6.9 Estimating Model Parameters Involving ODE's Using Fermentation Data

6.9.1 Concepts Demonstrated

Determination of rate expression parameters using experimental concentration data for a fermentation process which produces penicillin.

6.9.2 Numerical Methods Utilized

Integration of the differential equations for obtaining calculated concentration data in an inner loop and using a multivariable minimization program to obtain the optimal parameter values in an outer loop.

6.9.3 Problem Statement (Adapted from Constantinides[3])

When the microorganism *Penicillium Chrysogenum* is grown in a batch fermenter under carefully controlled conditions, the cells reproduce at a rate which can be modeled by the logistic law

$$\frac{dy_1}{dt} = b_1 y_1\left(1 - \frac{y_1}{b_2}\right) \tag{6-49}$$

where y_1 is the concentration of the cells expressed as percent dry weight. In addition, the rate of production of penicillin has been mathematically quantified by the equation where y_2 is the units of penicillin per mL.

$$\frac{dy_2}{dt} = b_3 y_1 - b_4 y_2 \tag{6-50}$$

(a) Use the experimental data in Table 6–22 to find the values of the parameters b_1, b_2, b_3 and b_4 which minimize the sum of squares of the differences between the calculated and experimental concentrations (y_1 and y_2) for all the data points. The following initial estimates can be used: $b_1 = 0.1$; $b_2 = 4.0$; $b_3 = 0.02$ and $b_4 = 0.02$.

(b) Plot the calculated and experimental values of y_1 and y_2 using the optimal parameter values.

Table 6–22 Penicillin Growth Data in Batch Fermenter - File **P6_9.txt**

Time h	Cell Concentration Percent Dry Weight y_1	Penicillin Concentration Units/mL y_2
0	0.18	0
10	0.12	0

Table 6–22 Penicillin Growth Data in Batch Fermenter - File **P6_9.txt**

Time h	Cell Concentration Percent Dry Weight y_1	Penicillin Concentration Units/mL y_2
22	0.48	0.0089
34	1.46	0.0642
46	1.56	0.2266
58	1.73	0.4373
70	1.99	0.6943
82	2.62	1.2459
94	2.88	1.4315
106	3.43	2.0402
118	3.37	1.9278
130	3.92	2.1848
142	3.96	2.4204
154	3.58	2.4615
166	3.58	2.283
178	3.34	2.7078
190	3.47	2.6542

6.9.4 Solution

The solution of the parameter estimation problem when the model is in the form of systems of ODEs can be divided into three stages.

1. Integration of the differential equations for a specified set of the parameter values in order to obtain the calculated dependent variable values at the same time (independent variable) intervals where experimental data is available.
2. Calculation of the sum of squares of the differences between the calculated and experimental values of the dependent variables.
3. Application of an optimization program which modifies the parameter values so as to obtain the minimum of the sum of squares.

These three stages can be most conveniently be carried out using MATLAB. First the model equations are introduced into POLYMATH and integrated using the parameter values that were provided as initial estimates. The POLYMATH set of equations is given in Table 6–23.

Table 6–23 Polymath Equation Set - File **P6-09.POL**

Line	Equation
1	d(y1)/d(t) = b1 * y1 * (1 - y1 / b2)
2	d(y2)/d(t) = b3 * y1 - b4 * y2
3	b1 = 0.1
4	b2 = 4
5	b3 = 0.02
6	b4 = 0.02

Table 6–23 Polymath Equation Set - File **P6-09.POL**

7	y1(0) = 0.18
8	y2(0) = 0
9	t(0) = 0
10	t(f) = 190

The set of equations from Polymath is then converted to a MATLAB function by selection of the MATLAB output within the Polymath software. The MATLAB function generated by Polymath must be modified by removing the specification of the numerical values of the parameters and adding the vector *Beta* to the list of the input parameters of the function. The resultant MATLAB function is given in Table 6–24.

Table 6–24 MATLAB Function for Integration

No.	Commands
1	function dYfuncvecdt = ODEfun(t,Yfuncvec,Beta);
2	y1 = Yfuncvec(1);
3	y2 = Yfuncvec(2);
4	b1=Beta(1);
5	b2=Beta(2);
6	b3=Beta(3);
7	b4=Beta(4);
8	dy1dt = b1 * y1 * (1 - y1 / b2);
9	dy2dt = b3 * y1 - b4 * y2;
10	dYfuncvecdt = [dy1dt; dy2dt];

Next the function that calculates the of the sum of squares of the differences between the calculated and experimental values of the dependent variables must be prepared. This function obtains the vector of the parameters, *Beta,* and the matrix of the experimental data, *data,* as input variables (see line 1 in Table 6–25) and returns the scalar *sum* which contains the sum of squares of the differences.

The first column of the matrix *data*, which contains the independent variable values (the time in hours in this particular case), is stored in the column vector *time* (line 4). The dependent variables measured values are stored in the matrix *ym* (line 5). The matrix *yc* will contain the calculated values of the dependent variables. In its first row the measured values are stored (line 6).

Table 6–25 MATLAB Function for Calculation of Sum-of-Squares

No.	Command/Comment
1	function sum=SumSqr(Beta,data)
2	global Feval SumSave
3	Feval=Feval+1;
4	time=data(:,1);
5	ym=data(:,2:end);

Table 6–25 MATLAB Function for Calculation of Sum-of-Squares

No.	Command/Comment
6	yc(1,:)=ym(1,:);
7	[nr,nc]=size(ym); % nr - umber of data points;
8	% nc - number of dependent variables
9	tspan = [time(1) time(2)]; % Range for the independent variable
10	y0 = ym(1,:); % Initial values for the dependent variables
11	for i=2:nr
12	[t,y]=ode45(@ODEfun,tspan,y0,[],Beta);
13	yc(i,:)=y(end,:);
14	if i<nr
15	tspan = [time(i) time(i+1)];
16	y0 = yc(i,:);
17	end
18	end
19	sum=0;
20	for ic=1:nc
21	sum=sum+(ym(:,ic)-yc(:,ic))'*(ym(:,ic)-yc(:,ic));
22	end
23	if sum< SumSave*0.9
24	disp([' Function evaluation No. ' num2str(Feval) ' Sum of Squares ' num2str(sum)]);
25	SumSave=sum;
26	end

The integration is carried out in a piecewise manner (in the for loop, in lines 11 through 18) in order to obtain the dependent variable calculated values at exactly the same intervals where measured values are available. Accordingly, the start of the first range of the independent variable is the first time value from the *time* vector and the end of the range is the second time value in the same vector (line 9). The initial values of the dependent variables are the values stored in the first row of the matrix *ym* (line 10). The differential equations are integrated using the *ode45* library function in line 12. Note that the vector of parameters *Beta* is added to the input parameters of the integration routine. The dependent variable values in the last row of the resultant *y* vector are stored as the next set of calculated values in the matrix *yc* (line 13).

If there are additional data points available the next time interval is substituted into the vector tspan (line 15) and the last calculated point of the dependent variables is substituted into the vector *y*0 (line 16) and the integration is repeated.

After the integration is finished for the entire range of the independent variable, the sum of squares of the differences between the experimental and calculated values are computed (in lines 19 through 22).

The function prints intermediate results (number of function evaluations and the sum of squares value) when there is 10% reduction in the sum of squares from the previously reported value. The commands in lines 2, 3 and 23 thorough 26 are related to the printing of the intermediate results.

An addition function is needed in order to create the data file, define initial estimates for the parameters, execute a minimization routine for finding the optimal values and display the results. Those tasks are accomplished in the following *Prob_6_9* function for this problem that is shown in Table 6–26.

Table 6–26 MATLAB Function for Minimization - File **Prob_6_9.m**

No.	Command/Comment
1	function Prob_6_9
2	clear, clc, format short g, format compact
3	global Feval SumSave
4	load P6_9.txt % Load experimental data as a text file.
5	% First column - independent variable. 2nd
6	% and higher columns dependent variables.
7	% All columns must be full. Missing values
8	% are not permitted
9	Beta=[0.1 4 0.02 0.02]; % Initial estimate for parameter values
10	Feval=0;
11	SumSave=realmax;
12	[BetaOpt,Fval]=fminsearch(@SumSqr,Beta,[],P6_9);
13	disp(' ');
14	disp('Optimal Results ');
15	disp(' Parm. No. Value')
16	for i=1:size(Beta,2)
17	disp([i BetaOpt(i)]);
18	end
19	disp([' Final Sum of Squares ' num2str(Fval) ' Function evaluations ' num2str(Feval)]);
20	tspan = [P6_9(1,1) P6_9(end,1)];
21	y0=[P6_9(1,2) P6_9(2,2)];
22	[t,y]=ode45(@ODEfun,tspan,y0,[],BetaOpt);
23	for i=1:size(y,2)
24	hold off
25	plot(t,y(:,i));
26	title([' Plot of dependent variable y' int2str(i)]);
27	xlabel(' Independent variable (t)');
28	ylabel([' Dependent variable y' int2str(i)]);
29	hold on
30	plot(P6_9(:,1),P6_9(:,i+1),'*')
31	pause
32	end

In this function the experimental data matrix is loaded from a text file named *P6_9.txt* (in line 4). The initial estimates for the parameters are set in line 9 and the minimization function is called in line 12. The *fminsearch* library function of *MATLAB* is used for minimization. This function uses the simplex method of Nelder and Mead[12] to find the minimum of the sum of squares. The input parameters are: 1) the name of the function that calculates the sum of squares 2) the vector of the initial estimates of the parameter values 3) the structure *Options* (in this case it is replaced by a null matrix [ ]) and 4) the experimental data matrix. The output parameters are 1) the vector of the optimal parameter values and 2) the sum of squares of the differences between the experimental and the calculated values at the optimum.

The results are displayed in lines 13 through 19. The plots of the calculated and experimental values of the dependent variables versus the independent variable are prepared and displayed in lines 20 through 32.

The complete MATLAB program file can be assembled by incorporating the MATLAB Function for Calculation of Sum-of-Squares of Table 6–25 and the MATLAB Function for Integration of Table 6–24 with the MATLAB Function for Minimization of Table 6–26 (file - **Prob_6_9.m)**.

The final results and some of the intermediate results are shown in Table 6–27. The initial sum of squares, 36.13, was reduced to the value 1.558 after 63 function evaluations. About 210 additional function evaluations were required for reducing the sum of squares by an additional 8% to 1.436.

Table 6–27 Intermediate and Final Results of the Sum of Squares Minimization

Function evaluation No. 1 Sum of Squares 36.1271	
Function evaluation No. 63 Sum of Squares 1.5582	
Optimal Results	
Parameter Number	Value
1	0.049875
2	3.634
3	0.020459
4	0.02652
Final Sum of Squares 1.4358 Function evaluations 268	

The optimal values of b_1, b_2, b_3 and b_4 (shown in Table 6–27) can be introduced into the model equations in order to obtain the calculated values of y_1 and y_2. The plot of the calculated and experimental values versus time for the cell concentration (y_1) is shown in Figure 6–6, and for the concentration of penicillin (y_2) is shown in Figure 6–7. The data representation is very good for the concentration profiles in both cases.

The problem solution files for the problem are found in directory CHAPTER 6 and designated **P6-09.POL**, **Prob_6_9.m**, and **P6_9.txt**.

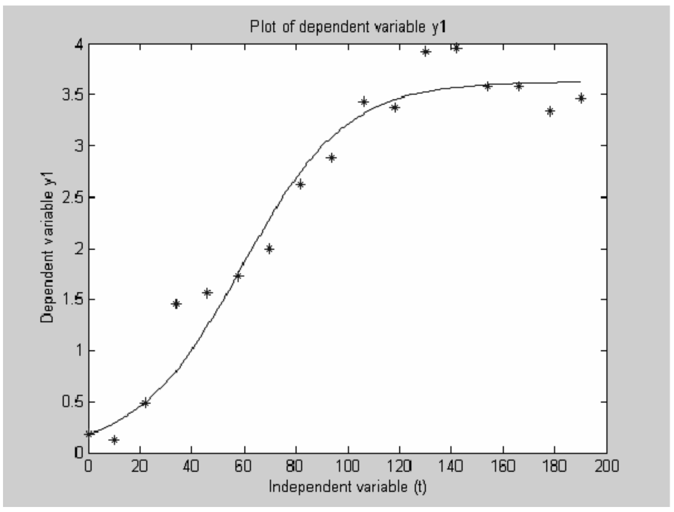

Figure 6–6 Calculated and Measured Cell Concentrations Versus Time

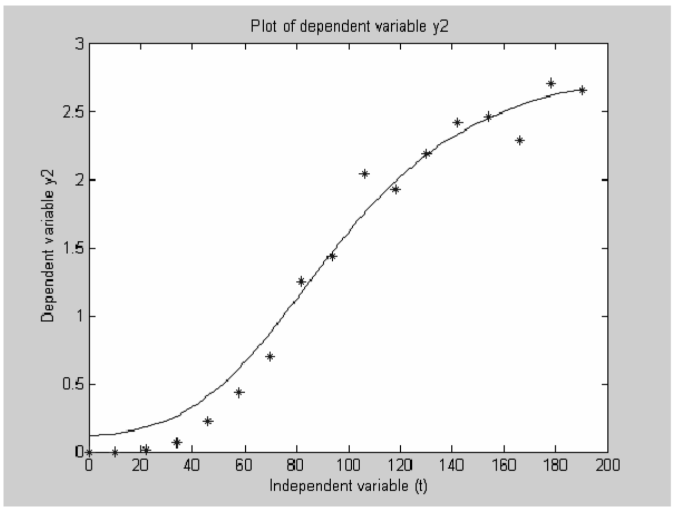

Figure 6–7 Calculated and Measured Penicillin Concentrations Versus Time