda^T \tag{6.46}$$

is the node admittance matrix, Y_b is the branch admittance matrix and Λ is the reduced incidence matrix of the network. Let

$$x \equiv Y_b v \tag{6.47}$$

where

$$v \triangleq \begin{bmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \tag{6.48}$$

Then

$$\begin{aligned} A &\equiv Y \\ y &\equiv V \\ b &\equiv I \\ A^T &\equiv Y^T \end{aligned}$$

and the adjoint network is given by

$$Y^T \hat{V} = \hat{I} \tag{6.49}$$

where

$$\hat{I} \equiv \frac{\partial f}{\partial V} \tag{6.50}$$

Network Sensitivity Evaluation for $f \equiv V_i$, I = constant

Here,

$$\left.\frac{\partial V_i}{\partial x}\right|_{Y\,V\,=\,I} = -\begin{bmatrix} V^T \dfrac{\partial Y^T}{\partial x_1}\,\hat{V}_i \\ V^T \dfrac{\partial Y^T}{\partial x_2}\,\hat{V}_i \\ \vdots \\ V^T \dfrac{\partial Y^T}{\partial x_k}\,\hat{V}_i \end{bmatrix} \tag{6.51}$$

where

$$\frac{\partial Y^T}{\partial x_j} = \Lambda\,\frac{\partial Y_b^T}{\partial x_j}\,\Lambda^T \tag{6.52}$$

But

$$V^T\,\Lambda\,\frac{\partial Y_b^T}{\partial x_j}\,\Lambda^T\,\hat{V}_i = V_b^T\,\frac{\partial Y_b^T}{\partial x_j}\,\hat{V}_{bi} \tag{6.53}$$

where V_b and $\hat{V}_{bi}$ are vectors of branch voltages in the original and adjoint networks, respectively. Here, we have used Kirchhoff's voltage law

$$\begin{aligned} V_b &= \Lambda^T\,V \\ \hat{V}_{bi} &= \Lambda^T\,\hat{V}_i \end{aligned} \tag{6.54}$$

and the fact that, for the adjoint network,

$$Y^T = (\Lambda\,Y_b\,\Lambda^T)^T = \Lambda\,Y_b^T\,\Lambda^T \tag{6.55}$$

In general, Y_b is very sparse and $\partial Y_b/\partial x_j$ has extremely few nonzero elements: *only those associated with a network element if x_j is the element itself.*

6.3.4 The Original/Adjoint Pair of Equations

Consider the pair of systems of linear equations

$$A\,y = b,\; A^T\,\hat{y} = u \tag{6.56}$$

to be solved for y and $\hat{y}$, respectively. These vectors are also conveniently referred to as *output or response vectors*. Here, b is the right-hand side of the given system of equations and u is the corresponding right-hand side of the second or adjoint system of equations. These vectors are also

conveniently referred to as *input or excitation vectors*.

Assume that matrix A is $n \times n$ and nonsingular and can be factorized into the lower and upper triangular form matrices

$$A = L\,U,\; A^T = U^T L^T \tag{6.57}$$

where

A, A^T is of the form $\begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix}$

L, U^T is of the form $\begin{bmatrix} \times & 0 & 0 \\ \times & \times & 0 \\ \times & \times & \times \end{bmatrix}$

U, L^T is of the form $\begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \end{bmatrix}$

Then, substituting the LU factors into the first of the pair of equations, we obtain

$$A\,y = (L\,U)y = L(U\,y) = b \tag{6.58}$$

Substituting the LU factors into the second of the pair of equations, we obtain

$$A^T \hat{y} = (U^T L^T)\hat{y} = U^T(L^T \hat{y}) = u. \tag{6.59}$$

6.3.5 Solution of the Original/Adjoint Equations

Let us define intermediate vectors z and $\hat{z}$ from the relations

$$U\,y = z \tag{6.60}$$

and

$$L^T \hat{y} = \hat{z} \tag{6.61}$$

respectively. We can then solve

$$L\,z = b,\; U^T \hat{z} = u \tag{6.62}$$

for the intermediate vectors z and $\hat{z}$, respectively, by *forward substitution*. This is followed by solving

$$U\,y = z,\ L^T \hat{y} = \hat{z} \tag{6.63}$$

for the desired solutions y and $\hat{y}$, respectively, by *backward substitution*.

Notice that only one LU decomposition is required, the major computational effort of the process. The two pairs of forward and backward substitutions are not normally considered to be significant in the computational effort. In effect we have solved the pair of related systems of linear equations for little more than the work required to solve just one system.

6.3.6 Multiple Right-Hand Sides

In the case of a system of equations which has to be solved for several right-hand sides (inputs or excitations), we have

$$A\,[y_1\ \ y_2\ \ \ldots] = [b_1\ \ b_2\ \ \ldots] \tag{6.64}$$

and we can organize the computations, after factorizing the matrix A, by considering

$$(L\,U)[y_1\ \ y_2\ \ \ldots] = [b_1\ \ b_2\ \ \ldots] \tag{6.65}$$

That is, we factorize once, define the intermediate vectors z_1, z_2 ..., then solve

$$L\,[z_1\ \ z_2\ \ \ldots] = [b_1\ \ b_2\ \ \ldots] \tag{6.66}$$

for z_1, z_2, ... (forward substitution). This is followed by solving

$$U\,[y_1\ \ y_2\ \ \ldots] = [z_1\ \ z_2\ \ \ldots] \tag{6.67}$$

for y_1, y_2, ... (backward substitution). These are the output or response vectors.

6.3.7 Matrix Inversion

A suitable approach to matrix inversion follows in a very natural way from the consideration of multiple right-hand sides (inputs, excitations). To see this let

$$\begin{aligned} b_1 &= u_1 \\ b_2 &= u_2 \\ &\vdots \\ b_n &= u_n \end{aligned}$$

where vectors u_1, u_2, ..., u_n, are unit vectors (unit inputs or excitations) having zeros everywhere except in rows 1, 2, ..., n, respectively, which contain 1, i.e., the unit matrix $\mathbf{1}$ is given by the ordered set of unit vectors as

$$\mathbf{1} = [u_1\ \ u_2\ \ \ldots\ \ u_n]$$

In this case,

$$Y \triangleq [y_1\ y_2\ \ldots\ y_n] = A^{-1} \tag{6.68}$$

i.e., the ordered set of solutions $y_1, y_2, \ldots, y_n$ constitutes the inverse of the matrix A, i.e., the ordered set of outputs or responses to n (orthogonal) independent inputs or excitations.

6.3.8 Superposition of Solutions

Consider a right-hand side vector b formed by the linear combination of right-hand side vectors $b_1, b_2, \ldots$, as

$$b = c_1 b_1 + c_2 b_2 + \ldots \tag{6.69}$$

where $c_1, c_2, \ldots$, are multipliers. Then the solution of $A y = b$ is given by

$$\begin{aligned} y &= A^{-1}b \\ &= A^{-1}(c_1 b_1 + c_2 b_2 + \ldots) \\ &= c_1 A^{-1} b_1 + c_2 A^{-1} b_2 + \ldots \\ &= c_1 y_1 + c_2 y_2 + \ldots \end{aligned} \tag{6.70}$$

Notice that the solution vector y is the corresponding linear combination of the individual solutions $y_1, y_2, \ldots$

6.3.9 Iterative Improvement

Suppose we wish to solve

$$A y = b \tag{6.71}$$

for y but we obtain a slightly wrong solution, namely $\bar{y} = y + \delta y$, where δy is unknown. Then substituting back into the equations, we get

$$A \bar{y} = A(y + \delta y) = b + \delta b \tag{6.72}$$

Hence,

$$A \delta y = \delta b = A\bar{y} - b \tag{6.73}$$

where A, $\bar{y}$, and b are known. We reuse the L and U factors computed during the solution to the original system. We solve this system for δy, then express

$$y = \bar{y} - \delta y \tag{6.74}$$

This process is designed to improve the inaccurate solution previously obtained.

6.3.10 Some Typical Systems

Symmetrical matrix

If the matrix A is symmetrical then

$$A^T \equiv A \tag{6.75}$$

and

$$A\,[y\ \hat{y}] = [b\ \ u] \tag{6.76}$$

i.e., one LU set is needed for the solution process without any further rearrangement. The adjoint system is exactly the same as the original system, except for the input vector (right-hand side).

Furthermore, if $b \equiv u$ then $y = \hat{y}$. Thus only one solution of the system suffices for both outputs or responses y and $\hat{y}$.

Tridiagonal system

A tridiagonal systems is basic and has received significant attention. It is of the form

$$A = \begin{bmatrix} \times & \times & 0 & & & & \\ \times & \times & \times & \cdot & & & \\ 0 & \times & \times & \cdot & \cdot & & \\ & \cdot & \cdot & \cdot & \cdot & \cdot & \\ & & \cdot & \cdot & \cdot & \cdot & \cdot \\ & & & \cdot & \cdot & \times & \times \\ & & & & \cdot & \times & \times \end{bmatrix} \tag{6.77}$$

Notice that only the diagonal elements, and the elements immediately above and below any diagonal element are possibly nonzero. In general, the inverse is dense, i.e.,

$$A^{-1} = \begin{bmatrix} \times & \times & \times & \cdot & \cdot & \cdot \\ \times & \times & \times & \cdot & \cdot & \cdot \\ \times & \times & \times & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \times & \times & \times \\ \cdot & \cdot & \cdot & \times & \times & \times \\ \cdot & \cdot & \cdot & \times & \times & \times \end{bmatrix} \tag{6.78}$$

But the LU decomposition retains the essential sparsity of the original tridiagonal matrix, namely,

$$\boldsymbol{L}\ \boldsymbol{U} = \begin{bmatrix} \times & & & & & \\ \times & \times & & & & \\ & \times & \times & & & \\ & & \cdot & \cdot & & \\ & & & \cdot & \times & \\ & & & & \times & \times \end{bmatrix} \begin{bmatrix} \times & \times & & & & \\ & \times & \times & & & \\ & & \times & \cdot & & \\ & & & \cdot & \cdot & \\ & & & & \times & \times \\ & & & & & \times \end{bmatrix} \tag{6.79}$$

Special techniques exist for such a system. Furthermore, storage for zero elements in $\boldsymbol{A}$ is not necessary.

6.3.11 Solution Sensitivities by Matrix Perturbation

To obtain the sensitivities of a solution (response) of a system due to changes in the matrix (system) we could proceed as follows. Consider a variable ϕ which affects $\boldsymbol{A}$, i.e., $\boldsymbol{A} \equiv \boldsymbol{A}(\phi)$. Let

$$\overline{\phi} = \phi + \Delta\phi \tag{6.80}$$

and

$$\Delta\boldsymbol{A} = \boldsymbol{A}(\overline{\phi}) - \boldsymbol{A}(\phi) \tag{6.81}$$

where the system

$$(\boldsymbol{A} + \Delta\boldsymbol{A})\overline{\boldsymbol{y}} = \boldsymbol{b} \tag{6.82}$$

is to be solved for $\overline{\boldsymbol{y}}$.

Then, for some response function

$$f = \boldsymbol{u}^T \boldsymbol{y} \tag{6.83}$$

where $\boldsymbol{u}$ is a selection vector, selecting a linear combination of the elements (responses) of $\boldsymbol{y}$, the change in f is given by

$$\Delta f = \boldsymbol{u}^T(\overline{\boldsymbol{y}} - \boldsymbol{y}) \tag{6.84}$$

In this case we have

$$\frac{\Delta f}{\Delta\phi} = \frac{\boldsymbol{u}^T(\overline{\boldsymbol{y}} - \boldsymbol{y})}{\Delta\phi} = \frac{\boldsymbol{u}^T\Delta\boldsymbol{y}}{\Delta\phi} \tag{6.85}$$

where

$$\Delta\boldsymbol{y} = \overline{\boldsymbol{y}} - \boldsymbol{y} \tag{6.86}$$

We need two complete solutions, namely, y and $\bar{y}$, in general. This is for arbitrary $\Delta\phi \neq 0$.

Suppose, however, that

$$\Delta A = \frac{\partial A}{\partial \phi} \Delta\phi \tag{6.87}$$

then the first-order sensitivity formula

$$\frac{\partial f}{\partial \phi} = - u^T A^{-1} \frac{\partial A}{\partial \phi} A^{-1} b \tag{6.88}$$

becomes

$$\frac{\partial f}{\partial \phi} = - (\hat{y}^T \Delta A y) \frac{1}{\Delta \phi} \tag{6.89}$$

This is an exact first-order sensitivity even for large $\Delta\phi$. This means that we need not keep track of exactly which elements of A are affected by the change of ϕ. We simply evaluate the product in brackets and divide the result by the corresponding $\Delta\phi$.

6.4 Circuit Interpretation of Derivative Evaluation

6.4.1 Nodal Formulation

We consider here a nodal formulation. The mesh formulation and results follow from the principle of duality.

Let

$$\bar{y} = u^T y \tag{6.90}$$

where u is a selection vector. Then

$$\frac{\partial \bar{y}}{\partial \phi} = u^T \frac{\partial y}{\partial \phi} \tag{6.91}$$

Let

$\bar{y}$ stand for port response
ϕ stand for branch element
u stand for the original response selector
y stand for the nodal response vector

Then Fig. 6.2 elaborates on an interpretation of the various operations involved in derivative evaluation for the solution of a linear system. The key nodes and branches in the original and adjoint networks are shown in Fig. 6.3.

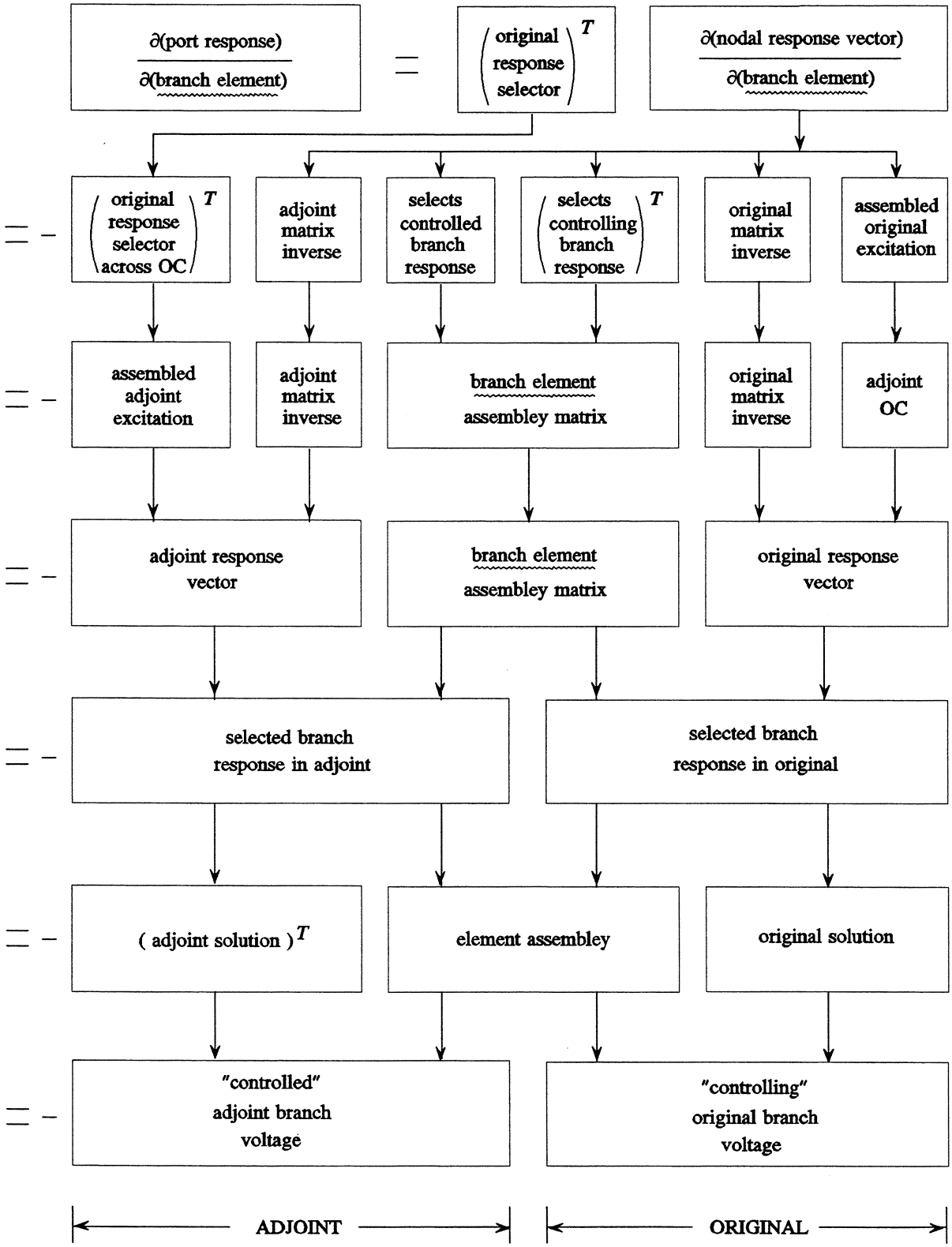

Fig. 6.2 Interpretation of derivative evaluation.

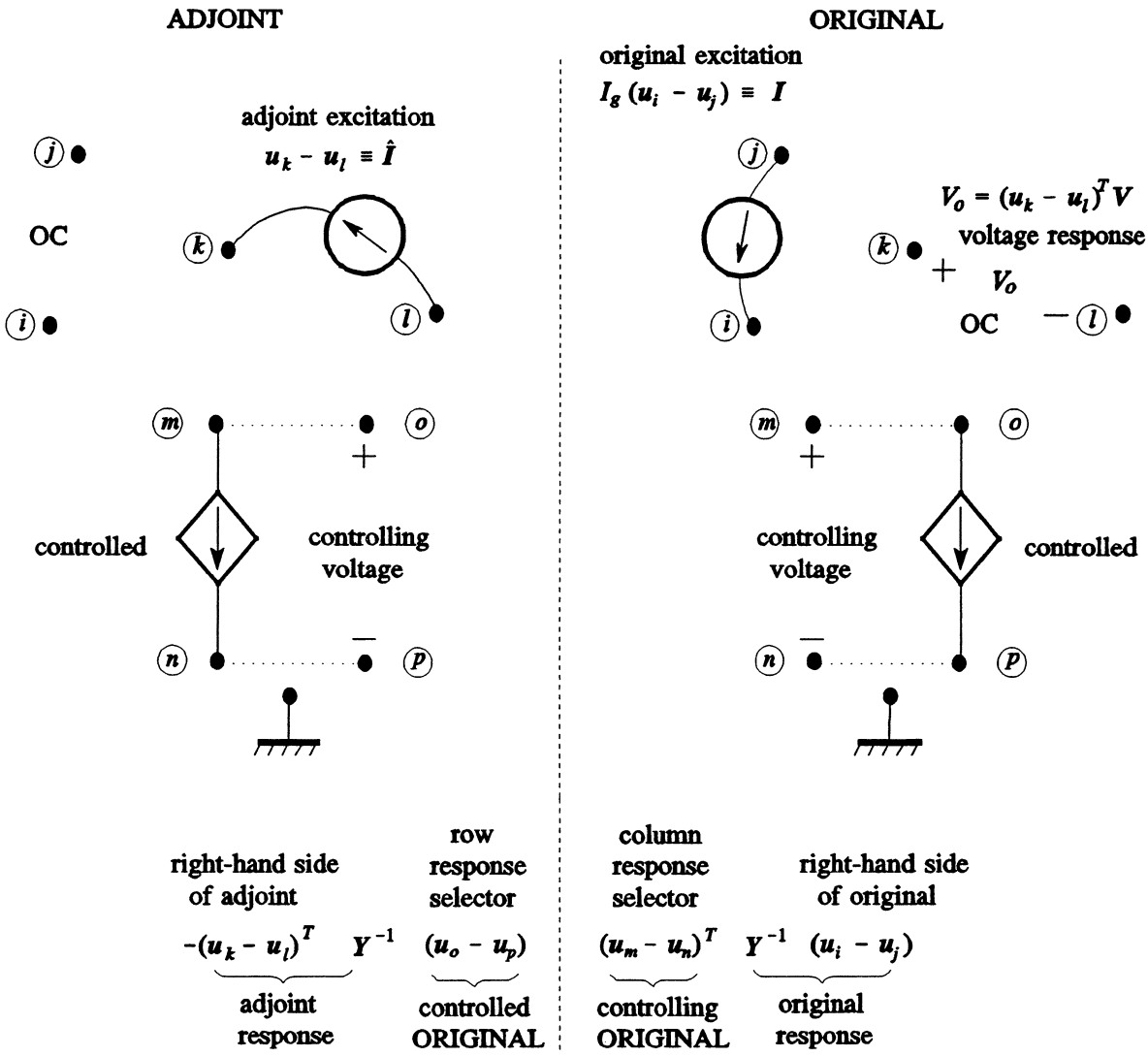

Fig. 6.3 Key nodes and branches in the original and adjoint networks. Note that original controlling nodes correspond to adjoint controlled nodes and original controlled nodes correspond to adjoint controlling nodes.

6.4.2 Examples

Example 1

An example of this analysis for a voltage controlled current source involves for the original element

$$(u_o - u_p)g_m(u_m - u_n)^T = \begin{bmatrix} g_m & -g_m \\ -g_m & g_m \end{bmatrix} \begin{matrix} \leftarrow o \\ \leftarrow p \end{matrix} \Big\} \text{ controlled nodes} \qquad (6.92)$$

$$\begin{matrix} \uparrow & \uparrow \\ m & n \end{matrix}$$

controlling nodes

and for the adjoint

$$(u_m - u_n)g_m(u_o - u_p)^T = \begin{bmatrix} g_m & -g_m \\ -g_m & g_m \end{bmatrix} \begin{matrix} \leftarrow m \\ \leftarrow n \end{matrix} \qquad (6.93)$$

$$\begin{matrix} \uparrow & \uparrow \\ o & p \end{matrix}$$

Example 2

Consider nodes n, p grounded. Let $m \equiv 1$, $o \equiv 2$.

$$u_o - u_p = u_2$$
$$u_m - u_n = u_1$$

$$\frac{\partial(\text{response})}{\partial(\text{element})} = -\hat{V}_2 \,\vdots\, V_1 \qquad (6.94)$$

Example 3

Consider Example 2 with $m \equiv o \equiv 4$, $n \equiv p \equiv 5$. Then

$$\frac{\partial(\text{response})}{\partial(\text{element})} = -(\hat{V}_4 - \hat{V}_5) \,\vdots\, (V_4 - V_5) \qquad (6.95)$$

Example 4

Let $m \equiv 1$, $n \equiv 2$, $o \equiv 3$, $p \equiv 4$. Then

$$(u_3 - u_4) \vdots (u_1 - u_2)^T$$
$$\Downarrow$$
$$-(\hat{V}_3 - \hat{V}_4) \vdots (V_1 - V_2) \tag{6.96}$$

Example 5

Let $V_{out} = V_{16} - V_{19}$. Then

$$\frac{\partial (V_{16} - V_{19})}{\partial \phi} = (u_{16} - u_{19})^T \frac{\partial V}{\partial \phi} \tag{6.97}$$

Comments

The sensitivity formula depends only on unperturbed original and adjoint voltages, type and locations of elements. It does not depend on types or locations of excitations and responses. The original circuit port current excitation implies an adjoint circuit open circuit. The adjoint circuit current excitation is placed at the original circuit response port.

Chain Rule

Use the chain rule for ϕ appearing in several elements, namely

$$\frac{\partial \bullet}{\partial \phi} = \sum_i \frac{\partial \bullet}{\partial (\text{element})_i} \frac{\partial (\text{element})_i}{\partial \phi} \tag{6.98}$$

Example 6

Let $Y = j\omega C$. Then

$$\frac{\partial \bullet}{\partial C} = \frac{\partial \bullet}{\partial Y} \frac{\partial Y}{\partial C} = \frac{\partial \bullet}{\partial (j\omega C)} \frac{\partial (j\omega C)}{\partial C} = j\omega \frac{\partial \bullet}{\partial (j\omega C)} \tag{6.99}$$

Example 7

For derivatives w.r.t. frequency ω we proceed as follows.

$$\begin{aligned} \frac{\partial \bullet}{\partial \omega} &= \frac{\partial \bullet}{\partial (j\omega C_1)} \frac{\partial (j\omega C_1)}{\partial \omega} + \frac{\partial \bullet}{\partial (j\omega C_2)} \frac{\partial (j\omega C_2)}{\partial \omega} + \cdots \\ &= jC_1 \frac{\partial \bullet}{\partial (j\omega C_1)} + jC_2 \frac{\partial \bullet}{\partial (j\omega C_2)} + \cdots \end{aligned} \tag{6.100}$$

Example 8

$$\nabla V_{out} = \begin{bmatrix} \frac{\partial V_{out}}{\partial G_1} \\ \frac{\partial V_{out}}{\partial G_2} \\ \vdots \\ \frac{\partial V_{out}}{\partial C_1} \\ \vdots \\ \frac{\partial V_{out}}{\partial g_m} \end{bmatrix} = \begin{bmatrix} -\hat{V}_{G_1} V_{G_1} \\ -\hat{V}_{G_2} V_{G_2} \\ \vdots \\ -j\omega \hat{V}_{C_1} V_{C_1} \\ \vdots \\ -\hat{V}_c V_d \end{bmatrix} \tag{6.101}$$

where V_d is the controlling branch voltage in the original and $\hat{V}_c$ is the controlling branch voltage in the adjoint.

6.4.3 Derivatives in Terms of Branch Voltages and Currents

Sensitivity of output voltage w.r.t. elements

Conductance, $\phi = G$

$$\frac{\partial V_{out}}{\partial G} = -\hat{V}V \tag{6.102}$$

where V and $\hat{V}$ are appropriate branch voltages.

Resistance, $\phi = 1/R$

$$G = 1/R \qquad \frac{\partial \cdot}{\partial R} = \frac{\partial \cdot}{\partial G}\frac{\partial G}{\partial R} = \frac{\partial \cdot}{\partial G}\left[-\frac{1}{R^2}\right] \tag{6.103}$$

Hence

$$\frac{\partial V_{out}}{\partial R} = -\hat{V}V\left[-\frac{1}{R^2}\right] = \hat{I}I \tag{6.104}$$

where I and $\hat{I}$ are branch currents corresponding to V and $\hat{V}$, respectively.

Capacitance, $\phi = j\omega C$

$$\frac{\partial V_{out}}{\partial C} = \frac{\partial V_{out}}{\partial (j\omega C)} \frac{\partial (j\omega C)}{\partial C} = j\omega \frac{\partial V_{out}}{\partial (j\omega C)} = -j\omega \hat{V} V \tag{6.105}$$

Inductance, $\phi = (1/j\omega L)$

$$\begin{aligned} \frac{\partial V_{out}}{\partial L} &= \frac{\partial V_{out}}{\partial (1/(j\omega L))} \frac{\partial (1/(j\omega L))}{\partial L} \\ &= -\frac{1}{j\omega L^2} \frac{\partial V_{out}}{\partial (1/(j\omega L))} \\ &= -\frac{\hat{V} V}{j\omega L^2} \\ &= j\omega \hat{I} I \end{aligned} \tag{6.106}$$

Mutual Conductance, $\phi = g_m$

$$I_2 = g_m V_1$$

$$\frac{\partial V_{out}}{\partial g_m} = -\hat{V}_2 V_1 \tag{6.107}$$

where

$$\hat{I}_1 = g_m \hat{V}_2$$

Sensitivity of output current w.r.t. elements

Here, we state derivatives which may be derived by duality.

Resistance, $\phi = R$

$$\frac{\partial I_{out}}{\partial R} = -\hat{I} I \tag{6.108}$$

Compare with (6.102).

Conductance, $\phi = 1/G$

$$\frac{\partial I_{out}}{\partial G} = \hat{V} V \tag{6.109}$$

Compare with (6.104).

Inductance, $\phi = j\omega L$

$$\frac{\partial I_{out}}{\partial L} = -j\omega \hat{I} I \tag{6.110}$$

Compare with (6.105).

Capacitance, $\phi = (1/j\omega C)$

$$\frac{\partial I_{out}}{\partial C} = j\omega \hat{V} V \tag{6.111}$$

Compare with (6.106).

Mutual Resistance, $\phi = r_m$

$$V_2 = r_m I_1$$

$$\frac{\partial I_{out}}{\partial r_m} = -\hat{I}_2 I_1 \tag{6.112}$$

where

$$\hat{V}_1 = r_m \hat{I}_2$$

Compare with (6.107).

6.5 Singular Value Decomposition

Theorem

A matrix A can be factorized as

$$A = U W V^T \tag{6.113}$$

where, in general

$$\begin{bmatrix} \\ A \\ \\ \end{bmatrix}_{m \times n} = \begin{bmatrix} \\ U \\ \\ \end{bmatrix}_{m \times n} \begin{bmatrix} w_1 & & & \\ & w_2 & & \\ & & \ddots & \\ & & & w_n \end{bmatrix}_{n \times n} \begin{bmatrix} \\ V^T \\ \\ \end{bmatrix}_{n \times n} \tag{6.114}$$

and where

$$\begin{bmatrix} & U^T & \end{bmatrix} \begin{bmatrix} U \\ \\ \end{bmatrix} = [\, V^T \,]\,[\, V \,] = [\, 1 \,]_{n \times n} \tag{6.115}$$

column orthonormal — *row and column orthonormal*

An appropriate decomposition of A returns U in place of A along with W and V separately.

If A is $n \times n$, then if A is not singular

$$A^{-1} = (V^T)^{-1} W^{-1} (U)^{-1} = V \begin{bmatrix} 1/w_1 & & & \\ & 1/w_2 & & \\ & & \ddots & \\ & & & 1/w_n \end{bmatrix} U^T \tag{6.116}$$

Then the solution to the system of equations $Ay = b$ is given by

$$y = V \,\mathrm{diag}\{1/w_j\}\, U^T b \tag{6.117}$$

A singular and some $w_j = 0$

A solution $\overline{y}$ of "smallest length" can be found such that

$$\overline{y} = V \overline{W} U^T b \tag{6.118}$$

where

$$\overline{W} \triangleq \mathrm{diag}\{1/\overline{w}_j\} \tag{6.119}$$

If $w_j \neq 0$ then $\overline{w}_j = w_j$. If $w_j = 0$ then $1/\overline{w}_j = 0$.

Underdetermined set: $m < n$

When the number of equations m is less than the number of unknowns n the system has the form

$$\begin{bmatrix} & A & \end{bmatrix} \begin{bmatrix} y \\ \\ \end{bmatrix} = [\, b \,] \tag{6.120}$$

In this case we augment the system as follows. Set

$$\begin{bmatrix} A \\ \cdots \\ 0 \end{bmatrix} \begin{bmatrix} y \\ \\ \end{bmatrix} = \begin{bmatrix} b \\ \cdots \\ 0 \end{bmatrix} \tag{6.121}$$

and proceed as with the singular case.

Overdetermined set: $m > n$

When the number of equations m is greater than the number of unknowns n we use

$$y = V \operatorname{diag}\{1/w_j\}\, U^T b \tag{6.122}$$

This turns out to be the least-squares solution to the set of equations, whose form is now

$$\left[A \right]_{m \times n} [y]_n = \left[b \right]_m \tag{6.123}$$

In this case premultiplying both sides of the equation $A\, y = b$ by A^T we have

$$A^T A\, y = A^T b \tag{6.124}$$

or, introducing the decomposition $A = U\, W\, V^T$ on both sides

$$V W^T \underbrace{U^T U}_{1} W V^T y = V W^T U^T b \tag{6.125}$$

and premultiplying both sides by V^T

$$\underbrace{V^T V}_{1} W^T W V^T y = \underbrace{V^T V}_{1} W^T U^T b \tag{6.126}$$

Noting that W^T is square and not singular

$$W^T W V^T y = W^T U^T b \tag{6.127}$$

from which

$$V^T y = W^{-1} U^T b \tag{6.128}$$

hence

$$y = V\, W^{-1}\, U^T b \tag{6.129}$$

provides the required least-squares solution to the overdetermined system.

Chapter 7

SENSITIVITY ANALYSIS OF LINEAR NETWORKS

The *adjoint network method* can be used to great advantage in evaluating the gradient vector of objective functions related to performance functions such as gain, insertion loss, reflection coefficient, or any other desired response. A very broad class of networks can be treated by this method. As will be seen, no more than two complete network analyses are required to evaluate the gradient vector regardless of the number of variable parameters. This results from considering two networks, the original one which is to be optimized and a topologically equivalent adjoint network.

Director and Rohrer first discussed the concept of the adjoint network and indicated its relevance in "automated design" of networks in the frequency and time domains [1,2]. In the frequency domain [2], they considered reciprocal and nonreciprocal, lumped, linear, and time-invariant elements. We will restrict ourselves here to the frequency domain, review Director and Rohrer's results, and extend them.

We will present the Tellegen Theorem approach [3]. We will present applications to evaluation of Thevenin equivalents and network sensitivity analysis. Sensitivity expressions for commonly used linear elements are presented. Some uniformly distributed elements are included [4,5]. Multiport elements and their general hybrid representation are stressed. Sensitivity evaluation suitable for least pth and minimax optimization is presented.

7.1 The Tellegen Theorem Approach [3]

Consider linear, time-invariant networks in the frequency domain. We let v or V denote a branch voltage and i or I the corresponding branch current, where the direction of i or I is associated with the direction of v or V. All branches of the network are thus represented by *associated reference directions*. With associated reference directions, the current always enters a branch at the plus sign and leaves at the minus sign.

We also consider a topologically equivalent network whose branch voltages are denoted by $\hat{v}$ or $\hat{V}$ and corresponding branch currents by $\hat{i}$ or $\hat{I}$. The context determines which variables are excitations and which are responses. When a voltage excitation is clearly identifiable it is denoted by e or E, a corresponding current excitation by j or J.

7.1.1 Tellegen's Theorem

General Statement

Let

$$\boldsymbol{v} \triangleq \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_b \end{bmatrix} \tag{7.1}$$

contain all the branch voltages in a network and

$$\boldsymbol{i} \triangleq \begin{bmatrix} i_1 \\ i_2 \\ \vdots \\ i_b \end{bmatrix} \tag{7.2}$$

contain all the corresponding branch currents using associated reference directions. $\boldsymbol{v}$ and $\boldsymbol{i}$ must satisfy Kirchhoff's voltage and current laws, respectively. Then Tellegen's Theorem states [6]

$$\sum_{\text{branches}} vi = \boldsymbol{v}^T\boldsymbol{i} = 0 \tag{7.3}$$

The summation indicates that the product of branch voltage (branch current) with the corresponding branch current (branch voltage) must be taken over all branches of the network.

As long as the topologies are the same, $\boldsymbol{v}$ can refer to one network and $\boldsymbol{i}$ to another. See the example in Fig. 7.1, where we have

$$\boldsymbol{v}^T\boldsymbol{i} = 24 + 32 - 70 + 14 = 0$$

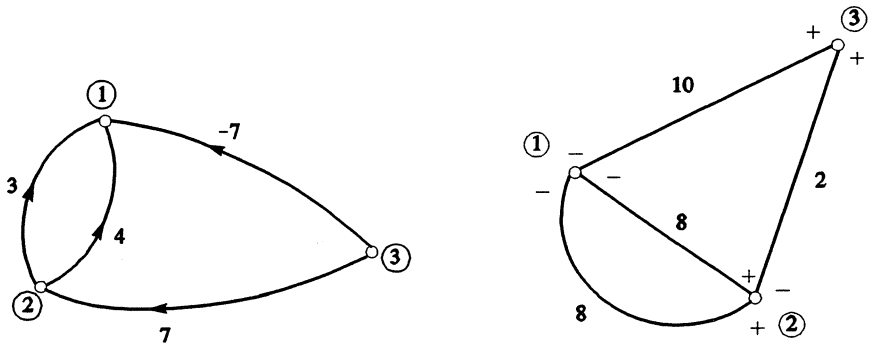

Fig. 7.1 Illustration of Tellegen's theorem applied to two networks of the same topology. Since the nature of the elements is immaterial, they are replaced by branches.

Phasor Representation

As mentioned earlier we will confine ourselves to a consideration of linear, time-invariant networks in the *frequency domain*. "Unhatted" variables V, I, etc., will thus denote phasors associated with the original network, whereas the "hatted" variables $\hat{V}$, $\hat{I}$, etc., the corresponding phasors associated with the adjoint network. By Tellegen's Theorem [6]

$$V_B^T \hat{I}_B = 0 \tag{7.4}$$

$$I_B^T \hat{V}_B = 0 \tag{7.5}$$

where the subscript B implies that the associated vectors contain all corresponding complex branch voltages and currents.

Difference Form

Subtracting (7.5) from (7.4) we obtain a difference form

$$V_B^T \hat{I}_B - I_B^T \hat{V}_B = 0 \tag{7.6}$$

which can be rewritten in an expanded difference form

$$\sum_{\text{branches}} V\hat{I} - \sum_{\text{branches}} I\hat{V} = 0 \tag{7.7}$$

where the generic notation of (7.3) has been implemented with phasors. This pair of summations can be divided into three groups, one associated with voltage sources, one with current sources and the third group with elements of the network. This leads to the expanded difference form

$$\sum_{\substack{\text{voltage}\\ \text{sources}}} (V\hat{I} - I\hat{V}) + \sum_{\substack{\text{current}\\ \text{sources}}} (V\hat{I} - I\hat{V}) + \sum_{\text{elements}} (V\hat{I} - I\hat{V}) = 0 \tag{7.8}$$

Notice that these groups are similar to each other.

Perturbed Difference Form

Perturbing only the elements in the first network, while the second network remains fixed gives

$$\Delta V_B^T \hat{I}_B = 0 \tag{7.9}$$

$$\Delta I_B^T \hat{V}_B = 0 \tag{7.10}$$

where we have applied the perturbation operator Δ only to the voltages and currents of the original network and we assume that Kirchhoff's voltage and current laws must also be applicable to ΔV_B and ΔI_B. Subtracting (7.10) from (7.9) gives

$$\Delta V_B^T \hat{I}_B - \Delta I_B^T \hat{V}_B = 0 \tag{7.11}$$

An expanded difference form similar to (7.8) can be written down. It is left to a later part of this

chapter.

7.1.2 Difference Form for Derivative Calculation

In order to simplify the notation, we will use *the subscript ϕ to denote* $\partial\bullet/\partial\phi$, i.e.,

$$\bullet_\phi \equiv \partial\bullet/\partial\phi \tag{7.12}$$

Applying this subscript to the foregoing expanded difference (7.8) form we obtain

$$\sum_{\substack{\text{voltage}\\\text{sources}}} (V_\phi \hat{I} - I_\phi \hat{V}) + \sum_{\substack{\text{current}\\\text{sources}}} (V_\phi \hat{I} - I_\phi \hat{V}) + \sum_{\text{elements}} (V_\phi \hat{I} - I_\phi \hat{V}) = 0 \tag{7.13}$$

We now proceed to simplify this expression, taking into account features of the analysis which we wish to invoke.

Eliminating Voltage Response Derivatives

Set any $\hat{I}$ to zero to eliminate the corresponding response derivative V_ϕ.

Eliminating Current Response Derivatives

Set any $\hat{V}$ to zero to eliminate the corresponding response derivative I_ϕ.

Constant Voltage Sources

$V_\phi \equiv 0$ for fixed voltage sources (independent of ϕ). These sources will be designated E.

Constant Current Sources

$I_\phi \equiv 0$ for fixed current sources (independent of ϕ). These sources will be designated J.

Branch Relations

Branch relations such as the general hybrid matrix description

$$\begin{bmatrix} I_a \\ V_b \end{bmatrix} = \begin{bmatrix} Y & A \\ M & Z \end{bmatrix} \begin{bmatrix} V_a \\ I_b \end{bmatrix} \tag{7.14}$$

must be expressed in derivative form to be useful in sensitivity evaluation.

Reduced Difference Form for Derivatives

After taking the foregoing rules into consideration the reduced difference form for derivatives becomes

$$\sum_{\substack{\text{voltage}\\\text{sources}}} I_\phi \hat{V} - \sum_{\substack{\text{current}\\\text{sources}}} V_\phi \hat{I} = \ldots + \begin{bmatrix} V_a \\ V_b \end{bmatrix}_\phi^T \begin{bmatrix} \hat{I}_a \\ \hat{I}_b \end{bmatrix} - \begin{bmatrix} I_a \\ I_b \end{bmatrix}_\phi^T \begin{bmatrix} \hat{V}_a \\ \hat{V}_b \end{bmatrix} + \ldots \tag{7.15}$$

Notice that the term of this sum corresponding to the element under consideration has been placed on the

right-hand side of the equation. Furthermore, it is expressed in matrix form.

Sensitivities of Response Currents

We define an adjoint voltage source $\hat{E}$ to expose a sensitivity of a response current w.r.t. ϕ, namely, I_ϕ, which we may be interested in.

Sensitivities of Response Voltages

We define an adjoint current source $\hat{J}$ to expose a sensitivity of a response voltage w.r.t. ϕ, namely, V_ϕ, which we may be interested in.

Difference Form for Response Calculations

Going back to the original difference form before we took derivatives we may express a difference form for responses simply as

$$-\sum_{\substack{\text{voltage}\\ \text{sources}}} (E\hat{I} - I\hat{V}) - \sum_{\substack{\text{current}\\ \text{sources}}} (V\hat{I} - J\hat{V}) = \ldots + \begin{bmatrix} V_a \\ V_b \end{bmatrix}^T \begin{bmatrix} \hat{I}_a \\ \hat{I}_b \end{bmatrix} - \begin{bmatrix} I_a \\ I_b \end{bmatrix}^T \begin{bmatrix} \hat{V}_a \\ \hat{V}_b \end{bmatrix} + \ldots \tag{7.16}$$

Again, the term of this sum corresponding to the element under consideration has been placed on the right-hand side of the equation.

Derivative of the Branch Relation

Applying the derivative operator w.r.t. ϕ to the hybrid branch relationship (7.14) we obtain

$$\begin{bmatrix} I_a \\ V_b \end{bmatrix}_\phi = \begin{bmatrix} Y & A \\ M & Z \end{bmatrix} \begin{bmatrix} V_a \\ I_b \end{bmatrix}_\phi + \begin{bmatrix} Y_\phi & A_\phi \\ M_\phi & Z_\phi \end{bmatrix} \begin{bmatrix} V_a \\ I_b \end{bmatrix} \tag{7.17}$$

Here, the left-hand side is ready for substitution into any expression which has an identical term.

7.1.3 Adjoint Element

The partial Tellegen sum for the element, i.e., the right-hand side of the reduced difference form for sensitivities, can be written as

$$\ldots + \begin{bmatrix} I_a \\ V_b \end{bmatrix}_\phi^T \begin{bmatrix} -\hat{V}_a \\ \hat{I}_b \end{bmatrix} + \begin{bmatrix} V_a \\ I_b \end{bmatrix}_\phi^T \begin{bmatrix} \hat{I}_a \\ -\hat{V}_b \end{bmatrix} + \ldots \tag{7.18}$$

in preparation for direct substitution of the left-hand side of the derivative of the branch relation (7.17). Upon this substitution the term becomes

$$\ldots + \begin{bmatrix} V_a \\ I_b \end{bmatrix}_\phi^T \begin{bmatrix} Y^T & M^T \\ A^T & Z^T \end{bmatrix} \begin{bmatrix} -\hat{V}_a \\ \hat{I}_b \end{bmatrix} + \begin{bmatrix} V_a \\ I_b \end{bmatrix}^T \begin{bmatrix} Y_\phi^T & M_\phi^T \\ A_\phi^T & Z_\phi^T \end{bmatrix} \begin{bmatrix} -\hat{V}_a \\ \hat{I}_b \end{bmatrix} + \begin{bmatrix} V_a \\ I_b \end{bmatrix}_\phi^T \begin{bmatrix} \hat{I}_a \\ -\hat{V}_b \end{bmatrix} + \ldots \tag{7.19}$$

Factorizing the foregoing expression we obtain

$$\ldots + \begin{bmatrix} V_a \\ I_b \end{bmatrix}_\phi^T \left(\begin{bmatrix} \hat{I}_a \\ -\hat{V}_b \end{bmatrix} + \begin{bmatrix} Y^T & M^T \\ A^T & Z^T \end{bmatrix} \begin{bmatrix} -\hat{V}_a \\ \hat{I}_b \end{bmatrix} \right) + \begin{bmatrix} V_a \\ I_b \end{bmatrix}^T \begin{bmatrix} -Y_\phi^T & M_\phi^T \\ -A_\phi^T & Z_\phi^T \end{bmatrix} \begin{bmatrix} \hat{V}_a \\ \hat{I}_b \end{bmatrix} + \ldots \qquad (7.20)$$

The expression in brackets should be set to zero in order to eliminate any consideration of the derivatives of V and I w.r.t ϕ. That is, we want to set

$$\begin{bmatrix} V_a \\ I_b \end{bmatrix}_\phi^T \left(\begin{bmatrix} \hat{I}_a \\ -\hat{V}_b \end{bmatrix} + \begin{bmatrix} Y^T & M^T \\ A^T & Z^T \end{bmatrix} \begin{bmatrix} -\hat{V}_a \\ \hat{I}_b \end{bmatrix} \right) = 0 \qquad (7.21)$$

This action defines the adjoint element in hybrid matrix form. This adjoint element relation can then be rewritten as

$$\begin{bmatrix} \hat{I}_a \\ \hat{V}_b \end{bmatrix} = \begin{bmatrix} Y^T & -M^T \\ -A^T & Z^T \end{bmatrix} \begin{bmatrix} \hat{V}_a \\ \hat{I}_b \end{bmatrix} \qquad (7.22)$$

7.1.4 Sensitivity Expression

What is left in the summation (7.20) is the sensitivity expression, namely

$$G_\phi = \begin{bmatrix} V_a \\ I_b \end{bmatrix}^T \begin{bmatrix} -Y_\phi^T & M_\phi^T \\ -A_\phi^T & Z_\phi^T \end{bmatrix} \begin{bmatrix} \hat{V}_a \\ \hat{I}_b \end{bmatrix} \qquad (7.23)$$

when ϕ affects the element. It is *zero* for ϕ not affecting the element.

Finally, we arrive at the reduced difference form for sensitivity evaluation, after restoring the left-hand side from (7.15), as

$$\sum_{\substack{\text{voltage} \\ \text{sources}}} I_\phi \hat{V} - \sum_{\substack{\text{current} \\ \text{sources}}} V_\phi \hat{I} = G_\phi \qquad (7.24)$$

where we assume only one element of the network is affected by ϕ. This is a very important result. Equation (7.24) basically relates changes in port responses due to changes in element values.

7.1.5 Thevenin Equivalent Evaluation

Reduced Difference Form for Response Evaluation

Suppose we only wish to obtain a reduced difference form for response evaluation. In this case, the difference form becomes simply

$$\sum_{\substack{\text{voltage}\\\text{sources}}} (E\hat{I} - I\hat{V}) + \sum_{\substack{\text{current}\\\text{sources}}} (V\hat{I} - J\hat{V}) = 0 \tag{7.25}$$

after the same definition of the adjoint element. This result can be derived in a manner similar to the derivation for (7.24).

Application to Thevenin Equivalent

Since we associate a current response with a voltage source and a voltage response with a current source we could equally write "current responses" in place of "voltage sources" and "voltage responses" in place of "current sources" under the summations. Suppose we select a port at which to define a Thevenin equivalent, as shown in Fig. 7.2. To obtain the voltage V_{TH} we assume a zero current excitation at the corresponding port of the original network. To simplify the analysis we also let the corresponding adjoint current excitation be -1. Further, we set remaining adjoint excitations to zero, yielding

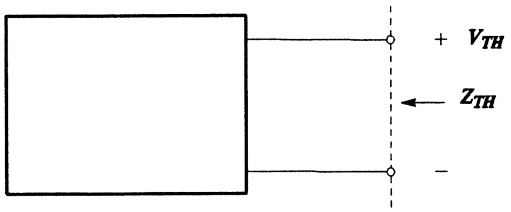

Fig. 7.2 Thevenin equivalent representation.

$$\sum_{\substack{\text{voltage}\\\text{sources}}} (E\hat{I} - I(0)) + \sum_{\substack{\text{current}\\\text{sources}}} (V(0) - J\hat{V}) = -[V_{TH}(-1) - (0)\hat{V}_{TH}] \tag{7.26}$$

where the terminating conditions have been substituted and the port at which the Thevenin is to be defined has been broken out and placed on the right-hand side. Rearranging we have

$$V_{TH} = \sum_{\substack{\text{voltage}\\\text{sources}}} E\hat{I} - \sum_{\substack{\text{current}\\\text{sources}}} J\hat{V} \tag{7.27}$$

Notice that we simply *evaluate the adjoint network* voltage and current responses in the branches corresponding to the original excitations and substitute into the formula. The values of E and J are assumed given.

To obtain the Thevenin impedance we need to excite the original network at its port of interest with one amp and, with all other original excitations set to zero, evaluate the corresponding voltage. Then this voltage is equivalent to the Thevenin impedance Z_{TH}. We further simplify by setting all adjoint

excitations to zero except for the port of interest. Here we apply, as before, a current of one amp. Thus

$$\sum_{\substack{\text{voltage}\\ \text{sources}}} ((0)\hat{I} - I(0)) + \sum_{\substack{\text{current}\\ \text{sources}}} (V(0) - (0)\hat{V}) = -[Z_{TH}(-1) - (-1)\hat{V}_{TH}] \tag{7.28}$$

where the entire left-hand side has now been reduced to 0. Observe that the *response of the adjoint network* $\hat{V}_{TH}$ is the required Z_{TH}. The Thevenin impedances of both original and adjoint networks are identical and are found through the same adjoint analysis, even though the original network is considered under different conditions. *Only the adjoint network requires analysis.*

7.1.6 Remarks and Some Extensions

Sensitivity Expression When ϕ Occurs in Several Elements

When the variable ϕ appears in several elements we could write down all the terms for the elements in question. We would finally arrive at a summation of all individual sensitivity expressions, namely

$$\sum_{\substack{\text{voltage}\\ \text{sources}}} I_\phi \hat{V} - \sum_{\substack{\text{current}\\ \text{sources}}} V_\phi \hat{I} = \sum_{\phi \text{ branches}} G_\phi \tag{7.29}$$

Compare this equation with (7.24).

General Formulas for Two-terminal Elements

See Fig. 7.3 for typical examples of two-terminal elements.

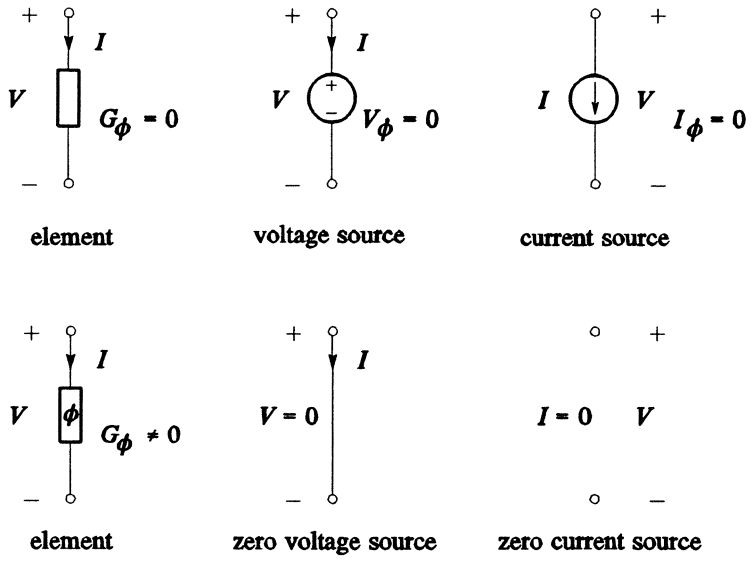

Fig. 7.3 Some-two-terminal elements.

Sensitivity Expressions

The following summarizes some of the results we have derived for the sensitivity expression G_ϕ.

ϕ does not affect the element $\Rightarrow G_\phi = 0$.

ϕ does affect the element $\Rightarrow G_\phi \neq 0$.

ϕ affects many elements $\Rightarrow G_\phi \rightarrow \Sigma\, G_\phi$, the summation being over all affected elements.

$\phi = \omega \Rightarrow \Sigma\, G_\omega$.

When We Have a Vector ϕ

Notice that the computation of sensitivity expressions depends only the results of the original network analysis, for which appropriate branch voltages and currents are needed, and the corresponding results for an adjoint network analysis. This is independent of the number of variables ϕ. Thus the equation of the form (7.24) for

$$\partial/\partial\phi \equiv \nabla$$

becomes

$$\sum_{\substack{\text{voltage}\\ \text{sources}}} \hat{V}\nabla I - \sum_{\substack{\text{current}\\ \text{sources}}} \hat{I}\nabla V = G \tag{7.30}$$

where vector G contains G_ϕ, $\Sigma\, G_\phi$ or $\Sigma\, G_\omega$ in appropriate rows.

7.2 Two Ports

7.2.1 Voltage Excitation

Consider the two-port network excited by voltage $E = V_{in}$ as shown in Fig. 7.4(a).

Calculation of $\partial V_{out}/\partial\phi$

Fig. 7.4(b) shows the adjoint for evaluation of $\partial V_{out}/\partial\phi$ leading to a special case of (7.30) as

$$(-1)\nabla V_{out} = G \tag{7.31}$$

Calculation of $\partial I_{in}/\partial\phi$

Fig. 7.4(c) shows the adjoint for evaluation of $\partial I_{in}/\partial\phi$ leading to the associated special case of (7.30) as

$$\nabla I_{in} = G \tag{7.32}$$

Reciprocity

In both cases, if the network is reciprocal then we consider original and adjoint networks identical. In the case of $\partial I_{in}/\partial\phi$, if $V_{in} = 1$, then not only is the adjoint network the same as the original, so is the adjoint solution!

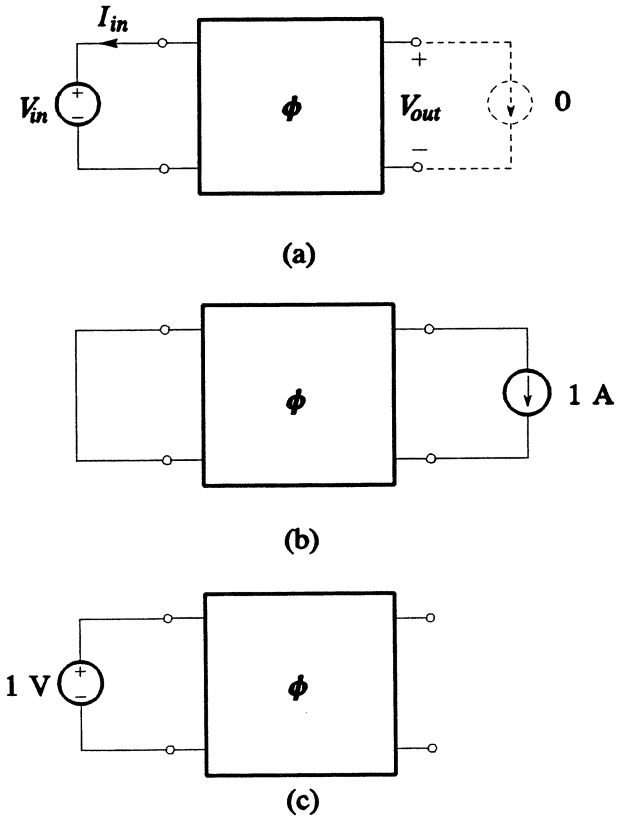

Fig. 7.4 (a) A two-port network excited by voltage V_{in}, (b) the adjoint network for evaluating $\partial V_{out}/\partial\phi$ and (c) the adjoint network for evaluating $\partial I_{in}/\partial\phi$.

7.2.2 Current Excitation

Consider the two-port network excited by current $J = I_{in}$ as shown in Fig. 7.5(a).

Calculation of $\partial V_{out}/\partial\phi$

Fig. 7.5(b) shows the adjoint for evaluation of $\partial V_{out}/\partial\phi$ leading to a special case of (7.30) as

$$(-1)\nabla V_{out} = G \tag{7.33}$$

Calculation of $\partial V_{in}/\partial\phi$

Fig. 7.5(c) shows the adjoint for evaluation of $\partial V_{in}/\partial\phi$ leading to the associated special case of (7.30) as

$$(-1)\nabla V_{in} = G \tag{7.34}$$

Reciprocity

In both cases, if the network is reciprocal then we consider original and adjoint networks identical. In the case of $\partial V_{in}/\partial\phi$, if $I_{in} = 1$, then, once again, not only is the adjoint network the same as the original, so is the adjoint solution.

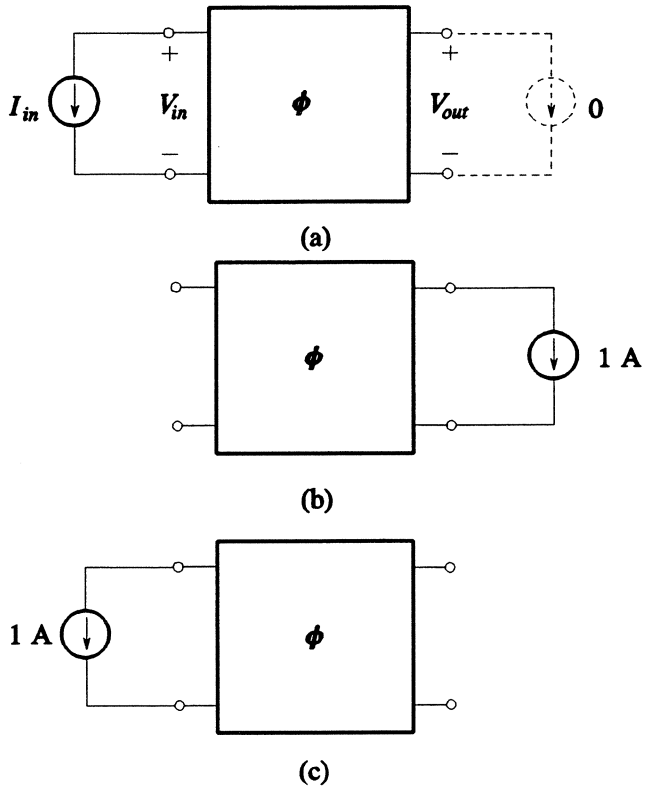

Fig. 7.5 (a) A two-port network excited by current I_{in}, (b) the adjoint network for evaluating $\partial V_{out}/\partial\phi$ and (c) the adjoint network for evaluating $\partial V_{in}/\partial\phi$.

7.3 Special Matrix Sensitivity Expressions [3]

7.3.1 Impedance Matrix Element

Fig. 7.6 shows N-port original and adjoint elements characterized in terms of open-circuit impedance matrices $\boldsymbol{Z}$ and $\hat{\boldsymbol{Z}}$, respectively. Let $\boldsymbol{V}$, $\boldsymbol{I}$, $\hat{\boldsymbol{V}}$, and $\hat{\boldsymbol{I}}$ denote N-element vectors containing the relevant port variables.

The original N-port element branch relation is given by

$$\boldsymbol{V} = \boldsymbol{Z}\boldsymbol{I} \tag{7.35}$$

This equation is a special case of the hybrid formulation (7.14) when we delete the "a" ports. The corresponding adjoint element is represented by

$$\hat{\boldsymbol{V}} = \hat{\boldsymbol{Z}}\hat{\boldsymbol{I}} \tag{7.36}$$

For derivatives w.r.t. ϕ we obtain, from (7.35)

$$\boldsymbol{V}_\phi = \boldsymbol{Z}_\phi\boldsymbol{I} + \boldsymbol{Z}\boldsymbol{I}_\phi \tag{7.37}$$

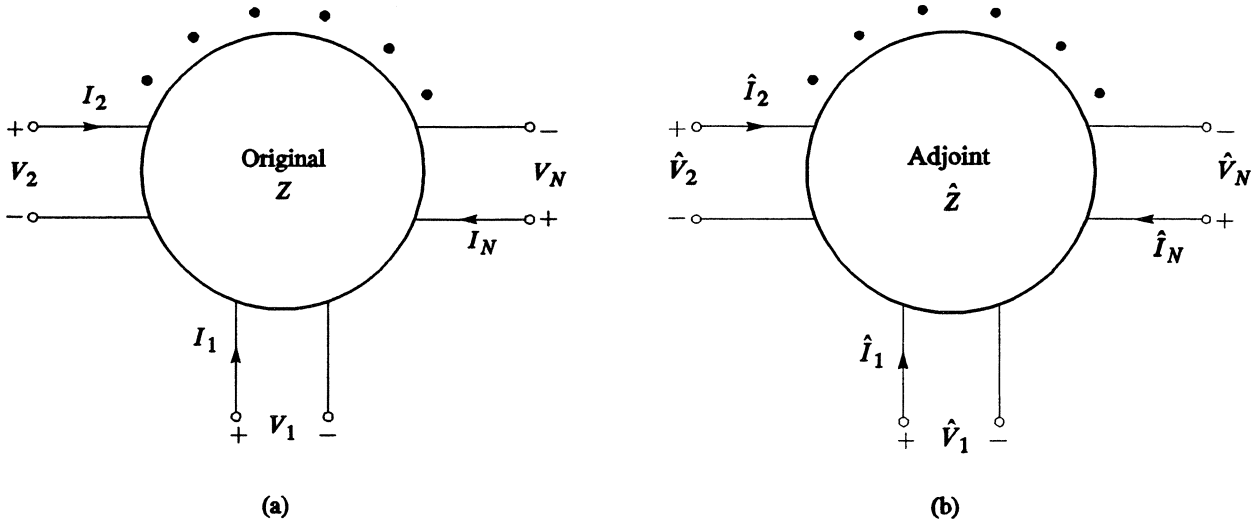

Fig. 7.6 Original and adjoint elements represented by impedance matrices. In general, many such elements suitably connected form the original and adjoint networks.

As indicated by Fig. 7.7 the port variables can be thought of as equivalent branch variables. Some *i*th

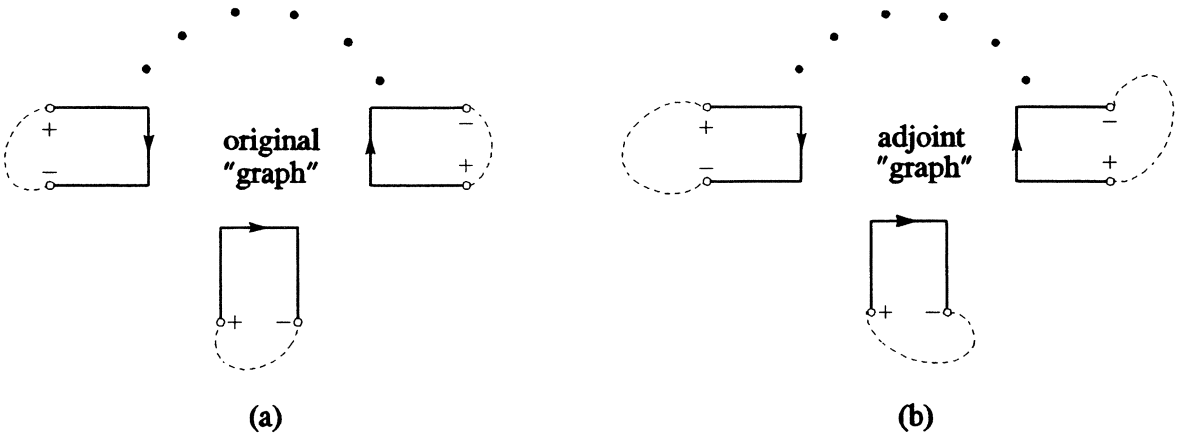

Fig. 7.7 Representation of the elements of Fig. 7.6 for application of Tellegen's theorem.

equivalent branch might consist of an impedance Z_{ii} in series with voltage generators of value $Z_{ij}I_j$, j = 1, 2, ... The Tellegen sum leads to

$$(I^T Z_\phi^T + I_\phi^T Z^T)\hat{I} - I_\phi^T \hat{V}$$

which reduces to

$$I^T Z_\phi^T \hat{I} \tag{7.38}$$

after factorization and setting the term associated with I_ϕ to **0** and setting

$$\hat{Z} \equiv Z^T \tag{7.39}$$

The result (7.38) is directly obtainable from (7.23) if we let

$$I_b \equiv I \tag{7.40}$$

$$\hat{I}_b \equiv \hat{I} \tag{7.41}$$

and delete the "a" ports.

(7.39) defines the adjoint element. Observe that expression (7.38), the only term of the Tellegen sum relating to the N-port element, does not contain I_ϕ or V_ϕ. Further, note that the adjoint of a reciprocal element is identical to the original, since $Z^T = Z$.

Clearly, any network may be thought of as consisting of the interconnection of a number of multiport elements. Thus, several terms of the form of expression (7.38) can appear in the Tellegen sum.

7.3.2 Admittance Matrix Element

For an admittance matrix representation we can show that

$$-V^T Y_\phi^T \hat{V} \tag{7.42}$$

corresponds to expression (7.38). Y^T is the admittance matrix of the adjoint. If we note that

$$V_a \equiv V \tag{7.43}$$

$$\hat{V}_a \equiv \hat{V} \tag{7.44}$$

then we see (7.42) as a special case of (7.23) with the "b" ports deleted.

7.3.3 Examples

Fig. 7.8 shows the results of a direct application of the sensitivity formulas to three commonly used elements. Table 7.1 summarizes sensitivity expressions for some commonly used lumped and distributed elements. An element consisting of a single branch is simply viewed as a one-port element.

Consider, for example, a uniformly distributed line (Fig. 7.9) having characteristic impedance Z_0 and electrical length θ. Since the element is reciprocal, we have

$$\hat{Z} = Z^T = Z = Z_0 \begin{bmatrix} \coth\theta & csch\,\theta \\ csch\,\theta & \coth\theta \end{bmatrix} \tag{7.45}$$

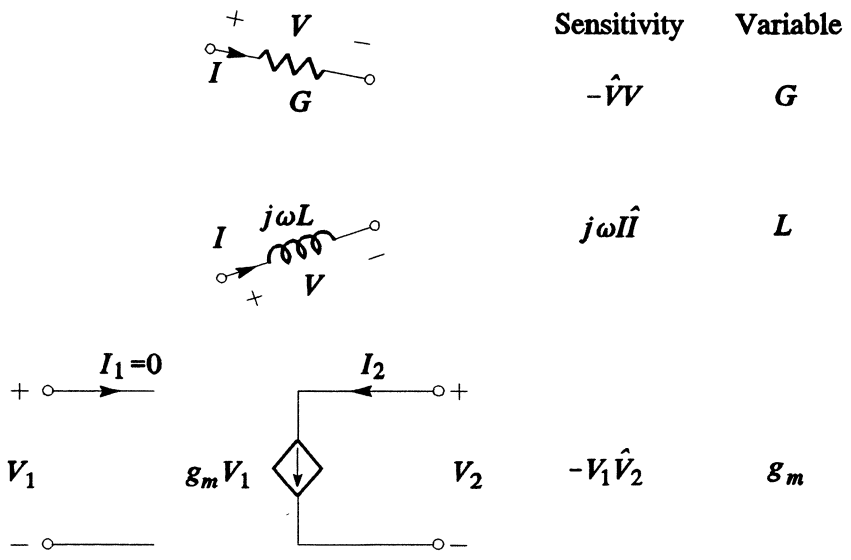

Fig. 7.8 Sensitivities for three common elements: a resistor of conductance G, an inductor of inductance L and a voltage-controlled current source with transfer conductance g_m.

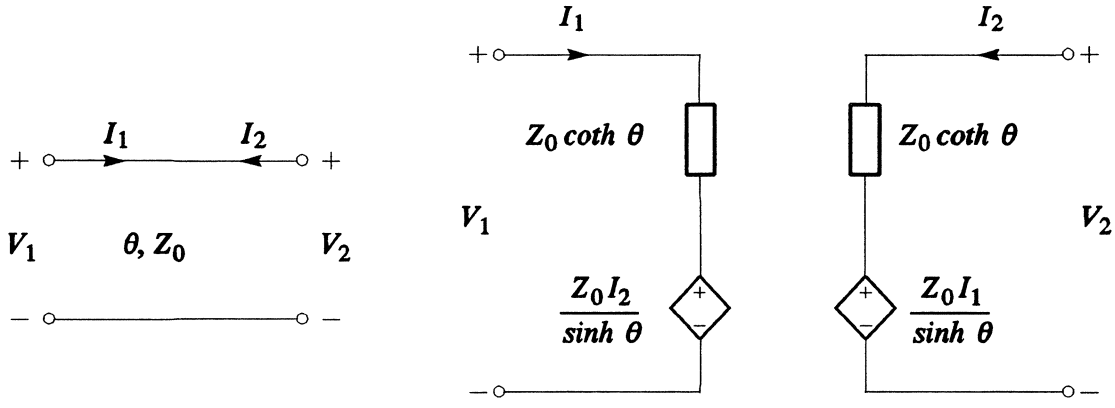

Fig. 7.9 Uniform line and convenient representation.

TABLE 7.1 SENSITIVITY EXPRESSIONS FOR SOME LUMPED AND DISTRIBUTED ELEMENTS

Element	Equation		Sensitivity (component of $\boldsymbol{G}$)	Variable (component of $\boldsymbol{\phi}$)
	Original	Adjoint		
Resistor	$V = RI$ $I = GV$	$\hat{V} = R\hat{I}$ $\hat{I} = G\hat{V}$	$I\hat{I}$ $-V\hat{V}$	R G
Inductor	$V = j\omega LI$ $I = \frac{1}{j\omega}\,\Gamma V$	$\hat{V} = j\omega L\hat{I}$ $\hat{I} = \frac{1}{j\omega}\,\Gamma\hat{V}$	$j\omega I\hat{I}$ $-\frac{1}{j\omega}V\hat{V}$	L Γ
Capacitor	$V = \frac{1}{j\omega}\,SI$ $I = j\omega CV$	$\hat{V} = \frac{1}{j\omega}\,S\hat{I}$ $\hat{I} = j\omega C\hat{V}$	$\frac{1}{j\omega}\,I\hat{I}$ $-j\omega V\hat{V}$	S C
Transformer	$\begin{bmatrix} V_1 \\ I_2 \end{bmatrix} = \begin{bmatrix} 0 & n \\ -n & 0 \end{bmatrix} \begin{bmatrix} I_1 \\ V_2 \end{bmatrix}$	$\begin{bmatrix} \hat{V}_1 \\ \hat{I}_2 \end{bmatrix} = \begin{bmatrix} 0 & n \\ -n & 0 \end{bmatrix} \begin{bmatrix} \hat{I}_1 \\ \hat{V}_2 \end{bmatrix}$	$V_2\,\hat{I}_1 + I_1\,\hat{V}_2$	n
Gyrator	$V = \begin{bmatrix} 0 & \alpha \\ -\alpha & 0 \end{bmatrix} I$	$\hat{V} = \begin{bmatrix} 0 & -\alpha \\ \alpha & 0 \end{bmatrix} \hat{I}$	$I_1\,\hat{I}_2 - I_2\,\hat{I}_1$	α
Voltage controlled voltage source	$\begin{bmatrix} I_1 \\ V_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ \mu & 0 \end{bmatrix} \begin{bmatrix} V_1 \\ I_2 \end{bmatrix}$	$\begin{bmatrix} \hat{I}_1 \\ \hat{V}_2 \end{bmatrix} = \begin{bmatrix} 0 & -\mu \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \hat{V}_1 \\ \hat{I}_2 \end{bmatrix}$	$V_1\,\hat{I}_2$	μ
Voltage controlled current source	$I = \begin{bmatrix} 0 & 0 \\ g_m & 0 \end{bmatrix} V$	$\hat{I} = \begin{bmatrix} 0 & g_m \\ 0 & 0 \end{bmatrix} \hat{V}$	$-V_1\,\hat{V}_2$	g_m
Current controlled voltage source	$V = \begin{bmatrix} 0 & 0 \\ r_m & 0 \end{bmatrix} I$	$\hat{V} = \begin{bmatrix} 0 & r_m \\ 0 & 0 \end{bmatrix} \hat{I}$	$I_1\,\hat{I}_2$	r_m

Table 7.1 *Continued*

Current controlled current source	$\begin{bmatrix} V_1 \\ I_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ \beta & 0 \end{bmatrix} \begin{bmatrix} I_1 \\ V_2 \end{bmatrix}$	$\begin{bmatrix} \hat{V}_1 \\ \hat{I}_2 \end{bmatrix} = \begin{bmatrix} 0 & -\beta \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \hat{I}_1 \\ \hat{V}_2 \end{bmatrix}$	$-I_1\ \hat{V}_2$	β
Short circuited uniformly distributed line	$V = Z_0(\tanh\theta)I$	$\hat{V} = Z_0(\tanh\theta)\hat{I}$	$(\tanh\theta)I\hat{I}$ $Z_0(sech^2\theta)I\hat{I}$	Z_0 θ
	$I = Y_0(\coth\theta)V$	$\hat{I} = Y_0(\coth\theta)\hat{V}$	$-(\coth\theta)V\hat{V}$ $Y_0(csch^2\theta)V\hat{V}$	Y_0 θ
Open circuited uniformly distributed line	$V = Z_0(\coth\theta)I$	$\hat{V} = Z_0(\coth\theta)\hat{I}$	$(\coth\theta)I\hat{I}$ $-Z_0\ (csch^2\theta)I\hat{I}$	Z_0 θ
	$I = Y_0(\tanh\theta)V$	$\hat{I} = Y_0(\tanh\theta)\hat{V}$	$-(\tanh\theta)V\hat{V}$ $Y_0\ (sech^2\theta)V\hat{V}$	Y_0 θ
Uniformly distributed line	$V = Z_0\begin{bmatrix} \coth\theta & \text{csch}\,\theta \\ \text{csch}\,\theta & \coth\theta \end{bmatrix} I$	same as original network equation but with $\hat{V}$ and $\hat{I}$ replacing V and I, respectively	$\frac{1}{Z_0}\ V^T\hat{I}$ $-\frac{1}{\sinh\theta}\ V^T\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}\hat{I}$	Z_0 θ
	$I = Y_0\begin{bmatrix} \coth\theta & -\text{csch}\,\theta \\ -\text{csch}\,\theta & \coth\theta \end{bmatrix} V$		$\frac{1}{Y_0}\ I^T\hat{V}$ $-\frac{1}{\sinh\theta}\ I^T\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}\hat{V}$	Y_0 θ

Table 7.1 *Continued*

Lossless transmission line	$V = -jZ_0\begin{bmatrix}\cot\beta l & \csc\beta l\\ \csc\beta l & \cot\beta l\end{bmatrix} I$	same as original network equation but with $\hat{V}$ and $\hat{I}$ replacing V and I, respectively	$\frac{1}{Z_0} V^T \hat{I}$	Z_0
			$-\frac{\beta}{\sin\beta l} V^T \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}\hat{I}$	l
	$I = -jY_0\begin{bmatrix}\cot\beta l & -\csc\beta l\\ -\csc\beta l & \cot\beta l\end{bmatrix} V$		$\frac{1}{Y_0} I^T \hat{V}$	Y_0
			$-\frac{\beta}{\sin\beta l} I^T \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}\hat{V}$	l
Uniform RC line	as for uniformly distributed line with $Z_0 = \sqrt{\frac{R}{sC}}$ and $\theta = \sqrt{sRC}$		$\frac{1}{2R} V^T \begin{bmatrix}1 & -\frac{\theta}{\sinh\theta}\\ -\frac{\theta}{\sinh\theta} & 1\end{bmatrix}\hat{I}$	R
			$-\frac{1}{2C} V^T \begin{bmatrix}1 & \frac{\theta}{\sinh\theta}\\ \frac{\theta}{\sinh\theta} & 1\end{bmatrix}\hat{I}$	C

Invoking expression (7.38) we obtain

$$\begin{aligned} I^T Z_\phi^T \hat{I} &= I^T \left[(Z_0)_\phi \begin{bmatrix} \coth\theta & csch\,\theta \\ csch\,\theta & \coth\theta \end{bmatrix} - \frac{Z_0\,\theta_\phi}{\sinh\theta} \begin{bmatrix} csch\,\theta & \coth\theta \\ \coth\theta & csch\,\theta \end{bmatrix} \right]^T \hat{I} \\ &= \left[\frac{(Z_0)_\phi}{Z_0} ZI - \frac{\theta_\phi}{\sinh\theta} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} ZI \right]^T \hat{I} \\ &= \frac{(Z_0)_\phi}{Z_0} V^T \hat{I} - \frac{\theta_\phi}{\sinh\theta} V^T \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \hat{I} \end{aligned} \tag{7.46}$$

when $\phi = Z_0$

$$G_\phi = \frac{1}{Z_0} V^T \hat{I} \tag{7.47}$$

when $\phi = \theta$

$$G_\phi = -\frac{1}{\sinh\theta} V^T \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \hat{I} \tag{7.48}$$

Observe that the sensitivities shown in Table 7.1 depend on currents and voltages present in the *unperturbed* original and adjoint networks. At most, two network analyses (using any suitable method) will, therefore, yield the information required to evaluate them. Note that if there is no excitation at a port, the appropriate source is set to zero. If the response at a port is of no interest, the appropriate adjoint excitation should be zero. Elements or parameters not to be varied are simply not represented in $\boldsymbol{G}$ or $\boldsymbol{\phi}$.

7.4 Applications of the Adjoint Network Method

7.4.1 Minimax Approximation

Consider the situation depicted in Fig. 7.10. Suppose we are given the problem: minimize a positive independent variable ϕ_{k+1} subject to

$$\phi_{k+1} \geq f(\boldsymbol{\phi}, \omega_i) \triangleq |\rho(\boldsymbol{\phi}, j\omega_i)|^2, \qquad \omega_i \in \Omega_d \tag{7.49}$$

where ρ is the input reflection coefficient, and Ω_d is a discrete set of frequencies in the band of interest. This problem then is effectively to minimize the maximum magnitude of the reflection coefficient over a band. Now

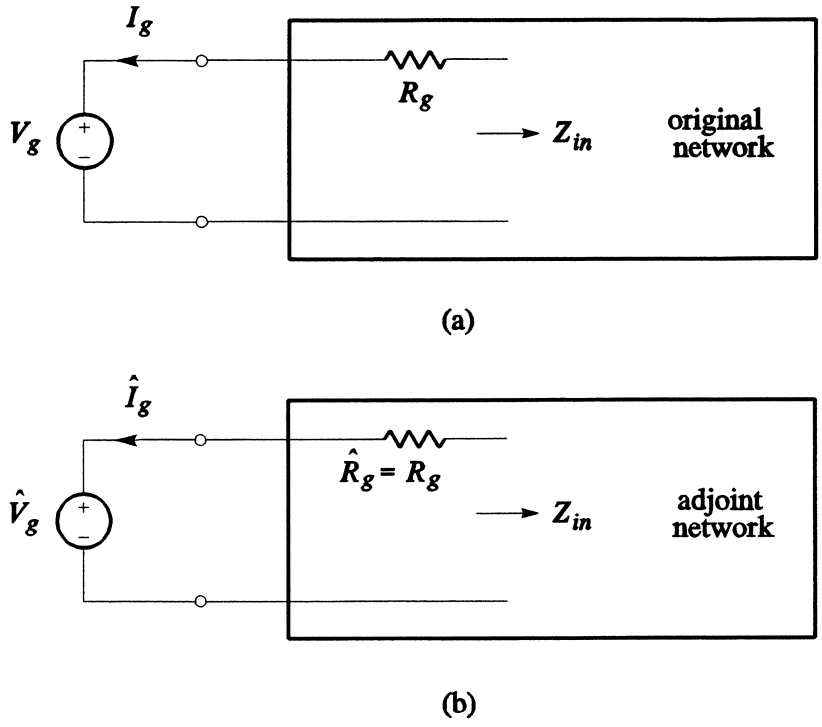

Fig. 7.10 Possible original and adjoint networks for design on the reflection coefficient basis.

$$\rho = \frac{Z_{in} - R_g}{Z_{in} + R_g} = 1 - \frac{2R_g}{Z_{in} + R_g} = 1 + \frac{2R_g I_g}{V_g} \tag{7.50}$$

so that

$$\begin{aligned} \nabla f(\phi, \omega_i) &= Re\{2\rho^*(\phi, j\omega_i)\ \nabla\rho(\phi, j\omega_i)\} \\ &= Re\left\{\frac{4R_g}{V_g}\ \rho^*(\phi, j\omega_i)\ \nabla I_g(\phi, j\omega_i)\right\} \end{aligned} \tag{7.51}$$

From (7.30)

$$\hat{V}_g \nabla I_g = \boldsymbol{G} \tag{7.52}$$

Hence

$$\nabla I_g = \frac{1}{\hat{V}_g}\ \boldsymbol{G} \tag{7.53}$$

and, finally,

$$\nabla f(\phi, \omega_i) = \mathrm{Re}\left\{ \frac{4R_g}{V_g \hat{V}_g} \rho^*(\phi, j\omega_i) G(\phi, \omega_i) \right\} \tag{7.54}$$

Observe that we are at liberty to set $\hat{V}_g = V_g$. If the original network is reciprocal so that the adjoint network is identical to the original, we need perform only one network analysis to obtain $\nabla f(\phi, \omega_i)$.

7.4.2 Least *p*th Approximation

If there are n_V independent voltage sources and n_I independent current sources, from (7.30) we have

$$G = \sum_{i=1}^{n_V} \hat{V}_i \nabla I_i - \sum_{i=n_V+1}^{n_V+n_I} \hat{I}_i \nabla V_i \tag{7.55}$$

Suppose we are given the objective function [3],

$$U = \sum_{i=1}^{n_V+n_I} \int_{\Omega} |e_i(\phi, j\omega)|^p \, d\omega \tag{7.56}$$

where Ω defines a frequency range of interest and were $e_i(\phi, j\omega)$ is an ith function of the form of (4.29) such that

$$F_i(\phi, j\omega) \triangleq \begin{cases} I_i(\phi, j\omega), & i = 1, 2, \ldots, n_V \\ V_i(\phi, j\omega), & i = n_V+1, \ldots, n_V+n_I \end{cases} \tag{7.57}$$

Equation (7.56) thus represents a summation of functions of the form of (5.75). The specified functions $S_i(j\omega)$ correspond to desired response currents and voltages. In general, $F_i(\phi, j\omega)$, $S_i(j\omega)$, and hence $e_i(\phi, j\omega)$ may be complex. Now, from (5.73)

$$\nabla U = \sum_{i=1}^{n_V+n_I} \int_{\Omega} Re\{p|e_i(\phi, j\omega)|^{p-2} w_i(\omega) e_i^*(\phi, j\omega) \nabla F_i(\phi, j\omega)\} d\omega \tag{7.58}$$

Comparing (7.55), (7.57) and (7.58), we see that if the adjoint network excitations are taken as

$$p|e_i(\phi, j\omega)|^{p-2} w_i(\omega) e_i^*(\phi, j\omega) = \begin{cases} \hat{V}_i(j\omega) & i = 1, 2, \ldots, n_V \\ -\hat{I}_i(j\omega) & i = n_V+1, \ldots, n_V+n_I \end{cases} \tag{7.59}$$

then

$$\nabla U = \int_{\Omega} Re\{G\}\, d\omega \tag{7.60}$$

The corresponding expression for the discrete case is

$$\nabla U = \sum_{\Omega_d} Re\{G\} \tag{7.61}$$

where Ω_d is the discrete set of frequencies.

7.4.3 Group Delay Computation

In group delay computations we are essentially interested in sensitivities with respect to frequency ω [7]. This parameter is different from others that we have considered in that it is common throughout the network. Specifically, let us distinguish variables associated with some jth element of an n-element network by the subscript j. Then, assuming only ω is varied, (7.30) can be written as

$$\sum_{\substack{\text{voltage}\\ \text{sources}}} \hat{V}\frac{\partial I}{\partial \omega} - \sum_{\substack{\text{current}\\ \text{sources}}} \hat{I}\frac{\partial V}{\partial \omega} = \sum_{j=1}^{n} \begin{bmatrix} V_{aj}^T & I_{bj}^T \end{bmatrix} \begin{bmatrix} -\dfrac{\partial Y_j^T}{\partial \omega} & \dfrac{\partial M_j^T}{\partial \omega} \\ -\dfrac{\partial A_j^T}{\partial \omega} & \dfrac{\partial Z_j^T}{\partial \omega} \end{bmatrix} \begin{bmatrix} \hat{V}_{aj} \\ \hat{I}_{bj} \end{bmatrix} \tag{7.62}$$

(7.62) can be more appropriately written as

$$\sum_{i=1}^{n_V} \hat{V}_i \frac{\partial I_i}{\partial \omega} - \sum_{i=n_V+1}^{n_V+n_I} \hat{I}_i \frac{\partial V_i}{\partial \omega} = \sum_{j=1}^{n} G_{\omega j} \tag{7.63}$$

where

$$G_{\omega j} \triangleq \begin{bmatrix} V_{aj}^T & I_{bj}^T \end{bmatrix} \begin{bmatrix} -\dfrac{\partial Y_j^T}{\partial \omega} & \dfrac{\partial M_j^T}{\partial \omega} \\ -\dfrac{\partial A_j^T}{\partial \omega} & \dfrac{\partial Z_j^T}{\partial \omega} \end{bmatrix} \begin{bmatrix} \hat{V}_{aj} \\ \hat{I}_{bj} \end{bmatrix} \tag{7.64}$$

If, in particular, the kth port is to be investigated, and this happens to be a current-excited port (The value of the excitation could, of course, be zero), then (7.63) reduces to

$$-\hat{I}_k \frac{\partial V_k}{\partial \omega} = \sum_{j=1}^{n} G_{\omega j} \tag{7.65}$$

if all adjoint excitations except $\hat{I}_k$ are set to zero. Evaluation of the sensitivity expression $G_{\omega j}$ is accomplished by the results of two network analyses. The sensitivity formulas from Table 7.1 may be used if appropriate, since

$$\begin{bmatrix} -\dfrac{\partial \boldsymbol{Y}_j^T}{\partial \omega} & \dfrac{\partial \boldsymbol{M}_j^T}{\partial \omega} \\ -\dfrac{\partial \boldsymbol{A}_j^T}{\partial \omega} & \dfrac{\partial \boldsymbol{Z}_j^T}{\partial \omega} \end{bmatrix} = \sum_r \begin{bmatrix} -\dfrac{\partial \boldsymbol{Y}_j^T}{\partial \phi_r} & \dfrac{\partial \boldsymbol{M}_j^T}{\partial \phi_r} \\ -\dfrac{\partial \boldsymbol{A}_j^T}{\partial \phi_r} & \dfrac{\partial \boldsymbol{Z}_j^T}{\partial \phi_r} \end{bmatrix} \frac{\partial \phi_r}{\partial \omega} \tag{7.66}$$

where the subscript r denotes some rth parameter in the jth element with respect to which a sensitivity expression is already available.

Consider, for example, $\theta = j\omega l/c = j\beta l$, where c is the velocity of propagation. Then the ω-sensitivity of a lossless transmission line is jl/c times the θ-sensitivity shown in Table 7.1. Consider an inductor as a second example. The left-hand side of (7.66) reduces immediately to jL using $Z = j\omega L$.

Finally, to compute the group delay $T_G(\omega)$ we note that

$$T_G(\omega) = -\mathrm{Im}\left\{ \frac{1}{V_k} \frac{\partial V_k}{\partial \omega} \right\} \tag{7.67}$$

where it is assumed that all sources have constant, frequency-independent phase angles. For convenience, letting the excitation $\hat{I}_k = 1/V_k$,

$$T_G(\omega) = \mathrm{Im}\left\{ \sum_{j=1}^{n} G_{\omega j} \right\} \tag{7.68}$$

Equation (7.68) is also valid for calculations of group delay if the kth port is a voltage-excited port (the value of the excitation could, of course, be zero). All one has to remember is to set all adjoint excitations to zero except $\hat{V}_k$ which is set to $-1/I_k$.

7.4.4 Extensions and Other Applications

An important point to remember about the adjoint network method is that the analysis of the adjoint, in general, can take considerably less effort than the analysis of the original network. If $\boldsymbol{Y}_n$ is, for example, the nodal admittance matrix of the original network, and its inverse $\boldsymbol{Y}_n^{-1}$ has been computed, then we can use the result $(\boldsymbol{Y}_n^T)^{-1} = (\boldsymbol{Y}_n^{-1})^T$. For a further discussion of possible computational efficiency, the reader is referred to Director [8].

Extensions to second-order sensitivities have been formulated [9], including group-delay sensitivities [7]. Of particular interest to filter designers are applications of the adjoint network concept to the computation of dissipation-induced loss distortion in both lumped and distributed networks [10, 11]. Further extensions include the exploitation of the adjoint network concept in first- and second-order sensitivity computation using wave variables rather than voltages and currents [12, 13]. These results

should also be of interest to filter designers.

7.5 References

[1] S.W. Director and R.A. Rohrer, "The generalized adjoint network and network sensitivities," *IEEE Trans. Circuit Theory*, vol. CT-16, 1969, pp. 318-323.

[2] S.W. Director and R.A. Rohrer, "Automated network design – the frequency-domain case," *IEEE Trans. Circuit Theory*, vol. CT-16, 1969, pp. 330-337.

[3] J.W. Bandler, "Computer-aided circuit optimization," in *Modern Filter Theory and Design*, G.C. Temes and S.K. Mitra, Eds. New York: Wiley-Interscience, 1973, Chapter 6.

[4] J.W. Bandler and R.E. Seviora, "Current trends in network optimization," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-18, 1970, pp. 1159-1170.

[5] J.W. Bandler and R.E. Seviora, "Computation of sensitivities for noncommensurate networks," *IEEE Trans. Circuit Theory*, vol. CT-18, 1971, pp. 174-178.

[6] C.A. Desoer and E.S. Kuh, *Basic Circuit Theory*. New York: McGraw-Hill, 1969, Chapter 9.

[7] G.C. Temes, "Exact computation of group delay and its sensitivities using adjoint-network concept," *Electronics Letters*, vol. 6, 1970, pp. 483-485.

[8] S.W. Director, "*LU* factorization in network sensitivity calculations," *IEEE Trans. Circuit Theory*, vol. CT-18, 1971, pp. 184-185.

[9] G.A. Richards, "Second-derivative sensitivity using the concept of the adjoint network," *Electronics Letters*, vol. 5, 1969, pp. 398-399.

[10] G. C. Temes and R. N. Gadenz, "Simple technique for the prediction of dissipation-induced loss distortion," *Electronics Letters*, vol. 6, 1970, pp. 836-837.

[11] R.N. Gadenz and G.C. Temes, "Computation of dissipation-induced loss distortion in lumped/distributed networks," *Electronics Letters*, vol. 7, 1971, pp. 258-260.

[12] J.W. Bandler and R.E. Seviora, "Wave sensitivities of networks," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-20, 1972, pp. 138-147.

[13] J.W. Bandler and R.E. Seviora, "Computation of equivalent wave source using the adjoint network," *Electronics Letters*, vol. 7, 1971, pp. 235-236.

McMASTER UNIVERSITY BOOKSTORE

CUSTOM COURSEWARE SERVICE

SKU