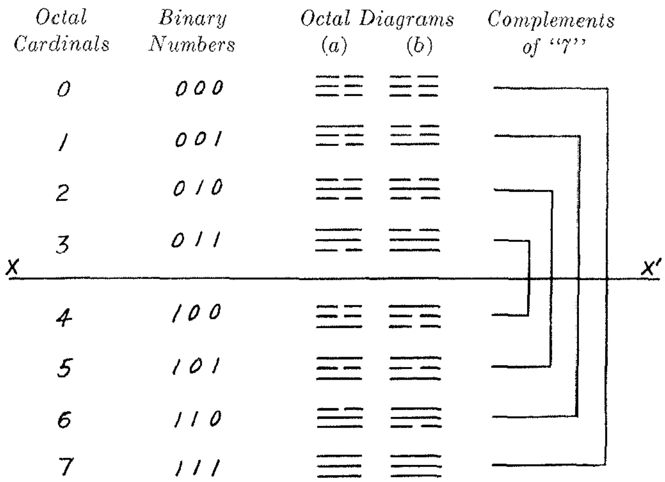

FIG. 2

9. While either the octal cardinals or their binary numbers have shown complementary characteristics of 7 about the dotted axis XX′, both octal diagrams (*a*) and (*b*) reveal likewise in exactly the same manner.

10. The fact that the binary trinities have the characteristics of being complements lend themselves to the addition of the complement of a number in lieu of its subtraction just as a purely binary machine does.

In closing, it may be noted with significant implications that, while the Chinese abacus [7] constitutes the oldest bi-quinary computing device, the binary trinities of the octal diagrams, which were derived by the Chinese ancients circa 26 centuries before the earliest prototype of the Chinese abacus, reveal all the necessary characteristics that are required in the internal number representation in an ultra-modern electronic digital computer.

REFERENCES

1. The Book of Changes; The Ghou Dynasty, "Axioms" (under the dual octal diagrams) by CHOU WEN WANG (before 1122 B.C.), "Corollaries" by CONFUCIUS (551–478 B.C.).
2. "Sacred Books of the East"; translated by James Legge; an English translation of "The Yih King" (The Book of Changes) appeared as Vol. 16 of this series, 1882.
3. JAMES LEGGE, *The Chinese Classics, with Translation, Critical and Exegetical Notes* . . . , 7 Vols., 2nd Ed., Clarendon Press, Oxford, 1895.
4. R. H. MATHEWS, *Chinese English Dictionary*; China Inland Mission and Presbyterian Mission Press, 1931; Harvard University Press, 1943, 1944, 1952; (a) p. 247, (b) p. 1165, (c) p. 520, (d) p. 670, (e) p. 436, (f) pp. 182, 441–2.
5. MAYERS' *Chinese Reader's Manual*.
6. *Chinese Cyclopoedia*, Chung Hua Book Company, Shanghai, Mar. 1947, 2nd Printing Oct. 1948; (a) p. 60, (b) p. 148, (c) p. 224, (d) p. 507.
7. SHU-T'IEN LI, Origin and development of the Chinese abacus, *J. Assoc. Comp. Mach. 6* (1959), 102–110.

Multi-Dimensional Least-Squares Polynomial Curve Fitting*

FRED H. LESH, *California Institute of Technology, Pasadena, Calif.*

Introduction

The theory of least-squares polynomial curve fitting is well-known and many computer programs have been written to fit polynomials of arbitrary order to data in one dimension. The extension of the theory to more than one dimension, however, is seldom seen in the literature; few programs have been written which will do multi-dimensional least-squares polynomial curve fitting. One reason for this may be that standard notations make the statement of the theory difficult to write. This paper presents a notation which makes the statement of the multi-dimensional theory almost as simple as that of the one-dimensional theory and shows, using this notation, how to construct a general multi-dimensional least squares polynomial curve fitting routine.

* This paper presents one phase of research carried out at the Jet Propulsion Laboratory, California Institute of Technology, under Contract No. NASw-6, sponsored by the National Aeronautics and Space Administration.

Statement of the Multi-Dimensional Theory

Suppose we have a function F defined at N points of Euclidean n-space R_n. Then a point in R_n has n coordinates which we call $x_1, x_2, \cdots, x_n$. Let X designate the vector $(x_1, \cdots, x_n)$, and let the N points at which F is defined be X_i, $i = 1, \cdots, N$. Let

$$F_i = F(X_i) = F(x_{1_i}, \cdots, x_{n_i}).$$

Define an exponent vector ϵ as a vector of positive integers $(\pi_1, \cdots, \pi_n)$ and define X^ϵ as $x_1^{\pi_1}, x_2^{\pi_2}, \cdots, x_n^{\pi_n}$.

A statement of the least-squares problem in R_n using this notation is as follows:

Given a set of exponent vectors $\epsilon_1, \cdots, \epsilon_m$ and a set of data X_i, F_i, $i = 1, \cdots, N$. Find the coefficients $c_1, \cdots, c_m$ of the polynomial

$$P(X) = c_1 X^{\epsilon_1} + \cdots + c_m X^{\epsilon_m}$$

which minimize E where

$$E = \sum_{i=1}^{N} [F_i - P(X_i)]^2.$$

The analysis now proceeds just as in the one-dimensional case. In order for E to be a minimum, all the partials of E with respect to the c_j must be zero.

$$\frac{\partial E}{\partial c_j} = 2 \sum_{i=1}^{N} [F_i - P(X_i)]^2 \frac{\partial P(X_i)}{\partial c_j} = 0$$

but

$$\frac{\partial P(X_i)}{\partial c_j} = X_i^{\epsilon_j}.$$

So the normal equations are

$$c_1 \sum_i X_i^{\epsilon_1 + \epsilon_j} + \cdots + c_m \sum_i X_i^{\epsilon_m + \epsilon_j} = \sum_i F_i X_i^{\epsilon_j}, \qquad j = 1, \cdots, m$$

or in matrix notation

$$A \gamma^T = b^T$$

where

$$A = \begin{pmatrix} \sum_i X_i^{\epsilon_1 + \epsilon_1} & \cdots & \sum_i X_i^{\epsilon_1 + \epsilon_m} \\ \vdots & & \vdots \\ \sum_i X_i^{\epsilon_m + \epsilon_1} & \cdots & \sum_i X_i^{\epsilon_m + \epsilon_m} \end{pmatrix}$$

$$\gamma = (c_1, \cdots, c_m)$$

$$b = \left(\sum F_i X_i^{\epsilon_1}, \cdots, \sum F_i X_i^{\epsilon_m}\right)$$

and u^T indicates u transpose.

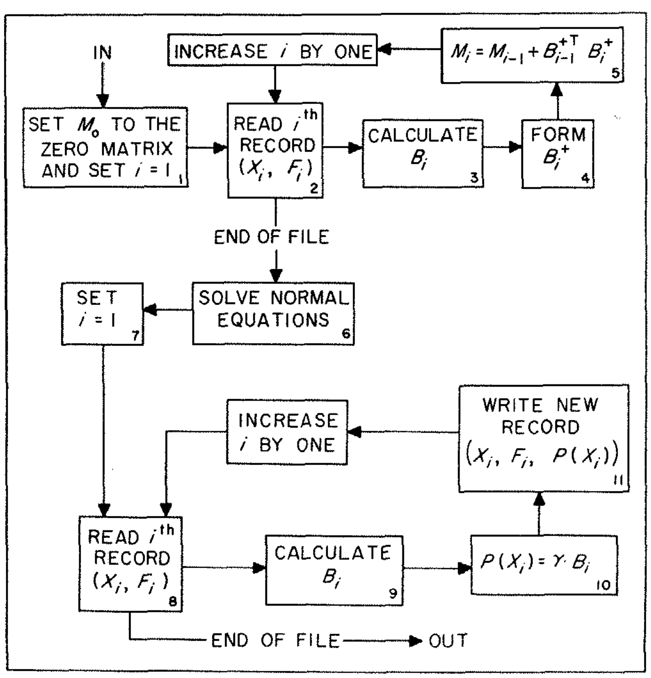

FIG. 1. Flow chart for machine program

It is important to the organization of the program to note that:

1. $A = \sum_i B_i^T B_i$
2. $b = \sum_i F_i B_i$
3. $P(X_i) = \gamma \cdot B_i$

where $B_i = (X_i^{\epsilon_1}, \cdots, X_i^{\epsilon_m})$.

Development of the Computer Program

The data are organized into records where each record contains X_i, F_i for one point, and these records are stored sequentially on paper tape, magnetic tape, magnetic drum, or magnetic core. After the last record is a mark called end of file which will enable the computer to tell when it has read in all the data.

The exponent vectors $\epsilon_1, \cdots, \epsilon_m$ are stored in the working memory of the computer since it is frequently necessary to try several different sets of ϵ for a given set of data in order to get a satisfactory fit.

Let B_i^+ be the vector $(X_i^{\epsilon_1}, \cdots, X_i^{\epsilon_m}, F_i)$, and note that the matrix $\sum_i B_i^{+T} B_i^+$ with the bottom row left off is the augmented matrix of the normal equations. Since the matrix $\sum_i B_i^T B_i$ is symmetric, only the elements on and above the main diagonal are needed and only those elements need be stored. The reason for constructing the vector B_i^+ is that the logic involved in forming the upper half of $B_i^{+T} B_i^+$ is much simpler than the logic necessary to form the upper half of $B_i^T B_i$ and insert the elements of $F_i B_i$ to augment it.

Figure 1 is a flow chart for the machine program. Note that in box 5 only the elements on and above the main diagonal and above the bottom row are calculated and stored. Note also that the final output of the subroutine consists of:

(1) The vector γ generated by box 6.
(2) The new file $[X_i, F_i, P(X_i)]$, $i = 1, \cdots, N$, generated by box 11.

This new file can be written on any medium desired.

The main programming effort is required in the writing of the routine for generating the vector B_i from the vector X_i and the set of vectors $\epsilon_1, \cdots, \epsilon_m$ for arbitrary m and n. Once the form of the exponent is decided on, and the program for calculating B_i has been written, the rest of the program is straightforward.

A subroutine is now being written at the Jet Propulsion Laboratory for fitting least-squares polynomials to data which may be distributed in any fashion whatever throughout a space of as many as seven dimensions. The fitting polynomial can consist of any desired combination of terms and the calling sequence will determine whether single or double precision arithmetic is used in the computation.