Chapter e02 – Curve and Surface Fitting

1. Scope of the Chapter

The main aim of this chapter is to assist the user in finding a function which approximates a set of data points. Typically the data contain random errors, as of experimental measurement, which need to be smoothed out. To seek an approximation to the data, it is first necessary to specify for the approximating function a mathematical form (a polynomial, for example) which contains a number of unspecified coefficients: the appropriate fitting routine then derives for the coefficients the values which provide the best fit of that particular form. The chapter deals mainly with curve and surface fitting (i.e. fitting with functions of one and of two variables) when a polynomial or a cubic spline is used as the fitting function, since these cover the most common needs. However, fitting with other functions and/or more variables can be undertaken by means of general linear or nonlinear routines (some of which are contained in other chapters) depending on whether the coefficients in the function occur linearly or nonlinearly. Cases where a graph rather than a set of data points is given can be treated simply by first reading a suitable set of points from the graph.

The chapter also contains routines for evaluating polynomial and spline curves and surfaces, and differentiating or integrating spline curves once the numerical values of their coefficients have been determined.

2. Background

2.1. Preliminary Considerations

In the curve-fitting problems considered in this chapter, we have a dependent variable y and an independent variable x, and we are given a set of data points (x_r, y_r), for $r = 1, 2, \ldots, m$. The preliminary matters to be considered in this section will, for simplicity, be discussed in this context of curve-fitting problems. In fact, however, these considerations apply equally well to surface and higher-dimensional problems. Indeed, the discussion presented carries over essentially as it stands if, for these cases, we interpret x as a vector of several independent variables and correspondingly each x_r as a vector containing the rth data value of each independent variable.

We wish, then, to approximate the set of data points as closely as possible with a specified function, $f(x)$ say, which is as smooth as possible: $f(x)$ may, for example, be a polynomial. The requirements of smoothness and closeness conflict, however, and a balance has to be struck between them. Most often, the smoothness requirement is met simply by limiting the number of coefficients allowed in the fitting function – for example, by restricting the degree in the case of a polynomial. Given a particular number of coefficients in the function in question, the fitting routines of this chapter determine the values of the coefficients such that the 'distance' of the function from the data points is as small as possible. The necessary balance is struck by the user comparing a selection of such fits having different numbers of coefficients. If the number of coefficients is too low, the approximation to the data will be poor. If the number is too high, the fit will be too close to the data, essentially following the random errors and tending to have unwanted fluctuations between the data points. Between these extremes, there is often a group of fits all similarly close to the data points and then, particularly when least-squares polynomials are used, the choice is clear: it is the fit from this group having the smallest number of coefficients.

The above process can be seen as the user minimizing the smoothness measure (i.e. the number of coefficients) subject to the distance from the data points being acceptably small. Some of the routines, however, do this task themselves. They use a different measure of smoothness (in each case one that is continuous) and minimize it subject to the distance being less than a threshold specified by the user. This is a much more automatic process, requiring only some experimentation with the threshold.

2.1.1. Fitting criteria: the l_2 norm

A measure of the above 'distance' between the set of data points and the function $f(x)$ is needed. The distance from a single data point (x_r, y_r) to the function can simply be taken as

$$\epsilon_r = y_r - f(x_r), \tag{1}$$

and is called the **residual** of the point. (With this definition, the residual is regarded as a function of the coefficients contained in $f(x)$; however, the term is also used to mean the particular value of ϵ_r which corresponds to the fitted values of the coefficients.) However, we need a measure of distance for the set of data points as a whole. The measure used by the routines of this chapter is the l_2 norm. With ϵ_r defined in (1), the l_2 norm is given by:

$$\sqrt{\sum_{r=1}^{m} \epsilon_r^2}. \tag{2}$$

Minimization of this norm usually provides the fitting criterion, the minimization being carried out with respect to the coefficients in the mathematical form used for $f(x)$: with respect to the b_i for example if the mathematical form is the power series in (4) below. The fit which results from minimizing (2) is the l_2 fit, the well-known least-squares fit (minimizing (2) is equivalent to minimizing the square of (2), i.e. the sum of squares of residuals, and it is the latter which is used in practice).

Some of the routines in this chapter do not minimize the l_2 norm itself but instead minimize some (intuitively acceptable) measure of smoothness subject to the norm being less than a user-specified threshold. These routines fit with cubic or bicubic splines (see (6) and (8) below) and the smoothing measures relate to the size of the discontinuities in their third derivatives. A much more automatic fitting procedure follows from this approach.

2.1.2. Weighting of data points

The use of the above norm assumes that the data values y_r are of equal (absolute) accuracy. Some of the routines enable an allowance to be made to take account of differing accuracies. The allowance takes the form of 'weights' applied to the y-values so that those values known to be more accurate have a greater influence on the fit than others. These weights, to be supplied by the user, should be calculated from estimates of the absolute accuracies of the y-values, these estimates being expressed as standard deviations, probable errors or some other measure which has the same dimensions as y. Specifically, for each y_r the corresponding weight w_r should be inversely proportional to the accuracy estimate of y_r. For example, if the percentage accuracy is the same for all y_r, then the absolute accuracy of y_r is proportional to y_r (assuming y_r to be positive, as it usually is in such cases) and so $w_r = K/y_r$, for $r = 1, 2, \ldots, m$, for an arbitrary positive constant K. (This definition of weight is stressed because often weight is defined as the square of that used here.) The norm (2) above is then replaced by

$$\sqrt{\sum_{r=1}^{m} w_r^2 \epsilon_r^2}. \tag{3}$$

Again it is the square of (3) which is used in practice rather than (3) itself.

2.2. Curve Fitting

When, as is commonly the case, the mathematical form of the fitting function is immaterial to the problem, polynomials and cubic splines are to be preferred because their simplicity and ease of handling confer substantial benefits. The **cubic spline** is the more versatile of the two. It consists of a number of cubic polynomial segments joined end to end with continuity in first and second derivatives at the joins. The third derivative at the joins is in general discontinuous. The x-values of the joins are called **knots**, or, more precisely, interior knots. Their number determines the number of coefficients in the spline, just as the degree determines the number of coefficients in a polynomial.

2.2.1. Representation of polynomials

Two different forms for representing a polynomial are used in different routines. One is the usual power-series form

$$f(x) \equiv b_0 + b_1 x + b_2 x^2 + \ldots + b_k x^k. \tag{4}$$

The other is the Chebyshev series form

$$f(x) \equiv \tfrac{1}{2} a_0 T_0(x) + a_1 T_1(x) + a_2 T_2(x) + \ldots + a_k T_k(x), \tag{5}$$

where $T_i(x)$ is the Chebyshev polynomial of the first kind of degree i (see Cox and Hayes (1973), page 9), and where the range of x has been normalised to run from -1 to $+1$. The use of either form leads theoretically to the same fitted polynomial, but in practice results may differ substantially because of the effects of rounding error. The Chebyshev form is to be preferred, since it leads to much better accuracy in general, both in the computation of the coefficients and in the subsequent evaluation of the fitted polynomial at specified points. This form also has other advantages: for example, since the later terms in (5) generally decrease much more rapidly from left to right than do those in (4), the situation is more often encountered where the last terms are negligible and it is obvious that the degree of the polynomial can be reduced (note that on the interval $-1 \le x \le 1$ for all i, $T_i(x)$ attains the value unity but never exceeds it, so that the coefficient a_i gives directly the maximum value of the term containing it). If the power-series form is used it is most advisable to work with the variable x normalised to the range -1 to $+1$, carrying out the normalisation before entering the relevant routine. This will often substantially improve computational accuracy.

2.2.2. Representation of cubic splines

A cubic spline is represented in the form

$$f(x) \equiv c_1N_1(x) + c_2N_2(x) + \ldots + c_pN_p(x), \tag{6}$$

where $N_i(x)$, for $i = 1,2,\ldots,p$, is a normalised cubic B-spline (see Hayes (1974)). This form, also, has advantages of computational speed and accuracy over alternative representations.

2.3. Surface Fitting

There are now two independent variables, and we shall denote these by x and y. The dependent variable, which was denoted by y in the curve-fitting case, will now be denoted by f. (This is a rather different notation from that indicated for the general-dimensional problem in the first paragraph of Section 2.1, but it has some advantages in presentation.)

Again, in the absence of contrary indications in the particular application being considered, splines are the approximating functions most commonly used.

2.3.1. Bicubic splines: definition and representation

The bicubic spline is defined over a rectangle R in the (x,y) plane, the sides of R being parallel to the x- and y-axes. R is divided into rectangular panels, again by lines parallel to the axes. Over each panel the bicubic spline is a bicubic polynomial, that is it takes the form

$$\sum_{i=0}^{3}\sum_{j=0}^{3} a_{ij}x^iy^j. \tag{7}$$

Each of these polynomials joins the polynomials in adjacent panels with continuity up to the second derivative. The constant x-values of the dividing lines parallel to the y-axis form the set of interior knots for the variable x, corresponding precisely to the set of interior knots of a cubic spline. Similarly, the constant y-values of dividing lines parallel to the x-axis form the set of interior knots for the variable y. Instead of representing the bicubic spline in terms of the above set of bicubic polynomials, however, it is represented, for the sake of computational speed and accuracy, in the form

$$f(x,y) = \sum_{i=1}^{p}\sum_{j=1}^{q} c_{ij}M_i(x)N_j(y), \tag{8}$$

where $M_i(x)$, for $i = 1,2,\ldots,p$, and $N_j(y)$, for $j = 1,2,\ldots,q$, are normalised B-splines (see Hayes and Halliday (1974) for further details of bicubic splines and Hayes (1974) for normalised B-splines).

3. Recommendations on Choice and Use of Available Routines

3.1. General

The choice of a routine to treat a particular fitting problem will depend first of all on the fitting function chosen. If there is only one independent variable, the user should choose a spline (Section

3.3) when the curve represented by the data is of complicated form, perhaps with several peaks and troughs. When the curve is of simple form, first try a polynomial (see Section 3.2) of low degree, say up to degree 5 or 6, and then a spline if the polynomial fails to provide a satisfactory fit. (Of course, if third-derivative discontinuities are unacceptable to the user, a polynomial is the only choice.) If the problem is one of surface fitting, one of the spline routines (Section 3.4.1) will be required.

3.1.1. Data considerations

A satisfactory fit cannot be expected by any means if the number and arrangement of the data points do not adequately represent the character of the underlying relationship: sharp changes in behaviour, in particular, such as sharp peaks, should be well covered. Data points should extend over the whole range of interest of the independent variable(s): extrapolation outside the data ranges is most unwise. Then, with polynomials, it is advantageous to have additional points near the ends of the ranges, to counteract the tendency of polynomials to develop fluctuations in these regions. When, with polynomial curves, the user can precisely choose the x-values of the data, the special points defined in Section 3.2.2 should be selected. With splines the choice is less critical as long as the character of the relationship is adequately represented. All fits should be tested graphically before accepting them as satisfactory.

For this purpose it should be noted that it is not sufficient to plot the values of the fitted function only at the data values of the independent variable(s); at the least, its values at a similar number of intermediate points should also be plotted, as unwanted fluctuations may otherwise go undetected. Such fluctuations are the less likely to occur the lower the number of coefficients chosen in the fitting function. No firm guide can be given, but as a rough rule, at least initially, the number of coefficients should not exceed half the number of data points (points with equal or nearly equal values of the independent variable, or both independent variables in surface fitting, counting as a single point for this purpose). However, the situation may be such, particularly with a small number of data points, that a satisfactorily close fit to the data cannot be achieved without unwanted fluctuations occurring. In such cases, it is often possible to improve the situation by a transformation of one or more of the variables, as discussed in the next section: otherwise it will be necessary to provide extra data points. Further advice on curve fitting is given in Cox and Hayes (1973) and, for polynomials only, in Hayes (1970). Much of the advice applies also to surface fitting; see also the routine documents.

3.1.2. Transformation of variables

Before starting the fitting, consideration should be given to the choice of a good form in which to deal with each of the variables: often it will be satisfactory to use the variables as they stand, but sometimes the use of the logarithm, square root, or some other function of a variable will lead to a better-behaved relationship. This question is customarily taken into account in preparing graphs and tables of a relationship and the same considerations apply when curve or surface fitting. The practical context will often give a guide. In general, it is best to avoid having to deal with a relationship whose behaviour in one region is radically different from that in another. A steep rise at the left-hand end of a curve, for example, can often best be treated by curve fitting in terms of $\log(x + c)$ with some suitable value of the constant c. A case when such a transformation gave substantial benefit is discussed in page 60 of Hayes (1970). According to the features exhibited in any particular case, transformation of either dependent variable or independent variable(s) or both may be beneficial. When there is a choice it is usually better to transform the independent variable(s): if the dependent variable is transformed, the weights attached to the data points must be adjusted. Thus (denoting the dependent variable by y, as in the notation for curves) if the y_r to be fitted have been obtained by a transformation $y = g(Y)$ from original data values Y_r, with weights W_r, for $r = 1, 2, \ldots, m$, we must take

$$w_r = W_r/(dy/dY), \tag{9}$$

where the derivative is evaluated at Y_r. Strictly, the transformation of Y and the adjustment of weights are valid only when the data errors in the Y_r are small compared with the range spanned by the Y_r, but this is usually the case.

3.2. Polynomial Curves

3.2.1. Least-squares polynomials: arbitrary data points

nag_1d_cheb_fit (e02adc) fits to arbitrary data points, with arbitrary weights, polynomials of all degrees up to a maximum degree k, which is a choice. If the user is seeking only a low-degree polynomial, up to degree 5 or 6 say, $k = 10$ is an appropriate value, providing there are about 20 data points or more. To assist in deciding the degree of polynomial which satisfactorily fits the data, the routine provides the root-mean-square residual s_i for all degrees $i = 1, 2, \ldots, k$. In a satisfactory case, these s_i will decrease steadily as i increases and then settle down to a fairly constant value, as shown in the example

i	s_i
0	3.5215
1	0.7708
2	0.1861
3	0.0820
4	0.0554
5	0.0251
6	0.0264
7	0.0280
8	0.0277
9	0.0297
10	0.0271

If the s_i values settle down in this way, it indicates that the closest polynomial approximation justified by the data has been achieved. The degree which first gives the approximately constant value of s_i (degree 5 in the example) is the appropriate degree to select. (Users who are prepared to accept a fit higher than sixth degree should simply find a high enough value of k to enable the type of behaviour indicated by the example to be detected: thus they should seek values of k for which at least 4 or 5 consecutive values of s_i are approximately the same.) If the degree were allowed to go high enough, s_i would, in most cases, eventually start to decrease again, indicating that the data points are being fitted too closely and that undesirable fluctuations are developing between the points. In some cases, particularly with a small number of data points, this final decrease is not distinguishable from the initial decrease in s_i. In such cases, users may seek an acceptable fit by examining the graphs of several of the polynomials obtained. Failing this, they may (a) seek a transformation of variables which improves the behaviour, (b) try fitting a spline, or (c) provide more data points. If data can be provided simply by drawing an approximating curve by hand and reading points from it, use the points discussed in Section 3.2.2.

3.2.2. Least-squares polynomials: selected data points

When users are at liberty to choose the x-values of data points, such as when the points are taken from a graph, it is most advantageous when fitting with polynomials to use the values $x_r = \cos(\pi r/n)$, for $r = 0, 1, \ldots, n$ for some value of n, a suitable value for which is discussed at the end of this section. Note that these x_r relate to the variable x after it has been normalised so that its range of interest is -1 to $+1$. nag_1d_cheb_fit (e02adc) may then be used as in Section 3.2.1 to seek a satisfactory fit. However, if the ordinate values are of equal weight, as would often be the case when they are read from a graph, nag_1d_cheb_interp_fit (e02afc) is to be preferred, as being simpler to use and faster. This latter algorithm provides the coefficients a_j, for $j = 0, 1, \ldots, n$, in the Chebyshev series form of the polynomial of degree n which interpolates the data. In a satisfactory case, the later coefficients in this series, after some initial significant ones, will exhibit a random behaviour, some positive and some negative, with a size about that of the errors in the data or less. All these 'random' coefficients should be discarded, and the remaining (initial) terms of the series be taken as the approximating polynomial. This truncated polynomial is a least-squares fit to the data, though with the point at each end of the range given half the weight of each of the other points. The following example illustrates a case in which degree 5 or perhaps 6 would be chosen for the approximating polynomial.

j	a_j
0	9.315
1	−8.030
2	0.303
3	−1.483
4	0.256
5	−0.386
6	0.076
7	0.022
8	0.014
9	0.005
10	0.011
11	−0.040
12	0.017
13	−0.054
14	0.010
15	−0.034
16	−0.001

Basically, the value of n used needs to be large enough to exhibit the type of behaviour illustrated in the above example. A value of 16 is suggested as being satisfactory for very many practical problems, the required cosine values for this value of n being given in Cox and Hayes (1973), page 11. If a satisfactory fit is not obtained, a spline fit should be tried, or, if the user is prepared to accept a higher degree of polynomial, n should be increased: doubling n is an advantageous strategy, since the set of values $\cos(\pi r/n)$, for $r = 0, 1, \ldots, n$, contains all the values of $\cos(\pi r/2n)$, for $r = 0, 1, \ldots, 2n$, so that the old data set will then be re-used in the new one. Thus, for example, increasing n from 16 to 32 will require only 16 new data points, a smaller number than for any other increase of n. If data points are particularly expensive to obtain, a smaller initial value than 16 may be tried, provided the user is satisfied that the number is adequate to reflect the character of the underlying relationship. Again, the number should be doubled if a satisfactory fit is not obtained.

3.3. Cubic Spline Curves

3.3.1. Least-squares cubic splines

nag_1d_spline_fit_knots (e02bac) fits to arbitrary data points, with arbitrary weights, a cubic spline with interior knots specified by the user. The choice of these knots so as to give an acceptable fit must largely be a matter of trial and error, though with a little experience a satisfactory choice can often be made after one or two trials. It is usually best to start with a small number of knots (too many will result in unwanted fluctuations in the fit, or even in there being no unique solution) and, examining the fit graphically at each stage, to add a few knots at a time at places where the fit is particularly poor. Moving the existing knots towards these places will also often improve the fit. In regions where the behaviour of the curve underlying the data is changing rapidly, closer knots will be needed than elsewhere. Otherwise, positioning is not usually very critical and equally-spaced knots are often satisfactory. See also the next section, however.

A useful feature of the routine is that it can be used in applications which require the continuity to be less than the normal continuity of the cubic spline. For example, the fit may be required to have a discontinuous slope at some point in the range. This can be achieved by placing three coincident knots at the given point. Similarly a discontinuity in the second derivative at a point can be achieved by placing two knots there. Analogy with these discontinuous cases can provide guidance in more usual cases: for example, just as three coincident knots can produce a discontinuity in slope, so three close knots can produce a rapid change in slope. The closer the knots are, the more rapid can the change be.

An example set of data is given in Figure 1. It is a rather tricky set, because of the scarcity of data on the right, but it will serve to illustrate some of the above points and to show some of the dangers to be avoided. Three interior knots (indicated by the vertical lines at the top of the diagram) are chosen as a start. We see that the resulting curve is not steep enough in the middle and fluctuates at both ends, severely on the right. The spline is unable to cope with the shape and more knots are needed.

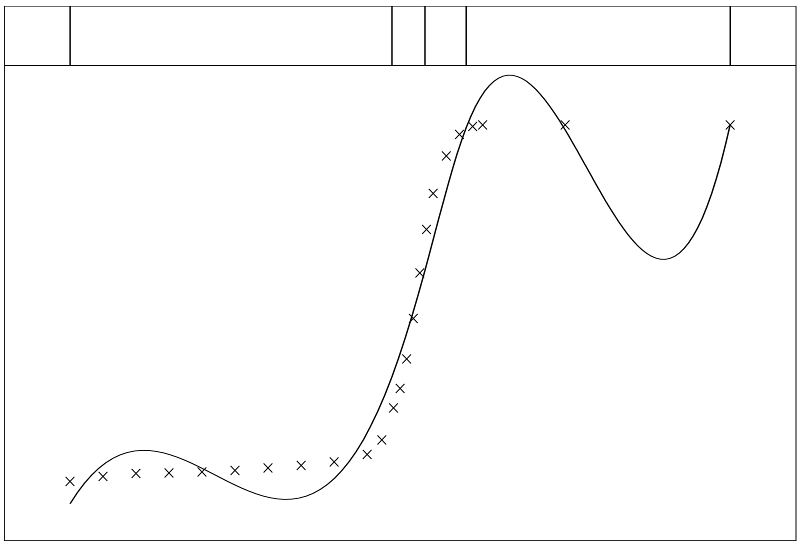

Figure 1

In Figure 2, three knots have been added in the centre, where the data shows a rapid change in behaviour, and one further out at each end, where the fit is poor. The fit is still poor, so a further knot is added in this region and, in Figure 3, disaster ensues in rather spectacular fashion.

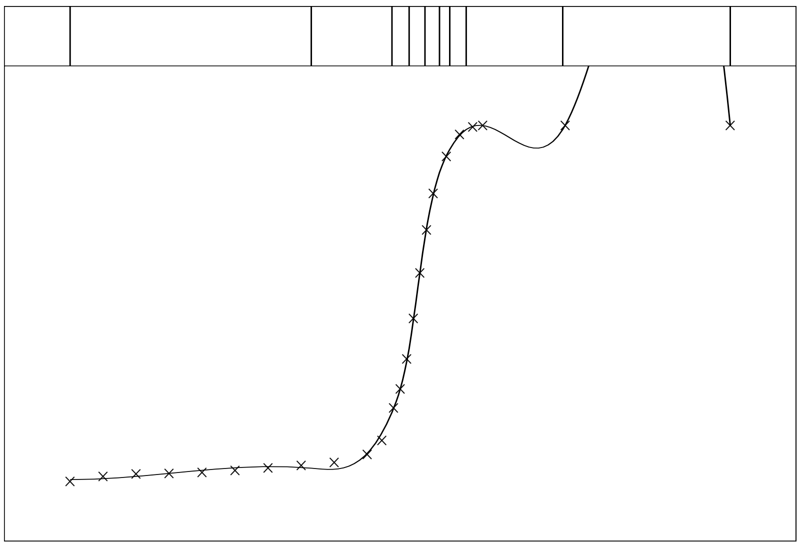

Figure 2

The reason is that, at the right-hand end, the fits in Figure 1 and Figure 2 have been interpreted as poor simply because of the fluctuations about the curve underlying the data (or what it is naturally assumed to be). But the fitting process knows only about the data and nothing else about the underlying curve, so it is important to consider only closeness to the data when deciding goodness of fit.

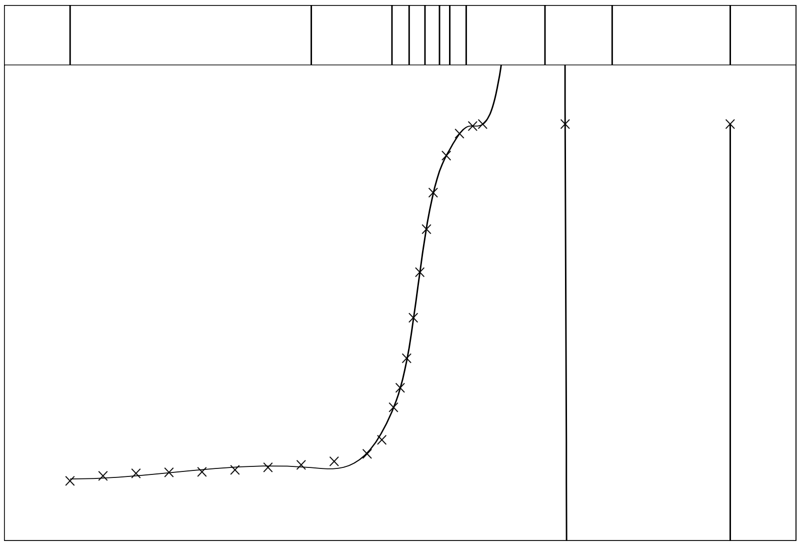

Figure 3

Thus, in Figure 1, the curve fits the last two data points quite well compared with the fit elsewhere, so no knot should have been added in this region. In Figure 2, the curve goes exactly through the last two points, so a further knot is certainly not needed here.

Figure 4 shows what can be achieved without the extra knot on each of the flat regions. Remembering that within each knot interval the spline is a cubic polynomial, there is really no need to have more than one knot interval covering each flat region.

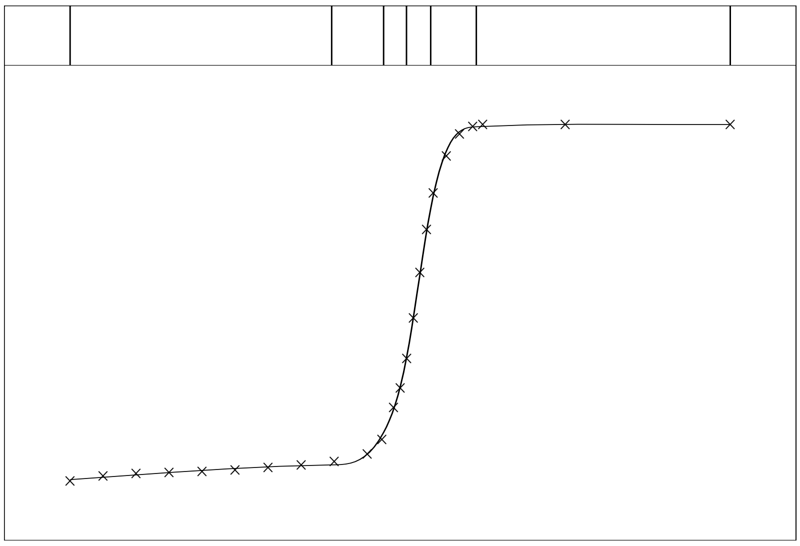

Figure 4

What we have, in fact, in Figure 2 and Figure 3 is a case of too many knots (so too many coefficients in the spline equation) for the number of data points. The warning in the second paragraph of Section 2.1 was that the fit will then be too close to the data, tending to have unwanted fluctuations between the data points. The warning applies locally for splines, in the sense that, in localities where there are plenty of data points, there can be a lot of knots, as long as there are few knots where there are few points, especially near the ends of the interval. In the present example, with so few data points on the right, just the one extra knot in Figure 2 is too many! The signs are clearly present, with the last two points fitted exactly (at least to the graphical accuracy and actually much closer than that) and fluctuations **within** the last two knot-intervals (cf. Figure 1, where only the final point is fitted exactly and one of the wobbles spans several data points).

The situation in Figure 3 is different. The fit, if computed exactly, **would** still pass through the last two data points, with even more violent fluctuations. However, the problem has become so ill-conditioned that all accuracy has been lost. Indeed, if the last interior knot were moved a tiny amount to the right, there would be no unique solution and an error message would have been caused. **Near-singularity** is, sadly, not picked up by the routine, but can be spotted readily in a graph, as Figure 3. B-spline coefficients becoming large, with alternating signs, is another indication. However, it is better to avoid such situations, firstly by providing, whenever possible, data adequately covering the range of interest, and secondly by placing knots only where there is a reasonable amount of data.

The example here could, in fact, have utilised from the start the observation made in the second paragraph of this section, that three close knots can produce a rapid change in slope. The example has two such rapid changes and so requires two sets of three close knots (in fact, the two sets can be so close that one knot can serve in both sets, so only five knots prove sufficient in Figure 4). It should be noted, however, that the rapid turn occurs within the range spanned by the three knots. This is the reason that the six knots in Figure 2 are not satisfactory as they do not quite span the two turns.

Some more examples to illustrate the choice of knots are given in Cox and Hayes (1973).

3.3.2. Automatic fitting with cubic splines

nag_1d_spline_fit (e02bec) also fits cubic splines to arbitrary data points with arbitrary weights but itself chooses the number and positions of the knots. The user has to supply only a threshold for the sum of squares of residuals. The routine first builds up a knot set by a series of trial fits in the l_2 norm. Then, with the knot set decided, the final spline is computed to minimize a certain smoothing measure subject to satisfaction of the chosen threshold. Thus it is easier to use than nag_1d_spline_fit_knots (e02bac) (see previous section), requiring only some experimentation with this threshold. It should therefore be first choice unless the user has a preference for the ordinary least-squares fit or, for example, wishes to experiment with knot positions, trying to keep their number down (nag_1d_spline_fit (e02bec) aims only to be reasonably frugal with knots).

3.4. Spline Surfaces

3.4.1. Automatic fitting with bicubic splines

nag_2d_spline_fit_scat (e02ddc) fits bicubic splines to arbitrary data points with arbitrary weights choosing the knot sets itself. The user has to supply only a threshold for the sum of squares of residuals. Just like the automatic curve nag_1d_spline_fit (e02bec) (Section 3.3.2), nag_2d_spline_fit_scat (e02ddc) then builds up the knot sets and finally fits a spline minimizing a smoothing measure subject to satisfaction of the threshold.

nag_2d_spline_fit_grid (e02dcc) is a very similar routine to nag_2d_spline_fit_scat (e02ddc) but deals with data points of equal weight which lie on a rectangular mesh in the (x, y) plane. This kind of data allows a very much faster computation and so is to be preferred when applicable. Substantial departures from equal weighting can be ignored if the user is not concerned with statistical questions, though the quality of the fit will suffer if this is taken too far. In such cases, the user should revert to nag_2d_spline_fit_scat (e02ddc).

3.5. Evaluation, Differentiation and Integration

Routines are available to evaluate, differentiate and integrate polynomials in Chebyshev-series form and cubic or bicubic splines in B-spline form. These polynomials and splines may have been produced by the various fitting routines or, in the case of polynomials, from prior calls of the differentiation and integration routines themselves.

nag_1d_cheb_eval (e02aec) evaluates polynomial curves, nag_1d_spline_evaluate (e02bbc) evaluates cubic spline curves, and nag_2d_spline_eval (e02dec) and nag_2d_spline_eval_rect (e02dfc) bicubic spline surfaces.

There are also two routines for computing derivatives and integrals of splines. nag_1d_spline_deriv (e02bcc) provides values of a cubic spline curve and its first three derivatives (the rest, of course, are zero) at a given value of x. nag_1d_spline_intg (e02bdc) computes the value of the definite integral of a cubic spline over its whole range. The routines can also be applied to surfaces of the form (8).

For example, if, for each value of j in turn, the coefficients c_{ij}, for $i = 1, 2, \ldots, p$, are supplied to nag_1d_spline_deriv (e02bcc) with $x = x_0$ and on each occasion we select from the output the value of the second derivative, d_j say, and if the whole set of d_j are then supplied to the same routine with $x = y_0$, the output will contain all the values at (x_0, y_0) of

$$\frac{\partial^2 f}{\partial x^2} \text{ and } \frac{\partial^{r+2} f}{\partial x^2 \partial y^r}, \quad r = 1, 2, 3.$$

Equally, if after each of the first p calls of nag_1d_spline_deriv (e02bcc) we had selected the function value (nag_1d_spline_evaluate (e02bbc) would also provide this) instead of the second derivative and we had supplied these values to nag_1d_spline_intg (e02bdc), the result obtained would have been the value of

$$\int_A^B f(x_0, y)\, dy,$$

where A and B are the end-points of the y interval over which the spline was defined.

4. Index

Automatic fitting,	
with bicubic splines	`e02dcc`
	`e02ddc`
with cubic splines	`e02bec`
Data on rectangular mesh	`e02dcc`
Differentiation,	
of cubic splines	`e02bcc`
Evaluation,	
of bicubic splines	`e02dec`
	`e02dfc`
of cubic splines	`e02bbc`
of cubic splines and derivatives	`e02bcc`
of definite integral of cubic splines	`e02bdc`
of polynomials,	
in one variable	`e02aec`
Integration,	
of cubic splines (definite integral)	`e02bdc`
Least-squares curve fit,	
with cubic splines	`e02bac`
with polynomials,	
arbitrary data points	`e02adc`
selected data points	`e02afc`

5. References

Baker G A (1975) *Essentials of Padé Approximants* Academic Press, New York

Cox M G and Hayes J G (1973) Curve Fitting: A guide and suite of algorithms for the non-specialist user *NPL Report NAC 26* National Physical Laboratory

Hayes J G (1974) Numerical methods for curve and surface fitting *Bull. Inst. Math. Appl.* **10** 144–152

Hayes J G (ed.) (1970) Curve fitting by polynomials in one variable *Numerical Approximation to Functions and Data* Athlone Press, London

Hayes J G (ed.) (1970) Fitting data in more than one variable *Numerical Approximation to Functions and Data* Athlone Press, London

Hayes J G and Halliday J (1974) The least-squares fitting of cubic spline surfaces to general data sets *J. Inst. Math. Appl.* **14** 89–103