Curve Fitting

R. J. Renka

Department of Computer Science & Engineering
University of North Texas

12/05/2016
There are three ways to represent a planar curve.

1. **Functional Form**: \( \{(x, y) \in \mathbb{R}^2 : y = f(x)\} \) for some function \( f : \mathbb{R} \rightarrow \mathbb{R} \).

2. **Parametric Form**: \( \{(x, y) \in \mathbb{R}^2 : x = f_1(t), y = f_2(t), t \in \mathbb{R}\} \) for a pair of functions \( f_1, f_2 : \mathbb{R} \rightarrow \mathbb{R} \).

3. **Implicit Form**: \( \{(x, y) \in \mathbb{R}^2 : f(x, y) = 0\} \) for a function \( f : \mathbb{R}^2 \rightarrow \mathbb{R} \).

The functional form is limited to one \( y \) value for each \( x \) value. Only the parametric form extends to space curves. All three forms extend to surfaces and hypersurfaces. The implicitly defined curves are referred to as *contours* or *level sets*.
There are three types of curve fitting problem.

1. **Approximation**: Given a function $g : [a, b] \rightarrow \mathbb{R}$, find a function $f$ in some function space $S$ such that $f$ approximates $g$ in some sense, such as minimizing $\|f - g\|$ for a suitable function space norm such as $\|f\|_{L_\infty[a,b]} \equiv \sup_{a \leq x \leq b} |f(x)|$.

2. **Data Fitting**: Given data points consisting of distinct abscissae $x_1 < x_2 \ldots < x_n$ with corresponding data values (ordinates) $\{y_i\}_{i=1}^n$ find a function $f \in S$ such that $\|f - y\|$ is minimized for some vector norm, where $f_i = f(x_i)$.

3. **CAGD**: Given control points $p_i = (x_i, y_i)$ or $p_i = (x_i, y_i, z_i)$ and basis functions $B_i(t), t \in [a, b]$, construct a parametric curve $C(t) = \sum_{i=0}^n B_i(t)p_i$. 

R. J. Renka

Curve Fitting
In all three cases the curve may be interpolatory: $f(x_i) = g(x_i)$ for discrete points $x_i \in [a, b]$, $f(x_i) = y_i$, or $C(t_i) = p_i$ for some parameter values $t_i$. In the case of approximation, the evaluation points for $g$ may be chosen to control the approximation error by making the maximum mesh width $h_i = x_{i+1} - x_i$ small. In the data fitting problem we cannot control the error. The data points may come from some underlying function $g$, but we cannot evaluate $g$ at arbitrary points. If there are significant errors in the data values, interpolation is not appropriate. A better method is a least squares fit:

$$\min_{f \in S} \sum_{i=1}^{n} w_i(y_i - f(x_i))^2, \quad w_i = \left(\frac{1}{\delta y_i}\right)^2$$

for standard deviation $\delta y_i$, and a linear space $S$ of dimension less than $n$. The dimension determines a tradeoff between smoothness and closeness to the data. A good choice for $S$ is polynomials unless the data suggests something else.
In the case of CAGD, the goal is a user-designed curve, perhaps a boundary curve for a user-designed surface, or the centerline of a ship hull. In the case of approximation or fitting data from some underlying function $g$, the purpose might be to

- Evaluate $g$ at points where it is expensive or impossible to obtain data,
- Save storage in a table,
- Find $g'$, $g''$, $\int_a^b g$, etc.
- Obtain a picture of the data by graphing \( y = g(x) \),
- Remove noise and find trends.
A polynomial of degree \( n - 1 \) (order \( n > 0 \)) is a function of the form
\[
p(x) = \sum_{j=1}^{n} c_j x^{j-1} = c_1 + c_2 x + \ldots + c_n x^{n-1}
\]
for \( x \in [a, b] \) or \( x \in \mathbb{R} \).

**Theorem:** Denote by \( P_{n-1}[a, b] \) the set of polynomials of degree less than or equal to \( n - 1 \). This is an \( n \)-dimensional linear space.

**proof:** We first show that the *monomials* \( \phi_j(x) = x^{j-1} \) are linearly independent, and hence form a basis. Suppose
\[
p(x) = \sum_{j=1}^{n} c_j \phi_j(x) = 0.
\]
Then \( p \) and all derivatives of \( p \) are identically zero (zero for all \( x \)). In particular, \( p(0) = c_1 = 0 \), \( p'(0) = c_2 = 0, \ldots, p^{(n-1)}(0) = (n-1)! c_n = 0 \), so that all coefficients are necessarily zero. Now suppose \( p = \sum_j c_j \phi_j \) and \( q = \sum_j d_j \phi_j \) are elements of \( P_{n-1}[a, b] \). Then \( \alpha p + \beta q = \sum_j e_j \phi_j \) for \( e_j = \alpha c_j + \beta d_j \), so that \( P_{n-1} \) is closed under linear combinations. \( \square \)
Theorem: Given $n$ distinct abscissae $x_1, x_2, \cdots, x_n$ with corresponding ordinates $\{y_i\}_{i=1}^n$, there is a unique polynomial $p \in P_{n-1}[x_1, x_n]$ that interpolates the data points: $p(x_i) = y_i$ for $1 \leq i \leq n$.

proof: $p(x_i) = \sum_{j=1}^{n} c_j x_i^{j-1} = y_i$ for all $i$ if and only if $A\mathbf{c} = \mathbf{y}$, where $A_{ij} = x_i^{j-1}$. Thus, there is a unique polynomial interpolant if and only if $A\mathbf{c} = \mathbf{y}$ has a unique solution; i.e., $A$ is nonsingular. Suppose $A\bar{\mathbf{c}} = \mathbf{0}$. Then $\bar{p}(x) = \sum_{j=1}^{n} \bar{c}_j x^{j-1}$ has $n$ distinct zeros $(x_1, x_2, \ldots, x_n)$, and is therefore identically zero by the Fundamental Theorem of Algebra. Hence, $\bar{\mathbf{c}} = \mathbf{0}$, and $A$ is nonsingular. □

Note that there are many good choices for basis functions other than the monomials. The polynomial interpolant is unique but has different coefficients for different basis functions.
Interpolation is a two-phase process. In the first phase we construct the interpolant by solving the linear system $Ac = y$, where $c$ and $y$ are the vectors of coefficients and data values, respectively, and $A$ is the order-$n$ matrix with components $A_{ij} = \phi_j(x_i)$ for polynomial basis functions $\phi_j$. In the case of monomial basis functions, $A$ is the Vandermonde matrix. It is nonsingular for distinct abscissae but tends to be ill-conditioned for large $n$. Solution by Gaussian elimination requires $n^3/3 + n^2 - n/3$ multiplies.

The second phase of interpolation involves evaluation of the interpolant and/or derivatives or integrals of the interpolant. The number of evaluation points is application-dependent. Evaluation of polynomials in the monomial or power form is efficient, but numerically unstable for large $n$. (Bernstein polynomial basis functions are optimal for stability.) Horner’s method requires $n - 1$ multiplies:

$$p(x) = c_1 + x(c_2 + x(c_3 + \ldots + x(c_{n-1} + c_n x)\ldots)).$$
A choice of basis that eliminates the cost of computing coefficients is the set of Lagrangian basis functions

\[ l_j(x) = \frac{\prod_{k \neq j} (x - x_k)}{\prod_{k \neq j} (x_j - x_k)} \Rightarrow l_j(x_i) = \delta_{ij}, \]

where \( \delta_{ij} \) is the Kronecker delta function. This makes \( A \) the identity matrix, so that the coefficients are the data ordinates. Evaluation, on the other hand, is relatively expensive, requiring \( 2n^2 - 3n \) multiplies and \( n \) divides for each evaluation point.

The Lagrangian basis functions have a property that the monomials lack and that is crucial for interactive curve design: they add to 1 at every point. To show this, let \( p(x) = (\sum_{j=1}^{n} l_j(x)) - 1 \). Then \( p \) is a polynomial of degree \( n - 1 \) with \( n \) zeros, hence identically zero.
Polynomial interpolation is a *global* data fitting method: the interpolant at every point depends on all of the data values, so that a perturbation of the data value at one end of the curve affects the other end of the curve. This is an undesirable effect unless \( n \) is small. Furthermore, a polynomial of degree \( n - 1 \) can have \( n - 3 \) inflection points, causing the curve to oscillate between the data points, especially near the ends. An extreme example of this occurs with interpolation of values of Runge’s function \( f(x) = 1/(1 + 25x^2) \) on a uniform grid in the interval \([-1, 1]\). Instead of the error approaching zero as the mesh width decreases, the error actually approaches \( \infty \). This is despite the Weierstrass Approximation Theorem which states that any continuous function can be uniformly approximated by a polynomial. The result is not true if the polynomial is constrained to interpolate a set of function values.
Splines

In order to localize the effect of perturbations in data points, we employ low-degree polynomials pieced together with some degree of continuity and smoothness (continuity of derivatives).

**Defn:** $C^k[a, b] = \{ f : [a, b] \to \mathbb{R} \text{ such that } f, f', \ldots, f^{(k)} \text{ are continuous} \}$. This is an infinite-dimensional linear space.

**Defn:** Give a sequence of distinct knots $x_1 < x_2 < \ldots < x_n$, the polynomial spline of degree $k$ associated with the knots is a polynomial of degree $k$ on each subinterval $[x_i, x_{i+1}]$ and an element of $C^{k-1}[x_1, x_n]$.

- $k = 1$: $C^0$ piecewise linear
- $k = 2$: $C^1$ piecewise quadratic
- $k = 3$: $C^2$ cubic spline

The cubic spline is the most popular choice for interpolation. It is global but the effect of perturbing a data value decreases rapidly with distance from the data point.
For $n$ knots, we have $n - 1$ subintervals and polynomial pieces, and $n - 2$ interior knots at which we must enforce continuity between the $C^\infty$ polynomial pieces. The number of degrees of freedom in a piecewise polynomial is the total number of coefficients or basis functions — $(k + 1)$ times the number of polynomial pieces. Each constraint, such as a specified value or a requirement for continuity at a point, reduces the number of degrees of freedom by 1.

- The number of degrees of freedom in the piecewise polynomial is $(k + 1)(n - 1)$.
- The number of degrees of freedom in the spline is $((k + 1)(n - 1)) - k(n - 2) = n + k - 1$.
- The number of degrees of freedom in the spline interpolant is $(n + k - 1) - n = k - 1$.
- The number of degrees of freedom in the piecewise polynomial interpolant is $(k + 1)(n - 1) - 2(n - 1) = kn - k - n + 1$.
- The number of degrees of freedom in the spline interpolant is $(kn - k - n + 1) - (k - 1)(n - 2) = k - 1$.
The $k - 1 = 2$ degrees of freedom in the cubic spline interpolant are eliminated by specifying end conditions. Choices include specified first or second derivative values, and the not-a-knot condition which specifies continuity of third derivative at $x_2$ and $x_{n-1}$. The reason for the term is that adjacent cubics with third-derivative continuity across the knot comprise a single cubic function. This follows from the fact that fourth and higher derivatives of a cubic function are identically zero. More generally, the definition of a spline enforces the maximum degree of continuity while preserving the piecewise nature of the function.
The first reference to mathematical splines appeared in a 1946 paper by I. J. Schoenberg. Mechanical splines were flexible strips of elastic material (thin beams) fixed at a set of knots for use as a drafting tool. The shape $y(x)$ naturally assumes minimum potential energy associated with bending:

$$E = \int \kappa^2 \, ds = \int \left( \frac{y''}{s'^3} \right)^2 \, s' \, dx = \int \frac{y''^2}{(1 + y'^2)^{5/2}} \, dx$$

for curvature $\kappa$. The minimizer of $E$ is referred to as the nonlinear spline, and was first studied by Euler. Given $n$ distinct abscissae $x_i$ with data values $y_i$, the solution to the problem of minimizing the linearized curvature $\int_{x_1}^{x_n} f''^2$ over functions in $H^{2,2}[x_1, x_n]$ that interpolate the data is the natural cubic spline with knots at the abscissae. With additional constraints such as specified endpoint first or second derivative values, the solution remains the cubic spline.
Hermite interpolation refers to interpolation of specified derivative values, along with data values. Suppose \( \{d_i\}_{i=1}^n \) are specified as required first derivative values at the abscissae \( x_1 < x_2 < \ldots < x_n \). A counting argument shows that a \( C^1 \) piecewise cubic function has just enough freedom to interpolate both the data values and knot first derivative values. The argument can be simplified to consideration of a single subinterval: there are four degrees of freedom in the cubic, and four constraints associated with the endpoint values and derivatives. The \( C^1 \) continuity follows from the interpolation conditions. This is a local method. Perturbing a data value or knot derivative value affects the interpolant only in the one or two subintervals that share the knot. The interpolant can be formulated in terms of basis functions with support on only two subintervals.
For $i = 1, \ldots, n - 1$, define $s_i(x) = x - x_i$ and $h_i = x_{i+1} - x_i > 0$ for distinct abscissae. Then for $x \in [x_i, x_{i+1}]$, the Hermite cubic interpolant is

$$f(x) = f_i(x) = H_0^i(x)y_i + H_1^i(x)d_i + H_2^i(x)d_{i+1} + H_3^i(x)y_{i+1},$$

where, omitting the subscripts and superscripts,

$$H_0(x) = \frac{h^3 - 3hs^2 + 2s^3}{h^3}, \quad H'_0(x) = \frac{-6hs + 6s^2}{h^3}, \quad H''_0(x) = \frac{-6h + 12s}{h^3},$$
$$H_1(x) = \frac{s^3 - 2hs^2 + h^2s}{h^2}, \quad H'_1(x) = \frac{3s^2 - 4hs + h^2}{h^2}, \quad H''_1(x) = \frac{6s - 4h}{h^2},$$
$$H_2(x) = \frac{s^3 - hs^2}{h^2}, \quad H'_2(x) = \frac{3s^2 - 2hs}{h^2}, \quad H''_2(x) = \frac{6s - 2h}{h^2},$$
$$H_3(x) = \frac{3hs^2 - 2s^3}{h^3}, \quad H'_3(x) = \frac{6hs - 6s^2}{h^3}, \quad H''_3(x) = \frac{6h - 12s}{h^3}.$$
Consider the following *shape properties* of a data set.

- **positivity** \( y_i > 0 \) for \( i = 1, \ldots, n \),
- **monotonicity** \( y_i < y_{i+1} \) for \( i = 1, \ldots, n - 1 \),
- **convexity** \( y_i - y_{i-1} < y_{i+1} - y_i \) for \( i = 2, \ldots, n - 1 \).

The properties may hold globally or locally (for a contiguous subset of the indices), analogous properties may be defined with the inequalities reversed, and the inequalities may be nonstrict. A fitting function \( f \) is said to preserve the shape of the data if it has the corresponding properties \( f(x) > 0 \) for all \( x \), \( f'(x) > 0 \) for all \( x \), and \( f''(x) > 0 \) for all \( x \). With a relaxed definition that allows for nonexistent derivatives at the knots, the piecewise linear interpolant of the data preserves the shape of the data. A smooth interpolant that preserves the shape of the data can also be constructed. Some applications require a shape-preserving fit. Probability density functions, for example, are monotonic.
The knot derivative values for the $C^1$ piecewise cubic Hermite interpolant $H$ need not be prespecified. Instead, they may be chosen so that $H$ preserves local monotonicity of the data. The method starts with the derivative at $x_i$ of the quadratic interpolant of the three points $(x_{i-1}, y_{i-1}), (x_i, y_i)$, and $(x_{i+1}, y_{i+1})$:

$$
\delta_i \equiv (\Delta_i h_{i-1} + \Delta_{i-1} h_i)/(h_{i-1} + h_i), \quad \text{where } \Delta_i = (y_{i+1} - y_i)/h_i \text{ for } i = 2, \ldots, n - 1
$$

with similar formulas for the endpoints. These approximations are then modified as necessary to satisfy the constraints. For some data sets this method produces a better fit than a cubic spline which can have extraneous inflection points.

A more powerful method involves the use of a tension factor in each subinterval. As tension is increased, the curve segment varies from cubic (zero tension) to linear (infinite tension). The tension factors can be automatically chosen just large enough to satisfy the constraints.
Cubic Spline Interpolation

A convenient method for computing a cubic spline interpolant $f$ is to start with the Hermite cubic form, and treat the knot derivative values $d_i$ as a set of $n$ free parameters. We obtain an order-$n$ linear system by specifying continuity of second derivative $f''$ at the $n-2$ interior knots, and adding a pair of linear end conditions. This provides the basis for a flexible software package in which knot derivatives may be user-specified, chosen by a local method to preserve monotonicity, or chosen by solving the system to produce a cubic spline. The equations are

$$f''(x_i) = -6\Delta_{i-1} + 2d_{i-1} + 4d_i = \frac{6\Delta_i - 4d_i - 2d_{i+1}}{h_i} = f''(x_i) \Rightarrow$$

$$\frac{2}{h_{i-1}}d_{i-1} + \frac{4}{h_i}d_i + \frac{4}{h_i}d_i + \frac{2}{h_{i+1}}d_{i+1} = \frac{6}{h_i}\Delta_i + \frac{6}{h_{i-1}}\Delta_{i-1} \Rightarrow$$

$$\frac{1}{h_{i-1}}d_{i-1} + \left(\frac{2}{h_{i-1}} + \frac{2}{h_i}\right)d_i + \frac{1}{h_i}d_{i+1} = 3\left(\frac{\Delta_{i-1}}{h_{i-1}} + \frac{\Delta_i}{h_i}\right)$$

for $i = 2, \ldots, n-1$, where $\Delta_i = (y_{i+1} - y_i)/h_i$. 
Cubic Spline Interpolation continued

For end conditions $f''(x_1) = y_1''$ and $f''(x_n) = y_n''$, we have

\[
\frac{1}{h_1} (6\Delta_1 - 4d_1 - 2d_2) = y_1'' \Rightarrow \frac{2}{h_1} d_1 + \frac{1}{h_1} d_2 = 3 \frac{\Delta_1}{h_1} - \frac{y_1''}{2},
\]

\[
\frac{1}{h_{n-1}} (-6\Delta_{n-1} + 2d_{n-1} + 4d_n) = y_n'' \Rightarrow \frac{1}{h_{n-1}} d_{n-1} + \frac{2}{h_{n-1}} d_n = 3 \frac{\Delta_{n-1}}{h_{n-1}} + \frac{y_n''}{2}.
\]

With $n = 5$ the system is

\[
\begin{bmatrix}
\frac{2}{h_1} & \frac{1}{h_1} & 0 & 0 & 0 \\
\frac{1}{h_1} & \frac{2}{h_1} + \frac{2}{h_2} & \frac{1}{h_2} & 0 & 0 \\
0 & \frac{1}{h_2} & \frac{2}{h_2} + \frac{2}{h_3} & \frac{1}{h_3} & 0 \\
0 & 0 & \frac{1}{h_3} & \frac{2}{h_3} + \frac{2}{h_4} & \frac{1}{h_4} \\
0 & 0 & 0 & \frac{1}{h_4} & \frac{2}{h_4}
\end{bmatrix}
\begin{bmatrix}
\begin{bmatrix}
d_1 \\
d_2 \\
d_3 \\
d_4 \\
d_5
\end{bmatrix}
\end{bmatrix}
= 3
\begin{bmatrix}
\frac{\Delta_1}{h_1} - \frac{y_1''}{6} \\
\frac{\Delta_1}{h_1} + \frac{\Delta_2}{h_2} \\
\frac{\Delta_2}{h_2} + \frac{\Delta_3}{h_3} \\
\frac{\Delta_3}{h_3} + \frac{\Delta_4}{h_4} \\
\frac{\Delta_4}{h_4} + \frac{y_n''}{6}
\end{bmatrix}.
\]
Cubic Spline Interpolation continued

For end conditions \( f'(x_1) = y'_1 \) and \( f'(x_n) = y'_n \), we have \( d_1 = y'_1 \) and \( d_n = y'_n \). There is no way to scale these equations that would retain symmetry in the linear system. The right approach is to reduce the system to order \( n - 2 \) by moving the given values of \( d_1 \) and \( d_n \) to the right hand side. For \( n = 5 \) we have

\[
\begin{bmatrix}
\frac{2}{h_1} + \frac{2}{h_2} & \frac{1}{h_2} & 0 \\
\frac{1}{h_2} & \frac{2}{h_2} + \frac{2}{h_3} & \frac{1}{h_3} \\
0 & \frac{1}{h_3} & \frac{2}{h_3} + \frac{2}{h_4}
\end{bmatrix}
\begin{bmatrix}
d_2 \\
d_3 \\
d_4
\end{bmatrix}
=\begin{bmatrix}
3\left(\frac{\Delta_1}{h_1} + \frac{\Delta_2}{h_2}\right) - \frac{y'_1}{h_1} \\
3\left(\frac{\Delta_2}{h_2} + \frac{\Delta_3}{h_3}\right) \\
3\left(\frac{\Delta_3}{h_3} + \frac{\Delta_4}{h_4}\right) - \frac{y'_n}{h_4}
\end{bmatrix}.
\]

In either case, the linear system is symmetric, tridiagonal, diagonally dominant, and positive definite. Hence, no pivoting is required for stability, and the operation count is \( O(n) \). The condition number depends on the distribution of abscissae. Large variations in \( h_i \) could result in an ill-conditioned system.
Periodic end conditions are required to obtain a closed curve in the parametric case. We extend \( f \) with the additional cubic function \( f_n \) on \( [x_n, x_{n+1}] \), where \( h_n = x_{n+1} - x_n > 0 \), \( f(x_{n+1}) = y_1 \), \( f'(x_{n+1}) = d_1 \), and \( f''(x_{n+1}) = f''(x_1) \). Thus

\[
\frac{1}{h_n} d_n + \left( \frac{2}{h_n} + \frac{2}{h_1} \right) d_1 + \frac{1}{h_1} d_2 = 3 \left( \frac{\Delta_n}{h_n} + \frac{\Delta_1}{h_1} \right) \quad \text{and} \quad \frac{1}{h_{n-1}} d_{n-1} + \left( \frac{2}{h_{n-1}} + \frac{2}{h_n} \right) d_n + \frac{1}{h_n} d_1 = 3 \left( \frac{\Delta_{n-1}}{h_{n-1}} + \frac{\Delta_n}{h_n} \right),
\]

where \( \Delta_n = (y_1 - y_n)/h_n \). The matrix retains symmetry, and is positive definite, but the work space requirement increases from \( n - 1 \) to \( 2(n - 1) \), the extra array being required to store the fill-in in the last row and last column.
For \( n = 5 \),

\[
\begin{bmatrix}
\frac{2}{h_5} + \frac{2}{h_1} & \frac{1}{h_1} & 0 & 0 & \frac{1}{h_5} \\
\frac{1}{h_1} & \frac{2}{h_1} + \frac{2}{h_2} & \frac{1}{h_2} & 0 & 0 \\
0 & \frac{1}{h_2} & \frac{2}{h_2} + \frac{2}{h_3} & \frac{1}{h_3} & 0 \\
0 & 0 & \frac{1}{h_3} & \frac{2}{h_3} + \frac{2}{h_4} & \frac{1}{h_4} \\
\frac{1}{h_5} & 0 & 0 & \frac{1}{h_4} & \frac{2}{h_4} + \frac{2}{h_5}
\end{bmatrix}
\begin{bmatrix}
d_1 \\
d_2 \\
d_3 \\
d_4 \\
d_5
\end{bmatrix} = 3
\begin{bmatrix}
\frac{\Delta_5}{h_5} + \frac{\Delta_1}{h_1} \\
\frac{\Delta_1}{h_1} + \frac{\Delta_2}{h_2} \\
\frac{\Delta_2}{h_2} + \frac{\Delta_3}{h_3} \\
\frac{\Delta_3}{h_3} + \frac{\Delta_4}{h_4} \\
\frac{\Delta_4}{h_4} + \frac{\Delta_5}{h_5}
\end{bmatrix}.
\]
Given a point \( x \) at which \( f \) or a derivative of \( f \) is to be evaluated, the first step is to find the index \( i \) of the interval \([x_i, x_{i+1}]\) containing \( x \). This can be done by a binary search. In some applications, such as displaying the graph of \( f \), there are many evaluation points, and they may be ordered. In that case each search requires only constant time.

The evaluation time can be reduced by storing two additional arrays containing vectors \( c \) and \( e \) of scaled second and third derivative values at the knots. For \( x \in [x_i, x_{i+1}] \)

\[
f_i(x) = y_i + d_i(x - x_i) + c_i(x - x_i)^2 + e_i(x - x_i)^3,
\]

where

- \( f_i(x_i) = y_i \),
- \( f'_i(x_i) = d_i \),
- \( (1/2)f''_i(x_i) = c_i = (1/h_i)(3\Delta_i - 2d_i - d_{i+1}) \),
- \( (1/6)f'''_i(x_i) = e_i = (1/h_i^2)(-2\Delta_i + d_i + d_{i+1}) \).
Suppose we wish to evaluate a cubic function

\[ f(t) = at^3 + bt^2 + ct + d \] at \( n + 1 \) points uniformly distributed in \([\alpha, \beta]\). Let \( \delta = (\beta - \alpha)/n \), \( t_i = \alpha + i\delta \), and \( f_i = f(t_i) \) for \( i = 0, \ldots, n \). Then, for \( i = 1, \ldots, n \),

\[ f_i = f_{i-1} + \Delta f_{i-1}, \quad \text{where} \]

\[
\begin{align*}
\Delta f_i &= f_{i+1} - f_i = f(t_i + \delta) - f(t_i) \\
&= a(t_i + \delta)^3 + b(t_i + \delta)^2 + c(t_i + \delta) - at_i^3 - bt_i^2 - ct_i \\
&= a(3t_i^2\delta + 3t_i\delta^2 + \delta^3) + b(2\delta t_i + \delta^2) + c\delta \\
&= 3a\delta t_i^2 + (3a\delta^2 + 2b\delta)t_i + a\delta^3 + b\delta^2 + c\delta.
\end{align*}
\]

\[ \Delta f_i = \Delta f_{i-1} + \Delta^2 f_{i-1}, \quad \text{where} \]

\[
\begin{align*}
\Delta^2 f_i &= \Delta f_{i+1} - \Delta f_i \\
&= 3a\delta(t_{i+1}^2 - t_i^2) + (3a\delta^2 + 2b\delta)(t_{i+1} - t_i) \\
&= 3a\delta^2(2t_i + \delta) + 3a\delta^3 + 2b\delta^2 = 6a\delta^2 t_i + 6a\delta^3 + 2b\delta^2.
\end{align*}
\]
Finally, we have

\[ \Delta^2 f_i = \Delta^2 f_{i-1} + \Delta^3 f, \]  

where

\[ \Delta^3 f = \Delta^2 f_i - \Delta^2 f_{i-1} = 6a\delta^3. \]

After initialization, the algorithm requires only 3n adds.

```
for i = 1 to n
    \[ f_i = f_{i-1} + \Delta f \]
    \[ \Delta f = \Delta f + \Delta^2 f \]
    \[ \Delta^2 f = \Delta^2 f + \Delta^3 f \]
endfor
```
Initialization is as follows.

\[
\begin{align*}
\delta &= \frac{(\beta - \alpha)}{n} \\
f_0 &= f(\alpha) = [(a\alpha + b)\alpha + c]\alpha + d \\
\Delta f &= \Delta f_0 = 3a\delta\alpha^2 + (3a\delta^2 + 2b\delta)\alpha + a\delta^3 + b\delta^2 + c\delta \\
    &= [a\delta^2 + (3a\alpha + b)\delta + (3a\alpha^2 + 2b\alpha + c)]\delta \\
    &= \delta([a(3\alpha + \delta) + b]\delta + \alpha(3a\alpha + 2b) + c) \\
\Delta^2 f &= \Delta^2 f_0 = 2\delta^2(3a\delta + b + 3a\alpha) \\
    &= 2\delta^2[3a(\alpha + \delta) + b] \\
\Delta^3 f &= 6a\delta^3
\end{align*}
\]
A parametric space curve is the range (image) of a function \( \mathbf{C} : [a, b] \to \mathbb{R}^3 \). Note that the function represents more than the curve; it associates points on the curve with parameter values. The function may be thought of as defining the trajectory of a particle moving through space, in which case the parameter is time. A parametric curve can always be reparameterized. If, for example, \( \bar{t} = a + b - t \), then the function \( \bar{\mathbf{C}} \) defined by \( \bar{\mathbf{C}}(t) \equiv \mathbf{C}(\bar{t}) \) has the same range as \( \mathbf{C} \) but the path is in the reverse direction. We generally require that \( \mathbf{C} \) be differentiable, and assume that the parameterization is \emph{regular}: \( \mathbf{C}'(t) \neq 0 \) for all \( t \in [a, b] \), so that the curve has a tangent direction at every point, and the particle cannot reverse direction.

When the emphasis is on the curve rather than the trajectory, the range of parameter \( t \) is irrelevant, and it is convenient to take \([a, b] = [0, 1]\).
Given control points $p_i = (x_i, y_i)$, $i = 0, \ldots, n$, suppose that we wish to construct an interpolatory parametric curve $C$ such that $C(t_i) = p_i$ for all $i$ and some choice of discrete parameter values $t_i$ (a knot sequence in the case of a spline fit). Some reasonable choices are

**Uniform**  \quad $t_i = i$

**Chord-length**  \quad $t_0 = 0$ and $t_i = \sum_{j=1}^{i} \|p_j - p_{j-1}\|$ for $i = 1, \ldots, n$

**Centripetal**  \quad $t_0 = 0$ and $t_i = \sum_{j=1}^{i} \sqrt{\|p_j - p_{j-1}\|}$ for $i = 1, \ldots, n$

Note that $t_{i-1} < t_i$ is required in order to preserve the order in which the control points are traversed. Note also that, in all three cases, the parameter values can be normalized to the interval $[0, 1]$ by simply scaling $t_i$ by $1/t_n$ for all $i$.
Uniformly distributed values are not appropriate when the distances between control points vary widely, resulting in widely varying speed, large acceleration, overshoot and undershoot. The cumulative chord-length method results in constant average speed between control points, but does not account for changes in velocity direction. The centripetal parameterization is derived from physical heuristics with the goal of smoothing out variations in the centripetal force acting on a moving particle. There are more elaborate methods, but centripetal is a good choice. Any method can be defeated by a suitably chosen set of control points. Loops are unavoidable for some data.
Geometric continuity of order \( k \), denoted \( G^k \), refers to continuity with respect to arc length, and differs from parametric continuity (component functions in \( C^k[a, b] \)): \( G^0 \) and \( C^0 \) are equivalent, but a curve may be \( G_1 \) and not \( C^1 \), or \( C^1 \) and not \( G^1 \), and a curve may be \( G^2 \) and not \( C^2 \), or \( C^2 \) and not \( G^2 \). Note, however, that a \( G^2 \) curve (with continuous tangent and curvature) can be reparameterized with arc length parameterization, making it \( C^2 \).

**Example:** The following is \( G^1 \) but not \( C^1 \):

\[
\begin{align*}
x(t) &= \begin{cases} 
12t & \text{if } 0 \leq t \leq 1 \\
4t + 8 & \text{if } 1 \leq t \leq 2 
\end{cases} \\
y(t) &= \begin{cases} 
9t & \text{if } 0 \leq t \leq 1 \\
3t + 6 & \text{if } 1 \leq t \leq 2 
\end{cases}
\end{align*}
\]

\[
\begin{align*}
x'(t) &= \begin{cases} 
12 & \text{if } 0 \leq t \leq 1 \\
4 & \text{if } 1 \leq t \leq 2 
\end{cases} \\
y'(t) &= \begin{cases} 
9 & \text{if } 0 \leq t \leq 1 \\
3 & \text{if } 1 \leq t \leq 2 
\end{cases}
\end{align*}
\]

The unit tangent is constant — \( T(t) = (.8, .6) \) for all \( t \), but the speed drops from 15 to 5 at \( t = 1 \).
If speed decreases continuously to zero, and then starts increasing in a different direction, we have $C^1$ parametric continuity but undefined tangent direction where velocity is zero (a non-regular parameterization). **Example:** The following is $C^1$ but not $G^1$:

\[
\begin{align*}
  x(t) &= \begin{cases} 
    2t - t^2 & \text{if } 0 \leq t \leq 1 \\
    t^2 - 2t + 2 & \text{if } 1 \leq t \leq 2
  \end{cases} \\
  y(t) &= \begin{cases} 
    2t - t^2 & \text{if } 0 \leq t \leq 1 \\
    2t - t^2 & \text{if } 1 \leq t \leq 2
  \end{cases}
\end{align*}
\]

\[
\begin{align*}
  x'(t) &= \begin{cases} 
    2 - 2t & \text{if } 0 \leq t \leq 1 \\
    2t - 2 & \text{if } 1 \leq t \leq 2
  \end{cases} \\
  y'(t) &= \begin{cases} 
    2 - 2t & \text{if } 0 \leq t \leq 1 \\
    2 - 2t & \text{if } 1 \leq t \leq 2
  \end{cases}
\end{align*}
\]

The values are $C(0) = (0, 0)$, $C(1) = (1, 1)$, and $C(2) = (2, 0)$, and the velocity vectors are $C'(0) = (2, 2)$, $C'(1) = (0, 0)$, and $C'(2) = (2, -2)$. The unit tangent vector is piecewise constant with values $(1/\sqrt{2})(1, 1)$ in $[0, 1)$ and $(1/\sqrt{2})(1, -1)$ in $(1, 2]$. 
We shift emphasis now from data fitting to curve design. Rather than fitting a curve to measured or computed data values, the curve is fit to a sequence of control points which an interactive user manipulates with the goal of producing a smooth curve appropriate to some application such as ship hull design. We consider both planar curves and space curves.

The standard format for a space curve is

$$C(u) = \sum_{i=0}^{n} B_i(u) p_i,$$

where $B_i$ is a basis function, and $p_i = (x_i, y_i, z_i)$ is a control point for $i = 0, \ldots, n$. We could interpret the components of $C$ as linear combinations of the basis functions, but it is more instructive to think of the curve as a weighted sum of the control points, where the weights are basis functions. (For each parameter value $u$, the basis function values $B_i(u)$ are weights associated with the point $C(u)$ on the curve.)
The degree-\(n\) Bernstein polynomials are defined by

\[
B_i(u) = \binom{n}{i} u^i (1-u)^{n-i}, \quad u \in [0,1] \quad \text{for} \quad \binom{n}{i} = \frac{n!}{i!(n-i)!}
\]

and \(i = 0, \ldots, n\), where \(0! = 1\) and \(u^0 = 1\) for \(u = 0\). Expressions for the polynomials on a general interval \([a, b]\) can be obtained by a change of variable \(t = a + (b-a)u\). For \(n = 3\) we have

\[
B_0(u) = (1 - u)^3, \quad B'_0(u) = -3(1 - u)^2, \quad B''_0(u) = 6(1 - u)
\]
\[
B_1(u) = 3u(1 - u)^2, \quad B'_1(u) = 3(1 - 3u)(1 - u), \quad B''_1(u) = -6(2 - 3u)
\]
\[
B_2(u) = 3u^2(1 - u), \quad B'_2(u) = 3u(2 - 3u), \quad B''_2(u) = 6(1 - 3u)
\]
\[
B_3(u) = u^3, \quad B'_3(u) = 3u^2, \quad B''_3(u) = 6u
\]

The midpoint \(u = (1/2)\) values are \(B_0 = B_3 = 1/8\) and \(B_1 = B_2 = 3/8\). Also, \(B_1\) has an inflection point at \(u = 2/3\), and \(B''_2 = 0\) at \(u = 1/3\).
Bernstein Polynomial Properties

1. $B_i \in P_n[0, 1]$ for $i = 0, \ldots, n$.

2. $B_0(0) = 1$, $B_i(0) = 0$ for $i \neq 0$, $B_n(1) = 1$, and $B_i(1) = 0$ for $i \neq n$. $B'_0(0) = -n$, $B'_1(0) = n$, $B'_i(0) = 0$ for $i > 1$, etc.

3. $\{B_i(u)\}_{i=0}^n$ is a basis for $P_n[0, 1]$. To prove the functions are linearly independent, suppose $p(u) = \sum_{i=0}^n \alpha_i B_i(u) = 0$. Then $p(0) = \alpha_0 = 0$ and $p(1) = \alpha_n = 0$. Using those results, we then get $p'(0) = \alpha_1 = 0$ and $p'(1) = \alpha_{n-1} = 0$. Proceeding by finite induction, we get $\alpha_i = 0$ for all $i$.

4. $B_i(u) \geq 0$ for all $u \in [0, 1]$.

5. Partition of unity: for every point $u$,

$$\sum_{i=0}^n B_i(u) = \sum_{i=0}^n \binom{n}{i} u^i (1-u)^{n-i} = (u + 1 - u)^n = 1.$$  

6. $B_i$ is maximized at $u = i/n$, and $B_i(i/n) > B_j(i/n)$ for $j \neq i$.

7. Reflection symmetry about $u = 1/2$:

$$B_{n-i}(1-u) = \binom{n}{n-i} (1-u)^{n-i} u^i = B_i(u).$$

8. $\text{spt}(B_i) = [0, 1]$. 

R. J. Renka

Curve Fitting
Bernstein Polynomial Evaluation

The following theorem provides a stable evaluation algorithm using recursion on the polynomial degree, now included as a superscript.

**Theorem** \( B_i^n(u) = (1 - u)B_{i-1}^{n-1}(u) + uB_i^{n-1}(u), \) \((i = 0, \ldots, n),\)

where \( B_0^0(u) = 1, \) and \( B_i^k(u) = 0 \) for \( i < 0 \) and \( i > k.\)

**proof**: First note that

\[
\binom{n-1}{i} + \binom{n-1}{i-1} = \frac{(n-1)!}{i!(n-1-i)!} + \frac{(n-1)!}{(i-1)!(n-i)!} = \frac{(n-1)!(n-i+i)}{i!(n-i)!} = \frac{n!}{i!(n-i)!} = \binom{n}{i}.
\]

Hence

\[
B_i^n(u) = \left( \binom{n-1}{i} \right) u^i (1-u)^{n-i} + \left( \binom{n-1}{i-1} \right) u^i (1-u)^{n-i} = (1-u)B_{i-1}^{n-1}(u) + uB_i^{n-1}(u). \quad \square
\]
The theorem implies an evaluation algorithm based on repeated computation of convex combinations. The number of operations required to compute the \( n + 1 \) basis function values at a single point \( u \) is \( n^2 + n - 2 \) multiplies and \( n(n - 1)/2 + 1 \) adds. This is less efficient than the \( n - 1 \) multiplies required by Horner’s method for monomial basis functions. The Bernstein polynomials, however, have an advantage in terms of numerical stability.

Given a set of coefficients defining a polynomial in terms of some basis functions, the problem of computing a value, derivative, or zero of the polynomial is ill-conditioned for monomial basis functions, but not for Bernstein basis functions. More precisely, a small perturbation in the coefficients has much more effect on the monomial form than on the Bézier form. The monomial form is especially problematic for high polynomial degree.
Bézier Curve Properties

Given control points \( p_i \in \mathbb{R}^d \) for \( d = 2 \) or \( d = 3 \), and \( i = 0, \ldots, n \), the Bézier curve associated with the control points is the range of

\[
C(u) = \sum_{i=0}^{n} B_i(u) p_i,
\]

where \( \{B_i : [0, 1] \rightarrow \mathbb{R}\}_{i=0}^{n} \) are the Bernstein polynomial basis functions. The curve properties follow from the properties of the basis functions (cited in square brackets).

1. Only the endpoints are interpolated: \( C(0) = p_0 \) and \( C(1) = p_n \) [2]. Also, \( C'(0) = n(p_1 - p_0) \), and \( C'(1) = n(p_n - p_{n-1}) \) so that the curve is tangent to the control polygon at the endpoints [2].

2. \( C \in (C^\infty[0, 1])^3 \) [1].

3. The curve is globally controlled by \( \{p_i\}_{i=0}^{n} \) [8], but we have pseudo-local control in the sense that the effect of moving \( p_i \) decreases with distance from \( C(i/n) \) [6].
C is coordinate-free. Let $T$ be an affine transformation on $\mathbb{R}^d$ defined by $T(p) = Ap + b$, $A$ linear. Then

$$T(C(u)) = T \left( \sum_{i=0}^{n} B_i(u)p_i \right) = A \left( \sum_{i=0}^{n} B_i(u)p_i \right) + b$$

$$= \left( \sum_{i=0}^{n} B_i(u)Ap_i \right) + b$$

$$= \sum_{i=0}^{n} B_i(u)Ap_i + b \sum_{i=0}^{n} B_i(u) \quad [5]$$

$$= \sum_{i=0}^{n} B_i(u)T(p_i),$$

so that the transformed curve is the Bézier curve associated with the transformed control points.
5. **C** has the *convex hull property*. Since the weights are nonnegative and add to 1 ([4] and [5]), \( C(u) \) is a convex combination of the control points for every \( u \in [0, 1] \); i.e., the curve is contained in the convex hull of the control points.

6. **C** has the *variation diminishing property*. The number of intersections of the curve with an arbitrary plane is less than or equal to the number of intersections of the control polygon with the plane. For example, a planar Bézier curve could be convex (have at most two intersections with any plane other than its containing plane) even though its control polygon is not convex, and a convex control polygon implies a convex curve. In the nonparametric case (functional form), the Bézier curve preserves monotonicity of the control polygon.
Degree Elevation Property

Since

\[(1 - u)B_i^n(u) = \frac{n + 1 - i}{n + 1} B_i^{n+1}(u)\]

and

\[uB_i^n(u) = \frac{i + 1}{n + 1} B_i^{n+1}(u),\]

\[C(u) = (1 - u)C(u) + uC(u)\]

\[= \sum_{i=0}^{n} \frac{n + 1 - i}{n + 1} B_i^{n+1}(u) p_i + \sum_{i=0}^{n} \frac{i + 1}{n + 1} B_i^{n+1}(u) p_i\]

\[= \sum_{i=0}^{n+1} \frac{n + 1 - i}{n + 1} B_i^{n+1}(u) p_i + \sum_{j=0}^{n+1} \frac{j}{n + 1} B_j^{n+1}(u) p_{j-1}\]

\[= \sum_{i=0}^{n+1} B_i^{n+1}(u) \hat{p}_i \quad \text{for} \quad \hat{p}_i = \frac{n + 1 - i}{n + 1} p_i + \frac{i}{n + 1} p_{i-1}.\]
Hand-drawing rule for cubics:

\[ C(1/2) = (1/8)(p_0 + 3p_1 + 3p_2 + p_3), \quad C'(1/2) = (3/4)(-p_0 - p_1 + p_2 + p_3). \]

Define

\[ M_1 = \frac{p_0 + p_1}{2}, \quad M_2 = \frac{p_1 + p_2}{2}, \quad M_3 = \frac{p_2 + p_3}{2}, \]
\[ M_4 = \frac{M_1 + M_2}{2} = \frac{p_0 + 2p_1 + p_2}{4}, \]
\[ M_5 = \frac{M_2 + M_3}{2} = \frac{p_1 + 2p_2 + p_3}{4}, \]
\[ M = \frac{M_4 + M_5}{2} = C \left( \frac{1}{2} \right), \quad M_5 - M_4 = \frac{1}{3} C' \left( \frac{1}{2} \right). \]

The curve then goes through \( M \) with tangent direction \( M_5 - M_4 \). Note the variation diminishing property.
Piecewise Bézier Curves

In the above construction, the Bézier curve associated with control points \( p_0, M_1, M_4, \) and \( M \) is \( \{ C(u) : 0 \leq u \leq (1/2) \} \), and the Bézier curve associated with \( M, M_5, M_3, \) and \( p_3 \) is \( \{ C(u) : (1/2) \leq u \leq 1 \} \). If \( M_1 \) or \( M_3 \) is altered, we then have a \( G^1 \) piecewise cubic Bézier curve. It remains \( G^1 \) as long as \( M_4, M, \) and \( M_5 \) are collinear. More generally, we obtain a locally controlled \( G^1 \) curve by piecing together low-degree (2 to 4) Bézier curves in such a way that tangents vary continuously. The pieces need not all have the same degree.

For piecewise cubics, we can take \( p_0, p_3, p_6, \ldots \) to be interpolated control points, and take \( p_1, p_2, p_4, p_5, p_7, \ldots \) to be auxiliary control points which a user manipulates to control the shape.

The theory was started in the late 1950’s by Pierre Bézier at Renault and Paul de Casteljau at Citroën working independently on design of auto bodies.
The hand-drawing rule is a special case (associated with $u = .5$) of a more general property of Bézier curves. For $u \in [0, 1]$, define $b_i^0(u) = p_i$ and

$$b_i^r(u) = (1 - u)b_i^{r-1}(u) + ub_{i+1}^{r-1}(u)$$

for $r = 1, \ldots, n$, $i = 0, \ldots, n - r$.

Then $b_0^n(u) = C(u)$.

In the case of cubics ($n=3$), we have the following tableau of control points.

$$
\begin{array}{cccc}
  b_0^0 & & & \\
  b_0^1 & b_0^1 & & \\
  b_1^1 & b_1^1 & b_2^2 & \\
  b_2^1 & b_2^1 & b_2^2 & b_3^3 \\
\end{array}
$$

The curve is subdivided at $u$: $\{C(t) : t \in [0, u]\}$ has control polygon $b_0^0, b_1^1, b_2^2, b_3^3$, while $\{C(t) : t \in [u, 1]\}$ has control polygon $b_3^3, b_2^2, b_1^1, b_0^0$. 
The Bernstein polynomials can be converted to monomial form for fast evaluation by Horner’s method (at a price in loss of stability). For the cubics

\[
B_0(t) = (1 - t)^3 = -t^3 + 3t^2 - 3t + 1 \\
B_1(t) = 3t(1 - t)^2 = 3t^3 - 6t^2 + 3t \\
B_2(t) = 3t^2(1 - t) = -3t^3 + 3t^2 \\
B_3(t) = t^3
\]

Hence

\[
C(t)^T = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_0^T \\ p_1^T \\ p_2^T \\ p_3^T \end{bmatrix}.
\]

An alternative, analogous to Horner’s method, for \( s = 1 - t \):

\[
C(t) = [(sp_0 + 3tp_1)s + 3t^2p_2]s + t^3p_3.
\]
Given knots $u_0 < u_1 < \ldots < u_n$, control points $p_i$ and derivative vectors $d_i$ in $\mathbb{R}^2$ or $\mathbb{R}^3$ for $i = 0, \ldots, n$, define basis functions as the piecewise cubic polynomials $\phi_i$ and $\psi_i$ on $[u_0, u_n]$ such that

$$\phi_i(u_j) = \delta_{ij}, \quad \phi'_i(u_j) = 0,$$

$$\psi_i(u_j) = 0, \quad \psi'_i(u_j) = \delta_{ij},$$

for $i, j = 0, \ldots, n$. Then the Hermite cubic polynomial interpolant is

$$C(u) = \sum_{i=0}^{n} \phi_i(u)p_i + \sum_{i=0}^{n} \psi_i(u)d_i.$$ 

Note that $g(u) = \sum_{i=0}^{n} \phi_i(u) - 1$ is a piecewise cubic with double zeros at the knots, and is therefore identically zero, so that $\sum_{i=0}^{n} \phi_i(u) = 1$ for all $u$. 

R. J. Renka Curve Fitting
If $T\mathbf{p} = A\mathbf{p} + \mathbf{b}$, where $A$ is linear, then

$$T(C(u)) = A \left( \sum_{i=0}^{n} \phi_i(u) \mathbf{p}_i \right) + \mathbf{b} + A \left( \sum_{i=0}^{n} \psi_i(u) \mathbf{d}_i \right)$$

$$= \sum_{i=0}^{n} \phi_i(u) T(\mathbf{p}_i) + \sum_{i=0}^{n} \psi(u) (A \mathbf{d}_i);$$

i.e., we transform the curve by applying $T$ to the control points and $A$ to the derivative vectors.
The Hermite cubic basis functions (restricted to their support) are

\[ \phi_i(u) = \begin{cases} H_{3}^{i-1} & \text{for } u \in [u_{i-1}, u_i] \\ H_{0}^i & \text{for } u \in [u_i, u_{i+1}] \end{cases}, \quad \psi_i(u) = \begin{cases} H_{2}^{i-1} & \text{for } u \in [u_{i-1}, u_i] \\ H_{1}^i & \text{for } u \in [u_i, u_{i+1}] \end{cases}, \]

where, for \( u \in [u_i, u_{i+1}] \),

\[ C(u) = H_{0}^i(u)p_i + H_{1}^i(u)d_i + H_{2}^i(u)d_{i+1} + H_{3}^i(u)p_{i+1}. \]

Let \( t = (u - u_i)/(u_{i+1} - u_i) \in [0, 1] \), and define

\[ \bar{C}(t) = C(u) = \bar{H}_{0}(t)\bar{C}(0) + \bar{H}_{1}(t)\bar{C}'(0) + \bar{H}_{2}(t)\bar{C}'(1) + \bar{H}_{3}(t)\bar{C}(1). \]

Expressions for \( \bar{H}_{0}, \ldots, \bar{H}_{3} \) are obtained by observing that \( \bar{C} \) is the cubic Bézier curve defined by control points \( \bar{p}_0, \bar{p}_1, \bar{p}_2, \) and \( \bar{p}_3 \), where \( \bar{C}(0) = \bar{p}_0, \bar{C}'(0) = 3(\bar{p}_1 - \bar{p}_0), \bar{C}'(1) = 3(\bar{p}_3 - \bar{p}_2), \) and \( \bar{C}(1) = \bar{p}_3. \)
Hermite Cubic Parametric Curves continued

We thus have

\[
\bar{C}(t) = \bar{H}_0(t)\bar{p}_0 + 3\bar{H}_1(t)(\bar{p}_1 - \bar{p}_0) + 3\bar{H}_2(t)(\bar{p}_3 - \bar{p}_2) + \bar{H}_3(t)\bar{p}_3.
\]

Equating weights, we obtain

\[
B_0 = \bar{H}_0 - 3\bar{H}_1,
\]

\[
B_1 = 3\bar{H}_1,
\]

\[
B_2 = -3\bar{H}_2,
\]

\[
B_3 = 3\bar{H}_2 + \bar{H}_3.
\]

Hence

\[
\bar{H}_0(t) = B_0(t) + B_1(t) = (1 - t)^3 + 3t(1 - t)^2 = 2t^3 - 3t^2 + 1,
\]

\[
\bar{H}_1(t) = (1/3)B_1(t) = t(1 - t)^2 = t^3 - 2t^2 + t,
\]

\[
\bar{H}_2(t) = -(1/3)B_2(t) = -t^2(1 - t) = t^3 - t^2,
\]

\[
\bar{H}_3(t) = B_2(t) + B_3(t) = 3t^2(1 - t) + t^3 = -2t^3 + 3t^2.
\]

Values of \(H^i_0, \ldots, H^i_3\) are then obtained by substituting

\[
(u - u_i)/(u_{i+1} - u_i)
\]

for \(t\) in the above expressions with \(\bar{H}_1(t)\) and \(\bar{H}_2(t)\) scaled by \(\frac{du}{dt} = u_{i+1} - u_i\).
Uniform Cubic B-splines

Given \( n + 5 \) knots \( u_i \) for \( i = -2, -1, \ldots, n + 2 \), uniformly distributed (e.g., \( u_i = i \)), define the cubic B-spline \( B_i \), sometimes denoted \( B_{i,3} \) or \( B_i^3 \), for \( i = 0, \ldots, n \), as the cubic spline with support \([u_{i-2}, u_{i+2}]\), triple zeros at the endpoints, and the property that the knot values sum to 1.

To see that \( B_i \) is uniquely defined, note that it has 16 degrees of freedom (a cubic on each of four intervals), and there are 16 linearly independent constraints:

\[
\begin{align*}
B_i(u_{i-2}) &= B_i'(u_{i-2}) = B_i''(u_{i-2}) = 0 \\
B_i(u_{i+2}) &= B_i'(u_{i+2}) = B_i''(u_{i+2}) = 0 \\
B_i & \text{ is continuous at } u_{i-1}, u_i, \text{ and } u_{i+1} \\
B_i' & \text{ is continuous at } u_{i-1}, u_i, \text{ and } u_{i+1} \\
B_i'' & \text{ is continuous at } u_{i-1}, u_i, \text{ and } u_{i+1} \\
B_i(u_{i-1}) + B_i(u_i) + B_i(u_{i+1}) &= 1
\end{align*}
\]
Uniform Cubic B-splines continued

With uniformly distributed knots, all $n + 1$ B-splines are defined by four cubic blending functions $f_i : [0, 1] \rightarrow \mathbb{R}$

1. $f_1(t) = (1/6)t^3$
2. $f_2(t) = (1/6)(-3t^3 + 3t^2 + 3t + 1)$
3. $f_3(t) = (1/6)(3t^3 - 6t^2 + 4)$
4. $f_4(t) = (1/6)(-t^3 + 3t^2 - 3t + 1)$

The mappings between B-splines and blending functions are defined by

1. $f_1(t) = B_i|[u_{i-2}, u_{i-1}]$ for $t = (u - u_{i-2})/(u_{i-1} - u_{i-2})$
2. $f_2(t) = B_i|[u_{i-1}, u_i]$ for $t = (u - u_{i-1})/(u_i - u_{i-1})$
3. $f_3(t) = B_i|[u_i, u_{i+1}]$ for $t = (u - u_i)/(u_{i+1} - u_i)$
4. $f_4(t) = B_i|[u_{i+1}, u_{i+2}]$ for $t = (u - u_{i+1})/(u_{i+2} - u_{i+1})$
Table of Knot Function Values

<table>
<thead>
<tr>
<th>u</th>
<th>(B_i(u))</th>
<th>(B'_i(u))</th>
<th>(B''_i(u))</th>
</tr>
</thead>
<tbody>
<tr>
<td>(u_{i-1})</td>
<td>1/6</td>
<td>1/2</td>
<td>1</td>
</tr>
<tr>
<td>(u_i)</td>
<td>2/3</td>
<td>0</td>
<td>-2</td>
</tr>
<tr>
<td>(u_{i+1})</td>
<td>1/6</td>
<td>-(1/2)</td>
<td>1</td>
</tr>
</tbody>
</table>

**Exercise** Compute the first, second, and third derivatives of the blending functions, and evaluate them at the endpoints to show that \(B_i\) satisfies the 16 constraints and has discontinuous third derivative.

Note the following properties:

- \(\sum_{i=0}^{n} B_i(u) = \sum_{i=1}^{4} f_i(t) = 1\) for all \(u\) and \(t\).
- Reflection symmetry about \(t = .5\): \(f_4(t) = f_1(1 - t)\) and \(f_3(t) = f_2(1 - t)\).
- Inflection points: \(f''_2(1/3) = f''_3(2/3) = 0\).
B-spline Curve

Given control points $p_i \in \mathbb{R}^d$ for $i = 0, \ldots, n$ and B-spline basis functions $B_i : [u_0, u_n] \rightarrow \mathbb{R}$, the B-spline curve is
\[
\{ C(u) : u \in [u_0, u_n] \}
\]
for the function
\[
C(u) = \sum_{i=0}^{n} B_i(u)p_i.
\]

Cubic B-splines have the following properties:

1. $C$ is a $C^2$ parametric cubic spline; i.e., its components are linear combinations of cubic splines, and therefore are cubic splines.

2. The control points are not interpolated. The closest point of the curve to $p_j$ is
\[
C(u_j) = (1/6)p_{j-1} + (2/3)p_j + (1/6)p_{j+1}.
\]

3. The curve is locally controlled. The portion of the curve affected by a perturbation of $p_j$ is that associated with the support of $B_j$ — \{ $C(u) : u_{j-2} \leq u \leq u_{j+2}$ \}. 

R. J. Renka
Curve Fitting
The curve is *coordinate-free* (affine invariant).

The curve has the *convex hull property*. Not only is each point $C(u)$ of the curve contained in the convex hull of the control points $\{p_i\}_{i=0}^n$, but we have the stronger property that, for $u \in [u_j, u_{j+1}]$,

$$C(u) = B_{j-1}(u)p_{j-1} + B_j(u)p_j + B_{j+1}(u)p_{j+1} + B_{j+2}(u)p_{j+2}$$

$$= f_4(t)p_{j-1} + f_3(t)p_j + f_2(t)p_{j+1} + f_1(t)p_{j+2}$$

for $t = (u - u_j)/(u_{j+1} - u_j)$, where $\sum_{i=1}^4 f_i(t) = 1$ and $f_i(t) \geq 0$ for all $t$, so that $C(u)$ is in the convex hull of four control points.

**Hand-drawing rule:** $C(u_j) = (2/3)p_j + (1/3)M$ for $M = (p_{j-1} + p_{j+1})/2$, and $C'(u_j) = (p_{j+1} - p_{j-1})/2$. 
For $u \in [u_j, u_{j+1}]$, $t = (u - u_j)/(u_{j+1} - u_j) \in [0, 1]$,

$$C(t) = f_4(t)p_{j-1} + f_3(t)p_j + f_2(t)p_{j+1} + f_1(t)p_{j+2}$$

and

$$C(t)^T = (1/6)[ t^3 \quad t^2 \quad t \quad 1 ] \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \begin{bmatrix} p^T_{j-1} \\ p^T_j \\ p^T_{j+1} \\ p^T_{j+2} \end{bmatrix}.$$

Partition $[0, 1]$ into $k$ subintervals $t_i = i/k$ ($i = 0, 1, \ldots, k$). For each geometry vector $[p_{j-1} \quad p_j \quad p_{j+1} \quad p_{j+2}]^T$, ($j = 0, \ldots, n - 1$), connect $C(t_{i-1})$ to $C(t_i)$ with a line segment for $i = 1, \ldots, k$. Note that this requires two *phantom control points* $p_{-1}$ and $p_{n+1}$.
The phantom control points define end conditions.

- Define \( p_{-1} = 2p_0 - p_1 \) and \( p_{n+1} = 2p_n - p_{n-1} \). Then 
  \[ C(u_0) = (1/6)p_{-1} + (2/3)p_0 + (1/6)p_1 = p_0 \] and 
  \[ C(u_n) = (1/6)p_{n-1} + (2/3)p_n + (1/6)p_{n+1} = p_n. \] Also, 
  \[ C'(u_0) = -(1/2)p_{-1} + (1/2)p_1 = p_1 - p_0 \] and 
  \[ C'(u_n) = -(1/2)p_{n-1} + (1/2)p_{n+1} = p_n - p_{n-1}. \] Thus, the first and last control points are interpolated, and the endpoint tangent are defined by the first two and last two control points.

- **Periodic control points:** \( p_{-1} = p_n \) and \( p_{n+1} = p_0 \).
- **Duplicate control points:** \( p_{-1} = p_0 \) and \( p_{n+1} = p_n \).
Duplicate control points in the interior reduce geometric continuity. The function retains $C^2$ parametric continuity, but the curve loses continuity of curvature (has only $G^1$ continuity) at a doubled control point $p_{j+1} = p_j$ where

$$C(u_j) = \frac{1}{6}p_{j-1} + \frac{5}{6}p_j, \quad C(u_{j+1}) = \frac{5}{6}p_j + \frac{1}{6}p_{j+2},$$

$$C'(u_j) = \frac{1}{2}(p_j - p_{j-1}), \quad C'(u_{j+1}) = \frac{1}{2}(p_{j+2} - p_{j+1}).$$

At a triple control point $p_{j-1} = p_j = p_{j+1}$ we have only $G^0$ continuity. $C(u_j) = p_j$ so that $p_j$ is interpolated, and

$$C(u_{j-1}) = \frac{1}{6}p_{j-2} + \frac{5}{6}p_j, \quad C(u_{j+1}) = \frac{5}{6}p_j + \frac{1}{6}p_{j+2}.$$

The tangent at $p_j$ is undefined:

$$C'(u_{j-1}) = \frac{1}{2}(p_j - p_{j-2}), \quad C'(u_j) = 0, \quad C'(u_{j+1}) = \frac{1}{2}(p_{j+2} - p_j).$$
We now consider B-splines of degree $k$ for $1 \leq k \leq k_{\text{max}}$ with nonuniformly distributed knots, including duplicate knots (knots with multiplicity greater than 1). We may either use knots outside the domain of the function $C$ or take the endpoint knots to have multiplicity $k$. We will use a recursive definition. Alternative means of defining B-splines include divided differences and convolution.

For $n \geq k_{\text{max}}$, assume we have a nondecreasing sequence of knots $u_i$ for $-1 \leq i \leq n + k_{\text{max}}$, and define the linear functions

$$L_i^k(u) = \frac{u - u_{i-1}}{u_{i+k-1} - u_{i-1}}$$

for $i = 0, \ldots, n + 1 + k - k_{\text{max}}$. Then

$$B_i^0(u) = \begin{cases} 1 & \text{if } u_{i-1} \leq u < u_i \\ 0 & \text{otherwise} \end{cases}$$

for $0 \leq i \leq n + k_{\text{max}}$, and, for $0 \leq i \leq n + k_{\text{max}} - k$,

$$B_i^k(u) = L_i^k(u)B_i^{k-1}(u) + (1 - L_{i+1}^k(u))B_{i+1}^{k-1}(u).$$
Nonuniform B-splines continued

The support of $B^k_i(u)$ is $[u_{i-1}, u_{i+k}]$. The linear B-spline is

$$B^1_i(u) = \frac{u - u_{i-1}}{u_i - u_{i-1}} B^0_i(u) + \left(1 - \frac{u - u_i}{u_{i+1} - u_i}\right) B^0_{i+1}(u).$$

The terms that are undefined due to duplicate knots are taken to be zeros. Just as duplicate control points reduce geometric continuity and move the curve closer to the control polygon in the case of uniform B-splines, duplication of interior knots results in loss of parametric continuity and moves the curve closer to the control polygon in the nonuniform case. The continuity class of a degree-$k$ B-spline is $C^{k-1}$ if there are no duplicate knots other than the endpoints, $C^{k-2}$ at a double knot (for $k \geq 2$), $C^{k-3}$ at a triple knot (for $k \geq 3$), etc. A cubic B-spline curve interpolates any control points associated with a knot of multiplicity 3.
Nonuniform Rational B-splines (NURBS)

Given control points $p_i \in \mathbb{R}^d$, degree-$k$ B-spline basis functions $B_{i}^{k} : [u_0, u_n] \to \mathbb{R}$, and nonnegative weights $w_i$ for $i = 0, \ldots, n$, the B-spline curve associated with control points $(w_ip_i, w_i) \in \mathbb{R}^{d+1}$ is

$$\sum_{i=0}^{n} w_i B_{i}^{k}(u) \begin{bmatrix} w_ip_i \\ w_i \end{bmatrix}.$$ 

We obtain a rational B-spline curve in $\mathbb{R}^d$ by projecting through the origin onto the hyperplane $w = 1$:

$$C(u) = \frac{\sum_{i=0}^{n} w_i B_{i}^{k}(u)p_i}{\sum_{i=1}^{n} w_i B_{i}^{k}(u)}.$$ 

Note that $C$ is a polynomial B-spline if $w_i = c$ for all $i$. The weights are shape parameters: increasing $w_i$ pulls the curve toward $p_i$. The curve has all of the properties of a polynomial B-spline curve with the additional property of invariance under a projective projection.
The following table summarizes the tradeoffs among curve properties. For the degree-k spline, assume that $k > 1$.

<table>
<thead>
<tr>
<th></th>
<th>Degree-1 Spline</th>
<th>Hermite Cubic</th>
<th>Degree-k Spline</th>
<th>Poly. Interp.</th>
<th>Bézier Curve</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuity</td>
<td>$C^0$</td>
<td>$C^1$</td>
<td>$C^{k-1}$</td>
<td>$C^\infty$</td>
<td>$C^\infty$</td>
</tr>
<tr>
<td>Interpolates</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Local Control</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Coord-free</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes/No</td>
<td>Yes</td>
</tr>
<tr>
<td>Convex hull</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

The column labeled 'Poly. Interp.' is a polynomial interpolant. For Lagrangian basis functions it is coordinate-free; with monomials it is not. A fundamental tradeoff is that $C^\infty$ continuity requires a global method. Also, with the exception of (nonsmooth) piecewise linear interpolation, the convex hull property excludes interpolation (except at the endpoints).
A parametric surface is of the form

\[ \{ \mathbf{S}(u, v) = (x(u, v), y(u, v), z(u, v)) : u \in [a, b], \ v \in [c, d] \}. \]

A regular parametric surface has linearly independent first partial derivatives \( \mathbf{S}_u \) and \( \mathbf{S}_v \) so that \( \mathbf{S}_u \times \mathbf{S}_v \neq 0 \), and the tangent plane and surface normal direction are therefore well-defined at every point. A \( G^1 \) surface has continuous tangent and normal, hence continuity of specular reflection (highlights). A tensor product surface is of the form

\[ \mathbf{S}(u, v) = \sum_i \sum_j B_{ij}(u, v) \mathbf{p}_{i,j}, \]

where the basis functions \( B_{ij} \) may be expressed as products \( B_{ij}(u, v) = f_i(u)g_j(v) \). 

R. J. Renka Curve Fitting
A parametric tensor product B-spline surface is constructed from tensor products of univariate B-splines:

\[ S(u, v) = \sum_{i=0}^{n} \sum_{j=0}^{m} B_i^k(u) B_j^l(v) p_{ij} \]

for knot sequences \( u_i \) and \( v_j \) and polynomial degrees \( k \) and \( l \). A rational B-spline surface is defined as the projection of a 4D tensor product B-spline surface but is not itself a tensor product surface. A B-spline surface may be closed in one direction, forming a cylinder, or both, forming a torus. No tensor product surface can have the topology of a sphere without degeneracies.

Just as NURBS curves allow the exact representation of conic sections, rational B-spline surfaces allow the exact representation of surfaces of revolution and of quadric surfaces (zeros of quadratic functions of three variables, including spheres and cylinders).
A *surface of revolution* is of the form

\[
S(u, v) = (r(v) \cos(2\pi u), r(v) \sin(2\pi u), z(v))
\]

for \( u, v \in [0, 1] \). The curve \((r(v), z(v))\) in the \(xz\)-plane is revolved about the \(z\)-axis. For fixed \(v\), the intersection of the surface with the plane \(z = z(v)\) is a circle of radius \(r(v)\). Another means of generating a surface from curves is a *generalized cylinder surface*, defined by

\[
S(u, v) = (1 - v)Y_0(u) + vY_1(u)
\]

for \(v \in [0, 1]\) and parametric curves \(Y_0(u)\) and \(Y_1(u)\). A *tube surface* has the form

\[
S(u, v) = C(v) + r(v) \cos(2\pi u)n(v) + r(v) \sin(2\pi u)b(v)
\]

for center curve \(C(v)\), radius \(r(v)\), and orthonormal vectors \(t(v), n(v)\), and \(b(v)\), where \(t(v) = C'(v)/\|C'(v)\|\). A surface of revolution is a tube surface with a line segment as center curve.