A LEAST-SQUARE COMPUTATION METHOD FOR SMOOTHING AND DIFFERENTIATION OF TWO-DIMENSIONAL DATA

S. KRISTYÁN

Institute of Physics Labratory of Surface Physics,
Technical University, Budapest, H—1521 Budapest

Received October 21, 1987
Presented by Prof Dr. J. Giber

Abstract

A computation method to smooth and differentiate data of the $z=f(x, y)$ kind is introduced. Requiring only that the datapoints be equi-distant in x and equi-distant in y, smoothing parameters can be calculated for general use. The greatest advantage of the method is that even higher-level mixed partial derivatives can be calculated directly from the datapoints.

Introduction

Random errors in experimental data have necessitated the development of mathematical and computation procedures for "filtering" the data; that is to say, removing a great part of the noise without losing any substantial amount of information.

A widely-used mathematical method is that of the least-squares. [2, 3] Smoothing of the datapoints by considering only their 3—15 closest neighbors in the calculations, often gives the expected "filtering," but, in case of many hunderds of datapoints, the method is extremely slow and uneconomical.

A simplified least-squares method, [4] which requires that the data be equi-distant in each variable, uses "smoothing" and "differentiation" parameters, providing an easy-to-use algorithm with dramatically reduced computer time.

Discussion

For the one-dimensional case, a polynomial $P(x)=a_0+a_1x+a_2x^2+\dots a_nx^n$ is used, in a way, that the measured (y_i) and the calculated $(P(x_i))$ values differ the least:

$$\sum_{i=1}^{m}(P(x_i)-y_i)^2 = \min \tag{1}$$

(1) is then differentiated by and solved for the a_i's. After that, the value of the polynomial can be calculated in any point.

If the basic system has equal distances in the abcissa (Δx=const., between any two neighboring datapoints), then the $P(x_i)$ values, and the derivatives of the $P(x_i)$'s are calculated directly from the measured y_i values.

If we choose a coordinate system in which $x_i=i$, and the distance between two neighboring points is 1, so the center of smoothing is $x_0=0$, its first left-side neighbor is $x_{-1}=-1$, the third right-side neighbor is $x_3=3$, and if only the nearest m of the left-side neighbors and the nearest m of the right-side neighbors are considered in the calculations for each point, then we have

$$M = \sum_{i=-M}^{i=m} (a_0 + a_1 \cdot i + \ldots + a_n \cdot i^n - y_i)^2 \tag{2}$$

which must be now minimized.

$$\frac{\partial M}{\partial a_0} = \sum_{i=-m}^{m} (a_0 + a_1 \cdot i + \ldots + a_n \cdot i^n - y_i) = 0 \tag{3}$$

$$\frac{\partial M}{\partial a_1} = \sum_{i=-m}^{m} (a_0 + a_1 \cdot i + \ldots + a_n \cdot i^n - y_i) i = 0$$

$$\vdots$$

$$\frac{\partial M}{\partial a_k} = \sum_{i=-m}^{m} (a_0 + a_1 \cdot i + \ldots + a_n \cdot i^n - y_i) i^k = 0$$

where we have $n+1$ linear equations. Now, we arrange the Σi^k's in a matrix $[I]$, the a_i's in a column-vector [A], and the $\Sigma y_i \cdot i^k$'s in a column-vector [Y]:

$$[I] \cdot [\mathrm{A}] = [\mathrm{Y}] \tag{4}$$

Furthermore, it is easy to see, that

$$P(0) = a_0 \tag{5}$$

$$\frac{\mathrm{d}P(0)}{\mathrm{d}i} = a_1$$

$$\vdots$$

$$\frac{\mathrm{d}^k P(0)}{\mathrm{d}i^k} = k!\, a_k$$

For a given m, $[I]$ is easily calculated and the system is solved for the a_i's. We find that

$$a_0 = P(0) = \frac{\sum_{i=-m}^{m} S_i^{(0)} y_i}{N^{(0)}} \tag{6}$$

$$\vdots$$

$$k!\, a_k = P^{(k)}(0) = \frac{\sum_{i=-m}^{m} S_i^{(k)} y_i}{N^{(k)}}$$

In (6), the N's are the normalization constants, the $S_i^{(0)}$'s are the smoothing parameters, and the $S_i^{(k)}$'s are the differentiation parameters. Since the S_i's and the y_i's are totally independent, the S_i's will be the same if the degree of the polynomial and the value of m are unchanged.

The greatest benefit (see (6)) is that the derivatives are calculated directly from the measured data.

When the parameters are used for smoothing only, we disregard the length of the interval (Δx), but for the rth derivatives ($r=1, 2, \ldots$) the result must be multiplied by Δx^{-r}.

Our objective has been to develop a computation method, based on the same ideas, for two- and higher [5] dimensional data. The following polynomials could be used:

$$P_0(x_1 y) = a_0 + a_1 x + a_2 y + a_3 xy + a_4 x^2 y + \ldots$$

$$P_0(x_1 y_1 z) = a_0 + a_1 x + a_2 y + a_3 z + \ldots + d_s xyz^2 + \ldots \quad \text{etc.}$$

Let us consider the two-dimensional case. First, the data must be equidistant in $x(\Delta x=\text{const.})$ and equi-distant in $y(\Delta y=\text{const.})$. The measured z_i values are put in a $k\times l$ matrix, where x changes along the rows and y changes along the columns. [1] In order to find the smoothing parameters, we use a coordinate system $x_i=i$ and $y_j=j$; the center of smoothing will always be the (0, 0) point.

Similarly to the one-dimensional case, the $2m$ nearest neighbors participate in the calculations, but now these are $2m$ neighbors in the x and $2m$ neighbors in the y directions; so, we shall use double summation. Putting i for x and j for y, and using the least-square method, we have

$$M = \sum_{i=-m}^{m} \sum_{j=-m}^{m} (a_0 + a_1 \cdot i + \ldots + a_s \cdot i^r j^k + \ldots - z_{ij})^2 \tag{7}$$

Then, the minimization gives

$$\frac{\partial M}{\partial a_0} = \sum_{i=-m}^{m} \sum_{j=-m}^{m} (a_0 + a_1 \cdot i + \ldots + a_s \cdot i^r j^k + \ldots - z_{ij}) = 0 \tag{8}$$

$$\vdots$$

$$\frac{\partial M}{\partial a_s} = \sum_{i=-m}^{m} \sum_{j=-m}^{m} (a_0 + a_1 \cdot i + \ldots + a_s \cdot i^r j^k + \ldots - z_{ij}) i^r j^k = 0$$

$$\vdots$$

Now we put the $\sum_{i=-m}^{m} \sum_{j=-m}^{m} i^r j^p$ values in matrix $[K]$, the a values in column-vector [A] and another column-vector, [Z] will have the $\sum_{i=-m}^{m} \sum_{j=-m}^{m} i^a j^b \cdot z_{ij}$ values. Then, we can write

$$[K]\cdot[\mathrm{A}] = [Z] \tag{9}$$

The results for the a's are again related to the derivatives (now partial derivatives):

$$r!\,p!\,a_s = \frac{\partial P(0,0)}{\partial i^r\, \partial j^p} = \frac{\sum_{i=-m}^{m} \sum_{j=-m}^{m} S_{ij}^{(r,p)} Z_{ij}}{N^{(r,p)}} \tag{10}$$

where $S_{ij}^{(0\ 0)}$'s are smoothing, the $S_{ij}^{(a,b)}$'s are differentiation parameters (if a or b is nonzero), the N's are the normalization constants and Z_{ij}'s are the values of the datapoints around $Z_{0,0}$ which is to be smoothed and/or differentiated.

If the x-interval is Δx and the y-interval is Δy, then the result must be multiplied by $\Delta x^{-r} \Delta y^{-p}$ for the mixed partial derivative, rth order in x and pth order in y.

The results can be generalized to higher dimensions. [5].

Results and discussion

Table 1 contains the parameters for one-dimensional smoothing, using third-degree polynomial, $P = a_0 + a_1 x + a_2 x^2 + a_3 x^3$, with $m = 2$.

Table I
Smoothing and Differentiation Parameters
for One-Dimensional Data

m	Smooth	dP/dx	d^2P/dx^2	d^3P/dx^3
−2	−3	1	2	−1
−1	12	−8	−1	2
0	17	0	−2	0
1	12	8	−1	−2
2	−3	−1	2	1
N	35	12	7	2

N is the normalization constant. To find the first derivative, for example, one has to multiply the second left-side value by 1, the first by −8, the point itself by 0, the first right-side neighbor by 8 and the second by −1, and then the sum of the calculated values is divided by 12 and divided by the stepsize (Δx).

For the two-dimensional case the data will be in a data-matrix and the smoothing and the differentiation parameters are in a matrix, too.

We have calculated the parameters for three different two-variable polynomials. Table II contains the numbers for the $P_2 = a_0 + a_1 \cdot x + a_2 \cdot y$ smoothing curve. Table III for the $P_3 = a_0 + a_1 \cdot x + a_2 \cdot y + a_3 \cdot x^2 + a_4 \cdot xy + a_5 \cdot y^2$, and Table IV for the $P_4 = a_0 + a_1 \cdot x + a_2 \cdot y + a_3 \cdot x^2 + a_4 xy + a_5 \cdot y^2 + a_6 \cdot x^3 + a_7 \cdot x^2 y + a_8 xy^2 + a_9 \cdot y^3$.

Depending on the data the user can choose the one that apparently best fits the measured values. For the sake of better understanding, we provide the mapping of the parameters, in case of $m=2$:

$$\begin{array}{cl} & (-2,2)(-1,2)(0,2)(1,2)(2,2) \\ \uparrow & (-2,1)(-1,1)(0,1)(1,1)(2,1) \\ Y & (-2,0)(-1,0)(0,0)(1,0)(2,0) \\ & (-2,-1)(-1,-1)(0,-1)(1,-1)(2,-1) \\ & (-2,-2)(-1,-2)(0,-2)(1,-2)(2,-2) \\ & \qquad\qquad X \rightarrow \end{array}$$

The first number in the parentheses is the j index and the second is the i index.

Table II

Smooting and Differentiation Parameters

$P_2 = a_0+a_1\cdot x+a_2\cdot y$

Smoothing:	1	1	1	1	1	
	1	1	1	1	1	
	1	1	1	1	1	$N = 25$
	1	1	1	1	1	
	1	1	1	1	1	
$\partial P/\partial x$:	−2	−1	0	1	2	
	−2	−1	0	1	2	
	−2	−1	0	1	2	$N = 50$
	−2	−1	0	1	2	
	−2	−1	0	1	2	
$\partial P/\partial y$:	2	2	2	2	2	
	1	1	1	1	1	
	0	0	0	0	0	$N = 50$
	−1	−1	−1	−1	−1	
	−2	−2	−2	−2	−2	

The parameters have been tested with generated data. The data-matrices in the size of 60×48 contained the calculated values of $z_1=8.0+273.0x^2+5.0\cdot xy^2$, $z_2=3.8+2.3x-4.5y$, and $z_3=-7.7+5.5\cdot x+11.3\cdot y^2$, with a random error ranging from -10% to $+10\%$ of the calculated values. As previously mentioned, the data-points are put in the data-matrix in a way that x changes along the rows and y changes along the columns. The stepsize for x was 0.1, for y it was 0.01.

The calculations were so lengthy that even the generated values had to be calculated by a Hewlett—Packard 85 personal computer.

For z_1, the parameters from Table IV were used; for z_2 from Table II; and for z_3 from Table III. The results proved to be excellent; the errors were reduced to 1/10—

Table III
Smoothing and Differentiation Parameters

$$P_3 = a_0+a_1x+a_2y+a_3x^2+a_4xy+a_5y^2$$

Smoothing:	−13	2	7	2	−13	
	2	17	22	17	2	
	7	22	27	22	7	$N = 175$
	2	17	22	17	2	
	−13	2	7	2	−13	
$\partial P/\partial x$:	−2	−1	0	1	2	
	−2	−1	0	1	2	
	−2	−1	0	1	2	$N = 50$
	−2	−1	0	1	2	
	−2	−1	0	1	2	
$\partial P/\partial y$:	2	2	2	2	2	
	1	1	1	1	1	
	0	0	0	0	0	$N = 50$
	−1	−1	−1	−1	−1	
	−2	−2	−2	−2	−2	
$\partial P^2/\partial x^2$:	2	−1	−2	−1	2	
	2	−1	−2	−1	2	
	2	−1	−2	−1	2	$N = 35$
	2	−1	−2	−1	2	
	2	−1	−2	−1	2	
$\partial P^2/\partial y^2$:	2	2	2	2	2	
	−1	−1	−1	−1	−1	
	−2	−2	−2	−2	−2	$N = 35$
	−1	−1	−1	−1	−1	
	2	2	2	2	2	
$\partial P^2/\partial x\partial y$:	−4	−2	0	2	4	
	−2	−1	0	1	2	
	0	0	0	0	0	$N = 100$
	2	1	0	−1	−2	
	4	2	0	−2	−4	

1/20 and even less fraction of the original. Further improvement could be made by using $m=3$ instead of 2. However, with $m=2$ 416 peripherial datapoints are lost (these are used for the calculations but they themselves can not be treated because they do not have enough neighboring datapoints around them) with $m=3$ even more would have been lost (608).

Table IV
Smoothing and Differentiation Parameters

$P_4 = a_0 + a_1x + a_2y + a_3x^2 + a_4xy + a_5y^2 + a_6x^3 + a_7x^2y + a_8xy^2 + a_9y^3$

Smoothing:	−13	2	7	2	−13	
	2	17	22	17	2	
	7	22	27	22	7	
	2	17	22	17	2	$N = 175$
	−13	2	7	2	−13	
$\partial P/\partial x$:	31	−44	0	44	−31	
	−5	−62	0	62	5	
	−17	−68	0	68	17	
	−5	−62	0	62	5	$N = 420$
	31	−44	0	44	−31	
$\partial P/\partial y$:	−31	5	17	5	−31	
	44	62	68	62	44	
	0	0	0	0	0	$N = 420$
	−44	−62	−68	−62	−44	
	31	−5	−17	−5	31	
$\partial^2 P/\partial x^2$:	2	−1	−2	−1	2	
	2	−1	−2	−1	2	
	2	−1	−2	−1	2	$N = 35$
	2	−1	−2	−1	2	
	2	−1	−2	−1	2	
$\partial^2 P/\partial y^2$:	2	2	2	2	2	
	−1	−1	−1	−1	−1	
	−2	−2	−2	−2	−2	$N = 35$
	−1	−1	−1	−1	−1	
	2	2	2	2	2	
$\partial^2 P/\partial x \partial y$:	−4	−2	0	2	4	
	−2	−1	0	1	2	
	0	0	0	0	0	$N = 100$
	2	1	0	−1	−2	
	4	2	0	−2	−4	
$\partial^3 P/\partial x^3$:	−1	2	0	−2	1	
	−1	2	0	−2	1	
	−1	2	0	−2	1	$N = 10$
	−1	2	0	−2	1	
	−1	2	0	−2	1	
$\partial^3 P/\partial y^3$:	1	1	1	1	1	
	−2	−2	−2	−2	−2	
	0	0	0	0	0	$N = 10$
	2	2	2	2	2	
	−1	−1	−1	−1	−1	

Table IV continued

$\partial^3 P/\partial x^2 \partial y$:	4	−2	−4	−2	4	
	2	−1	−2	−1	2	
	0	0	0	0	0	$N = 70$
	−2	1	2	1	−2	
	−4	2	4	2	−4	
$\partial^3 P/\partial x \partial y^2$:	−4	−2	0	2	4	
	2	1	0	−1	−2	
	4	2	0	−2	−4	$N = 70$
	2	1	0	−1	−2	
	−4	−2	0	2	4	

Conclusion

Using the calculated parameters, the computations have proved to be extremely fast and very accurate. However, if the polynomials of the datapoints had not been known, there would have been the question, immediately: Which parameters to use? This is the case with experimental datapoints, too. The best way seems to be to start calculating the third-order partial derivatives with the parameters of Table IV, and if the results are all zeros (or very close to zero), but the second-order derivatives are not zeroes, then the data show a P_3-behavior. If the second-order derivatives are zeros, too, then the data must be the $P_2 = a_0 + a_1 x + a_2 \cdot y$ type.

References

1. KNORR, F. J. and HARRIS, J. M.: Anal. Chem. *53*, 272, 1981.
2. GUEST, P. G.: "Numerical Methods of Curve Fitting", p. 349, University Press, Cambridge, England, 1961.
3. CHENEY, W., KINCAID, D.: "Numerical Mathematics and Computing", p. 267, Brooks/Cole, Monterey, California, 1980.
4. SAVITZKY, A., GOLAY, M. T.: Anal. Chem. 36, 1627, 1964.
5. SZAMOSI, J., KRISTYAN, S.: Yet unpublished calculations.

Dr. Sándor KRISTYÁN H-1521, Budapest