0273-2289/97/63–65—0339$10.50

Experimental Data Analysis

An Algorithm for Determining Rates and Smoothing Data

K. Thomas Klasson

Chemical Technology Division, Oak Ridge National Laboratory, Oak Ridge, TN 37831-6044*

ABSTRACT

Reaction rate determination from experimental data is generally an essential part of evaluating enzyme or microorganism growth kinetics and the effects on them. Commonly used methods include forward, centered, or backward finite difference equations using two or more data points. Another commonly applied method for determining rates is least-square regression techniques, and when the sought function is unknown, polynomials are often applied to represent the data. The cubic spline functions presented in this article represent a versatile method of evaluating rates. The advantage in using this method is that experimental error may be largely accounted for by the incorporation of a smoothing step of the experimental data without force-fitting of the data. It also works well when data are unevenly spaced (often the case for experiments running over long periods of time). The functions are easily manipulated, and the algorithm can be written concisely for computer programming. The development of spline functions to determine derivatives as well as integrals is presented.

Index Entries: Cubic splines; computer programming; derivatives; smoothing data.

INTRODUCTION

Reaction rate determination from experimental data is generally an essential part of evaluating enzyme or microorganism growth kinetics and effects on them. Commonly used methods include simple forward,

*Managed by Lockheed Martin Energy Research Corp. for the U.S. Department of Energy under contract DE-AC05-96OR22464. The submitted manuscript has been authored by a contractor of the U.S. Government. Accordingly, the U.S. Government retains a nonexclusive, royal-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.

centered, or backward finite difference equations using two or more data points. Another method was presented by LeDuy and Zajic *(1)*, and involves the construction of a circle with the three neighboring points on the perimeter, finding the center coordinates, and the normal to the equation between the center coordinates and the second (middle) point. This method works well with well-behaved data evenly spaced. The advantage in using the method described in this article is that experimental error may be largely accounted for by the incorporation of a smoothing step of the experimental data. It also works well when data are unevenly spaced (often the case for experiments running over long periods of time). The difference between the numerical method proposed and the manual-graphical method of finding derivatives (dX/dt) may be represented as the following:

Manual or graphical method
1. Plot experimental data. (X as a function of t)
2. Draw a smooth curve through or close to the points.
3. Take many readings of curve and find derivatives at each of the original values of t using finite differences.

Numerical method
1. Construct cubic spline with a "smoothness factor."
2. Use spline functions to find derivatives at original t values.

The construction of cubic splines is simple with today's personal computers and available programming languages.

Cubic Spline Theory

A cubic spline joins two neighboring points with a third-order polynomial *(2)*. Thus, if there are N experimental data points, the overall curve through the points may be described by $N - 1$ equations, each in the form of $f_i(t) = \alpha_i t^3 + b_i t^2 + c_i t + d_i$ (*see* Fig. 1). The third-order equation contains four constants, and therefore, we need to determine a total of $4(N - 1)$ constants. In order to do this and to assure a continuous smooth curve without breaks and abrupt changes in the derivative at the points, a few criteria must be followed.

1. The equation values must be equal at the interior points ($2N - 4$ conditions). In other words, the spline must pass through the data points.
2. The first and last equations must pass through the end points (two conditions).
3. The first and second derivatives at the interior points must be equal (2N – 4 conditions). In other words, the spline must be smooth as it passes through the data points.
4. The second derivatives at the end points are zero (two conditions).

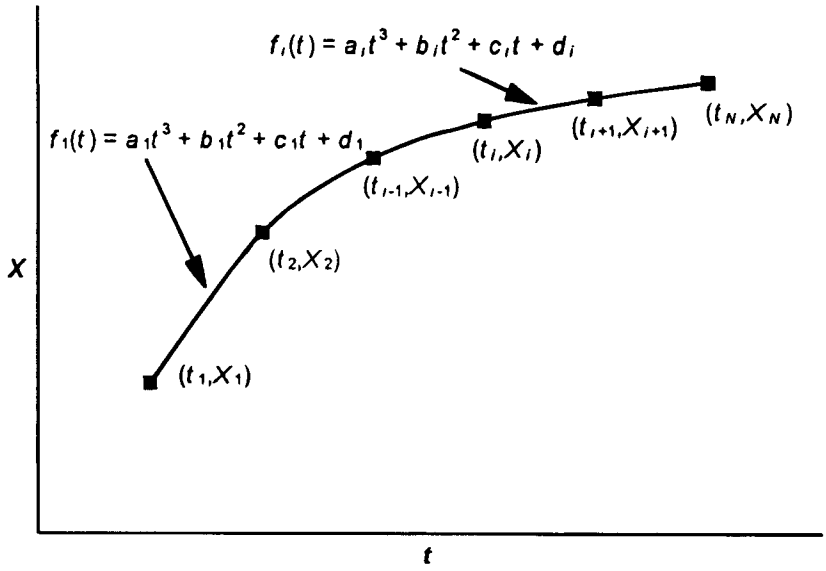

Fig. 1. Example of cubic spline construction.

Even though the above system is completely defined, it is more efficient to restructure the equations to contain fewer unknown constants (*2*). If the experimental data correspond to the points $(t_1, X_1), (t_2, X_2), \ldots (t_i, X_i), \ldots (t_N, X_N)$, then the i:th polynomial between the points i and $i + 1$ may be described in the terms of surrounding points by:

$$f_i(t) = \frac{f''(t_i)}{6(t_{i+1} - t_i)}(t_{i+1} - t)^3 + \frac{f''(t_{i+1})}{6(t_{i+1} - t_i)}(t - t_i)^3 + \left[\frac{X_i}{(t_{i+1} - t_i)} - \frac{f''(t_i)(t_{i+1} - t_i)}{6}\right](t_{i+1} - t) + \left[\frac{X_{i+1}}{(t_{i+1} - t_i)} - \frac{f''(t_{i+1})(t_{i+1} - t_i)}{6}\right](t - t_i) \quad (1)$$

This equation contains only two unknowns, the second derivatives, which may be evaluated using:

$$(t_{i+1} - t_i)\ f''(t_i) + 2(t_{i+2} - t_i)\ f''(t_{i+1}) + (t_{i+2} - t_{i+1})\ f''(t_{i+2}) = \frac{6}{(t_{i+2} - t_{i+1})}(X_{i+2} - X_{i+1}) - \frac{6}{(t_{i+1} - t_i)}(X_{i+1} - X_i) \quad (2)$$

Equation (2) was derived by taking the first derivative of Eq. (1) for both the i:th and $(i + 1)$:th polynomial and setting them equal at the shared point (t_{i+1}, X_{i+1}). When Eq. (2) is written for all interior points, we will have

$N-2$ simultaneous equations with $N-2$ unknown if criterion four above is incorporated. The equation system may be written in the following format:

$$\begin{aligned} a_2 f_1'' + b_2 f_2'' + c_2 f_3'' &= r_2 \\ a_3 f_2'' + b_3 f_3'' + c_3 f_4'' &= r_3 \\ \downarrow & \\ a_{N-1} f_{N-2}'' + b_{N-1} f_{N-1}'' + c_{N-1} f_N'' &= r_{N-1} \end{aligned}$$

Since $f_1'' = [f''(t_1)] = 0$ and $f_N'' = [f''(t_N)] = 0$ (according to criterion four), the equation system will form a matrix with a tridiagonal band. This type of matrix may easily be solved using simplified Gaussian elimination with a few lines of computer programming. Once the second derivatives have been evaluated at the interior points, they may be used to define completely each of the polynomials, according to Eq. (1).

DETERMINATION OF DERIVATIVES AND INTEGRALS

Equation (1) may be differentiated to determine the derivative (e.g., when estimating rates) at any point on the splines yielding:

$$f_i'(t) = \frac{f''(t_{i+1})}{2(t_{i+1}-t_i)}(t-t_i)^2 - \frac{f''(t_i)}{2(t_{i+1}-t_i)}(t_{i+1}-t)^2 + \frac{(X_{i+1}-X_i)}{(t_{i+1}-t_i)} - \frac{(f''(t_{i+1})-f''(t_i))}{6}(t_{i+1}-t_i) \quad (3)$$

Equation (3) may be used for the points 1 through $N-1$, and the derivative for the last point may be evaluated using the $(N-1)$:th equation. Note that Eq. (3) is further simplified, since t is substituted by t_i when evaluating derivatives at the data points.

Equation (1) may also be integrated to determine the integral under the data points. This may be desired when estimating cell and product yield uptake using an integrated equation vs a differentiated equation *(3)*. The integral between data points i and $i+1$ connected by the cubic spline function $f_i'(t)$ is given by:

$$\int_{t_i}^{t_{i+1}} f_i(t)\,dt = \tfrac{1}{2}(t_{i+1}-t_i)(X_i + X_{i+1}) - \tfrac{1}{24}(t_{i+1}-t_i)^3(f''(t_i) + f''(t_{i+1})) \quad (4)$$

SMOOTHING OF DATA

The procedure for smoothing of data is based on the above cubic spline procedure with a few modifications and is presented by the following algorithm:

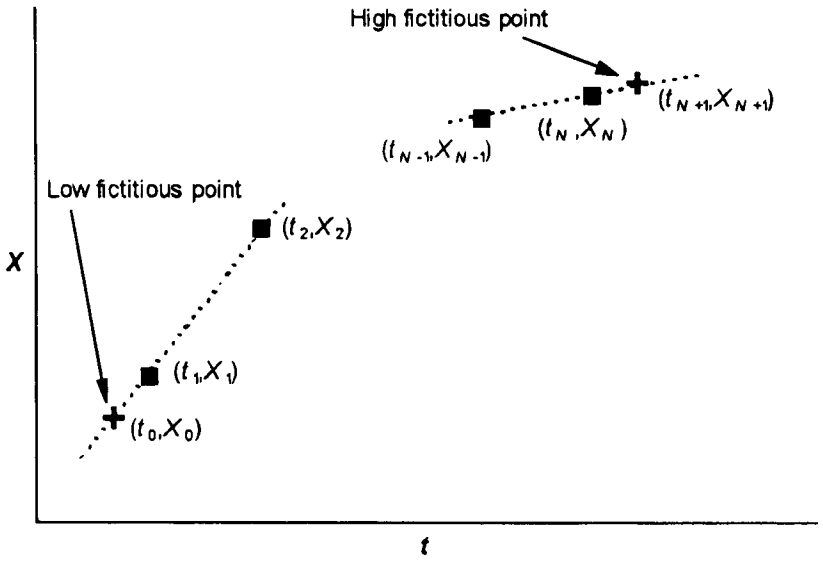

Fig. 2. Determination of fictitious points slightly outside data intervals.

1. Determine positions of two fictitious points outside the range of t by constructing a straight line between the first and second point, and finding a point on this line with a value of t slightly lower than t_1. Do the same for the fictitious point in the upper region of t (*see* Fig. 2).
2. Connect all points including fictitious points with straight lines, and find the center point for each interval of t equal to $(t_{i+1} + t_i)/2$. This will give $N + 1$ new values of t and X.
3. Create cubic splines through the new set of data (the midpoints).
4. Using the original values of t and the spline functions, calculate values of X for all the points. The values will correspond to smooth data and have slightly different values than the original "raw data."

The above procedure may be invoked iteratively by using the smooth data obtained in step 4 as starting values in step 1. One completed sequence may be seen as a smoothness factor (SF) of 1. Similarly, the original cubic spine has a SF of 0 (zero). Visually, one may see the smoothing of the original spline as a method of straightening the curve (*see* Fig. 3). An infinite SF will theoretically result in a straight line. The derivatives of the smooth data may of course be evaluated as described in the previous section.

AN EXAMPLE

In 1959, Luedeking and Piret published a bacterial product-formation equation (*4*), which has since been used repeatedly to describe production in various bioprocesses. In their original study, they used a graphical

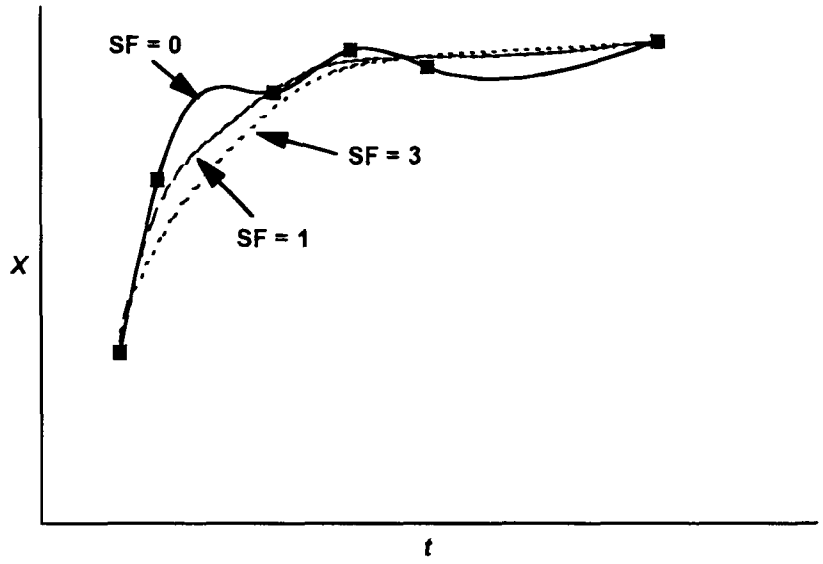

Fig. 3. Examples of cubic splines with different "smoothness" factor.

method, and reduced their 55 raw data points (time and cell concentration) to 25 data points by plotting their raw data, manually constructing a curve through the data, and finally taking readings off this curve. This graphical data were then the starting point for determination of growth rate (dN/dt) by graphical differentiation.

In Fig. 4, I have plotted Luedeking and Piret's determined growth rate (which they determined from 25 points) together with the growth rate calculated based on cubic splines with SFs of zero and one. As is noted when SF = 0 (cubic spline through the 55 raw data points), the calculated derivatives (growth rates) vary substantially from the graphically determined derivatives, but when the raw data are smoothed with an SF = 1, the derivatives compare well with Luedeking and Piret's. The reason that the cubic spline with SF = 0 produces a somewhat scattered result stems from the built-in strength of the spline to adjust itself to go through the points; this may result in an oscillating behavior (as noted in Fig. 3 with SF = 0) that is reflected in derivatives at the points. These oscillations are especially noted when there is experimental error in the data. Smoothing of data reduces oscillations and may be thought of as a method to adjust for experimental errors. Care must be taken not to overdo smoothing and jeopardize accurate representation of the data; the use of correlation coefficients, residuals, and visual inspection (as in Fig. 3) can help alleviate this concern.

Another commonly applied method for smoothing data is least-square regression techniques, and when the sought function is unknown, polynomials are often applied to represent the data. Derivatives (or rates) may then be analytically derived from the regression equation. In Fig. 5, the data from Fig 4. have been represented again, but this time, the deriv-

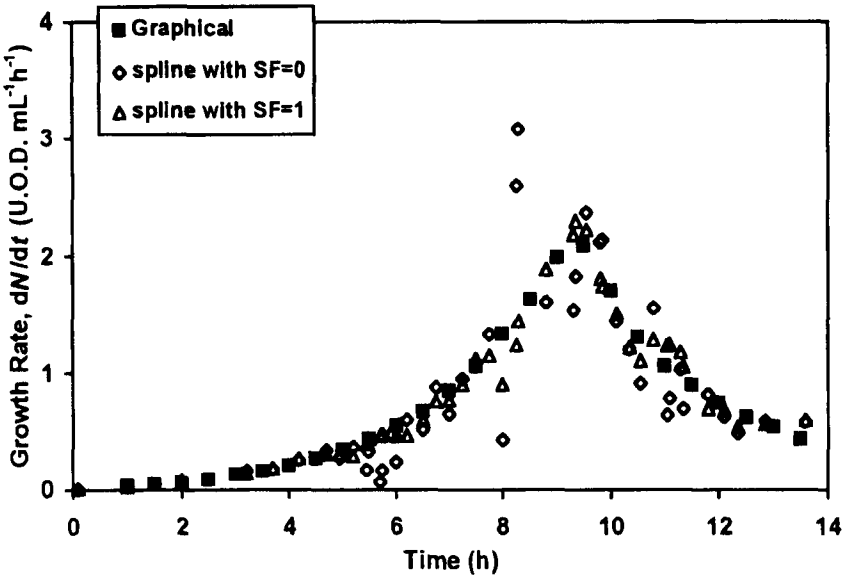

Fig. 4. Comparison between graphical and cubic spline methods when determining derivatives.

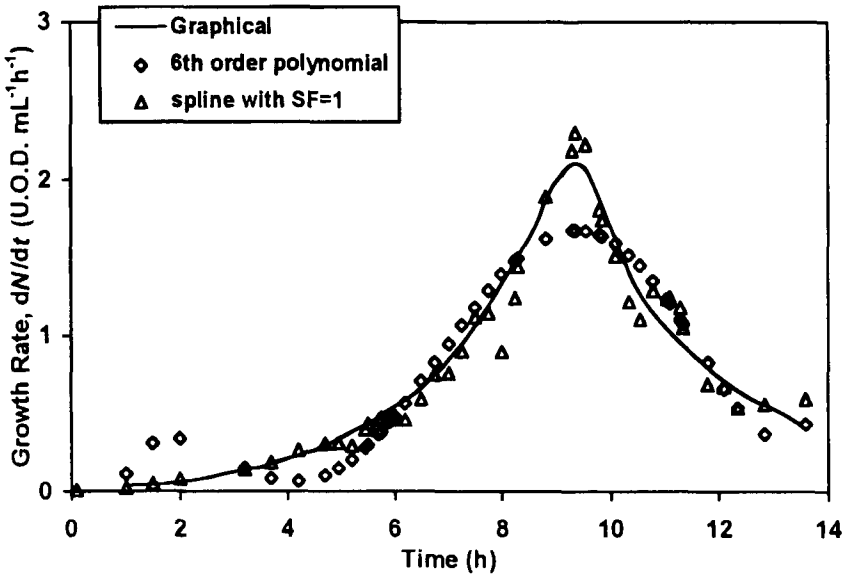

Fig. 5. Comparison of derivatives derived from cubic splines and a sixth-order polynomial.

atives from a spline with SF = 1 are compared with derivatives calculated from a sixth-order polynomial fitted (using method of least-squares) to the raw data. The polynomial fits the data quite well with a correlation coefficient of 0.998, but the calculated derivatives at the points are quite distant from the values obtained by Luedeking and Piret (represented

with curve in Fig. 5). This indicates that it is important not to force-fit a set of data to any one function as a means to evaluate derivatives.

ERROR IN SMOOTHING

Since the smooth curve generated through the use of cubic splines with SF > 0 does not go through the data points, it is appropriate that we define a correlation coefficient as *(5)*:

$$r^2 = \frac{\sum_1^N (X_i - \bar{X})^2 - \sum_1^N (X_i - \hat{X}_i)^2}{\sum_1^N (X_i - \bar{X})^2} = \frac{2\sum_1^N X_i \hat{X}_i - \sum_1^N \hat{X}_i^2 - \frac{1}{N}\left(\sum_1^N X_i\right)^2}{\sum_1^N X_i^2 - \frac{1}{N}\left(\sum_1^N X_i\right)^2} \quad (5)$$

where X_i is the i:th data value, $\hat{X}_i$ is the predicted value, and $\bar{X}$ is the average of X_i. The residual $(X_i - \hat{X}_i)$ is another useful parameter for estimating data correlation. In the example presented in Fig. 4, the correlation coefficient was equal to 1.00 for SF = 1; thus, the smoothed values were very close to the raw data.

COMPUTER PROGRAMMING

The programming was done in Microsoft Q-Basic for simplicity, but may easily be rewritten to any other programming language. The computer code is listed below and is available via electronic mail by contacting the author.

```
'Initiation
DECLARE SUB TRIMATRIX (t!(), X!(), i0!, N!, r!())
DIM t(50), X(50), tmid(50), Xmid(50), traw(50), Xraw(50), r(50), dXdt(50)
'
'Simple data input
INPUT "Number of points"; N
PRINT "Input in order of increasing t,"
FOR i = 1 TO N
   PRINT "(t,X) for point"; i;
   INPUT traw(i), Xraw(i)
   t(i) = traw(i): X(i) = Xraw(i)
NEXT i
'
'Smoothing of data
CLS
INPUT "Input Smoothness Factor (SF)"; SF
'
FOR SFcounter = 1 TO SF
   'Step 1. Find fictitious points
   t(0) = t(1) - .05 * (t(2) - t(1))
   X(0) = X(1) + (X(2) - X(1)) * (t(0) - t(1)) / (t(2) - t(1))
```

```
    t(N + 1) = t(N) + .05 * (t(N) – t(N – 1))
    X(N + 1) = X(N) + (X(N – 1) – X(N)) * (t(N + 1) – t(N)) / (t(N – 1) – t(N))
    '
    'Step 2. Find mid-points on connecting lines
    FOR i = 0 TO N
       tmid(i) = (t(i + 1) + t(i))/2
       Xmid(i) = (X(i + 1) + X(i))/2
    NEXT i
    '
    'Step 3. Create cubic splines with mid-points
    CALL TRIMATRIX(tmid(), Xmid(), 0, N, r())
    '
    'Step 4. Find smooth X-values on splines at original t-values
    FOR i = 0 TO N – 1
      t = traw(i + 1)
      Xtemp = (r(i) * (tmid(i + 1) – t) ^ 3 + r(i + 1) * (t – tmid(i)) ^ 3) / 6 / (tmid(i + 1) –
      tmid(i))
      Xtemp = Xtemp + (Xmid(i) / (tmid(i + 1) – tmid(i)) – r(i) * (tmid(i + 1) – tmid(i))
      / 6) * (tmid(i + 1) – t)
      X(i + 1) = Xtemp + (Xmid(i + 1) / (tmid(i + 1) – tmid(i)) – r(i + 1) * (tmid(i + 1)
      – tmid(i)) / 6) * (t – tmid(i))
NEXT i
NEXT SFcounter
'
CALL TRIMATRIX(t(), X(), 1, N, r())
'
'Calculate derivatives based on spline equations
FOR i = 1 TO N – 1
   dXdt(i) = –r(i) * (t(i + 1) – t(i)) / 2 + (X(i + 1) – X(i)) / (t(i + 1) – t(i)) – (r(i + 1) –
   r(i)) / 6 * (t(i + 1) – t(i))
NEXT i
'Derivative at last point (N) must be calculated separately
dXdt(N) = (X(N) – X(N – 1)) / (t(N) – t(N – 1)) + r(N – 1) * (t(N) – t(N – 1)) / 6
'
'Print smooth values and derivatives at smooth values
PRINT "These are your raw and smooth values plus derivatives with SF="; SF
PRINT "t-raw", "X-raw", "t-smooth", "X-smooth", "dX/dt"
FOR i = 1 TO N
   PRINT traw(i), Xraw(i), t(i), X(i), dXdt(i)
   'Calculate the summations needed for r2 calculation
   sum1 = sum1 + X(i) * Xraw(i): sum2 = sum2 + X(i)^ 2: sum3 = sum3 + Xraw(i):
   sum4 = sum4 + Xraw(i)^ 2
NEXT i
PRINT "For SF="; SF; ", the correlation coefficient, r2 ="; (2 * sum1 – sum2 –
   sum3 ^ 2 / N) / (sum4 – sum3 ^ 2 / N)
END
'
SUB TRIMATRIX (t(), X(), i0, N, r())
   DIM a(50), b(50), c(50) 'Local variables
   'Create constants in Equation <2> based on points, numbered i0 to N
   FOR i = i0 TO N – 2
      a(i + 1) = t(i + 1) – t(i)
      b(i + 1) = 2 * (t(i + 2) – t(i))
```

```
    c(i + 1) = t(i + 2) – t(i + 1)
    r(i + 1) = 6 * (X(i + 2) – X(i + 1)) / (t(i + 2)
       – t(i + 1)) – 6 * (X(i + 1) – X(i)) / (t(i +
       1) – t(i))
  NEXT i
  r(i0) = 0: r(N) = 0 'Criterion 4
  '
  'Solve triagonal augmented matrix
  FOR i = i0 + 1 TO N – 2
    b(i + 1) = b(i + 1) – a(i + 1) / b(i) * c(i)
    r(i + 1) = r(i + 1) – a(i + 1) / b(i) * r(i)
  NEXT i
  FOR i = 1 TO N – 1 – i0
    r(N – i) = (r(N – i) – c(N – i) * r(N – i + 1))
      / b(N – i)
  NEXT i
  '
  'r() contains the solution = values for secondary derivatives
END SUB
```

CONCLUSION

Cubic spline functions represent a powerful method of evaluating rates and/or smoothing data. The functions are easily manipulated, and the algorithm can be written concisely. The data smoothing can potentially adjust for experimental error and compares well to manual graphical methods. The method presented constitutes an improvement over force-fitting.

REFERENCES

1. LeDuy, A. and Zajic, J. E. (1973), *Biotechnol. Bioeng.* **15**, 805–810.
2. Cheney, W. and Kincaid, D. (1980), in *Numerical Mathematics and Computing*, Brooks/Cole, Monterey, CA, pp. 339–343.
3. Klasson, K. T., Clausen, E. C., and Gaddy, J. L. (1989), *Appl. Biochem. Biotechnol.* **20**, 491–509.
4. Luedeking, R. and Piret, E. L. (1959), *J. Biochem. Microbiol. Technol. Eng.* **1**, 393–412.
5. Walpole, R. E. and Myers, R. H. (1989), in *Probability and Statistics for Engineers and Scientists*, 4th ed., Macmillan, New York, p. 422.