11.2 Least Squares Approximation

For many applications, many data points are given and a cubic B-spline curve is desired which approximates their shape. The most popular method to find such an approximating curve is that of *least squares approximation*, much in the spirit of a similar method of Section 5.4.

Suppose we want to approximate a data set using a cubic B-spline curve with L polynomial segments. Recall from Section 10.1 that a cubic with only simple domain knots has the relationship $K = L + 5$, where K is the number of knots. Thus, the first step is to construct a knot sequence $u_0, \ldots, u_{K-1}$. We are given P data points $\mathbf{p}_0, \ldots, \mathbf{p}_{P-1}$, each $\mathbf{p}_i$ being associated with a parameter value v_i.[1] We wish to find a cubic B-spline curve $\mathbf{x}(u)$ such that the distances $\|\mathbf{p}_i - \mathbf{x}(v_i)\|$ are small. Figure 11.3 illustrates the geometry.

Ideally, we would have $\mathbf{p}_i = \mathbf{x}(v_i)$; $i = 0, \ldots, P-1$. If our B-spline curve $\mathbf{x}(u)$ is of the form

$$\mathbf{x}(u) = \mathbf{d}_0 N_0^3(u) + \ldots + \mathbf{d}_{D-1} N_{D-1}^3(u),$$

we would like the following to hold:

$$\begin{aligned} \mathbf{d}_0 N_0^3(v_0) + \ldots + \mathbf{d}_n N_{D-1}^3(v_0) &= \mathbf{p}_0 \\ &\vdots \\ \mathbf{d}_0 N_0^3(v_{P-1}) + \ldots + \mathbf{d}_n N_{D-1}^3(v_{P-1}) &= \mathbf{p}_{P-1}. \end{aligned}$$

This may be condensed into matrix form:

$$\begin{bmatrix} N_0^3(v_0) & \ldots & N_{D-1}^3(v_0) \\ & \vdots & \\ & \vdots & \\ N_0^3(v_{P-1}) & \ldots & N_{D-1}^3(v_{P-1}) \end{bmatrix} \begin{bmatrix} \mathbf{d}_0 \\ \vdots \\ \mathbf{d}_{D-1} \end{bmatrix} = \begin{bmatrix} \mathbf{p}_0 \\ \vdots \\ \vdots \\ \mathbf{p}_{P-1} \end{bmatrix}. \tag{11.1}$$

Or, even shorter:

$$M\mathbf{D} = \mathbf{P}. \tag{11.2}$$

Since we assume the number P of data points is larger than the number D of curve control points, this linear system is clearly *overdetermined*. We attack it, just as in the Bézier case, by simply multiplying both sides by M^{T}:

$$M^{\mathrm{T}} M\mathbf{D} = M^{\mathrm{T}}\mathbf{P}. \tag{11.3}$$

Our development lacked some details that are essential when implementing "real life" examples. How many segments L should the curve have? How should the knots u_j and the parameter values v_i be chosen?

There are no universal answers to these problems. But you might want to consider the following:

- Choose the parameters v_i according to the chord length method explained in Section 5.5.
- Select $L \approx P/10$.
- Choose the knots u_i such that approximately ten v_j fall in each interval domain knot interval $[u_i, u_{i+1}]$.

11.3 Shape Equations

A common measure for polygon shape is the use of second differences. These are of the form

$$\Delta^2 \mathbf{d}_i = \mathbf{d}_i - 2\mathbf{d}_{i+1} + \mathbf{d}_{i+2}.$$

A polygon is considered "nice" if the sum

$$\|\Delta^2 \mathbf{d}_0\| + \ldots + \|\Delta^2 \mathbf{d}_{D-3}\|$$

is small.

The following example should give credibility to this concept.

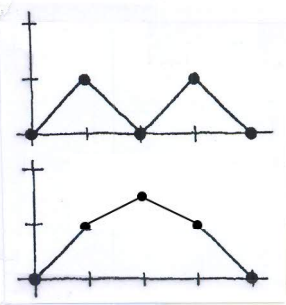
EXAMPLE 11.1

Let a polygon be given by the 2D points

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 0 \end{bmatrix}, \begin{bmatrix} 3 \\ 1 \end{bmatrix}, \begin{bmatrix} 4 \\ 0 \end{bmatrix}$$

and a second one by

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 1.5 \end{bmatrix}, \begin{bmatrix} 3 \\ 1 \end{bmatrix}, \begin{bmatrix} 4 \\ 0 \end{bmatrix}.$$



Both polygons are shown in Sketch 96; visually, the first one is "rougher" than the second one.

We compute the second differences of the first polygon:

$$\Delta^2 \mathbf{d}_0 = \begin{bmatrix} 0 \\ -2 \end{bmatrix}, \quad \Delta^2 \mathbf{d}_1 = \begin{bmatrix} 0 \\ 2 \end{bmatrix}, \quad \Delta^2 \mathbf{d}_2 = \begin{bmatrix} 0 \\ -2 \end{bmatrix}.$$

For the second polygon, we find

$$\Delta^2 \mathbf{d}_0 = \begin{bmatrix} 0 \\ -0.5 \end{bmatrix}, \quad \Delta^2 \mathbf{d}_1 = \begin{bmatrix} 0 \\ -1 \end{bmatrix}, \quad \Delta^2 \mathbf{d}_2 = \begin{bmatrix} 0 \\ -0.5 \end{bmatrix}.$$

The sum of the lengths of the first polygon's difference vectors is six; that of the second polygon is only two: the second one is smoother! This confirms that second differences tell us something about the shape of a polygon.

Returning to the topic of least squares approximation, it seems reasonable to add *shape equations* to the overdetermined system (11.2). These would be of the form

$$\begin{aligned} \mathbf{d}_0 - 2\mathbf{d}_1 + \mathbf{d}_2 &= \mathbf{0} \\ &\vdots \\ \mathbf{d}_{D-3} - 2\mathbf{d}_{D-2} + \mathbf{d}_{D-1} &= \mathbf{0}. \end{aligned}$$

With the addition of these equations, our overdetermined linear system becomes even more overdetermined. However, this is not detrimental at all—we still form the normal equations of the form (11.3) and solve them for the control points. Figure 11.4 illustrates the effect of the shape equations.

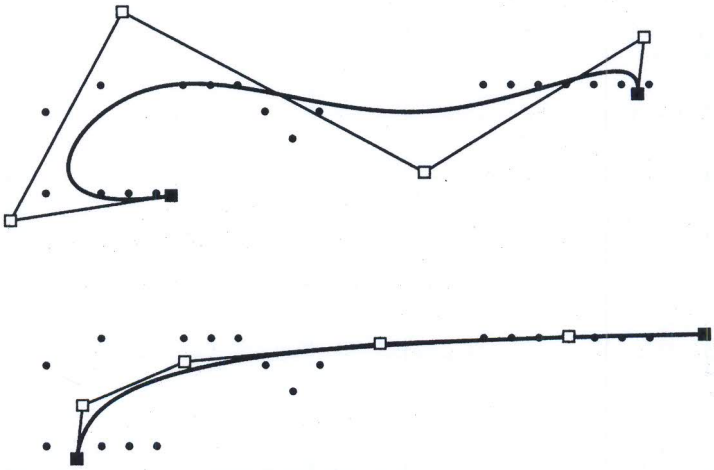


Figure 11.4.
The effect of shape equations. Top: without shape equations, and below: with shape equations.

11.4 Cubic Spline Interpolation

The most popular way of constructing cubic spline curves is through interpolation. This means that the number of data items (also called *interpolation constraints*) equals that of the unknown control points.

The task at hand is to interpolate to P given data points $\mathbf{p}_0, \ldots, \mathbf{p}_{P-1}$ with a cubic B-spline curve which has end knots of multiplicity three:

$$u_0 = u_1 = u_2, \quad u_3, \ldots, u_{K-4}, \quad u_{K-3} = u_{K-2} = u_{K-1}.$$

The junction points of the curve will be paired to the given data points; for example $\mathbf{p}_0$ will correspond to u_2, $\mathbf{p}_1$ will correspond to u_3, etc. Therefore, we will need $P-1$ curve segments, and thus $K = P+4$. A method for determining the knots based on the given data can be found in Section 5.5.

Because of the relationship between the number of knots and control points, the interpolating curve has $D = P+2$ control points, and needs $P+2$ data items to determine it. Example 11.2 should give you a feeling for the number of control points and curve segments.

EXAMPLE 11.2

Suppose we have $P = 5$ data points, and we consider the knot sequence

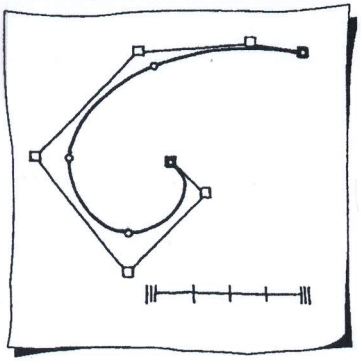
$$0, 0, 0, 1, 2, 3, 4, 4, 4$$

with $K = 5 + 4 = 9$ knots, thus having five junction points[3].

A control polygon for a cubic B-spline curve over this knot sequence is given by the $D = 7$ control points

$$\mathbf{d}_0, \ldots, \mathbf{d}_6.$$

Thus, one needs seven interpolation constraints to determine the curve. Sketch 97 shows a configuration like the one just described.



Sketch 97.
A cubic B-spline curve with four cubic segments.

Typically, one "throws in" two more data items at the ends of the curve, namely the derivatives

$$\mathbf{t}_s = \dot{\mathbf{x}}(u_2) \quad \text{and} \quad \mathbf{t}_e = \dot{\mathbf{x}}(u_{K-3}).$$

Here, $\mathbf{t}_s$ and $\mathbf{t}_e$ stand for the start and end tangents of the curve. These are called *end conditions*. The knots u_2 and u_{K-3} are the first and last domain knots, and simple expressions for these derivatives are given in (10.8) and (10.7). A good method for generating these tangents is called *Bessel tangents*. These tangents are extracted from the interpolating parabolas through the first and last three data points. They are given by

$$\mathbf{t}_s = -\frac{2\Delta_2 + \Delta_3}{\Delta_2(\Delta_2 + \Delta_3)}\mathbf{p}_0 + \frac{(\Delta_2 + \Delta_3)}{\Delta_2\Delta_3}\mathbf{p}_1 - \frac{\Delta_2}{\Delta_3(\Delta_2 + \Delta_3)}\mathbf{p}_2 \tag{11.4}$$

$$\begin{aligned}\mathbf{t}_e = &\frac{\Delta_{K-4}}{\Delta_{K-5}(\Delta_{K-5} + \Delta_{K-4})}\mathbf{p}_{L-2} \\ &- \frac{(\Delta_{K-5} + \Delta_{K-4})}{\Delta_{K-5}\Delta_{K-4}}\mathbf{p}_{L-1} \\ &+ \frac{(2\Delta_{K-4} + \Delta_{K-5})}{(\Delta_{K-5} + \Delta_{K-4})\Delta_{K-4}}\mathbf{p}_L\end{aligned} \tag{11.5}$$

Now we have as many knowns as unknowns, namely $D = P + 2$. Our interpolation conditions become

$$\begin{aligned}\mathbf{p}_0 &= \mathbf{x}(u_2) \\ \mathbf{t}_s &= \dot{\mathbf{x}}(u_2) \\ \mathbf{p}_1 &= \mathbf{x}(u_3) \\ &\vdots \\ \mathbf{t}_e &= \dot{\mathbf{x}}(u_{K-3}) \\ \mathbf{p}_{P-1} &= \mathbf{x}(u_{K-3}).\end{aligned} \tag{11.6}$$

However, as you might have noticed in Sketch 97, cubic spline interpolation with triple end knots will result in

$$\mathbf{d}_0 = \mathbf{p}_0 \quad \text{and} \quad \mathbf{d}_{D-1} = \mathbf{p}_{P-1}.$$

This clearly eliminates two unknowns, and therefore, we can eliminate two equations, and (11.6) becomes

$$\begin{aligned}\mathbf{t}_s &= \dot{\mathbf{x}}(u_2) \\ \mathbf{p}_1 &= \mathbf{x}(u_3) \\ &\vdots \\ \mathbf{p}_{P-2} &= \mathbf{x}(u_{K-4}) \\ \mathbf{t}_e &= \dot{\mathbf{x}}(u_{K-3}),\end{aligned} \tag{11.7}$$

for the $P = D - 2$ unknowns $\mathbf{d}_1, \ldots, \mathbf{d}_{D-2}$.

In theory, each data point yields an equation of the form

$$\mathbf{p}_i = \mathbf{d}_0 N_0^3(u_{2+i}) + \ldots + \mathbf{d}_{D-1} N_{D-1}^3(u_{2+i}).$$

But due to the local support property of B-spline curves, this reduces to

$$\mathbf{p}_i = \mathbf{d}_i N_i^3(u_{2+i}) + \mathbf{d}_{i+1} N_{i+1}^3(u_{2+i}) + \mathbf{d}_{i+2} N_{i+2}^3(u_{2+i}). \qquad (11.8)$$

This gives the system a *tridiagonal* structure.[4] The first and last equation in the system, the end conditions, involve only the first and last unknowns, respectively, in order to maintain this structure.

For the special case of equally spaced interior knots, we have

$$6\mathbf{p}_i = \mathbf{d}_i + 4\mathbf{d}_{i+1} + \mathbf{d}_{i+2}$$

for each equation involving a data point.

EXAMPLE 11.4

Returning to our example, notice that the knots are equally spaced. The end tangent equations are

$$\mathbf{t}_s = 3(\mathbf{d}_1 - \mathbf{d}_0) \quad \text{and} \quad \mathbf{t}_e = 3(\mathbf{d}_6 - \mathbf{d}_5),$$

where $\mathbf{d}_0$ and $\mathbf{d}_6$ are known. Therefore, the linear system is

$$\begin{bmatrix} 1 & & & & & & \\ 3/2 & 7/2 & 1 & & & & \\ & 1 & 4 & 1 & & & \\ & & & 1 & 7/2 & 3/2 & \\ & & & & & 1 & \end{bmatrix} \begin{bmatrix} \mathbf{d}_1 \\ \mathbf{d}_2 \\ \mathbf{d}_3 \\ \mathbf{d}_4 \\ \mathbf{d}_5 \end{bmatrix} = \begin{bmatrix} \mathbf{d}_0 + \frac{1}{3}\mathbf{t}_s \\ 6\mathbf{p}_1 \\ 6\mathbf{p}_2 \\ 6\mathbf{p}_3 \\ \mathbf{d}_6 - \frac{1}{3}\mathbf{t}_e \end{bmatrix}$$

An example is illustrated in Figure 11.5.