CHAPTER 6

Parametric Spline Curves

When we introduced splines in Chapter 1 we focused on spline curves, or more precisely, vector valued spline functions. In Chapters 2 and 4 we then established the basic theory of spline functions and B-splines, and in Chapter 5 we studied a number of methods for constructing spline functions that approximate given data. In this chapter we return to spline curves and show how the approximation methods in Chapter 5 can be adapted to this more general situation.

We start by giving a formal definition of parametric curves in Section 6.1, and introduce parametric spline curves in Section 6.2.1. In the rest of Section 6.2 we then generalise the approximation methods in Chapter 5 to curves.

6.1 Definition of Parametric Curves

In Section 1.2 we gave an intuitive introduction to parametric curves and discussed the significance of different parameterisations. In this section we will give a more formal definition of parametric curves, but the reader is encouraged to first go back and reread Section 1.2 in Chapter 1.

6.1.1 Regular parametric representations

A parametric curve will be defined in terms of *parametric representations.*

Definition 6.1. *A vector function or mapping* $\boldsymbol{f} : [a,b] \mapsto \mathbb{R}^s$ *of the interval* $[a,b]$ *into* $\mathbb{R}^s$ *for* $s \geq 2$ *is called a parametric representation of class* C^m *for* $m \geq 1$ *if each of the* s *components of* $\boldsymbol{f}$ *has continuous derivatives up to order* m. *If, in addition, the first derivative of* $\boldsymbol{f}$ *does not vanish in* $[a,b]$,

$$D\boldsymbol{f}(t) = \boldsymbol{f}'(t) \neq 0, \qquad \text{for } t \in [a,b],$$

then $\boldsymbol{f}$ *is called* a regular *parametric representation of class* C^m.

A parametric representation will often be referred to informally as a curve, although the term parametric curve will be given a more precise meaning later.

In this chapter we will always assume the parametric representations to be sufficiently smooth for all operations to make sense.

Note that a function $y = h(x)$ always can be considered as a curve through the parametric representation $\boldsymbol{f}(u) = \big(u, h(u)\big)$.

If we imagine travelling along the curve and let u denote the elapsed time of our journey, then the length of $\boldsymbol{f}'(u)$ which we denote by $||\boldsymbol{f}'(u)||$, gives the velocity with which we travel at time u, while the direction of $\boldsymbol{f}'(u)$ gives the direction in which we travel, in other words the tangent to the curve at time u. With these interpretations a regular curve is one where we never stop as we travel along the curve.

The straight line segment

$$\boldsymbol{f}(u) = (1-u)\boldsymbol{p}_0 + u\boldsymbol{p}_1, \quad \text{for } u \in [0,1],$$

where $\boldsymbol{p}_0$ and $\boldsymbol{p}_1$ are points in the plane, is a simple example of a parametric representation. Since $\boldsymbol{f}'(u) = \boldsymbol{p}_1 - \boldsymbol{p}_0$ for all u, we have in fact that $\boldsymbol{f}$ is a regular parametric representation, provided that $\boldsymbol{p}_0 \neq \boldsymbol{p}_1$. The tangent vector is, as expected, parallell to the curve, and the speed along the curve is constant.

As another example, let us consider the unit circle. It is easy to check that the mapping given by

$$\boldsymbol{f}(u) = \bigl(x(u), y(u)\bigr) = (\cos u, \sin u)$$

satisfies the equation $x(u)^2 + y(u)^2 = 1$, so that if u varies from 0 to 2π, the whole unit circle will be traced out. We also have $||\boldsymbol{f}'(u)|| = 1$ for all u, so that $\boldsymbol{f}$ is a regular parametric representation.

One may wonder what the significance of the regularity condition $\boldsymbol{f}'(u) \neq 0$ is. Let us consider the parametric representation given by

$$\boldsymbol{f}(u) = \begin{cases} (0, u^2), & \text{for } u < 0; \\ (u^2, 0), & \text{for } u \geq 0; \end{cases}$$

in other words, for $u < 0$ the image of $\boldsymbol{f}$ is the positive y-axis and for $u > 0$, the image is the positive x-axis. A plot of $\boldsymbol{f}$ for $u \in [-1, 1]$ is shown in Figure 6.1 (a). The geometric figure traced out by $\boldsymbol{f}$ clearly has a right angle corner at the origin, but $\boldsymbol{f}'$ which is given by

$$\boldsymbol{f}'(u) = \begin{cases} (0, 2u), & \text{for } u < 0; \\ (2u, 0), & \text{for } u > 0; \end{cases}$$

is still continuous for all u. The source of the problem is the fact that $\boldsymbol{f}'(0) = 0$. For this means that as we travel along the curve, the velocity becomes zero at $u = 0$ and cancels out the discontinuity in the tangent direction, so that we can manage to turn the corner. On the other hand, if we consider the unit tangent vector $\boldsymbol{\theta}(u)$ defined by

$$\boldsymbol{\theta}(u) = \boldsymbol{f}'(u)/||\boldsymbol{f}'(u)||,$$

we see that

$$\boldsymbol{\theta}(u) = \begin{cases} (0, -1), & \text{for } u < 0; \\ (1, 0), & \text{for } u > 0. \end{cases}$$

As expected, the unit tangent vector is discontinuous at $u = 0$.

A less obvious example where the same problem occurs is shown in Figure 6.1 (b). The parametric representation is $\boldsymbol{f}(u) = (u^2, u^3)$ which clearly has a continuous tangent, but again we have $\boldsymbol{f}'(0) = (0, 0)$ which cancels the discontinuity in the unit tangent vector at $u = 0$. To avoid the problems that may occur when the tangent becomes zero, it is common, as in Definition 6.1, to assume that the parametric representation is regular.

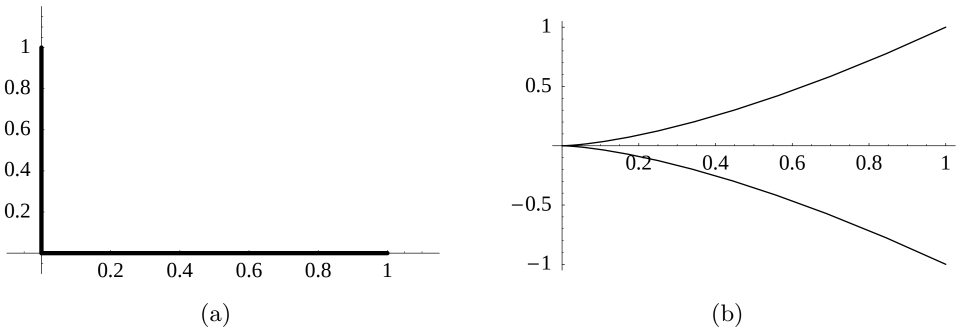

Figure 6.1. A parametric representation with continuous first derivative but discontinuous unit tangent (a), and the parametric representation $\boldsymbol{f}(u) = (u^2, u^3)$ (b).

6.1.2 Changes of parameter and parametric curves

If we visualise a parametric representation through its graph as we have done here, it is important to know whether the same graph may be obtained from different parametric representations. It is easy to see that the answer to this question is yes. Consider for example again the unit circle $\boldsymbol{f}(u) = (\cos u, \sin u)$. If we substitute $u = 2\pi v$, we obtain the parametric representation

$$\hat{\boldsymbol{r}}(v) = (\cos 2\pi v, \sin 2\pi v).$$

As v varies in the interval $[0, 1]$, the original parameter u will vary in the interval $[0, 2\pi]$ so that $\hat{\boldsymbol{r}}(v)$ will trace out the same set of points in $\mathbb{R}^2$ and therefore yield the same graph as $\boldsymbol{f}(u)$. The mapping $u = 2\pi v$ is called a *change of parameter.*

Definition 6.2. *A real function $u(v)$ defined on an interval I is called an allowable change of parameter of class C^m if it has m continuous derivatives, and the derivative $u'(v)$ is nonzero for all v in I. If $u'(v)$ is positive for all v then it is called an orientation preserving change of parameter. If $\boldsymbol{f}(u)$ is a parametric representation, then $\boldsymbol{g}(v) = \boldsymbol{f}\big(u(v)\big)$ is called a parametrisation of $\boldsymbol{f}$.*

From the chain rule we see that

$$\boldsymbol{g}'(v) = u'(v)\boldsymbol{f}'\big(u(v)\big).$$

From this it is clear that even if $\boldsymbol{f}$ is a regular parametric representation, we can still have $\boldsymbol{g}'(v) = 0$ for some v if $u'(v)$ becomes zero for some v. This is avoided by requiring $u'(v) \neq 0$ as in Definition 6.2.

If $u'(v) > 0$ for all v, the points on the graph of the curve are traced in the same order both by $\boldsymbol{f}$ and $\boldsymbol{g}$, the two representations have the same orientation. If $u'(v) < 0$ for all v, then $\boldsymbol{f}$ and $\boldsymbol{g}$ have opposite orientation, the points on the graph are traced in opposite orders. The change of parameter $u(v) = 2\pi v$ of the circle above was orientation preserving.

Note that since $u'(v) \neq 0$, the function $u(v)$ is one-to-one so that the inverse $v(u)$ exists and is an allowable change of parameter.

The redundancy in the representation of geometric objects can be resolved in a standard way. We simply say that two parametric representations are equivalent if they are related by a change of parameter (in this situation we will often say that one representation is a reparametrization of the other).

Definition 6.3. *A regular parametric curve is the equivalence class of parameterisations of a given regular parametric representation. A particular parametric representation of a curve is called a parametrisation of the curve.*

We will use this definition very informally. Most of the time we will just have a representation $\boldsymbol{f}$ which we will refer to as a parametrisation of a curve or simply a curve.

As an interpretation of the different parameterisations of a curve it is constructive to extend the analogy to travelling along a road. As mentioned above, we can think of the parameter u as measuring the elapsed time as we travel along the curve, and the length of the tangent vector as the velocity with which we travel. The road with its hills and bends is fixed, but there are still many ways to travel along it. We can both travel at different velocities and in different directions. This corresponds to different parameterisations.

A natural question is now whether there is a preferred way of travelling along the road. A mathematician would probably say that the best way to travel is to maintain a constant velocity, and we shall see later that this does indeed simplify the analysis of a curve. On the other hand, a physicist (and a good automobile driver) would probably say that it is best to go slowly around sharp corners and faster along straighter parts of the curve. For the purpose of constructing spline curves it turns out that this latter point of view usually give the best results.

6.1.3 Arc length parametrisation

Let us end this brief introduction to parametric curves by a discussion of parameterisations with constant velocity. Suppose that we have a parametrisation such that the tangent vector has constant unit length along the curve. Then the difference in parameter value at the beginning and end of the curve would equal the length of the curve, which is reason enough to study such parameterisations. This justifies the next definition.

Definition 6.4. *A regular parametric curve $\boldsymbol{g}(\sigma)$ in $\mathbb{R}^s$ is said to be parametrised by arc length if $||\boldsymbol{g}'(\sigma)|| = 1$ for all σ.*

Let $\boldsymbol{f}(u)$ be a given regular curve with $u \in [a, b]$, and let $\boldsymbol{g}(\sigma) = \boldsymbol{f}(u(\sigma))$ be a reparametrisation such that $||\boldsymbol{g}'(\sigma)|| = 1$ for all σ. Since $\boldsymbol{g}'(\sigma) = u'(\sigma)\boldsymbol{f}'(u(\sigma))$, we see that we must have $|u'(\sigma)| = 1/||\boldsymbol{f}'(u(\sigma))||$ or $|\sigma'(u)| = ||\boldsymbol{f}'(u)||$ (this follows since $u(\sigma)$ is invertible with inverse $\sigma(u)$ and $u'(\sigma)\sigma'(u) = 1$). The natural way to achieve this is to define $\sigma(u)$ by

$$\sigma(u) = \int_a^u ||\boldsymbol{f}'(v)||\, dv. \tag{6.1}$$

We sum this up in a proposition.

Proposition 6.5. *Let $\boldsymbol{f}(u)$ be a given regular parametric curve. The change of parameter given by (6.1) reparametrises the curve by arc length, so that if $\boldsymbol{g}(\sigma) = \boldsymbol{f}\big(u(\sigma)\big)$ then $||\boldsymbol{g}'(\sigma)|| = 1$.*

Note that $\sigma(u)$ as given by (6.1) gives the length of the curve from the starting point $\boldsymbol{f}(a)$ to the point $\boldsymbol{f}(u)$. This can be seen by sampling $\boldsymbol{f}$ at a set of points, computing the length of the piecewise linear interpolant to these points, and then letting the density of the points go to infinity.

Proposition 6.6. *The length of a curve $\boldsymbol{f}$ defined on an interval $[a,b]$ is given by*

$$L(\boldsymbol{f}) = \int_a^b \big\|\boldsymbol{f}'(u)\big\|\, du$$

It should be noted that parametrisation by arc length is not unique. The orientation can be reversed and the parameterisation may be translated by a constant. Note also that if we have a parametrisation that is constant but not arc length, then arc length parametrisation can be obtained by a simple scaling.

Parametrisation by arc length is not of much practical importance in approximation since the integral in (6.1) very seldom can be expressed in terms of elementary functions, and the computation of the integral is usually too expensive. One important exception is the circle. As we saw at the beginning of the chapter, the parametrisation $\boldsymbol{r}(u) = (\cos u, \sin u)$ is by arc length.

6.2 Approximation by Parametric Spline Curves

Having defined parametric curves formally, we are now ready to define parametric spline curves. This is very simple, we just let the coefficients that multiply the B-splines be points in $\mathbb{R}^s$ instead of real numbers. We then briefly consider how the spline approximation methods that we introduced in for spline functions can be generalised to curves.

6.2.1 Definition of parametric spline curves

A spline curve $\boldsymbol{f}$ must, as all curves, be defined on an interval I and take its values in $\mathbb{R}^s$. There is a simple and obvious way to achieve this.

Definition 6.7. *A parametric spline curve in $\mathbb{R}^s$ is a spline function where each B-spline coefficient is a point in $\mathbb{R}^s$. More specifically, let $\boldsymbol{t} = (t_i)_{i=1}^{n+d+1}$ be a knot vector for splines of degree d. Then a parametric spline curve of degree d with knot vector $\boldsymbol{t}$ and coefficients $\boldsymbol{c} = (\boldsymbol{c}_i)_{i=1}^{n}$ is given by*

$$\boldsymbol{g}(u) = \sum_{i=1}^{n} \boldsymbol{c}_i B_{i,d,\boldsymbol{t}}(u),$$

where each $\boldsymbol{c}_i = (c_i^1, c_i^2, \ldots, c_i^s)$ is a vector in $\mathbb{R}^s$. The set of all spline curves in $\mathbb{R}^s$ of degree d with knot vector $\boldsymbol{t}$ is denoted by $\mathbb{S}_{d,\boldsymbol{t}}^s$.

In Definition 6.7, a spline curve is defined as a spline function where the coefficients are points in $\mathbb{R}^s$. From this it follows that

$$\begin{aligned}\boldsymbol{g}(u) &= \sum_i \boldsymbol{c}_i B_i(u) = \sum_i (c_i^1, \ldots, c_i^s) B_i(u) \\ &= \Big(\sum_i c_i^1 B_i(u), \ldots, \sum_i c_i^s B_i(u)\Big) \\ &= \big(g^1(u), \ldots, g^s(u)\big),\end{aligned} \tag{6.2}$$

so that $\boldsymbol{g}$ is a vector of spline functions. This suggests a more general definition of spline curves where the degree and the knot vector in the s components need not be the same, but this is not common and seems to be of little practical interest.

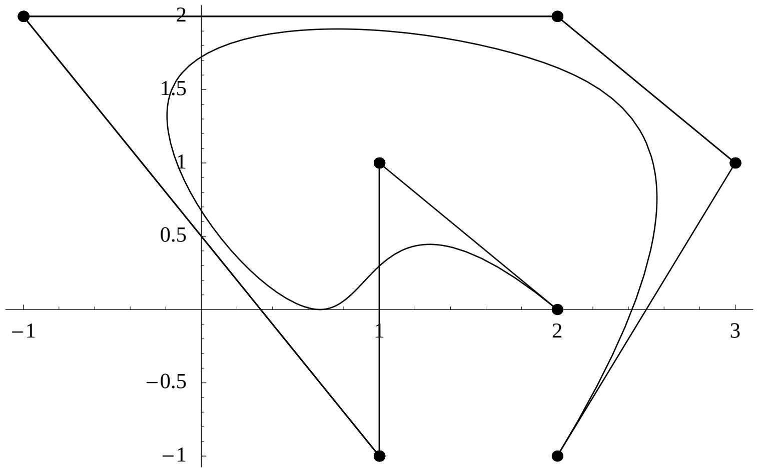

Figure 6.2. A cubic parametric spline curve with its control polygon.

Since a spline curve is nothing but a vector of spline functions as in (6.2), it is simple to compute $\boldsymbol{f}(u)$. Just apply a routine like Algorithm 2.20 to each of the component spline functions $g^1, \ldots, g^s$. If the algorithm has been implemented in a language that supports vector arithmetic, then evaluation is even simpler. Just apply Algorithm 2.20 directly to $\boldsymbol{g}$, with vector coefficients. The result will be the vector $\boldsymbol{g}(u) = \big(g^1(u), \ldots, g^s(u)\big)$.

Example 6.8. As an example of a spline curve, suppose that we are given n points $\boldsymbol{p} = (\boldsymbol{p}_i)_{i=1}^n$ in the plane with $\boldsymbol{p}_i = (x_i, y_i)$, and define the knot vector $\boldsymbol{t}$ by

$$\boldsymbol{t} = (1, 1, 2, 3, 4, \ldots, n-2, n-1, n, n).$$

Then the linear spline curve

$$\boldsymbol{g}(u) = \sum_{i=1}^{n} \boldsymbol{p}_i B_{i,1,\boldsymbol{t}}(u) = \Big(\sum_{i=1}^{n} x_i B_{i,1,\boldsymbol{t}}(u), \sum_{i=1}^{n} y_i B_{i,1,\boldsymbol{t}}(u)\Big)$$

is a representation of the piecewise linear interpolant to the points $\boldsymbol{p}$.

An example of a cubic spline curve with its control polygon is shown in Figure 6.2, and this example gives a good illustration of the fact that a spline curve is contained in the convex hull of its control points. This, we remember, is clear from the geometric construction of spline curves in Chapter 1.

Proposition 6.9. *A spline curve $\boldsymbol{g} = \sum_{i=1}^n \boldsymbol{c}_i B_{i,d,\boldsymbol{t}}$ defined on a $d+1$-extended knot vector $\boldsymbol{t}$ is a subset of the convex hull of its coefficients,*

$$\boldsymbol{g}(u) \in \mathbb{CH}(\boldsymbol{c}_1, \ldots, \boldsymbol{c}_n), \qquad \textit{for any } u \in [t_{d+1}, t_{n+1}].$$

If u is restricted to the interval $[t_\mu, t_{\mu+1}]$ then

$$\boldsymbol{g}(u) \in \mathbb{CH}(\boldsymbol{c}_{\mu-d}, \ldots, \boldsymbol{c}_\mu).$$

To create a spline curve, we only have to be able to create spline functions, since a spline curve is only a vector with spline functions in each component. All the methods described in previous chapters for approximation with spline functions can therefore also be utilised for construction of spline curves. To differentiate between curve approximation and function approximation, we will often refer to the methods of Chapter 5 as *functional approximation methods*.

6.2.2 The parametric variation diminishing spline approximation

In Section 5.4, we introduced the variation diminishing spline approximation to a function. This generalises nicely to curves.

Definition 6.10. *Let $\boldsymbol{f}$ be a parametric curve defined on the interval $[a,b]$, and let $\boldsymbol{t}$ be a $d+1$-extended knot vector with $t_{d+1}=a$ and $t_{n+1}=b$. The parametric variation diminishing spline approximation $V\boldsymbol{f}$ is defined by*

$$(V\boldsymbol{f})(u)=\sum_{i=1}^{n}\boldsymbol{f}(t_i^*)B_{i,d,\boldsymbol{t}}(u),$$

where $t_i^*=(t_{i+1}+\cdots t_{i+d})/d$.

Note that the definition of $V\boldsymbol{f}$ means that

$$V\boldsymbol{f}=(Vf^1,\dots,Vf^s).$$

If we have implemented a routine for determining the variation diminishing approximation to a scalar function ($s=1$), we can therefore determine $V\boldsymbol{f}$ by calling the scalar routine s times, just as was the case with evaluation of the curve at a point. Alternatively, if the implementation uses vector arithmetic, we can call the function once but with vector data.

A variation diminishing approximation to a half circle is shown in Figure 6.3.

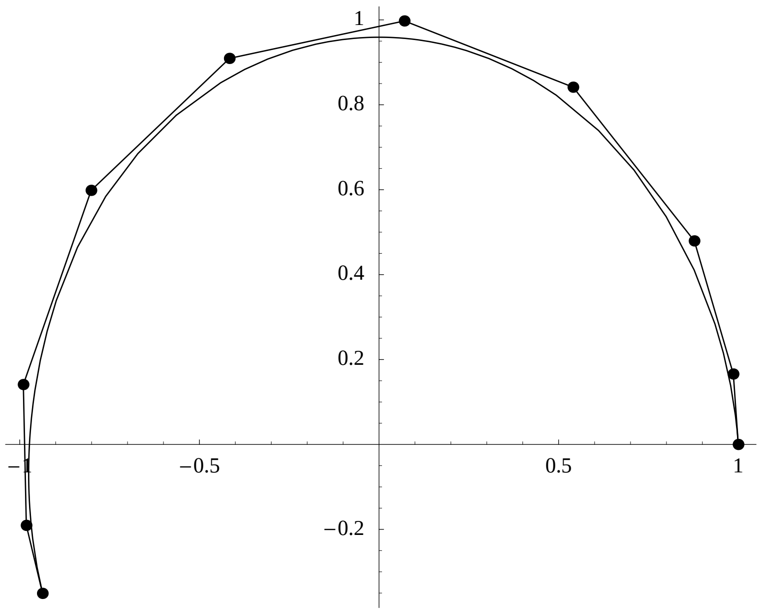

Figure 6.3. A cubic variation diminishing approximation to part of a circle.

It is much more difficult to employ the variation diminishing spline approximation when only discrete data are given, since somehow we must determine a knot vector. This is true for functional data, and for parametric data we have the same problem. In addition, we must also determine a parametrisation of the points. This is common for all parametric approximation schemes when they are applied to discrete data and is most easily discussed for cubic spline interpolation where it is easy to determine a knot vector.

6.2.3 Parametric spline interpolation

In Section 5.2, we considered interpolation of a given function or given discrete data by cubic splines, and we found that the cubic C^2 spline interpolant in a sense was the best of all C^2 interpolants. How can this be generalised to curves?

Proposition 6.11. *Let $\big(u_i, \boldsymbol{f}(u_i)\big)_{i=1}^m$ be given data sampled from the curve $\boldsymbol{f}$ in $\mathbb{R}^s$, and form the knot vector*

$$\boldsymbol{t} = (u_1, u_1, u_1, u_1, u_2, \dots, u_{m-1}, u_m, u_m, u_m, u_m).$$

Then there is a unique spline curve $\boldsymbol{g} = I_N \boldsymbol{f}$ in $\mathbb{S}^s_{3,\boldsymbol{t}}$ that satisfies

$$\boldsymbol{g}(u_i) = \boldsymbol{f}(u_i), \qquad \text{for } i = 1, \dots, m, \tag{6.3}$$

with the natural end conditions $\boldsymbol{g}''(u_1) = \boldsymbol{g}''(u_m) = \boldsymbol{0}$, and this spline curve $\boldsymbol{g}$ uniquely minimises

$$\Big\| \int_{u_1}^{u_m} \boldsymbol{h}''(u)\, du \Big\|$$

when $\boldsymbol{h}$ varies over the class of C^2 parametric representations that satisfies the interpolation conditions (6.3).

Proof. All the statements follow by considering the s functional interpolation problems separately. ∎

Note that Proposition 6.11 can also be expressed in the succinct form

$$I_N \boldsymbol{f} = (I_N f^1, \dots, I_N f^s).$$

This means that the interpolant can be computed by solving s functional interpolation problems. If we go back to Section 5.2.2, we see that the interpolant is determined by solving a system of linear equations. If we consider the s systems necessary to determine $I_N \boldsymbol{f}$, we see that it is only the right hand side that differs; the coefficient matrix $\boldsymbol{A}$ remains the same. This is known to be of great advantage since the LU-factorisation of the coefficient matrix can be computed once and for all and the s solutions computed by back substitution; for more information consult a text on Numerical Linear Algebra. As for evaluation and the variation diminishing approximation, this makes it very simple to implement cubic spline interpolation in a language that supports vector arithmetic: Simply call the routine for functional interpolation with vector data.

We have focused here on cubic spline interpolation with natural end conditions, but Hermite and free end conditions can be treated completely analogously.

Let us turn now to cubic parametric spline interpolation in the case where the data are just given as discrete data.

Problem 6.12. *Let $(\boldsymbol{p}_i)_{i=1}^m$ be a set of points in $\mathbb{R}^s$. Find a cubic spline $\boldsymbol{g}$ in some spline space $\mathbb{S}_{3,\boldsymbol{t}}$ such that*

$$\boldsymbol{g}(u_i) = \boldsymbol{p}_i, \qquad \text{for } i = 1, \dots, m,$$

for some parameter values $(u_i)_{i=1}^m$ with $u_1 < u_2 < \cdots < u_m$.

Problem 6.12 is a realistic problem. A typical situation is that somehow a set of points on a curve has been determined, for instance through measurements; the user then wants the computer to draw a 'nice' curve through the points. In such a situation the knot vector is of course not known in advance, but for functional approximation it could easily be determined from the abscissae. In the present parametric setting this is a fundamentally more difficult problem. An example may be illuminating.

Example 6.13. Suppose that m points in the plane $\boldsymbol{p} = (\boldsymbol{p}_i)_{i=1}^m$ with $\boldsymbol{p}_i = (x_i, y_i)$ are given. We seek a cubic spline curve that interpolates the points $\boldsymbol{p}$. We can proceed as follows. Associate with each data point $\boldsymbol{p}_i$ the parameter value i. If we are also given the derivatives (tangents) at the ends as (x_1', y_1') and (x_m', y_m'), we can apply cubic spline interpolation with Hermite end conditions to the two sets of data $(i, x_i)_{i=1}^n$ and $(i, y_i)_{i=1}^n$. The knot vector will then for both of the two components be

$$\boldsymbol{t} = (1, 1, 1, 1, 2, 3, 4, \ldots, m-2, m-1, m, m, m, m).$$

We can then perform the two steps

(i) Find the spline function $p^1 \in \mathbb{S}_{3,\boldsymbol{t}}$ with coefficients $\boldsymbol{c}^1 = (c_i^1)_{i=1}^{m+2}$ that interpolates the points $(i, x_i)_{i=1}^m$ and satisfies $Dp^1(1) = x_1'$ and $Dp^1(m) = x_m'$.

(ii) Find the spline function $p^2 \in \mathbb{S}_{3,\boldsymbol{t}}$ with coefficients $\boldsymbol{c}^2 = (c_i^2)_{i=1}^{m+2}$ that interpolates the points $(i, y_i)_{i=1}^m$ and satisfies $Dp^2(1) = y_1'$ and $Dp^2(m) = y_m'$.

Together this yields a cubic spline curve

$$\boldsymbol{g}(u) = \sum_{i=1}^{m+2} \boldsymbol{c}_i B_{i,3,\boldsymbol{t}}(u)$$

that satisfies $\boldsymbol{g}(i) = \boldsymbol{p}_i$ for $i = 1, 2, \ldots, m$.

The only part of the construction of the cubic interpolant in Example 6.13 that is different from the corresponding construction for spline functions is the assignment of the parameter value i to the point $\boldsymbol{f}_i = (x_i, y_i)$ for $i = 1, 2, \ldots, n$, and therefore also the construction of the knot vector. When working with spline functions, the abscissas of the data points became the knots; for curves we have to choose the knots specifically by giving the parameter values at the data points. Somewhat arbitrarily we gave point number i parameter value i in Example 6.13, this is often termed *uniform parametrisation.*

Going back to Problem 6.12 and the analogy with driving, we have certain places that we want to visit (the points $\boldsymbol{p}_i$) and the order in which they should be visited, but we do not know when we should visit them (the parameter values u_i). Should one for example try to drive with a constant speed between the points, or should one try to make the time spent between points be constant? With the first strategy one might get into problems around a sharp corner where a good driver would usually slow down, and the same can happen with the second strategy if two points are far apart (you must drive fast to keep the time), with a sharp corner just afterwards.

In more mathematical terms, the problem is to guess how the points are meant to be parametrised—which parametric representation are they taken from? This is a difficult problem that so far has not been solved in a satisfactory way. There are methods available though, and in the next section we suggest three of the simplest.

6.2.4 Assigning parameter values to discrete data

Let us recall the setting. We are given m points $(\boldsymbol{p}_i)_{i=1}^m$ in $\mathbb{R}^s$ and need to associate a parameter value u_i with each point that will later be used to construct a knot vector for spline approximation. Here we give three simple alternatives.

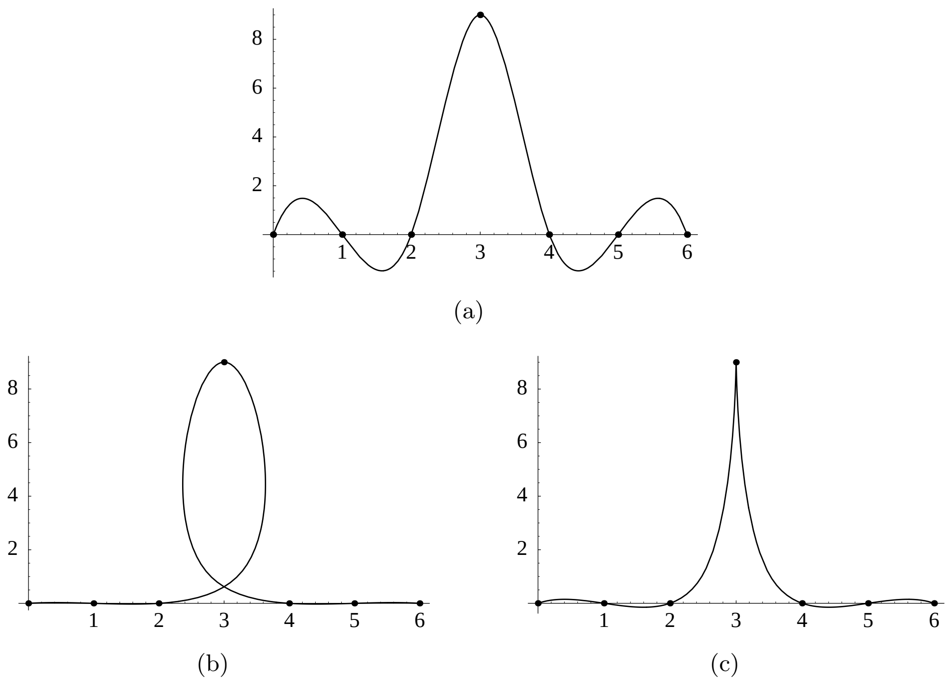

Figure 6.4. Parametric, cubic spline interpolation with uniform parametrisation (a), cord length parametrisation (b), and centripetal parametrisation (c).

1. **Uniform parametrisation** which amounts to $u_i = i$ for $i = 1, 2, \ldots, m$. This has the shortcomings discussed above.

2. **Cord length parametrisation** which is given by

$$u_1 = 0 \qquad \text{and} \qquad u_i = u_{i-1} + ||\boldsymbol{p}_i - \boldsymbol{p}_{i-1}|| \quad \text{for } i = 2, 3, \ldots, m.$$

   If the final approximation should happen to be the piecewise linear interpolant to the data, this method will correspond to parametrisation by arc length. This often causes problems near sharp corners in the data where it is usually wise to move slowly.

3. **Centripetal parametrisation** is given by

$$u_1 = 0 \qquad \text{and} \qquad u_i = u_{i-1} + ||\boldsymbol{p}_i - \boldsymbol{p}_{i-1}||^{1/2} \quad \text{for } i = 2, 3, \ldots, m.$$

   For this method, the difference $u_i - u_{i-1}$ will be smaller than when cord length parametrisation is used. But like the other two methods it does not take into consideration sharp corners in the data, and may therefore fail badly on difficult data.

There are many other methods described in the literature for determining good parameter values at the data points, but there is no known 'best' method. In fact, the problem of finding good parameterisations is an active research area.

Figures 6.4 (a)–(c) show examples of how the three methods of parametrisation described above perform on a difficult set of data.

6.2.5 General parametric spline approximation

In Chapter 5, we also defined other methods for spline approximation like cubic Hermite interpolation, general spline interpolation and least squares approximation by splines. All these and many other methods for functional spline approximation can be generalised very simply to parametric curves. If the data is given in the form of a parametric curve, the desired functional method can just be applied to each component function of the given curve. If the data is given as a set of discrete points $(\boldsymbol{p}_i)_{i=1}^m$, a parametrisation of the points must be determined using for example one of the methods in Section 6.2.4. Once this has been done, a functional method can be applied to each of the s data sets $(u_i, p_i^j)_{i,j=1,1}^{m,d}$. If we denote the functional approximation scheme by A and denote the data by $\boldsymbol{f}$, so that $\boldsymbol{f}_i = (u_i, \boldsymbol{p}_i)$ for $i = 1, \ldots, m$, the parametric spline approximation satisfies

$$A\boldsymbol{f} = (Af^1, \ldots, Af^s), \tag{6.4}$$

where f^j denotes the data set $(u_i, p_i^j)_{i=1}^m$ which we think of as $\big(u_i, f^j(u_i)\big)_{i=1}^m$. As we have seen several times now, the advantage of the relation (6.4) is that the parametric approximation can be determined by applying the corresponding functional approximation scheme to the s components, or, if we use a language that supports vector arithmetic, we simply call the routine for functional approximation with vector data. In Chapter 7, we shall see that the functional methods can be applied repeatedly in a similar way to compute tensor product spline approximations to surfaces.