CHAPTER 5

Empirical Modeling with Data Fitting

In this chapter the model building is *empirical* in nature, i.e., the functional form of the relationship between the dependent and independent variables is found by direct examination of data related to the process.

Data fitting problems have several common elements. The *model* has the general form

$$y = f(x_1, \dots, x_N; w_1, \dots, w_M)$$

and the *parameters* w_i are determined empirically from the *observations*

$$\{(x^{(i)}, y^{(i)})\}_{i=1}^{P}$$

by requiring f to be such that

$$y^{(i)} = f(x_1^{(i)}, \dots, x_N^{(i)}; w)$$

or at least that the *error function* $E(w)$ defined by

$$S(w) = \sum_i (y^{(i)} - f(x_1^{(i)}, \dots, x_N^{(i)}; w_1, \dots, w_M))^2$$

be *small*. This error function seeks to minimize the sum of squares of the *residuals.* As we shall see, other error functions are possible but least squares is certainly the most widely used and we will focus on this approach at the outset. Later in this chapter in Section 5.3 we will also consider the important case of uniform approximation.

EXAMPLE 5.1

A *Radial basis function* model has the form

$$f(x; w, c) = w_0 + \sum_{k=1}^{N_c} w_k \phi(\|x - c_k\|)$$

where the w_k are the weights and the c_k are the centers of the basis functions. An example of a radial basis function is

$$\phi(r) = \exp(-r^2)$$

The norm $\|\cdot\|$ is generally taken to be the Euclidean distance.

5.1 LINEAR LEAST SQUARES

In this section we begin by revisiting an example from the previous chapter followed by a general formulation of linear least squares and some simple extensions to exponential fits.

5.1.1 The Mammalian Heart Revisited

Recall from Example 2.11 in Subsection 2.3.2 that a sequence of proportionalities produced the model

$$r = kw^{-1/3}$$

where w is the body weight of a mammal and r is its heart rate. The data on Figure 2.8 corresponds to observations

$$\{(w_i^{-1/3}, r_i)\}$$

collected for various measured rates and weights. The residual error for the ith measurement is

$$\epsilon_i = r_i - kw_i^{-1/3}$$

and the total squared error is

$$E = \sum_{i=1}^{P} \epsilon_i^2$$

We rewrite this error as a function of the unknown slope parameter k as

$$E(k) = \sum_{i=1}^{P} (r_i - kw_i^{-1/3})^2$$

To minimize E as a function of k we compute the derivative of E w.r.t. k, i.e.,

$$\frac{dE}{dk} = \sum_{i=1}^{P} 2(r_i - kw_i^{-1/3}) \cdot (-w_i^{-1/3}) = 0$$

From which it follows that

$$k = \frac{\sum_{i=1}^{P} r_i w_i^{-1/3}}{\sum_{i=1}^{P} w_i^{-2/3}}$$

Thus we can obtain an estimate for the slope of the line empirically from the data.

5.1.2 General Formulation

In this section we focus our attention to one of the most widely used models

$$f(x; m, b) = mx + b$$

To clean-up the notation we now use subscripts to label points for domain data that is one dimensional; we used superscripts in the previous section when the dimension

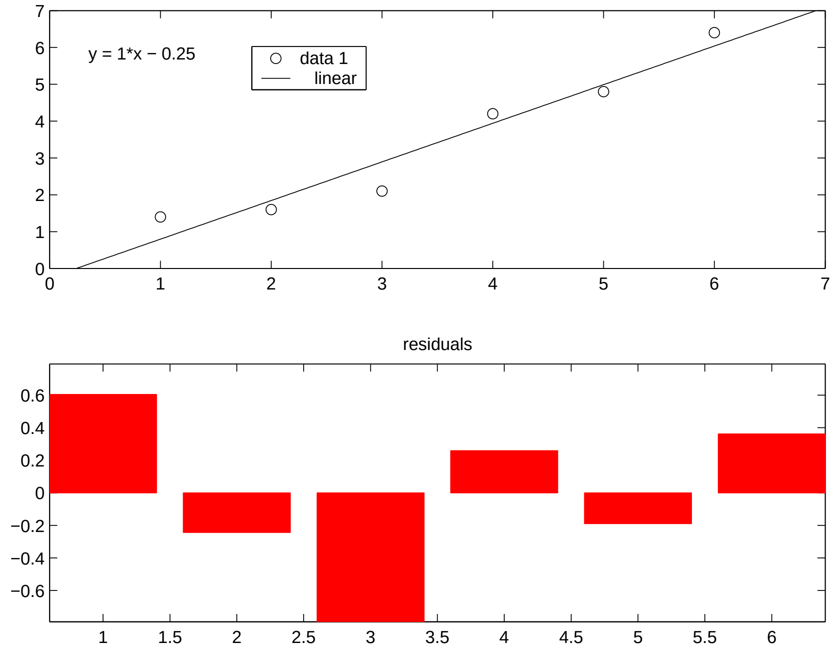


FIGURE 5.1: Linear least squares. The line $y = mx + b$ is determined such that the residuals ϵ_i^2 are minimized.

of the domain could exceed one. For a set of observations $\{(x_i, y_i)\}$, $i = 1, \dots, P$, the total squared error is given by

$$E(m, b) = \sum_{i=1}^{P} (y_i - mx_i - b)^2 \tag{5.1}$$

Now because there are two parameters that determine the error function the necessary condition for a minimum is now

$$\frac{\partial E}{\partial m} = 0$$
$$\frac{\partial E}{\partial b} = 0$$

Solving the above equations gives the slope of the line as

$$m = \frac{(\sum y_i)(\sum x_i) - P \sum y_i x_i}{(\sum x_i)^2 - P \sum x_i^2}$$

and its intercept to be

$$b = \frac{-(\sum y_i)(\sum x_i^2) + (\sum x_i)(\sum y_i x_i)}{(\sum x_i)^2 - P \sum x_i^2}$$

Interpolation Condition. In this section we present another route to the equations for m and b produced in the previous section. Again, the input data is taken as $\{x_i\}$, the output data is $\{y_i\}$ and the model equation is $y = mx + b$. Applying the *interpolation condition* for each observation we have

$$\begin{aligned} y_1 &= mx_1 + b \\ y_2 &= mx_2 + b \\ y_3 &= mx_3 + b \\ &\vdots \\ y_P &= mx_P + b \end{aligned}$$

In terms of matrices

$$\begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_P \end{pmatrix} \begin{pmatrix} b \\ m \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_P \end{pmatrix}$$

In terms of matrices we can summarize the above as

$$Xb = y$$

We can reveal the relationship between the previous approach using calculus and this approach with the interpolation condition by hitting both sides of the above matrix equation with the transpose X^T

$$\begin{pmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_P \end{pmatrix} \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_P \end{pmatrix} \begin{pmatrix} b \\ m \end{pmatrix} = \begin{pmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_P \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_P \end{pmatrix}$$

In terms of the matrices,

$$X^T X b = X^T y$$

Multiplying out produces the equations that are seen to be the same as those in the above section, i.e.,

$$\begin{pmatrix} P & \sum x_i \\ \sum x_i & \sum x_i^2 \end{pmatrix} \begin{pmatrix} b \\ m \end{pmatrix} = \begin{pmatrix} \sum y_i \\ \sum x_i y_i \end{pmatrix}$$

In linear algebra these equations are referred to as the *normal* equations.

There are many algorithms in the field of numerical linear algebra developed precisely for solving the problem

$$Xb = y$$

We will consider these more in the problems.

5.1.3 Exponential Fits

We have already seen models of the form

$$y = kx^n$$

where n was given. How about if n is unknown? Can it be determined empirically from the data? Note now that the computation of the derivative of the error function w.r.t. n is now quite complicated. This problem is resolved by converting it to a linear least squares problem now in terms of logarithms. Specifically,

$$\begin{aligned}\ln y &= \ln(kx^n)\\ &= \ln k + \ln x^n\\ &= \ln k + n \ln x\end{aligned}$$

This is now seen to be a linear least squares problem

$$y' = nx' + k'$$

where we have made the substitutions $y' = \ln y, k' = \ln k$ and $x' = \ln x$. Now one can apply the standard least squares solution to determine n and k'. The value of k can be found as well by

$$k = \exp(k')$$

5.1.4 Fitting Data with Polynomials

In the previous section we consider the basic linear model $f(x; c_0, c_1) = c_0 + c_1 x$. The simplest extension to this is the second order polynomial

$$f(x; c_0, c_1, c_2) = c_0 + c_1 x + c_2 x^2$$

Note first that adding the term $c_2 x^2$ will change the least square fit values of the coefficients c_0, c_1 obtain from the linear model and hence all the coefficients c_0, c_1 and c_2 must be computed. The least squares procedure follows along lines similar to the previous section. We assume a set of observations $\{(x_i, y_i)\}$ and define the sum of the squares of the model residuals to be the error function, i.e.,

$$E(c_0, c_1, c_2) = \sum_{i=1}^{P} (y_i - c_0 - c_1 x_i - c_2 x_i^2)^2 \tag{5.2}$$

Now requiring

$$\begin{aligned}\frac{\partial E}{\partial c_0} &= 0\\ \frac{\partial E}{\partial c_1} &= 0\\ \frac{\partial E}{\partial c_2} &= 0\end{aligned}$$

The resulting necessary conditions, written in terms of matrices, are then

$$\begin{pmatrix} P & \sum x_i & \sum x_i^2 \\ \sum x_i & \sum x_i^2 & \sum x_i^3 \\ \sum x_i^2 & \sum x_i^3 & \sum x_i^4 \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} \sum y_i \\ \sum x_i y_i \\ \sum y_i x_i^2 \end{pmatrix} \tag{5.3}$$

As for the linear model, it is possible to solve for the parameters c_0, c_1 and c_2 analytically, i.e., in closed form. It is less cumbersome to write Equation (5.3) as

$$Xc = z$$

where c is the column vector made up of the elements (c_0, c_1, c_2) and z is the column vector comprised of the elements $(\sum y_i, \sum x_i y_i, \sum y_i x_i^2)$ and X is the 3×3 matrix on the left of Equation (5.3). Now a computer package can be used to easily solve the resulting matrix equation.

Lagrange Polynomials. Consider the first degree polynomial defined by

$$P_1(x) = \frac{x - x_2}{x_1 - x_2} y_1 + \frac{x - x_1}{x_2 - x_1} y_2$$

By construction, it is clear that

$$P_1(x_1) = y_1$$

and

$$P_1(x_2) = y_2$$

So we have found the unique line passing through the points $\{(x_1, y_1), (x_2, y_2)\}$.

This procedure may be continued analogously for more points. A second degree polynomial passing through three prescribed points is given by

$$P_2(x) = \frac{(x - x_2)(x - x_3)}{(x_1 - x_2)(x_1 - x_3)} y_1 + \frac{(x - x_1)(x - x_3)}{(x_2 - x_1)(x_2 - x_3)} y_2 + \frac{(x - x_1)(x - x_2)}{(x_3 - x_1)(x_3 - x_2)} y_3$$

By construction, it again may be verified that

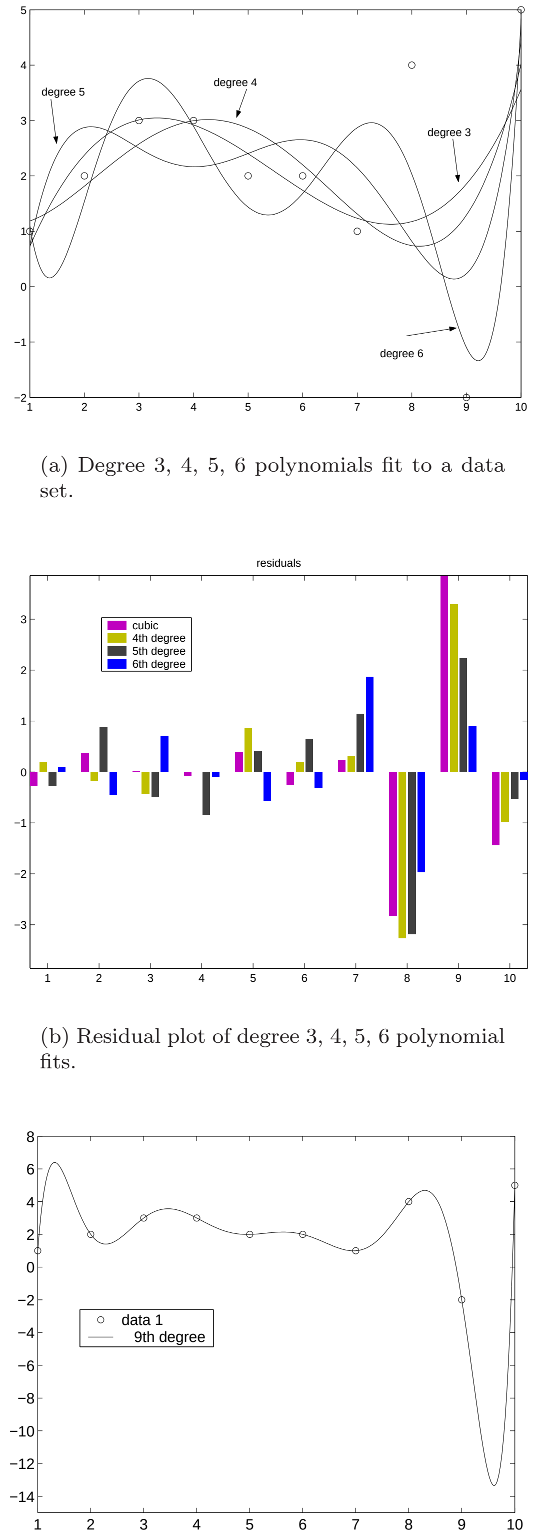
$$P_2(x_1) = y_1,$$

$$P_2(x_2) = y_2,$$

and

$$P_2(x_3) = y_3.$$

A pattern has emerged that makes it apparent that this simple procedure may be employed to fit a polynomial of degree n through a set of $n + 1$ points.



(a) Degree 3, 4, 5, 6 polynomials fit to a data set.

(b) Residual plot of degree 3, 4, 5, 6 polynomial fits.

(c) Degree 9 polynomial fit.

FIGURE 5.2: Comparative polynomial approximation to a 10 point data set.

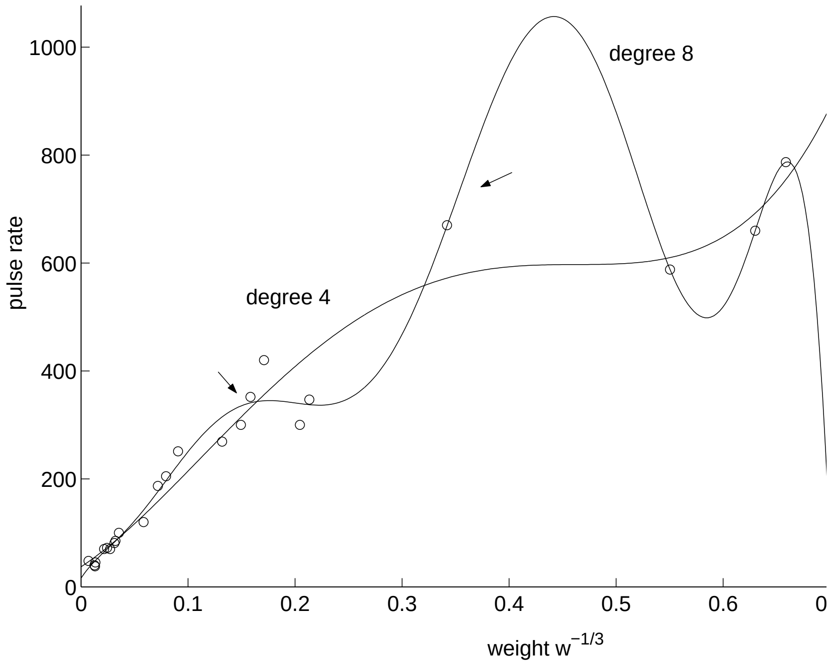


FIGURE 5.3: Degree 4 and degree 8 polynomials applied to the data $(w_i^{-1/3}, r_i)$.

5.1.5 Interpolation versus Least Squares

Dangers of Higher Order Polynomials. It is clear from the preceding section that we can always find a polynomial of degree n to exactly fit (i.e., satisfy the interpolation condition) $n+1$ points. So why not simply use high order polynomials? To answer this question consider the model

$$f(x) = c_{20}x^{20} + c_{19}x_{19} + \cdots + c_1x_1 + c_0$$

If one of the coefficients is perturbed by a small value ϵ the resulting model may predict wildly different results. For example, let

$$g(x) = (c_{20} + \epsilon)x^{20} + c_{19}x_{19} + \cdots + c_1x_1 + c_0$$

Then

$$g(x) = \epsilon x^{20} + f(x)$$

So, even if ϵ is small the difference between $f(x)$ and $g(x)$ is potentially very large. This is a manifestation of ill-conditioning of high degree polynomials, i.e., small changes in parameters may result in large changes in the function.

See Figure 5.3 for an example of the oscillations that appear with higher order polynomials.

5.2 SPLINES

One obvious procedure for reducing the need for higher order polynomials is to restrict each polynomial for the description of limited contiguous data subsets. This is the central idea behind *splines*. A spline is a piecewise defined function

$$S(x) = \begin{cases} S_1(x) & \text{if} \quad x_1 \le x < x_2, \\ S_2(x) & \text{if} \quad x_2 \le x < x_3, \\ \vdots & \vdots \\ S_j(x) & \text{if} \quad x_j \le x < x_{j+1}, \\ \vdots & \vdots \\ S_n(x) & \text{if} \quad x_n \le x < x_{n+1} \end{cases} \tag{5.4}$$

that satisfies the interpolation conditions for all the data points

$$S(x_j) = y_j = S_j(x_j) \tag{5.5}$$

Furthermore, the piecewise defined models may be joined by enforcing *auxiliary conditions* to be described. We begin with the simplest case.

5.2.1 Linear Splines

For linear splines the piecewise function takes the form

$$S_j(x) = c_0^j + c_1^j x$$

and this is valid over the interval $x_j \le x < x_{j+1}$. (See Figure 5.4.)

For $x_1 \le x < x_2$ the function $S_1(x)$ is the line passing through the points (x_1, y_1) and (x_2, y_2). The interpolation condition $S_1(x_1) = y_1$ requires

$$c_0^1 + c_1^1 x_1 = y_1.$$

The matching condition

$$S_1(x_2) = S_2(x_2) = y_2$$

requires

$$c_0^1 + c_1^1 x_2 = y_2$$

This system of two equations in two unknowns has the solutions

$$c_0^1 = \frac{x_2 y_1 - x_1 y_2}{x_2 - x_1}$$

and

$$c_1^1 = \frac{-y_1 + y_2}{x_2 - x_1}$$

Similarly for $S_2(x)$

$$S_2(x) = c_0^2 + c_1^2 x$$

The parameters c_0^2 and c_1^2 are then determined by conditions

$$\begin{cases} S_2(x_2) = y_2 \\ S_2(x_3) = S_3(y_3) = y_3 \end{cases} \tag{5.6}$$

which can be found to be

$$c_0^2 = \frac{x_3 y_2 - x_2 y_3}{x_3 - x_2}$$

and

$$c_1^2 = \frac{-y_2 + y_3}{x_3 - x_2}$$

In general, it follows

$$\begin{cases} S_j(x_j) = y_j \\ S_j(x_{j+1}) = S_{j+1}(x_{j+1}) = y_{j+1} \end{cases} \tag{5.7}$$

which can be found to be

$$c_0^j = \frac{x_{j+1} y_j - x_j y_{j+1}}{x_{j+1} - x_j}$$

and

$$c_1^j = \frac{-y_j + y_{j+1}}{x_{j+1} - x_j}$$

5.2.2 Cubic Splines

Notice that with linear splines that the function matches at interpolation points, i.e.,

$$S_j(x_{j+1}) = S_{j+1}(x_{j+1})$$

but that in general the derivative does not match, i.e.,

$$S_j^{'}(x_{j+1}) = S_{j+1}^{'}(x_{j+1})$$

This potential problem may be overcome by employing cubic splines (quadratic splines do not provide enough parameters).

For cubic splines the piecewise function takes the form

$$S_j(x) = c_0^j + c_1^j x + c_2^j x^2 + c_3^j x^3$$

and this is valid over the interval $x_j \leq x < x_{j+1}$. To simplify notation we will only consider two segments of $S(x)$

$$S(x) = \begin{cases} S_1(x) & \text{if} \quad x_1 \leq x < x_2, \\ S_2(x) & \text{if} \quad x_2 \leq x < x_3 \end{cases} \tag{5.8}$$

Now

$$S_1(x) = c_0^1 + c_1^1 x + c_2^1 x^2 + c_3^1 x^3$$

and

$$S_2(x) = c_0^2 + c_1^2 x + c_2^2 x^2 + c_3^2 x^3$$

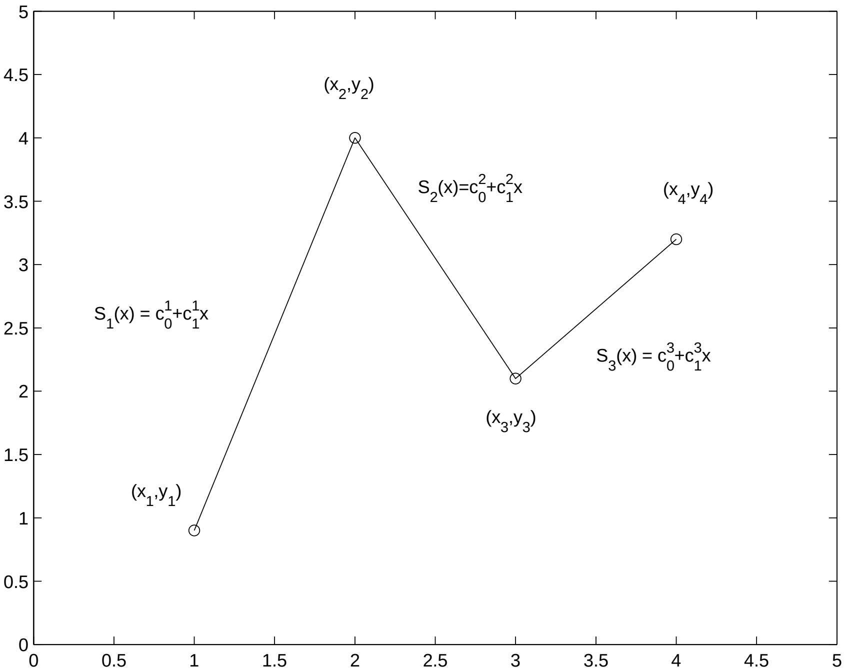


FIGURE 5.4: Linear splines.

And we observe that there are 8 parameters.

Three equations are obtained by employing the interpolation conditions

$$\begin{cases} S_1(x_1) = y_1 \\ S_2(x_2) = y_2 \\ S_2(x_3) = y_3 \end{cases} \tag{5.9}$$

The matching condition for the function value is

$$S_1(x_2) = S_2(x_2) = y_2$$

Given there are 8 parameters and only 4 equations specified we require 4 more relations.

We will require that first and second derivatives match at the interior points,

$$\begin{cases} S_1^{'}(x_2) = S_2^{'}(x_2) \\ S''{}_1(x_2) = S''{}_2(x_2) \end{cases} \tag{5.10}$$

The additional two parameters may be obtained by applying conditions on the derivatives at the endpoints x_1 and x_3. One possibility is to require

$$\begin{cases} S_1^{''}(x_1) = 0 \\ S_2^{''}(x_3) = 0 \end{cases} \tag{5.11}$$

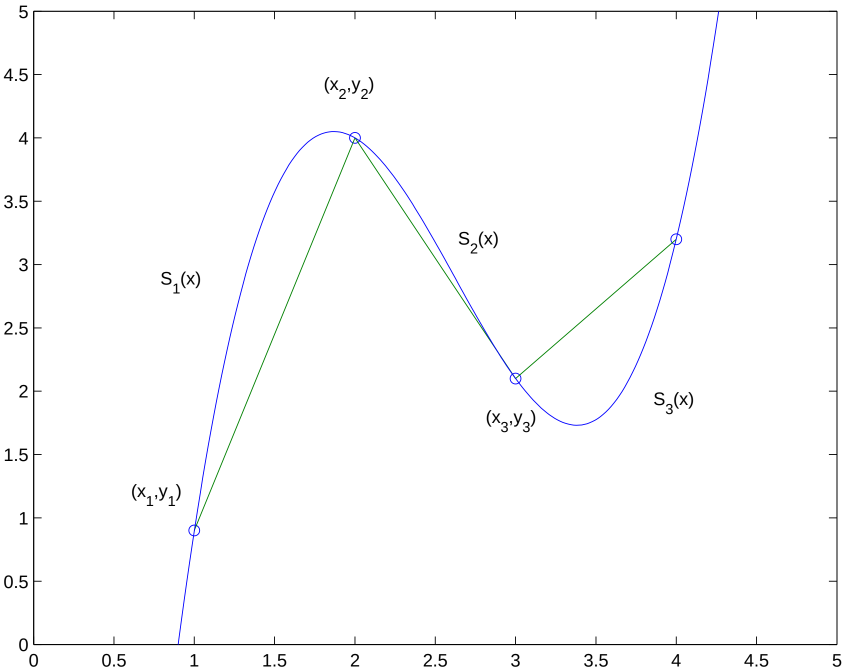


FIGURE 5.5: Cubic splines.

Since no data matching is involved these are called *natural splines*.

For a comparison of how linear and cubic splines fit a simple data set see Figure 5.5.

5.3 DATA FITTING AND THE UNIFORM APPROXIMATION

The topic of fitting a model to a data set can also be put into the framework of a linear program. Again, if we have a data set of domain (input) values $\{x_i\}$ and associated range (output) values $\{y_i\}$ we would like to determine the parameters $\{w_1, \ldots, w_k\}$ such that

$$y_i = f(x_i; w_1, \ldots, w_k)$$

An alternative to requiring that the sum of the squares of the residuals be zero is to simply minimize the maximum residual. This approach is known as the *uniform approximation*, or *Chebyshev* criterion. Since large negative residuals would make this meaningless we minimize the maximum absolute value of the residual.

To implement this idea, we first compute each residual ϵ_i as

$$\epsilon_i = y_i - f(x_i; w_1, \ldots, w_k)$$

and from all these determine the largest

$$\epsilon_{\max} = \max_i |\epsilon_i|$$

which will serve as our objective function. So the programming problem is

$$\min_{i} \epsilon_{\max}$$

where based on the definition of $\epsilon_{\max}$ we have the side constraints

$$|\epsilon_i| \leq \epsilon_{\max}$$

So, for a linear model, $f(x) = ax + b$ we have

$$\epsilon_i = y_i - ax_i - b$$

so the constraints become

$$|y_i - ax_i - b| \leq \epsilon_{\max}$$

or

$$-\epsilon_{\max} \leq y_i - ax_i - b \leq \epsilon_{\max}$$

Thus the linear program is to

$$\min \epsilon_{\max}$$

subject to the constraints

$$-\epsilon_{max} - y_i + ax_i + b \leq 0$$

$$-\epsilon_{\max} + y_i - ax_i - b \leq 0$$

where $i = 1, \ldots, P$.

In matrix notation we may rewrite the constraints as

$$\begin{pmatrix} -x_1 & -1 & -1 \\ x_1 & +1 & -1 \\ \vdots & \vdots & \vdots \\ -x_i & -1 & -1 \\ x_i & +1 & -1 \\ \vdots & \vdots & \vdots \\ -x_P & -1 & -1 \\ x_P & +1 & -1 \end{pmatrix} \begin{pmatrix} a \\ b \\ \epsilon_{\max} \end{pmatrix} \leq \begin{pmatrix} -y_1 \\ +y_1 \\ \vdots \\ -y_i \\ +y_i \\ \vdots \\ -y_P \\ +y_P \end{pmatrix}$$

Now solving this linear program to implement the uniform approximation approach on the uniform noise data of Table 5.1 produces the linear model equation

$$y = 0.9937x - 0.275,$$

while the least squares error criterion produces the model

$$y = 0.8923x - 0.5466.$$

For the uniform noise data the squared error was found to be 11.543 for the uniform approximation model while for the least squares model it is 9.998. Please see the errors in Table 5.1 and the comparative plot of the two models in Figure 5.6.

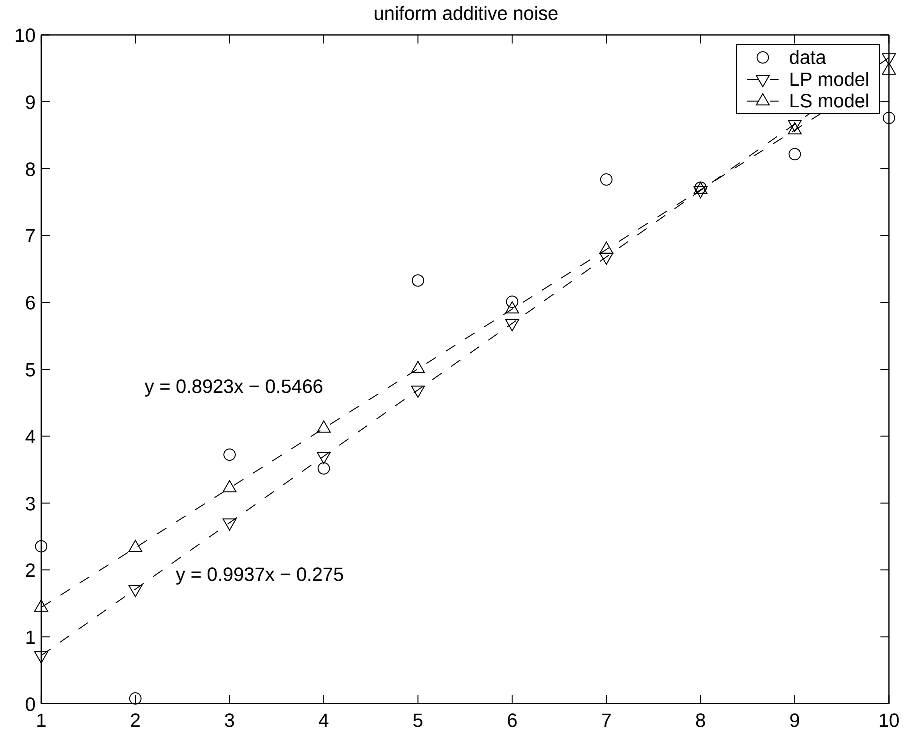


FIGURE 5.6: Least squares and uniform approximation to a linear trend with uniformly distributed additive noise.

x_i	y_i	ϵ_i uniform	ϵ_i least squares
1	2.3525	1.6338	0.9136
2	0.0786	-1.6338	-2.2527
3	3.7251	1.0191	0.5016
4	3.5179	-0.1818	-0.5979
5	6.3272	1.6338	1.3190
6	6.0113	0.3242	0.1108
7	7.8379	1.1571	1.0451
8	7.7156	0.0411	0.0305
9	8.2185	-0.4496	-0.3589
10	8.7586	-0.9032	-0.7111

TABLE 5.1: The data to be fit comprise the first two columns. The point-wise error for the uniform approximation and least squares approximation are in columns three and four, respectively. The underlined errors are those with maximum magnitude for each model. As expected, the uniform approximation has a smaller maximum error.

5.3.1 Error Model Selection?

Now we have seen that there is no unique way to compute the coefficients that fit a given model to data. The model is dependent on the way we measure the error (note that if our model is exact–e.g., the interpolation condition is satisfied–then the coefficients are unique and the error is zero). So the natural question arises: given a collection of data what is the appropriate error measure. The answer to this question lies partly in the nature of the data. If your data is very accurate, then a uniform approximation is indeed appropriate. If your data contains statistical outliers or lots of noise, a least squares approximation may be more robust.

The ultimate decision factor in what error term to use is in the added value of the model. If the predictive value of the model is superior in one error measure than another the choice is clear. Establishing superiority can often be challenging in practice.

PROBLEMS

5.1. Find the parameters a and b such that the models

$$y = ax$$

and

$$y = bx^2$$

fit the data $\{(0,0),(1,1),(2,3)\}$ according to the least squares criterion and compare the errors of the models. Without doing the calculation, can you predict what the model error would be for the model

$$y = cx + dx^2$$

Give your reasoning. *Hint:* You need not explicitly calculate c and d; considering the equations that produce them will be sufficient.

5.2. Find the line $y = b + mx$ of best fit through the data

$$\{(.1,.2),(.2,.3),(.3,.7),(.5,.2),(.75,.8)\}$$

using the least squares criterion.

5.3. Consider the model

$$f(x;c_0,c_1) = c_0 x^{-3/2} + c_1 x^{5/2}$$

Use the least squares approach to determine equations for c_0 and c_1 in terms of available data $(x_i, y_i)_{i=1}^{P}$.

5.4. Consider the mammalian pulse rate data $(r_i, w_i)_{i=1}^{24}$ provided in Table 2.1 in Subsection 2.3.2. Match the data to the models

(a) $r = b + mw^{-1/3}$

(b) $r = kw^n$

using least squares and compute the corresponding error terms given by Equation (5.1). You may use the Matlab least square codes provided, but you will first need to take appropriate transformations of the data.

5.5. Write MATLAB code to fit a second order polynomial

$$f(x;c_0,c_1,c_2) = c_0 + c_1 x + c_2 x^2$$

to the *linearized* mammalian heart data consisting of components $\{(w_i^{-1/3}, r_i)\}$. Compute the total squared error $E(c_0,c_1,c_2)$ given by Equation (5.2) and compare with the error term $E(m,b)$ found in Problem 5.4 (a). You may modify the code provided for this problem.

5.6. Derive the matrix Equation (5.3).

5.7. Rederive the matrix Equation (5.3) using the interpolation approach of Subsection 5.1.2.

5.8. Derive the 4×4 system of equations required to fit the model

$$f(x;c_0,c_1,c_2,c_3) = c_0 + c_1 x + c_2 x^2 + c_3 x^3$$

Put this system into matrix form and compare with the result in Equation (5.3). Can you extend this pattern to write down the matrix form of the least squares equations for the model

$$f(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_9 x^9$$

No exact derivation is required for this last part.

5.9. Reread Section 5.1.4 and propose a formula for a polynomial of degree 3, $P_3(x)$, such that the interpolation conditions $P_3(x_1) = y_1$, $P_3(x_2) = y_2$, $P_3(x_3) = y_3$, and $P_3(x_4) = y_4$ are satisfied.

5.10. Find the linear spline through points $\{(0,0),(2,1),(3,-2),(5,2)\}$.

5.11. Apply MATLAB's cubic spline routine to the Fort Collin's daily temperature data provided on odd days in September 2002; see Table 5.2. Use this model to predict the temperature on the even days. Compare your predictions with the actual temperature values provided in Table 5.3. Plot your results.

Day	5pm temperature
1	87.0
3	83.3
5	89.7
7	82.3
9	65.6
11	59.8
13	65.5
15	77.1
17	69.1
19	63.8
21	51.1

TABLE 5.2: Fort Collins' temperatures on odd days in September, 2002. All temperatures recorded at 5pm. Use this data to build spline model.

Day	5pm temperature
2	80.3
4	87.1
6	90.1
8	79.2
10	64.4
12	60.2
14	71.2
16	73.9
18	58.4
20	73.5

TABLE 5.3: Fort Collins' temperatures on even days in September, 2002. All temperatures recorded at 5pm. Use this data to test spline model.

5.12. Consider the data model

$$f(x) = a\sigma(bx + c)$$

where the *sigmoidal* function is given by

$$\sigma(x) = \frac{1}{1 + \exp(-x)}$$

Given a data set of input output pairs (x_i, y_i) write down the least squares optimization criterion for the unknown model parameters a, b, c. Show that the resulting system for a, b, c is nonlinear. Describe briefly how would you solve for them (without actually doing it).

5.13. Using the model in Exercise 5.12 compute a, b and c for the sample data set $\{(-.3, -.5), (-.1, -.2), (0, .1), (.2, .3), (.6, .6)\}$. Is this a good model for the data? How might you improve it?