# A method of storing the orthogonal polynomials used for curve and surface fitting

*By* D. G. Hayes*

**This paper describes a method of calculating and storing the orthogonal polynomials used in curve and surface fitting. The method saves considerable store space and time, and makes the amount of space needed independent of the number of data points.**

(Received August 1967, revised July 1968)

This paper is concerned with the problem of finding a polynomial approximation to a function of one or two variables, given the values of the function at a number of data points. If the $x$-coordinates of the data points are not distributed according to any definite law, one uses the method of Forsythe (1957). This method makes use of polynomials orthogonal over the given data points.

Forsythe suggests that the values at each of the data points of all the orthogonal polynomials used in the recurrence relation should be stored. This requires considerable store space. The space problem is worse in the case of surface fitting (Weisfeld, 1957), because the recurrence relation now contains more than three terms, and the number of points is usually greater.

The method of storing polynomials, to be described here, makes the space needed independent of the number of data points, as well as shortening the time of computation.

## Curve fitting

Let the data points be $(x_0, y_0), (x_1, y_1), \ldots$. Forsythe's method consists of generating polynomials $P_r(x)$ orthogonal over the data points using the recurrence relation

$$P_r(x) = \lambda_r(x - \alpha_r)P_{r-1}(x) - \beta_r P_{r-2}(x) \quad (1)$$

with initial polynomials

$$P_0(x) = \lambda_0 \qquad P_1(x) = \lambda_1\lambda_0(x - \bar{x})$$

where $\alpha$ and $\beta$ are given by

$$\alpha_r = \frac{(xP_{r-1}, P_{r-1})}{(P_{r-1}, P_{r-1})} \quad \beta_r = \frac{\lambda_r(P_{r-1}, P_{r-1})}{\lambda_{r-1}(P_{r-2}, P_{r-2})}. \quad (2)$$

The notation $(F, G)$ is used here and henceforth to denote the scalar product $\sum_i F(x_i)G(x_i)$ summed over all the given data points, for any two functions $F(x)$, $G(x)$.

The polynomial approximation $H_n(x)$ of degree $n$ is then given by

$$H_n(x) = \sum_{j=0}^{n} c_j P_j(x) \quad (3)$$

where $$c_j = (P_j, y)/(P_j, P_j). \quad (4)$$

## Surface fitting

In the case of surface fitting, we choose orthogonal polynomials of the following form, where $x$ and $y$ are the independent variables, and $z$ the dependent variable.

$$\begin{aligned} P_{0,0} &= a_0 \\ P_{1,0} &= a_1 + b_1x \\ P_{1,1} &= a_2 + b_2x + c_2y \\ P_{2,0} &= a_3 + b_3x + c_3y + d_3x^2 \\ P_{2,1} &= a_4 + b_4x + c_4y + d_4x^2 + e_4xy \\ &\ldots\ldots\ldots\ldots\ldots\ldots \end{aligned}$$

In general, $P_{r,s}$ is a polynomial of degree $r$. It may contain any term of degree less than $r$, and also terms in $x^r$, $x^{r-1}y$, $x^{r-2}y^2, \ldots, x^{r-s}y^s$, but no terms in $x^{r-s-1}y^{s+1}, \ldots, y^r$.

These polynomials may be found from the relation

$$P_{n,m} = \lambda_{n,m}xP_{n-1,m} - \sum_{r,s} \alpha_{r,s}P_{r,s} \quad (5)$$

where the summation extends over

(i) $r = n, s < m$
(ii) $r = n - 1$, all $s$
(iii) $r = n - 2, s \geqslant m$

and $$\alpha_{r,s} = \lambda_{n,m}(xP_{n-1,m}, P_{r,s})/(P_{r,s}, P_{r,s}). \quad (6)$$

A different recurrence relation must be used for the polynomial $P_{n,n}$. This polynomial is given by

$$P_{n,n} = \lambda_{n,n}yP_{n-1,n-1} - \sum_{r,s} \alpha_{r,s}P_{r,s} \quad (7)$$

where the summation extends over

(i) $r = n, s < n$
(ii) $r = n - 1$, all $s$
(iii) $r = s = n - 2$,

and $$\alpha_{r,s} = \lambda_{n,n}(yP_{n-1,n-1}, P_{r,s})/(P_{r,s}, P_{r,s}). \quad (8)$$

The polynomial approximation is now given by

$$H_n(x, y) = \sum_{r=0}^{n} \sum_{s=0}^{r} c_{r,s}P_{r,s}$$

where $$c_{r,s} = (P_{r,s}, z)/(P_{r,s}, P_{r,s}). \quad (9)$$

## Modified method of calculating and storing the orthogonal polynomials

The method of calculating and storing the orthogonal polynomials will be described in detail as applied to curve fitting, although it is likely to be of greater use for surface fitting. In the case of surface fitting, the equations which need to be used are similar to those used for curve fitting, but more cumbersome to write down.

One obvious method of storing the polynomials is to store the values of the polynomials at each of the data

* *Mathematical Computing Department, Brush Electrical Engineering Co. Ltd., Loughborough (at present at Caius College, Cambridge)*

points. Equation (3) then gives the approximating function $H_n(x)$ as the sum of a series of orthogonal polynomials. It is usual to convert this into a series of powers of $x$ by a further use of the recurrence relation (1).

Alternatively, $H_n(x)$ may be expressed as a series of Chebyshev polynomials (Clenshaw, 1960). The same author recommends the storing of the orthogonal polynomials in terms of the coefficients in their Chebyshev expansions throughout the computation. These expansions are used to recalculate the values of the orthogonal polynomials at the data points when necessary.

The modified method makes use of the quantities $a_0, a_1, \ldots, a_n$ and $b_0, b_1, b_2, \ldots$, defined for any polynomial $Q(x)$ by

$$Q(x) = a_0T_0(x) + a_1T_1(x) + \ldots + a_nT_n(x) \quad (10)$$

where $n$ = degree of $Q(x)$

$$(Q, T_r) = b_r \quad (11)$$

$T_r(x)$ denotes the Chebyshev polynomial of degree $r$.

The method consists of calculating and storing the values of $a_r$, and as many as necessary of the values of $b_r$, for each orthogonal polynomial.

It will be shown that the recurrence relation (1) can operate entirely in terms of the values of $a$ and $b$ of the orthogonal polynomials. There is thus no need to store the coordinates of the data points, or the values of any orthogonal polynomial at the data points.

## Properties of the quantities *a* and *b*

Given the values of $a$ and $b$ for any two polynomials $Q(x)$, $\bar{Q}(x)$, one can find the value of $(Q, \bar{Q})$ as follows:

$$\begin{aligned}(Q, \bar{Q}) &= (Q, \bar{a}_0T_0(x) + \bar{a}_1 T_1(x) + \bar{a}_2 T_2(x) + \ldots) \\ &= \bar{a}_0b_0 + \bar{a}_1b_1 + \ldots \\ &= a_0\bar{b}_0 + a_1\bar{b}_1 + \ldots \quad (12)\end{aligned}$$

In the same way one can find the value of $(Q, y)$ given the values of $d_r$ defined by $d_r = (T_r, y)$ from

$$(Q, y) = a_0d_0 + a_1 d_1 + a_2d_2 + \ldots \quad (13)$$

The Chebyshev polynomials have the properties

$$2xT_n = T_{n+1} + T_{n-1} \quad (n \geqslant 1)$$
$$xT_0 = T_1.$$

By definition $Q(x) = \sum\limits_{r=0} a_rT_r(x)$

hence $xQ = \frac{1}{2} \sum\limits_{r=1} a_r(T_{r+1} + T_{r-1}) + a_0T_1$

which may be re-arranged to give

$$xQ = \tfrac{1}{2} \sum_{r=2} (a_{r+1} + a_{r-1})T_r + \tfrac{1}{2}a_1T_0 + (a_0 + \tfrac{1}{2}a_2)T_1.$$

If we let $\bar{Q} = xQ$

then

$$\bar{a}_0 = \tfrac{1}{2}a_1 \quad (14)$$
$$\bar{a}_1 = \tfrac{1}{2}a_2 + a_0 \quad (15)$$
$$\bar{a}_r = \tfrac{1}{2}(a_{r+1} + a_{r-1}) \quad (r \geqslant 2). \quad (16)$$

Again, $(Q, T_r) = b_r$

which gives

$$(xQ, T_r) = (Q, xT_r) = \tfrac{1}{2}(b_{r+1} + b_{r-1}) \quad (r \geqslant 1)$$
$$(xQ, T_0) = b_1.$$

We then have

$$\bar{b}_0 = b_1 \quad (17)$$
$$\bar{b}_r = \tfrac{1}{2}(b_{r+1} + b_{r-1}) \quad (r \geqslant 1). \quad (18)$$

Given any set of polynomials expressed in terms of their values of $a$ and $b$, we can thus perform the following operations on them.

(i) Given $Q(x)$ find $xQ(x)$
i.e. given the values of $a$ and $b$ for $Q$, find the values of $a$ and/or $b$ for $xQ$. This is done as shown in equations (14)–(18).

(ii) Given any two polynomials $Q(x)$, $\bar{Q}(x)$, find $(Q, \bar{Q})$. This is done using (12).

(iii) Given $Q(x)$, find $(Q, y)$.
This is done using (13).

(iv) Given two polynomials $Q(x)$, $\bar{Q}(x)$, find $\lambda Q(x) + \mu\bar{Q}(x)$. If we let $\bar{\bar{Q}} = \lambda Q + \mu\bar{Q}$, then we have simply

$$\bar{\bar{a}}_r = \lambda a_r + \mu\bar{a}_r \qquad \bar{\bar{b}}_r = \lambda b_r + \mu\bar{b}_r.$$

Thus one can carry out all the operations necessary to use the recurrence relation (1) and equations (2), (3), and (4) entirely in terms of the values of the $a$ and $b$.

For the orthogonal polynomial $P_j$, the quantities $b_0, b_1, \ldots, b_{j-1}$ are zero, for the following reason. Any polynomial, and in particular any Chebyshev polynomial, of degree less than $j$ can be expressed in the form $\sum\limits_{i=0}^{s} \lambda_iP_i(x)$, where $s < j$. Thus if $T_r = \sum\limits_{i=0}^{r} \lambda_iP_i(x)$, then from definition (11) we have

$$b_r = (T_r, P_j) = \Big(\sum_{i=0}^{r} \lambda_iP_i, P_j\Big) = \sum_{i=0}^{r} \lambda_i(P_i, P_j)$$

$= 0$ if $r < j$ since the polynomials $P_k$ are orthogonal. Thus $b_0, b_1, \ldots, b_{j-1}$ are zero. Similarly for the polynomial $xP_j$ we have $b_0, \ldots, b_{j-2}$ equal to zero.

The number of values of the $b_r$ which need to be stored for any polynomial depends on the polynomial, and also on what degree the final polynomial approximation is to be. If the final approximation is to be of degree $n$, then for the polynomial $P_j$, we must store $b_j, b_{j+1}, \ldots, b_{2n-j}$. For the polynomial $xP_j$, we must store $b_{j-1}, \ldots, b_{2n-j-1}$.

## Surface fitting using the modified method

When surface fitting, a similar method can be used. The definitions corresponding to (10) and (11) become

$$\begin{aligned}Q(x, y) = {} & a_{00}T_0(x)T_0(y) + a_{10}T_1(x)T_0(y) \\ & + a_{20}T_2(x)T_0(y) + \ldots \\ & + a_{01}T_0(x)T_1(y) + a_{11}T_1(x)T_1(y) + \ldots \\ & + \ldots\end{aligned}$$

and $(Q, T_r(x)T_s(y)) = b_{rs}$.

The equations (12)–(18) can easily be generalised to deal with the case of surface fitting.

For the polynomial $P_{rs}$, the values of $b$ which need to be stored are $b_{r-s,s}, \ldots, b_{0,2n-r}$, i.e. all non-zero values of $b$ with the sum of the two suffices not greater than $2n - r$.

## Advantages of the modified method

Using the modified method, the coordinates of the individual points are each used once only, in calculating the values of $b$ for the zero degree polynomial, and the values of $d$. The coordinates, and the values of the orthogonal polynomials at the data points, need not be stored. Thus considerable store space is saved, especially if the number of points is large.

Let $n$ be the degree of the polynomial approximation required, and $m$ be the number of data points. In the case of curve fitting, the modified method needs approximately $5n$ locations of store. A method which stored the values of the polynomials at each of the data points would need approximately $4m + n$ locations.

In the case of surface fitting, the number of store locations needed for the two methods are approximately $\frac{6}{5}(n + 1)^3$ and $m(2n + 3) + \frac{2}{3}(n + 1)^3$ respectively.

In the case of curve fitting, the two methods take approximately the same amount of computer time. For surface fitting, the modified method requires less time. The time saving may be illustrated by an example in which a surface of degree 8 was fitted to 150 points. On an ICL 1905E computer, a program written by the author which used the values of the orthogonal polynomials at each of the data points took 23 seconds. A program using the modified method took 8 seconds, of which 2 seconds were spent finding the values of $d$, and the values of $b$ for the zero degree orthogonal polynomial.

## Numerical problems

When curve fitting, one can normalise the $x$-coordinates of the points so that they lie in the range $(-1, 1)$. If the points are not correctly normalised, and if any points lie outside this range, this causes ill-conditioning. Ill-conditioning also occurs if any large portion of the range is unoccupied by points.

When surface fitting, one must normalise the points so that they lie within the square $|x| < 1$, $|y| < 1$, and so that there are no large regions within this square unoccupied by points. If the points occupy a region in the shape of a parallelogram, then such normalisation is possible. However, if, for example the points occupy the interior of a triangle, then they cannot be satisfactorily normalised. The best one can do is to choose a parallelogram whose boundary lies as close as possible to the boundary of a triangle, and normalise the points as if they occupied the interior of that parallelogram. In the former case numerical difficulties are not likely to arise. In the latter case it may be necessary to restrict the degree of polynomial approximation to prevent numerical difficulties.

The modified method of surface fitting makes use of products of Chebyshev polynomials in $x$ and $y$. These polynomials have been chosen because they may be calculated using a simple recurrence relation. If the data points can be satisfactorily normalised, they do not cause ill-conditioning. It may be possible to prevent ill-conditioning in certain other cases by using some other polynomials in $x$ and $y$. The author has not investigated this possibility.

**Acknowledgements**

Acknowledgement is made to Mr. W. S. Blaschke for helpful comments.

The author is grateful to the Director of Engineering, Brush Electrical Engineering Company Limited, Loughborough, for permission to publish this paper.

**References**

Forsythe, G. E. (1957). Generation and Use of Orthogonal Polynomials for Data Fitting with a Digital Computer, *J. Soc. Indust. Appl. Math.*, Vol. 5, p. 74.

Weisfeld, M. (1957). Personal Communication to G. E. Forsythe, cf. ref. 1.

Clenshaw, C. W. (1960). Curve Fitting with a Digital Computer, *Computer Journal*, Vol. 2, p. 170.

## Book Review

*Decision Analysis*, by Howard Raiffa, 1968; 309 pages. (*Addison-Wesley.*)

The author, a self confessed 'Bayesian' and a very persuasive one I might add, has transposed a series of lectures on decision analysis into a very readable text book. These lectures were given at various times and not always in the continuous entity that comprises this work. This, however, does not mitigate against cohesiveness and the text flows in a calm, logical and very acceptable manner. Almost unique is the author's amiability in footnoting any section or chapter not vital to the theme of his argument so that the reader may bypass it if he so wishes. In fact, Mr Raiffa openly declares when he digresses and rides a personal hobby horse. One could wish for this in all authors.

One could also wish that the subject of 'Decision Analysis in Conditions of Uncertainty', as outlined so admirably, might find an audience and indeed be practised in the senior management levels of most UK companies. The ideas of PERT, network analysis and resource allocation are gaining adherents rapidly, and so they should. However, the decision point must have been reached before these disciplines can be of use. One suspects that in many cases the decision point may well have been arrived at by almost caveman-like processes of thought or hunch or just plain stubbornness.

A first casual flick through the pages might daunt the non-mathematician for there is an apparent profusion of algebraic formulae. This is not the case; in the main it is quite basic and easily comprehended. The work progresses from the first analysed elements of a basic decision, highlighting decision points or where chance prevails, through probability assessments, payoffs and the use of judgemental probability. The concluding two chapters deal with implementation of real rather than experimental situations and a final bibliographical 'walk through' that is far better than a mere list of references. This book can be recommended as reading for all practising decision makers for it is not merely an exercise in philosophical argument. For those who have a sincere desire for a disciplined approach to decisions, it is a must; for those who are inclined to jump in with both feet it will be a salutary exercise.

Lou Francis (Birmingham)