NAG Library Chapter Introduction

e01 – Interpolation

Contents

1 Scope of the Chapter 2

2 Background to the Problems 2

3 Recommendations on Choice and Use of Available Functions 3

3.1 General 3
3.2 One Independent Variable 3
3.2.1 Interpolated values: data without derivatives 3
3.2.2 Interpolating function: data without derivatives 4
3.2.3 Data containing derivatives 4
3.3 Two Independent Variables 5
3.3.1 Data on a rectangular mesh 5
3.3.2 Arbitrary data 5
3.4 Three Independent Variables 5
3.4.1 Arbitrary data 5
3.5 Four and Five Independent Variables 5
3.5.1 Arbitrary data 5
3.6 Multidimensional interpolation 6
3.6.1 Arbitrary data 6

4 Decision Tree 6

5 Functionality Index 6

6 Auxiliary Functions Associated with Library Function Arguments 8

7 Functions Withdrawn or Scheduled for Withdrawal 8

8 References 8

1 Scope of the Chapter

This chapter is concerned with the interpolation of a function of one or more variables. When provided with the value of the function (and possibly one or more of its lowest-order derivatives) at each of a number of values of the variable(s), the NAG C Library functions provide either an interpolating function or an interpolated value. For some of the interpolating functions, there are supporting NAG C Library functions to evaluate, differentiate or integrate them.

2 Background to the Problems

In motivation and in some of its numerical processes, this chapter has much in common with Chapter e02 (Curve and Surface Fitting). For this reason, we shall adopt the same terminology and refer to dependent variable and independent variable(s) instead of function and variable(s). Where there is only one independent variable, we shall denote it by x and the dependent variable by y. Thus, in the basic problem considered in this chapter, we are given a set of distinct values $x_1, x_2, \ldots, x_m$ of x and a corresponding set of values $y_1, y_2, \ldots, y_m$ of y, and we shall describe the problem as being one of interpolating the data points (x_r, y_r), rather than interpolating a function. In modern usage, however, **interpolation** can have either of two rather different meanings, both relevant to functions in this chapter. They are

(a) the determination of a function of x which takes the value y_r at $x = x_r$, for $r = 1, 2, \ldots, m$ (an **interpolating function** or **interpolant**),

(b) the determination of the value (**interpolated value** or **interpolate**) of an interpolating function at any given value, say $\hat{x}$, of x within the range of the x_r (so as to estimate the value at $\hat{x}$ of the function underlying the data).

The latter is the older meaning, associated particularly with the use of mathematical tables. The term ‘function underlying the data’, like the other terminology described above, is used so as to cover situations additional to those in which the data points have been computed from a known function, as with a mathematical table. In some contexts, the function may be unknown, perhaps representing the dependency of one physical variable on another, say temperature upon time.

Whether the underlying function is known or unknown, the object of interpolation will usually be to approximate it to acceptable accuracy by a function which is easy to evaluate anywhere in some range of interest. Polynomials, rational functions (ratios of two polynomials) and piecewise polynomials, such as cubic splines (see Section 2.2 in the e02 Chapter Introduction for definitions of terms in the latter case), being easy to evaluate and also capable of approximating a wide variety of functions, are the types of function mostly used in this chapter as interpolating functions. An interpolating polynomial is taken to have degree $m - 1$ when there are m data points, and so it is unique. It is called the **Lagrange interpolating polynomial**. The rational function, in the special form used, is also unique. An interpolating spline, on the other hand, depends on the choice made for the knots.

One way of achieving the objective in (b) above is, of course, through (a), but there are also methods which do not involve the explicit computation of the interpolating function. Everett's formula and Aitken's successive linear interpolation (see Dahlquist and BjÎrck (1974)) provide two such methods. Both are used in this chapter and determine a value of the Lagrange interpolating polynomial.

It is important to appreciate, however, that the Lagrange interpolating polynomial often exhibits unwanted fluctuations between the data points. These tend to occur particularly towards the ends of the data range, and to get larger with increasing number of data points. In severe cases, such as with 30 or 40 equally spaced values of x, the polynomial can take on values several orders of magnitude larger than the data values. (Closer spacing near the ends of the range tends to improve the situation, and wider spacing tends to make it worse.) Clearly, therefore, the Lagrange polynomial often gives a very poor approximation to the function underlying the data. On the other hand, it can be perfectly satisfactory when its use is restricted to providing interpolated values away from the ends of the data range from a reasonably small number of data values.

In contrast, a cubic spline which interpolates a large number of data points can often be used satisfactorily over the whole of the data range. Unwanted fluctuations can still arise but much less frequently and much less severely than with polynomials. Rational functions, when appropriate, would also be used over the whole data range. The main danger with these functions is that their polynomial

denominators may take zero values within that range. Unwanted fluctuations are avoided altogether by a function using piecewise cubic polynomials having only first derivative continuity. It is designed especially for monotonic data, but for other data still provides an interpolant which increases, or decreases, over the same intervals as the data.

The concept of interpolation can be generalized in a number of ways. Firstly, at each x, the interpolating function may be required to take on not only a given value but also given values for all its derivatives up to some specified order (which can vary with r). This is the Hermite–Birkoff interpolation problem. Secondly, we may be required to estimate the value of the underlying function at a value $\hat{x}$ outside the range of the data. This is the process of **extrapolation**. In general, it is a good deal less accurate than interpolation and is to be avoided whenever possible.

Interpolation can also be extended to the case of two or more independent variables. If the data values are given at the intersections of a regular two-dimensional mesh bicubic splines (see Section 2.3.2 in the e02 Chapter Introduction) are very suitable and usually very effective for the problem. For other cases, perhaps where the data values are quite arbitrarily scattered, polynomials and splines are not at all appropriate and special forms of interpolating function have to be employed. Many such forms have been devised and two of the most successful are in functions in this chapter. They both have continuity in first, but not higher, derivatives.

3 Recommendations on Choice and Use of Available Functions

3.1 General

Before undertaking interpolation, in other than the simplest cases, you should seriously consider the alternative of using a function from Chapter e02 to approximate the data by a polynomial or spline containing significantly fewer coefficients than the corresponding interpolating function. This approach is much less liable to produce unwanted fluctuations and so can often provide a better approximation to the function underlying the data.

When interpolation is employed to approximate either an underlying function or its values, you will need to be satisfied that the accuracy of approximation achieved is adequate. There may be a means for doing this which is particular to the application, or the function used may itself provide a means. In other cases, one possibility is to repeat the interpolation using one or more extra data points, if they are available, or otherwise one or more fewer, and to compare the results. Other possibilities, if it is an interpolating function which is determined, are to examine the function graphically, if that gives sufficient accuracy, or to observe the behaviour of the differences in a finite difference table, formed from evaluations of the interpolating function at equally-spaced values of x over the range of interest. The spacing should be small enough to cause the typical size of the differences to decrease as the order of difference increases.

3.2 One Independent Variable

3.2.1 Interpolated values: data without derivatives

When the underlying function is well represented by data points on both sides of the value, $\hat{x}$, at which an interpolated value is required, nag_1d_everett_interp (e01abc) should be tried first if the data points are equally spaced, nag_1d_aitken_interp (e01aac) if they are not. Both compute a value of the Lagrange interpolating polynomial, the first using Everett's formula, the second Aitken's successive linear interpolation. The first function requires an equal (or nearly equal) number of data points on each side of $\hat{x}$; such a distribution of points is preferable also for the second function. If there are many data points, this will be achieved simply by using only an appropriate subset for each value of $\hat{x}$. Ten to twelve data points are the most that would be required for many problems. Both functions provide a means of assessing the accuracy of an interpolated value, with nag_1d_everett_interp (e01abc) by examination of the size of the finite differences supplied, with nag_1d_aitken_interp (e01aac) by intercomparison of the set of interpolated values obtained from polynomials of increasing degree.

In other cases, or when the above functions fail to produce a satisfactory result, one of the functions discussed in the next section should be used. The spline and other piecewise polynomial functions are the most generally applicable. They are particularly appropriate when interpolated values towards the

ends of the range are required. They are also likely to be preferable, for reasons of economy, when many interpolated values are required.

nag_1d_aitken_interp (e01aac) above, and three of the functions discussed in the next section, can be used to compute extrapolated values. These three are nag_1d_cheb_interp (e01aec), nag_monotoni c_interpolant (e01bec) and nag_1d_ratnl_interp (e01rac) based on polynomials, piecewise polynomials and rational functions respectively. Extrapolation is not recommended in general, but can sometimes give acceptable results if it is to a point not far outside the data range, and only the few nearest data points are used in the process. nag_1d_ratnl_interp (e01rac) is most likely to be successful.

3.2.2 Interpolating function: data without derivatives

nag_1d_cheb_interp (e01aec) computes the Lagrange interpolating polynomial by a method (based on **Newton's formula with divided differences** (see FrÎberg (1970)) which has proved numerically very stable. However, the likelihood of the polynomial having unwanted fluctuations, particularly near the ends of the data range when a moderate or large number of data points are used, should be remembered.

Such fluctuations of the polynomial can be avoided if you are at liberty to choose the x values at which to provide data points. In this case, a function from Chapter e02, namely nag_1d_cheb_interp_fit (e02afc), should be used in the manner and with the x values discussed in Section 3.2.2 in the e02 Chapter Introduction.

Usually however, when the whole of the data range is of interest, it is preferable to use a cubic spline as the interpolating function. nag_1d_spline_interpolant (e01bac) computes an interpolating cubic spline, using a particular choice for the set of knots which has proved generally satisfactory in practice. If you wish to choose a different set, a cubic spline function from Chapter e02, namely nag_1d_spline_ fit_knots (e02bac), may be used in its interpolating mode, setting **spline**.**n** $= mdist + 4$ and all elements of the argument **weights** to unity.

The cubic spline does not always avoid unwanted fluctuations, especially when the data shows a steep slope close to a region of small slope, or when the data inadequately represents the underlying curve. In such cases, nag_monotonic_interpolant (e01bec) can be very useful. It derives a piecewise cubic polynomial (with first derivative continuity) which, between any adjacent pair of data points, either increases all the way, or decreases all the way (or stays constant). It is especially suited to data which is monotonic over the whole range.

In this function, the interpolating function is represented simply by its value and first derivative at the data points. Supporting functions compute its value and first derivative elsewhere, as well as its definite integral over an arbitrary interval. The other functions mentioned, namely nag_1d_cheb_interp (e01aec) and nag_1d_spline_interpolant (e01bac), provide the interpolating function either in Chebyshev series form or in B-spline form (see Sections 2.2.1 and 2.2.2 in the e02 Chapter Introduction). Functions for evaluating, differentiating and integrating these forms are discussed in Section 3.7 in the e02 Chapter Introduction. The splines and other piecewise cubics will normally provide better estimates of the derivatives of the underlying function than will interpolating polynomials, at any rate away from the central part of the data range.

nag_1d_ratnl_interp (e01rac) computes an interpolating rational function. It is intended mainly for those cases where you know that this form of function is appropriate. However, it is also worth trying in cases where the other functions have proved unsatisfactory. nag_1d_ratnl_eval (e01rbc) is available to compute values of the function provided by nag_1d_ratnl_interp (e01rac).

3.2.3 Data containing derivatives

nag_1d_cheb_interp (e01aec) (see Section 3.2.2) can also compute the polynomial which, at each x_r, has not only a specified value y_r but also a specified value of each derivative up to order p_r.

3.3 Two Independent Variables

3.3.1 Data on a rectangular mesh

Given the value f_{qr} of the dependent variable f at the point $\left(x_q, y_r\right)$ in the plane of the independent variables x and y, for each $q = 1, 2, \ldots, m$ and $r = 1, 2, \ldots, n$ (so that the points $\left(x_q, y_r\right)$ lie at the $m \times n$ intersections of a rectangular mesh), nag_2d_spline_interpolant (e01dac) computes an interpolating bicubic spline, using a particular choice for each of the spline's knot-set. This choice, the same as in nag_1d_spline_interpolant (e01bac), has proved generally satisfactory in practice. If, instead, you wish to specify your own knots, a function from Chapter e02, namely nag_2d_spline_fit_panel (e02dac), may be used (it is more cumbersome for the purpose, however, and much slower for larger problems). Using m and n in the above sense, the argument **m** must be set to $m \times n$, **spline.nx** and **spline.ny** must be set to $m + 4$ and $n + 4$ respectively and all elements of **w** should be set to unity. The recommended value for **eps** is zero.

3.3.2 Arbitrary data

As remarked at the end of Section 2, specific methods of interpolating are required for this problem, which can often be difficult to solve satisfactorily. Two of the most successful are employed in nag_2d_shep_interp (e01sgc) and nag_2d_triang_interp (e01sjc), the two functions which (with their respective evaluation functions nag_2d_shep_eval (e01shc) and nag_2d_triang_eval (e01skc)) are provided for the problem. Definitions can be found in the function documents. Both interpolants have first derivative continuity and are 'local', in that their value at any point depends only on data in the immediate neighbourhood of the point. This latter feature is necessary for large sets of data to avoid prohibitive computing time. nag_2d_shep_eval (e01shc) allows evaluation of the interpolant and its first partial derivatives.

The relative merits of the two methods vary with the data and it is not possible to predict which will be the better in any particular case.

nag_2d_shep_interp (e01sgc) and nag_2d_triang_interp (e01sjc) each perform a triangulation of the scattered data points and then calculate a bicubic interpolant based on this triangulation and on the function values at the scattered points (which can be evaluated by nag_2d_shep_eval (e01shc) and nag_2d_triang_eval (e01skc) respectively). Where derivative continuity is not essential and where bilinear interpolated values are sufficient, nag_2d_triangulate (e01eac) (which performs the same triangulation as nag_2d_shep_interp (e01sgc) and nag_2d_triang_interp (e01sjc)) and nag_2d_triang_bary_eval (e01ebc) (which performs barycentric interpolation using the set of function values) may be used.

3.4 Three Independent Variables

3.4.1 Arbitrary data

The function nag_3d_shep_interp (e01tgc) and its evaluation function nag_3d_shep_eval (e01thc) are provided for interpolation of three-dimensional scattered data. As in the case of two independent variables, the method is local, and produces an interpolant with first derivative continuity. nag_3d_shep_eval (e01thc) allows evaluation of the interpolant and its first partial derivatives.

3.5 Four and Five Independent Variables

3.5.1 Arbitrary data

The function nag_4d_shep_interp (e01tkc) and its evaluation function nag_4d_shep_eval (e01tlc) allow interpolation of four-dimensional scattered data, while the function nag_5d_shep_interp (e01tmc) and its evaluation function nag_5d_shep_eval (e01tnc) allow interpolation of five-dimensional scattered data. nag_4d_shep_interp (e01tkc) and nag_5d_shep_interp (e01tmc) are higher dimensional analogues to the functions nag_2d_shep_interp (e01sgc) and nag_3d_shep_interp (e01tgc), while nag_4d_shep_eval (e01tlc) and nag_5d_shep_eval (e01tnc) are analogous to nag_2d_shep_eval (e01shc) and nag_3d_shep_eval (e01thc).

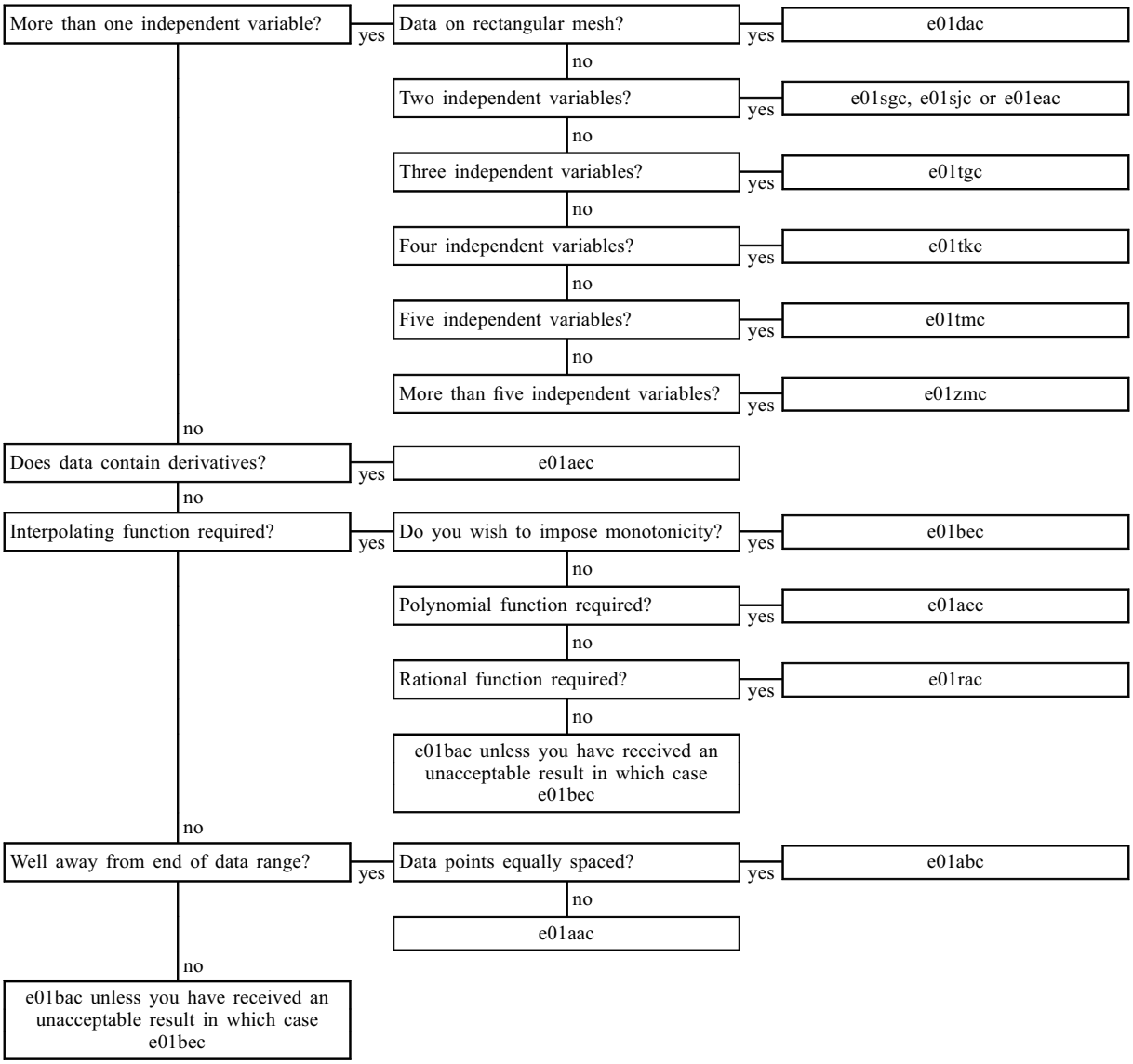
3.6 Multidimensional interpolation

3.6.1 Arbitrary data

Interpolation of scattered data in d-dimensions, where $d > 2$, is provided by function nag_nd_shep_interp (e01zmc). This extends the local method of nag_3d_shep_interp (e01tgc) and nag_4d_shep_interp (e01tkc) to higher dimensions. Evaluation of the interpolant, which has continuous first derivatives, is carried out by function nag_nd_shep_eval (e01znc).

4 Decision Tree

Tree 1



5 Functionality Index

Derivative,
of interpolant,
from nag_monotonic_interpolant (e01bec) ..... nag_monotonic_deriv (e01bgc)
from nag_2d_shep_interp (e01sgc) ..... nag_2d_shep_eval (e01shc)
from nag_3d_shep_interp (e01tgc) ..... nag_3d_shep_eval (e01thc)
from nag_4d_shep_interp (e01tkc) ..... nag_4d_shep_eval (e01tlc)
from nag_5d_shep_interp (e01tmc) ..... nag_5d_shep_eval (e01tnc)
from nag_nd_shep_interp (e01zmc) ..... nag_nd_shep_eval (e01znc)

Evaluation,
of interpolant,
from nag_monotonic_interpolant (e01bec) ..... nag_monotonic_evaluate (e01bfc)
from nag_1d_ratnl_interp (e01rac) ..... nag_1d_ratnl_eval (e01rbc)
from nag_2d_shep_interp (e01sgc) ..... nag_2d_shep_eval (e01shc)
from nag_2d_triang_interp (e01sjc) ..... nag_2d_triang_eval (e01skc)
from nag_3d_shep_interp (e01tgc) ..... nag_3d_shep_eval (e01thc)
from nag_4d_shep_interp (e01tkc) ..... nag_4d_shep_eval (e01tlc)
from nag_5d_shep_interp (e01tmc) ..... nag_5d_shep_eval (e01tnc)
from nag_nd_shep_interp (e01zmc) ..... nag_nd_shep_eval (e01znc)
from triangulation from nag_2d_triangulate (e01eac) ..... nag_2d_triang_bary_eval (e01ebc)

Extrapolation,
one variable,
piecewise cubic ..... nag_monotonic_interpolant (e01bec)
polynomial,
data with or without derivatives ..... nag_1d_cheb_interp (e01aec)
general data ..... nag_1d_aitken_interp (e01aac)
rational function ..... nag_1d_ratnl_interp (e01rac)

Integration (definite) of interpolant from nag_monotonic_interpolant (e01bec)
..... nag_monotonic_intg (e01bhc)

Interpolated values,
d variables,
from interpolant from nag_nd_shep_interp (e01zmc) ..... nag_nd_shep_eval (e01znc)
five variables,
from interpolant from nag_5d_shep_interp (e01tmc) ..... nag_5d_shep_eval (e01tnc)
four variables,
from interpolant from nag_4d_shep_interp (e01tkc) ..... nag_4d_shep_eval (e01tlc)
one variable,
from interpolant from nag_monotonic_interpolant (e01bec) ..... nag_monotonic_evaluate (e01bfc)
from interpolant from nag_monotonic_interpolant (e01bec) (including derivative)
..... nag_monotonic_deriv (e01bgc)
from polynomial,
equally spaced data ..... nag_1d_everett_interp (e01abc)
general data ..... nag_1d_aitken_interp (e01aac)
from rational function ..... nag_1d_ratnl_eval (e01rbc)
three variables,
from interpolant from nag_3d_shep_interp (e01tgc) ..... nag_3d_shep_eval (e01thc)
two variables,
barycentric, from triangulation from nag_2d_triangulate (e01eac)
..... nag_2d_triang_bary_eval (e01ebc)
from interpolant from nag_2d_shep_interp (e01sgc) ..... nag_2d_shep_eval (e01shc)
from interpolant from nag_2d_triang_interp (e01sjc) ..... nag_2d_triang_eval (e01skc)

Interpolating function,
d variables,
modified Shepard method ..... nag_nd_shep_interp (e01zmc)
five variables,
modified Shepard method ..... nag_5d_shep_interp (e01tmc)
four variables,
modified Shepard method ..... nag_4d_shep_interp (e01tkc)
one variable,
cubic spline ..... nag_1d_spline_interpolant (e01bac)
other piecewise polynomial ..... nag_monotonic_interpolant (e01bec)
polynomial,
data with or without derivatives ..... nag_1d_cheb_interp (e01aec)
rational function ..... nag_1d_ratnl_interp (e01rac)

three variables,
modified Shepard method ..... nag_3d_shep_interp (e01tgc)
two variables,
bicubic spline ..... nag_2d_spline_interpolant (e01dac)
modified Shepard method ..... nag_2d_shep_interp (e01sgc)
other piecewise polynomial ..... nag_2d_triang_interp (e01sjc)
triangulation ..... nag_2d_triangulate (e01eac)

6 Auxiliary Functions Associated with Library Function Arguments

None.

7 Functions Withdrawn or Scheduled for Withdrawal

The following lists all those functions that have been withdrawn since Mark 23 of the Library or are scheduled for withdrawal at one of the next two marks.

Withdrawn Function	Mark of Withdrawal	Replacement Function(s)
nag_2d_scat_interpolant (e01sac)	23	nag_2d_shep_interp (e01sgc) or nag_2d_triang_interp (e01sjc)
nag_2d_scat_eval (e01sbc)	23	nag_2d_shep_eval (e01shc) or nag_2d_triang_eval (e01skc)
nag_2d_scat_free (e01szc)	23	No longer required.

8 References

Dahlquist G and BjÎrck Đ (1974) *Numerical Methods* Prentice–Hall

FrÎberg C E (1970) *Introduction to Numerical Analysis* Addison–Wesley