Béziers and B-splines as Multiaffine Maps

Lyle Ramshaw

Systems Research Center, Digital Equipment Corporation
130 Lytton Avenue, Palo Alto, California 94301 USA

Abstract

It is a classical principle in mathematics that polynomials in a single variable of degree n are essentially equivalent to symmetric polynomials in n variables that are linear in each variable separately. We shall apply this principle to the Bézier and B-spline curves and surfaces that are used in computer aided geometric design. The main result is a method of labeling the Bézier points that control a curve segment or surface patch or the de Boor points that control a B-spline curve with symmetric, multivariate labels. The properties of these labels make it simple to understand or to reconstruct the basic algorithms in this area, such as the de Casteljau Algorithm and the de Boor Algorithm.

Key words

B-spline, Bézier curve, Bézier triangle, computer aided geometric design, de Boor Algorithm, de Casteljau Algorithm, interpolation, multilinearity.

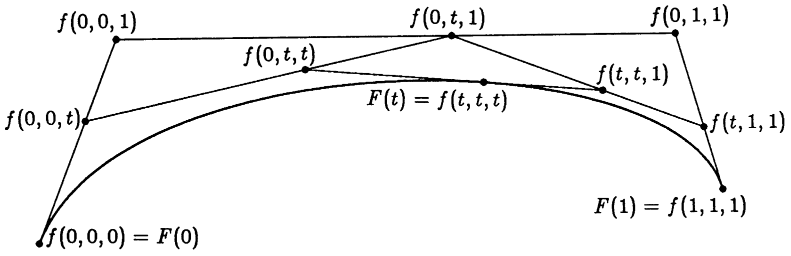


Fig. 1. The de Casteljau Algorithm for a cubic curve

1. Introduction

The standard explanations of the theory underlying Bézier and B-spline curves and surfaces aren't as simple as they should be. They have beautiful figures, and those figures clearly illustrate the geometry of the associated algorithms. But the labels on the points in the figures are a disaster. In particular, there is no easy way to tell, from the labels, what geometric relationships hold among the labeled points. Fortunately, there is a fix.

The best introduction to this fix is through examples. Fig. 1 shows a Bézier cubic curve segment $F([0,1])$ that has been subdivided into the two subsegments $F([0,t])$ and $F([t,1])$ by means of the de Casteljau Algorithm. (For definitions of Bézier curves and the de Casteljau Algorithm, see the excellent survey article by Böhm, Farin, and Kahmann [2], and the sources cited therein.) The function $f(u_1, u_2, u_3)$ appearing in the labels is a symmetric function of three real arguments

NATO ASI Series, Vol. F40
Theoretical Foundations of Computer Graphics and CAD
Edited by R.A. Earnshaw
© Springer-Verlag Berlin Heidelberg 1988

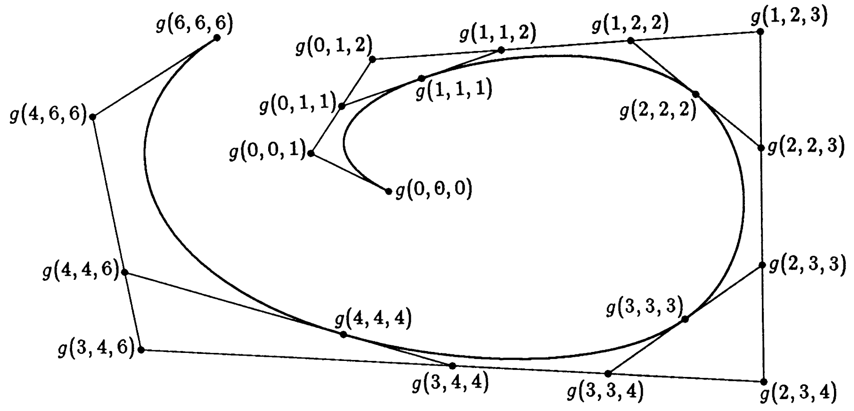


Fig. 2. The Böhm-Sablonnière Rules for a cubic spline curve

that is related to the curve F by the identity $F(u) = f(u,u,u)$. Note that the incidence structure of the points and lines in the figure is reflected in the labels: two points lie on the same line if and only if two out of the three arguments in their labels are the same. The distance ratios are also implied by the labels: for example, the point $f(t,t,1)$ lies t of the way from $f(0,t,1)$ to $f(1,t,1)$ (note that $f(1,t,1) = f(t,1,1)$). In fact, whenever we vary one of the arguments of f while holding the other two fixed, the value of f varies along a straight line at a constant rate—in fancier language, the function f is *triaffine*.

In the other direction, suppose that we were told that f was a symmetric, triaffine function and that the four outer vertices in Fig. 1 were the points $f(0,0,0)$, $f(0,0,1)$, $f(0,1,1)$, and $f(1,1,1)$. And suppose that we wished to compute $f(t,t,t)$. We would be able to reconstruct Fig. 1 from this information, even if we had never heard of the de Casteljau Algorithm. We would linearly interpolate between the four outer points in pairs to compute the three points with one t in their labels. A second stage of interpolation, taking these three points in pairs, would give us the two points $f(0,t,t)$ and $f(t,t,1)$. And a third and final stage of interpolation would give us the desired point $f(t,t,t) = F(t)$.

A similar flavor of multiaffine labeling also works for spline curves. Fig. 2 shows a cubic B-spline curve G with five segments, whose underlying knot sequence is $(0,0,0,0,1,2,3,4,6,6,6,6)$. As before, the labeling function g is a symmetric function of three arguments, related to the spline G by the identity $G(u) = g(u,u,u)$. Furthermore, the function g behaves triaffinely. For example, as u varies from 2 to 6, the point $g(u,3,4)$ moves at a constant rate along the line segment joining $g(2,3,4)$ to $g(6,3,4)$. Note that the de Boor points of G are precisely the points with labels of the form $g(r,s,t)$ where (r,s,t) is a triple of consecutive knots. Furthermore, for each pair (s,t) of consecutive knots, the Bézier points of the cubic polynomial segment $G([s,t])$ are the points $g(s,s,s)$, $g(s,s,t)$, $g(s,t,t)$, and $g(t,t,t)$. Thus, the multiaffine labels help to clarify the relationship between the de Boor points of a spline curve and the Bézier points of its segments, a relationship first explored by Böhm [1] and Sablonnière [6].

A word of warning: since $G(u)$ is a spline curve rather than a single polynomial

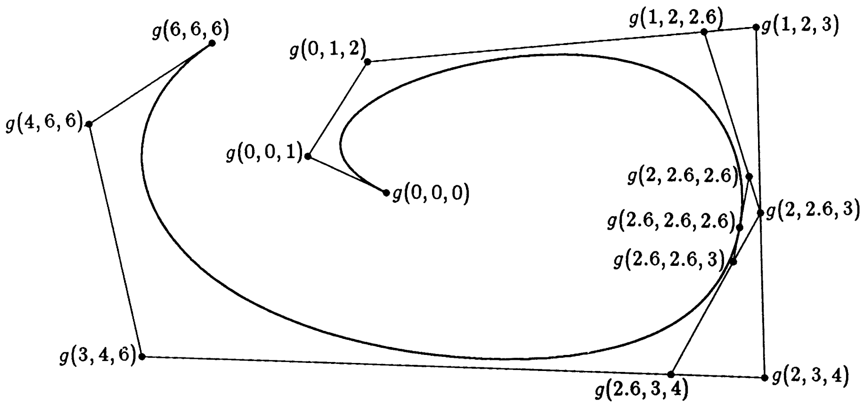


Fig. 3. The de Boor Algorithm for a cubic spline curve

cubic, the function g can't be as simple as the f in Fig. 1. The extra complexity shows up in the fact that $g(u_1, u_2, u_3)$ is well-defined only when its three arguments are fairly close together—in particular, when they don't skip over any knots. For example, the value $g(u, 3, 4)$ is well-defined only for u between 2 and 6.

Fig. 3 shows the same B-spline curve as Fig. 2 with the auxiliary points that we would draw to compute $G(2.6)$ using the de Boor Algorithm [2]. Starting from the de Boor points, we construct new points that have more and more copies of 2.6 among their g-arguments by performing linear interpolations between previously constructed points. Comparing our Fig. 3 with the corresponding Fig. 18 in the survey paper [2], it is amazing how well their figure is able to communicate the de Boor Algorithm under the severe handicap that its points are just labeled "d_i^j".

2. The Blossoming Principle

The Blossoming Principle is the core idea behind these new labelings. It allows us to replace a function of high degree in one variable by an essentially equivalent function of degree one but with lots of variables. This idea is not new; the new things in this paper are the name "blossoming" and the application of blossoming to the polynomial curves and surfaces used in computer aided geometric design.

Consider the cubic polynomial $Q(t) = 7t^3 + 6t^2 - 3t + 5$. Suppose that we want to construct a trivariate polynomial $q(t_1, t_2, t_3)$ of the form

$$q(t_1, t_2, t_3) = at_1t_2t_3 + bt_1t_2 + ct_2t_3 + dt_3t_1 + et_1 + ft_2 + gt_3 + h$$

that satisfies the identity $q(t, t, t) = Q(t)$; that is, we want the main diagonal of $q(t_1, t_2, t_3)$ to agree with Q. To achieve this correspondence, we must have $a = 7$, $b + c + d = 6$, $e + f + g = -3$, and $h = 5$. We can determine q uniquely if we also demand that $q(t_1, t_2, t_3)$ be a symmetric function of its three arguments. This symmetry condition forces us to make $b = c = d$ and $e = f = g$, resulting in the unique choice

$$q(t_1, t_2, t_3) := 7t_1t_2t_3 + 2t_1t_2 + 2t_2t_3 + 2t_3t_1 - t_1 - t_2 - t_3 + 5.$$

This argument can be generalized quite a bit; but first, a question of nomenclature. What shall we call a polynomial, such as the q above, that has degree one in each of its variables separately? The name "multilinear" pops to mind, but there is the problem that the word "linear" often implies "homogeneous" as well as "of degree one", and q would have to have $b = c = \cdots = h = 0$ in order to be multilinear in the homogeneous sense. To avoid confusion, we shall use the word "affine" instead of "linear" when we mean "of degree one, but not necessarily homogeneous". In particular, we shall say that q is *multiaffine*.

The argument above shows that, if $Q(t)$ is any cubic polynomial, there exists a unique symmetric triaffine polynomial $q(t_1, t_2, t_3)$ that satisfies $q(t,t,t) = Q(t)$. Going backwards from q to Q is even easier: if $q(t_1, t_2, t_3)$ is triaffine, its diagonal $Q(t) := q(t,t,t)$ is always a cubic polynomial, whether q is symmetric or not. Thus, univariate cubic polynomials and symmetric triaffine polynomials are essentially equivalent. Of course, that doesn't mean that they are equally easy to work with. In many cases, the triaffine version reveals the underlying structure more clearly than does the cubic version of the same abstract entity. Based on this observation, we shall refer to the process of deriving q from Q as *blossoming*, and we shall refer to q as the *blossom* of Q.

Any function defined by polynomials can be blossomed. For computer graphics applications, we shall extend our blossoming expertise in two ways. First, we shall generalize from degree 3 to degree n. Second, we shall deal with functions from affine spaces to affine spaces rather than with functions from the reals to the reals.

Recall that an *affine space* is like a linear space except for the lack of a distinguished origin. For the purposes of this paper, we shall restrict ourselves to affine spaces that are finite dimensional. An *affine combination* of points in an affine space is a linear combination whose coefficients sum to 1. If a parameter space P and an object space O are affine spaces, a function $h: P \to O$ is called *affine* if it preserves affine combinations. A function $f: P^n \to O$ is called *multiaffine* if it is an affine function of each argument when the others are held fixed. In the finite-dimensional case, a function $h: P \to O$ is affine if and only if, when we choose origins and bases for P and O, each coordinate of the point $h(\mathbf{u})$ can be written as a polynomial of total degree no greater than 1 in the coordinates of the argument point $\mathbf{u}$. Similarly, a function $f: P^n \to O$ is multiaffine if and only if, when we choose origins and bases, each coordinate of $f(\mathbf{u}_1, \ldots, \mathbf{u}_n)$ can be written as a polynomial in the coordinates of the argument points $\mathbf{u}_i$ in which each term includes at most one of the coordinates of any particular $\mathbf{u}_i$ as a factor.

If P and O are finite-dimensional affine spaces, a function $F: P \to O$ is called *polynomial of degree n* if, when we choose origins and bases for P and O, each coordinate of the point $F(\mathbf{u})$ can be written as a polynomial of degree no greater than n in the coordinates of the argument point $\mathbf{u}$. The special names *polynomial curve* and *polynomial surface* are used for polynomial functions $F: P \to O$ whose domains P have dimensions 1 and 2, respectively.

The Blossoming Principle (affine variant). *If P and O are finite-dimensional affine spaces, polynomial functions $F: P \to O$ of degree n and symmetric n-affine maps $f: P^n \to O$ are essentially equivalent. In particular, given a function of either type, a unique function of the other type exists satisfying $F(\mathbf{u}) = f(\mathbf{u}, \mathbf{u}, \ldots, \mathbf{u})$.*

Proof. Since there is no interaction between the different coordinates of the points in the object space O, it suffices to consider each coordinate separately, or equivalently, to consider the case where O is the real numbers $\mathbf{R}$.

Let p be the dimensionality of the parameter space P; choose an origin for

P; let the points $\mathbf{r}^1$ through $\mathbf{r}^p$ be a basis for the resulting linear space; and let $\mathbf{u} = u^1\mathbf{r}^1 + \cdots + u^p\mathbf{r}^p$ be the expansion of the point $\mathbf{u}$ into coordinates with respect to this basis. We are using superscripts to enumerate the p dimensions of P in order to leave subscripts free to enumerate the n arguments of f.

By our definitions, the map $F\colon P \to \mathbf{R}$ is a polynomial function of degree n if and only if the quantity $F(\mathbf{u})$ is given by a polynomial of total degree no greater than n in the variables u^1 through u^p. A term of this polynomial has the form

$$T_F := c(u^1)^{k_1}(u^2)^{k_2}\cdots(u^p)^{k_p}$$

for some real coefficient c and for some nonnegative integer exponents k_j satisfying $\sum_{j=1}^{p} k_j \le n$. Similarly, the function $f\colon P^n \to \mathbf{R}$ is n-affine if and only if we can express the value $f(\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n)$ as a polynomial in the np variables v_i^j whose general term has the form

$$T_f := c\prod_{i=1}^{n}\prod_{j=1}^{p}(v_i^j)^{e_{ij}}$$

where each e_{ij} is either 0 or 1 and we have $\sum_j e_{ij} \le 1$ for each fixed i.

Suppose that are given F and want to construct f satisfying $f(\mathbf{u}, \ldots, \mathbf{u}) = F(\mathbf{u})$. Note that the term T_f will contribute to T_F when we substitute $\mathbf{u}$ for each of the $\mathbf{v}_i$ if and only if $\sum_{i=1}^{n} e_{ij} = k_j$ for each j. If we let l denote the difference $l := n - \sum_j k_j$, the number of terms T_f that contribute to the particular term T_F is given by the multinomial coefficient $\binom{n}{k_1 \ldots k_p\ l}$. If we want f to be symmetric, our only choice is to split up each term T_F into $\binom{n}{k_1 \ldots k_p\ l}$ symmetric terms T_f in f with equal coefficients.

Going the other way, suppose that we are given f, whose terms T_f satisfy $\sum_j e_{ij} \le 1$ by assumption. When we substitute $\mathbf{u}$ for each of the $\mathbf{v}_i$, the total degree of the result will be $\sum_i \sum_j e_{ij} \le n$. Hence, the identity $F(\mathbf{u}) := f(\mathbf{u}, \ldots, \mathbf{u})$ will define a polynomial function of degree n or less, and it is obviously unique. This completes the proof.

There is an analogous Blossoming Principle that holds in the linear world: if we start with a homogeneous polynomial function F, the resulting blossom f is multilinear instead of just being multiaffine. The only difference in the proof is that we always have $l = 0$ in the homogeneous case. The particular case $n = 2$ and $O = \mathbf{R}$ of the linear variant of the Blossoming Principle is quite a famous theorem: it states that quadratic forms on a vector space are essentially equivalent to symmetric bilinear forms.

Our proof of the Blossoming Principle was done in terms of explicit bases. We could have reduced the basis-dependence by using the inclusion-exclusion formula

$$f(\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n) := \frac{1}{n!}\sum_{\substack{S\subseteq\{1,2,\ldots,n\}\\ 0<k=|S|}} (-)^{n-k}k^n F\left(\frac{1}{k}\sum_{i\in S}\mathbf{u}_i\right),$$

which expresses the blossom f directly in terms of values of the polynomial function F. But we had no hope of avoiding bases entirely, since our definition of the concept of a polynomial function involved bases. More abstract texts turn the Blossoming Principle from a theorem into a definition: they define a "polynomial function" to be the diagonal of a multiaffine function. The argument above is then used to show that, in the finite-dimensional case, these "polynomial functions" are precisely the functions that are given by polynomials when expanded in terms of a basis.

3. The blossoms of Béziers

Polynomial curves and surfaces are basic objects in computer aided design, and the affine variant of the Blossoming Principle applies to them directly. In this section, we shall relate blossoms to the Bézier theory of curves and surfaces.

Curves are the simplest case. Let $F: \mathbf{R} \to O$ be a polynomial curve of degree n, and let $f: \mathbf{R}^n \to O$ be its blossom; that is, let f be the unique n-affine map that satisfies $F(u) = f(u, \ldots, u)$. Suppose that we chose a reference interval $[s, t] \subset \mathbf{R}$, and consider the value $f(u_1, u_2, \ldots, u_n)$. We can write the first argument u_1 of f as an affine combination of the numbers s and t as follows:

$$u_1 = \left(\frac{t-u_1}{t-s}\right) s + \left(\frac{u_1-s}{t-s}\right) t.$$

Since f is affine in its first argument, we can then conclude that

$$f(u_1, u_2, \ldots, u_n) = \left(\frac{t-u_1}{t-s}\right) f(s, u_2, \ldots, u_n) + \left(\frac{u_1-s}{t-s}\right) f(t, u_2, \ldots, u_n).$$

The next step is conceptually straightforward, although notationally somewhat clumsy: we want to simultaneously expand all n of the arguments u_1 through u_n as affine combinations of s and t. The resulting sum has 2^n terms:

$$f(u_1, u_2, \ldots, u_n) = \sum_{\substack{I \cap J = \emptyset \\ I \cup J = \{1,2,\ldots,n\}}} \prod_{i \in I} \left(\frac{t-u_i}{t-s}\right) \prod_{j \in J} \left(\frac{u_j-s}{t-s}\right) f(\underbrace{s, \ldots, s}_{|I|}, \underbrace{t, \ldots, t}_{|J|}).$$

Note that we have taken advantage of the symmetry of f in order to put all of the s's before all of the t's in the arguments to f. The interesting thing about this formula is that it expresses every value of f in terms of just the $n+1$ special values whose arguments consist entirely of s's and t's. Thus, a symmetric n-affine function $f: \mathbf{R}^n \to O$ is completely determined once we know the $n+1$ particular values $f(s, \ldots, s)$, $f(s, \ldots, s, t)$, $\ldots$, $f(t, \ldots, t)$ as points in O. Furthermore, there are no constraints on these special values: if $\mathbf{x}_0$ through $\mathbf{x}_n$ are any $n+1$ points in O, the formula

$$f(u_1, u_2, \ldots, u_n) := \sum_{\substack{I \cap J = \emptyset \\ I \cup J = \{1,2,\ldots,n\}}} \prod_{i \in I} \left(\frac{t-u_i}{t-s}\right) \prod_{j \in J} \left(\frac{u_j-s}{t-s}\right) \mathbf{x}_{|J|}$$

defines a symmetric n-affine function $f: \mathbf{R}^n \to O$ that satisfies

$$f(\underbrace{s, \ldots, s}_{n-k}, \underbrace{t, \ldots, t}_{k}) = \mathbf{x}_k.$$

Thus, there is a one-to-one correspondence between symmetric n-affine functions $f: \mathbf{R}^n \to O$ and $(n+1)$-tuples $(\mathbf{x}_0, \ldots, \mathbf{x}_n)$ of points in O.

If we consider the special case where all of the arguments to the blossom f are equal, we find that we have simply reinvented the Bézier theory for polynomial curves, with the points $\mathbf{x}_k$ being the Bézier points. We have

$$F(u) = f(u, \ldots, u) = \sum_{0 \le k \le n} \binom{n}{k} \left(\frac{t-u}{t-s}\right)^{n-k} \left(\frac{u-s}{t-s}\right)^k f(\underbrace{s, \ldots, s}_{n-k}, \underbrace{t, \ldots, t}_{k}).$$

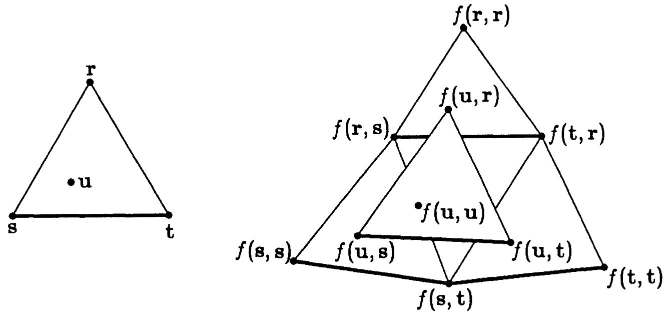


Fig. 4. The de Casteljau Algorithm for a quadratic surface

Therefore, if $F\colon \mathbf{R} \to O$ is a polynomial curve of degree n and $[s,t] \subset \mathbf{R}$ is any reference interval, the Bézier points of the n-ic segment $F([s,t])$ are precisely the blossom values $f(s,\ldots,s)$, $f(s,\ldots,s,t)$, $\ldots$, $f(t,\ldots,t)$.

We can begin to develop a geometric intuition for the behavior of blossoms from this result. Consider a quadratic polynomial curve in the plane, say $F(u) = (u, u^2)$. For any s and t, we know from the theory above that the three Bézier points of the parabolic segment $F([s,t])$ are the blossom values $f(s,s)$, $f(s,t)$, and $f(t,t)$. Hence, the point $f(s,t)$ must be the intersection of the tangents to the parabola $F(u)$ at $u = s$ and $u = t$. As a consequence, the range of the blossom f is precisely the parabola itself together with its exterior.

Surfaces aren't much harder than curves. Recall that a polynomial surface is a polynomial function $F\colon P \to O$ where P is two-dimensional. Let $\Delta\mathbf{rst} \subset P$ be a reference triangle. We shall use the three vertices $\mathbf{r}$, $\mathbf{s}$, and $\mathbf{t}$ of this reference triangle to form an affine frame for the two-dimensional parameter space P. Every point $\mathbf{u}$ in P can be written uniquely as an affine combination of the three points in this frame, in the form $\mathbf{u} = \beta_{\mathbf{r}}(\mathbf{u})\mathbf{r} + \beta_{\mathbf{s}}(\mathbf{u})\mathbf{s} + \beta_{\mathbf{t}}(\mathbf{u})\mathbf{t}$; the coefficients $\beta_{\mathbf{r}}(\mathbf{u})$, $\beta_{\mathbf{s}}(\mathbf{u})$ and $\beta_{\mathbf{t}}(\mathbf{u})$ in this expansion are called the ***barycentric coordinates*** of the point $\mathbf{u}$.

Let $f\colon P^n \to O$ be the blossom of the polynomial surface F. If we express all n of the arguments of f as affine combinations of $\mathbf{r}$, $\mathbf{s}$, and $\mathbf{t}$ and expand out, we get a sum for $f(\mathbf{u}_1,\ldots,\mathbf{u}_n)$ that has 3^n terms:

$$\sum_{\substack{I,J,K \text{ any} \\ \text{partition of} \\ \{1,2,\ldots,n\}}} \prod_{i\in I} \beta_{\mathbf{r}}(\mathbf{u}_i) \prod_{j\in J} \beta_{\mathbf{s}}(\mathbf{u}_j) \prod_{k\in K} \beta_{\mathbf{t}}(\mathbf{u}_k) f(\underbrace{\mathbf{r},\ldots,\mathbf{r}}_{|I|},\underbrace{\mathbf{s},\ldots,\mathbf{s}}_{|J|},\underbrace{\mathbf{t},\ldots,\mathbf{t}}_{|K|}).$$

This formula expresses every value of f in terms of the $\binom{n+2}{2}$ special values in which all of the arguments to f are either $\mathbf{r}$'s, $\mathbf{s}$'s, or $\mathbf{t}$'s. If we let all of the $\mathbf{u}_i$ be equal, we get

$$f(\mathbf{u},\ldots,\mathbf{u}) = \sum_{i+j+k=n} \binom{n}{i\ j\ k} \beta_{\mathbf{r}}(\mathbf{u})^i \beta_{\mathbf{s}}(\mathbf{u})^j \beta_{\mathbf{t}}(\mathbf{u})^k f(\underbrace{\mathbf{r},\ldots,\mathbf{r}}_{i},\underbrace{\mathbf{s},\ldots,\mathbf{s}}_{j},\underbrace{\mathbf{t},\ldots,\mathbf{t}}_{k}).$$

This shows that the special values of the blossom f are precisely the Bézier points of the triangular surface patch formed by restricting $F(\mathbf{u}) = f(\mathbf{u},\ldots,\mathbf{u})$ to $\Delta\mathbf{rst}$. Fig. 4 shows an application of the de Casteljau Algorithm to a quadratic triangular surface patch in which all points have been labeled as blossom values.

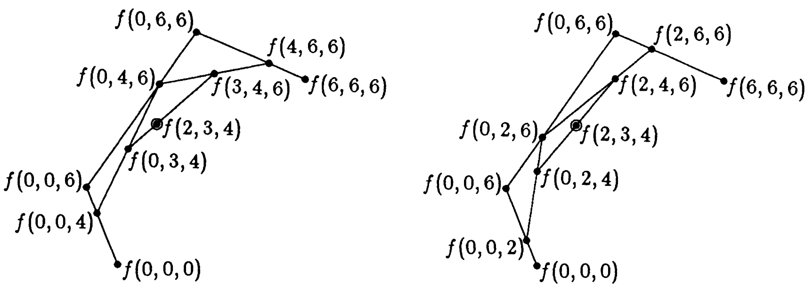


Fig. 5. Different ratios in different stages of the de Casteljau Algorithm

For a parameter space P of dimension $p > 2$, were one to arise, we would choose a reference p-simplex. A polynomial function $F: P \to O$ would have $\binom{n+p}{p}$ Bézier points, each of which was the result of choosing, with replacement, n points from among the $p+1$ simplex vertices and then evaluating the blossom on those n points.

4. The geometric view of a blossom's n arguments

What do the n different arguments to a blossom mean? Can we think of the blossom value $f(u_1, \dots, u_n)$ as the result of evaluating the curve $F(u)$ at n different times? In the next three sections, we shall study this question from three perspectives: geometric, differential, and algebraic.

From a geometric point of view, the meaning of the original time parameter u of a curve $F(u)$ is a choice of interpolation ratio. Given the Bézier points of the curve segment $F([s,t])$ and the ratio of the lengths of the line segments into which the number u divides the interval $[s,t]$, the de Casteljau Algorithm tells us how to construct the point $F(u) = f(u, \dots, u)$ by performing n stages of linear interpolations with that ratio.

It makes perfect sense, however, to use different ratios for the different stages of the de Casteljau Algorithm. If we use the ratio corresponding to u_i during the ith stage, the point that we end up constructing is precisely the blossom value $f(u_1, \dots, u_n)$. For example, the left half of Fig. 5 shows the construction of the point $f(4,3,2) = f(2,3,4)$ from the Bézier points of a cubic segment $F([0,6])$ by interpolating with ratios $2/3$, $1/2$, and $1/3$. The symmetry of f implies that using the same bag (multiset) of ratios in a different order will result in the same final point, even though the intermediate points and lines will be different; the right half of Fig. 5 uses the ratios $1/3$, $2/3$, and $1/2$. Thus, the geometric way to interpret the n arguments of a curve's blossom is that each of them provides the ratio to use during one of the n stages of the de Casteljau Algorithm.

This insight works just as well for surfaces as it does for curves. If $F: P \to O$ is a polynomial surface, any point $\mathbf{u}$ in P specifies a two-dimensional interpolation ratio. That is, the location of $\mathbf{u}$ with respect to the vertices of the reference triangle for P gives a way of choosing a corresponding point in the plane determined by any three points. Given the Bézier points of F, the de Casteljau Algorithm performs n stages of these two-dimensional interpolations in order to construct the point $F(\mathbf{u})$. Once again, it makes perfect sense to use different interpolation ratios during the different stages, and the final result does not depend on the order we choose. Ronald N. Goldman gets the credit for realizing that two-dimensional interpolation stages with different ratios commute with each other [5].

5. The differential view of a blossom's n arguments

Recall that, for a quadratic curve $F(u)$, the blossom value $f(s,t)$ is simply the intersection of the tangent lines to the parabola at $u = s$ and at $u = t$. For a second perspective on the meaning of the n arguments of a blossom, we would like to generalize this observation. In order to do so, we must study how differentiating interacts with blossoming.

As a warmup, let $F: \mathbf{R} \to O$ be a cubic polynomial curve. One possible correct formula for $F'(u)$ in terms of the blossom f is $F'(u) = 3\big(f(u+1,u,u) - f(u,u,u)\big)$. Since f is affine in its first argument, there are lots of other ways to write this, including $F'(u) = f(u+3,u,u) - f(u,u,u)$ and $F'(u) = 3\big(f(v+1,u,u) - f(v,u,u)\big)$ for any v in $\mathbf{R}$. Before we discuss why these formulas are correct, we have to discuss what they mean. Both F and f have values in the affine object space O; but a difference of two points, such as $f(u+1,u,u) - f(u,u,u)$, is not an affine combination—the sum of the coefficients is 0, not 1.

Associated with any affine space A is a linear space of the same dimension, which we shall write A_Δ. The elements of A_Δ are called *vectors* (more precisely, *free vectors*), to distinguish them from the elements of A itself, which are called *points*. Any linear combination of vectors is a vector. A linear combination of points represents a point if the sum of its coefficients is 1; it represents a vector if the sum of its coefficients is 0.

We can now make sense of the subtraction operation in the formula $F'(u) = 3\big(f(u+1,u,u) - f(u,u,u)\big)$ by realizing that the derivative of the curve F is a vector-valued map $F': \mathbf{R} \to O_\Delta$. We can also make sense of the addition in the term "$u+1$" by agreeing that the numeral "1" in that term denotes the unit vector in $\mathbf{R}_\Delta$, rather than the unit point in the affine space $\mathbf{R}$. To actually verify that the formula is true, however, we will need some multivariate calculus.

If $h: P \to O$ is any smooth function between affine spaces P and O, if $\mathbf{x}$ is a point in P, and if ξ is a vector in P_Δ, let $D_\xi h(\mathbf{x})$ denote the directional derivative of h in the ξ direction, evaluated at the point $\mathbf{x}$; the value $D_\xi h(\mathbf{x})$ is a vector in O_Δ. If h is an affine function, its directional derivatives are independent of $\mathbf{x}$. In fact, they are given by a linear map h_Δ, called the *linear part* of h, defined by $h_\Delta(\xi) = h_\Delta(\mathbf{v} - \mathbf{u}) = h(\mathbf{v}) - h(\mathbf{u})$, where $\mathbf{v} - \mathbf{u}$ is any way of writing the vector $\xi = \mathbf{v} - \mathbf{u}$ in A_Δ as a difference of points in A.

We can now compute as follows: $F'(u) = D_1 F(u) = D_{(1,1,\ldots,1)} f(u,u,\ldots,u)$. Breaking the diagonal vector $(1,1,\ldots,1)$ up into its axial components $(1,0,\ldots,0)$, $(0,1,0,\ldots,0)$, and the like, we can use the linearity of D in its subscript and the symmetry of f to deduce that $F'(u) = nD_{(1,0,\ldots,0)} f(u,u,\ldots,u)$. Let $g: \mathbf{R} \to O$ denote f as a function of its first argument when its other arguments are held fixed at u; that is, $g(v) := f(v,u,\ldots,u)$. We have $F'(u) = nD_1 g(u) = ng_\Delta(1) = n\big(g(u+1) - g(u)\big) = n\big(f(u+1,u,\ldots,u) - f(u,u,\ldots,u)\big)$. Note where the annoying factor of n arose: when we vary the single argument u of $F(u)$, all n of the arguments of the blossom $f(u_1,\ldots,u_n)$ vary in parallel. To first order, this causes the resulting value to move n times faster than it would have moved had we varied only one of the blossom arguments.

The directional derivatives of a polynomial surface F work the same way: if ξ is some vector in P_Δ, we have $D_\xi F(\mathbf{u}) = n\big(f(\mathbf{u}+\xi,\mathbf{u},\ldots,\mathbf{u}) - f(\mathbf{u},\mathbf{u},\ldots,\mathbf{u})\big)$. To compute derivatives of higher order, we must take higher-order differences of blossom values. For the second derivative of a curve, we have

$$F''(u) = n^{\underline{2}}(f(u+1,u+1,u,\ldots,u) - 2f(u+1,u,u,\ldots,u) + f(u,u,u,\ldots,u))$$

where $n^{\underline{2}} = n(n-1)$ is an instance of the *falling factorial power*, given by $n^{\underline{k}} = n(n-1)\cdots(n-k+1)$. For a second-order derivative of a surface, we have

$$\begin{aligned}D_\xi D_\eta F(\mathbf{u}) = n^{\underline{2}}(f(\mathbf{u}+\xi,\mathbf{u}+\eta,\mathbf{u},\ldots,\mathbf{u}) - f(\mathbf{u}+\xi,\mathbf{u},\mathbf{u},\ldots,\mathbf{u})\\ - f(\mathbf{u},\mathbf{u}+\eta,\mathbf{u},\ldots,\mathbf{u}) + f(\mathbf{u},\mathbf{u},\mathbf{u},\ldots,\mathbf{u})).\end{aligned}$$

The general formula for the kth derivative of a curve is

$$F^{(k)}(u) = n^{\underline{k}} \sum_{0\le j\le k} \binom{k}{j} (-)^{k-j} f(\underbrace{u+1,\ldots,u+1}_{j},\underbrace{u,\ldots,u}_{n-j}).$$

We won't write out the general formula for the kth-order derivatives of a surface.

One conclusion that we can draw from these formulas is that the more we want to know about the behavior of a function F near a point, the more arguments to its blossom we must be willing to vary away from that point. In particular, computing a kth derivative of F at the point $\mathbf{u}$ involves varying k of the blossom arguments away from $\mathbf{u}$. If we fix all of the blossom arguments at $\mathbf{u}$, our knowledge of F is restricted to zeroth order, that is, to the value $F(\mathbf{u})$ itself. If we can vary all n of the blossom arguments, we can compute F to nth order, that is, we know F completely.

We can sharpen this insight by rephrasing it in terms of osculating flats. For any k, let $\operatorname{Osc}_k F(\mathbf{u})$ denote the smallest affine subspace A with the properties that A contains $F(\mathbf{u})$ and that A_Δ contains all of the vectors $D_\xi F(\mathbf{u})$, $D_\xi D_\eta F(\mathbf{u})$, and the like up through kth order, where ξ, η, and the like are arbitrary vectors in P_Δ. The flat $\operatorname{Osc}_k F(\mathbf{u})$ is said to *osculate F to kth order* at $\mathbf{u}$. The above formulas imply that the mapping

$$(\mathbf{v}_1,\ldots,\mathbf{v}_k) \mapsto f(\mathbf{v}_1,\ldots,\mathbf{v}_k,\underbrace{\mathbf{u},\ldots,\mathbf{u}}_{n-k})$$

has range precisely $\operatorname{Osc}_k F(\mathbf{u})$. In particular, if a point $\mathbf{u}$ occurs m times among the blossom arguments, the resulting blossom value must lie in the flat $\operatorname{Osc}_{n-m} F(\mathbf{u})$.

This concept of osculating flats leads to a very natural differential intuition for the meaning of the n arguments of a blossom. For example, let F be a nondegenerate polynomial cubic, and hence a space curve; and let r, s, and t be distinct numbers. The blossom value $f(r,s,t)$ is the unique point that lies in the intersection of the three osculating planes $\operatorname{Osc}_2 F(r)$, $\operatorname{Osc}_2 F(s)$, and $\operatorname{Osc}_2 F(t)$. The value $f(s,s,t)$ is the intersection of the tangent line $\operatorname{Osc}_1 F(s)$ with the osculating plane $\operatorname{Osc}_2 F(t)$. More generally, for any nondegenerate polynomial curve or surface F, we can determine the blossom value $f(\mathbf{u}_1,\ldots,\mathbf{u}_n)$ uniquely by intersecting the osculating flats corresponding to all of the distinct points $\mathbf{u}$ that occur among the arguments $\mathbf{u}_i$.

The differential intuition is so simple that one might be tempted to use it to define blossoms, instead of manipulating the coefficients of polynomials as we did in Section 2. Unfortunately, that idea runs into trouble if the polynomial curve or surface is degenerate, that is, if its Bézier points are not affinely independent. For example, if F is a polynomial cubic curve that happens to line in a plane (that is, has zero torsion), all of its osculating planes $\operatorname{Osc}_2 F(u)$ are coincident, and hence we can't use their intersections to define the blossom values $f(r,s,t)$.

Note that the osculating flats of surfaces interact with each other somewhat more subtly than the osculating flats of curves. For example, let $F(\mathbf{u})$ be a non-degenerate quadratic polynomial surface. Since F has six Bézier points, the affine span of F is a 5-flat. The blossom value $f(\mathbf{s},\mathbf{t})$ is indeed the intersection of the two tangent planes $\mathrm{Osc}_1\, F(\mathbf{s})$ and $\mathrm{Osc}_1\, F(\mathbf{t})$. The subtle point is that we would expect two planes in a 5-flat to be skew; since these two planes intersect in the point $f(\mathbf{s},\mathbf{t})$, they aren't in general position.

6. The algebraic view of a blossom's n arguments

The third important perspective on the meaning of a blossom's n arguments comes from algebra, in particular, from the factoring of polynomials. The best way to investigate this perspective involves supplementing the Blossoming Principle with a symmetric, affine variant of the tensor product construction, which we might call the Tensoring Principle; but that is another story for another paper. In this paper, we shall keep things simple by limiting our application of the algebraic perspective to a particular family of prototypical Bézier curves and surfaces.

We begin with curves. Let S and T be two formal variables. We are going to consider homogeneous polynomials $Q(S,T)$ in the variables S and T. Just for the purposes of this section, let us refer to the quantity $Q(1,1)$, which is the sum of all of the coefficients in $Q(S,T)$, as the *flavor* of Q. Let $H_1^n[S,T]$ denote the set of all 1-flavored polynomials that are homogeneous of degree n in S and T. Note that $H_1^n[S,T]$ is an affine space of dimension n; the monomials S^n, $S^{n-1}T$, …, T^n form an obvious affine frame for it. Now, for each n, we can define a prototypical Bézier curve of degree n: let $\Phi_n\colon H_1^1[S,T] \to H_1^n[S,T]$ be the map that raises its argument to the nth power; that is, for B in $H_1^1[S,T]$, let $\Phi_n(B) := B^n$.

An arbitrary Bézier curve $F\colon \mathbf{R} \to O$ of degree n isn't that different from Φ_n. If $[s,t]$ is a reference interval in $\mathbf{R}$, then the function $g\colon \mathbf{R} \to H_1^1[S,T]$ given by

$$g(u) := \left(\frac{t-u}{t-s}\right) S + \left(\frac{u-s}{t-s}\right) T$$

is an affine isomorphism between $\mathbf{R}$ and the domain of Φ_n. On the codomain side, we can define a map $h\colon H_1^n[S,T] \to O$ by taking each monomial to the corresponding Bézier point,

$$h(S^{n-i}T^i) := f(\underbrace{s,\ldots,s}_{n-i},\underbrace{t,\ldots,t}_{i}),$$

and then extending so as to make h affine. We then have $F(u) = h(\Phi_n(g(u)))$. Thus, we aren't losing much when we decide to study the prototypical curve Φ_n instead of an arbitrary curve F.

The advantage of Φ_n is that its blossom $\varphi_n\colon (H_1^1[S,T])^n \to H_1^n[S,T]$ is just given by multiplication: $\varphi_n(B_1,\ldots,B_n) := B_1B_2\cdots B_n$. To prove this claim, it is enough to note that the function φ_n defined by multiplication is symmetric, is n-affine, and has the correct values on the diagonal.

Here, then, is the algebraic perspective on the meaning of a blossom's n arguments. A point $F(u)$ on a Bézier curve corresponds to a homogeneous bivariate formal polynomial that is the nth power of a linear polynomial corresponding to u. When we blossom that curve, we are widening our point of view to include all of the surrounding polynomials that are products of n linear factors. The n arguments of the blossom correspond to those n factors.

This algebraic perspective is particularly valuable when we try to understand what the range of the blossom looks like sitting inside the object space. When $n = 2$, the range of the blossom f is the parabola F itself together with its exterior. The points in the interior of the parabola correspond to the polynomials in $H_1^2[S,T]$ that don't split into linear factors over the reals, in particular, the ones that have conjugate complex roots. When $n = 3$, the geometry is more interesting. A nondegenerate cubic polynomial curve is a twisted space curve sitting in a 3-space. The range of its blossom is a 3-dimensional variety that spirals away from that space curve. The *umbilic bracelet* of catastrophe theory, neatly analyzed by E. C. Zeeman [7], is a double covering of this variety.

All of this theory works equally well for surfaces; we just start off with three formal variables R, S, and T, instead of two. The prototypical Bézier surface of degree n is the map $\Psi_n: H_1^1[R,S,T] \to H_1^n[R,S,T]$ given by raising to the nth power; its blossom ψ_n is just multiplication. The main new phenomenon that arises for surfaces is that the range of the blossom is smaller. For example, the affine span of the quadratic surface Ψ_2 is the entire space $H_1^2[R,S,T]$, which is 5-dimensional. But the blossom ψ_2 takes two arguments, each with two degrees of freedom, and hence has a range that is at most 4-dimensional. This dimensional discrepancy reflects the fact that homogeneous polynomials in more than two variables generally don't split into linear factors, even over the complex numbers.

7. Tensor product surfaces

There are two types of surfaces defined by polynomials that are used in computer aided geometric design, and we have been focusing entirely on one of those two types so far. In this section, we shall redress this imbalance.

Let $F: \mathbf{R} \times \mathbf{R} \to O$ be a polynomial function; that is, each coordinate of the point $F(u,v)$ is given by a polynomial in the two real variables u and v. There are two different types of degree bound that we could put on these coordinate polynomials [2]. In a ***tensor product surface*** of degree $(m;n)$, the coordinates of $F(u,v)$ are required to have degree no higher than m in the variable u and degree no higher than n in the variable v separately. In a ***triangular patch surface*** of degree n, the coordinates of $F(u,v)$ are required to have total degree no higher than n in the variables u and v jointly.

Triangular patch surfaces are precisely what we have been calling simply ***polynomial surfaces***. That is, if we place a bound on the total degree, the resulting condition is independent of the choice of basis for the domain space $P = \mathbf{R} \times \mathbf{R}$, and we can reinterpret F as a polynomial function $F(u,v) = F(\mathbf{u})$ defined on a two-dimensional affine space P.

For a tensor product surface of degree $(m;n)$, we place separate bounds on the degrees in u and in v. The resulting condition on F does depend on the choice of basis in the domain plane. Indeed, a tensor product surface is often best thought of in Curried fashion as a curve of curves, that is, as a function $F: \mathbf{R} \to (\mathbf{R} \to O)$. The following paragraph states, without proofs, the basic results about blossoming tensor product surfaces. (By the way, the name "bipolynomial surface" might be better than "tensor product surface"; note that, when $m = n = 3$, such surfaces are already called "bicubics".)

The blossom f of a tensor product surface F of degree $(m;n)$ is an $(m+n)$-affine function $f: \mathbf{R}^m \times \mathbf{R}^n \to O$ that is symmetric in its first m arguments, symmetric in its last n arguments, and satisfies the identity $F(u,v) = f(u,\ldots,u;v,\ldots,v)$. Another variant of the Blossoming Principle tells us that every tensor product surface has a unique blossom. If $[p,q] \times [r,s]$ is a reference rectangle for the domain plane $\mathbf{R} \times \mathbf{R}$, the $(m+1)(n+1)$ Bézier points of the surface F are the blossom values

$$f(\underbrace{p,\ldots,p}_{m-i},\underbrace{q,\ldots,q}_{i};\underbrace{r,\ldots,r}_{n-j},\underbrace{s,\ldots,s}_{j})$$

for i in $[0,m]$ and j in $[0,n]$. The de Casteljau Algorithm for a tensor product surface computes the blossom value $f(u_1,\ldots,u_m;v_1,\ldots,v_n)$ by performing $m+n$ stages of linear interpolations, with each blossom argument providing the ratio for one of these stages. The stages can be performed in any order.

8. Degree raising

There is one final property of the Bézier theory that is worth reconsidering from the multiaffine point of view: degree raising. For example, consider once again the standard parabola $F(u) = (u, u^2)$. This parabola is a polynomial curve of degree two, so it has a biaffine blossom $f(u,v)$. But we can also consider $F(u)$ to be a degenerate example of a cubic polynomial curve. Viewed in this light, the blossom of $F(u)$ is a triaffine function, say $g(u,v,w)$. We know that $F(u) = f(u,u) = g(u,u,u)$, of course. Can we easily compute the other values of g?

Indeed we can; we have

$$g(u,v,w) = \frac{f(u,v) + f(v,w) + f(w,u)}{3}.$$

To prove this claim, it is enough to note that the right-hand side is a symmetric, triaffine function whose diagonal agrees with F. For arbitrary degree n and an arbitrary parameter space P, we have

$$g(\mathbf{u}_1,\ldots,\mathbf{u}_{n+1}) = \frac{1}{n+1}\sum_{i=1}^{n+1} f(\mathbf{u}_1,\ldots,\hat{\mathbf{u}}_i,\ldots,\mathbf{u}_n),$$

where the hat over the argument $\mathbf{u}_i$ indicates that the ith argument is omitted.

From this result, we can verify the well-known formula that, given the Bézier points of an n-ic curve F, tells what the Bézier points of that same curve would be if it were viewed as a degenerate case of a curve of degree $n+1$. Let f be the n-blossom of F and let g be its $(n+1)$-blossom. Then, for i in $[0, n+1]$, we have

$$g(\underbrace{s,\ldots,s}_{n+1-i},\underbrace{t,\ldots,t}_{i}) = \left(1 - \frac{i}{n+1}\right) f(\underbrace{s,\ldots,s}_{n-i},\underbrace{t,\ldots,t}_{i}) + \frac{i}{n+1} f(\underbrace{s,\ldots,s}_{n+1-i},\underbrace{t,\ldots,t}_{i-1}).$$

9. On blossoms and joints

Blossoming is a useful tool for working with polynomial curves and surfaces; but a single polynomial curve or surface is pretty simple no matter how you think about it. The real power of the multiaffine approach becomes apparent only when it is applied to splines. In this section, we shall begin to consider spline curves.

Suppose that $\mathbf{R}$, our parameter space, has been partitioned into intervals by a certain increasing sequence of points $\{t_i\}$, which can be finite, infinite, or bi-infinite. For each i, suppose in addition that we are given a certain polynomial curve $F_i \colon \mathbf{R} \to O$ of degree no greater than n. We can assemble a spline curve

F of degree no greater than n out of the given n-ic curves F_i by specifying that $F(u) := F_i(u)$ whenever $t_i < u < t_{i+1}$. Note that, from a formal point of view, a spline curve is the same thing as a piecewise polynomial curve. The difference is that, when dealing with spline curves, we usually enforce some sort of continuity conditions at the joints between adjacent segments.

If $F\colon (r,s) \to O$ and $G\colon (s,t) \to O$ are polynomial curves defined on adjacent intervals in $\mathbf{R}$, there are two different types of continuity constraints that we could impose on the joint at s between F and G. This joint is called *parametrically continuous of order k*, or C^k continuous, if F and G agree to kth order at s. Parametric continuity is the natural notion to use in situations where the parameterizations of the curves are important as well as their shapes. In many case in computer aided geometric design, however, the particular parameterizations of the curves or surfaces are not meaningful. In such cases, the notion of *geometric continuity* is more natural, in which we allow ourselves to reparameterize F and G before checking to see that they agree to kth order [4].

Parametric continuity works out quite neatly from the multiaffine point of view, and it is the only type of continuity constraint that we will deal with in this paper. In particular, the order of parametric continuity at a joint between two n-ic curve segments corresponds in a simple way to the extent to which the blossoms of those two segments happen to agree.

Proposition 1. *Two n-ic polynomial curves $F\colon (r,s) \to O$ and $G\colon (s,t) \to O$ join with C^k continuity at s if and only if the n-blossoms f of F and g of G satisfy the identity*

$$f(u_1,\ldots,u_k,\underbrace{s,\ldots,s}_{n-k}) = g(u_1,\ldots,u_k,\underbrace{s,\ldots,s}_{n-k}),$$

that is, if and only if f and g agree on all argument bags (multisets) that include at least $n-k$ copies of s.

Intuitively speaking, each new order of parametric continuity means that one more blossom argument can be varied away from s without destroying the agreement between the blossoms of the joining curves. Fig. 6 shows examples of cubic segments joining with various levels of continuity. Note that, when the joint has C^k continuity, the portion of the diagram on which the labels given by the two blossoms f and g agree forms a de Casteljau Diagram with k shells.

In one direction, we already know the proof of Proposition 1. Section 5 showed that the derivatives of F up through kth order can be evaluated by varying at most k of the arguments of the blossom f away from the joint point s. If the blossoms f and g agree on all such argument bags, then F and G must agree to kth order.

The reverse direction is only slightly harder. Assume that $F^{(i)}(s) = G^{(i)}(s)$ for $0 \le i \le k$, where

$$F^{(i)}(s) = n^{\underline{i}} \sum_{0\le j\le i} \binom{i}{j} (-)^{i-j} f(\underbrace{s+1,\ldots,s+1}_{j},\underbrace{s,\ldots,s}_{n-j})$$

and similarly for $G^{(i)}(s)$. By induction on i, we can easily conclude that

$$f(\underbrace{s+1,\ldots,s+1}_{i},\underbrace{s,\ldots,s}_{n-i}) = g(\underbrace{s+1,\ldots,s+1}_{i},\underbrace{s,\ldots,s}_{n-i})$$

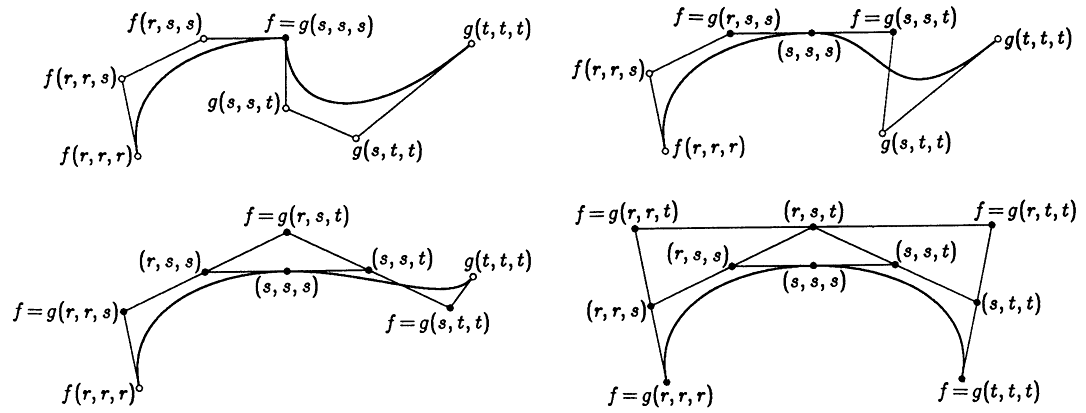


Fig. 6. Two cubic curves joining with C^0, C^1, C^2, and C^3 continuity

for $0 \leq i \leq k$. Note that the points on the left-hand sides of these $k+1$ equations are precisely the Bézier points with respect to the reference interval $[s, s+1]$ of the polynomial curve of degree k whose k-blossom is given by

$$(u_1, \ldots, u_k) \mapsto f(u_1, \ldots, u_k, \underbrace{s, \ldots, s}_{n-k}).$$

Similarly, the points on the right-hand sides are the Bézier points of the curve whose blossom is given by

$$(u_1, \ldots, u_k) \mapsto g(u_1, \ldots, u_k, \underbrace{s, \ldots, s}_{n-k}).$$

Since these two nameless curves have the same Bézier points, they must be the same curve. Hence, they must have the same blossom, which is a fancy way of restating the claim about f and g that we wanted to prove.

Proposition 1 is even simpler when restated in the language of knots and multiplicities [3]. Suppose that the n-ic curves $F\colon (r,s) \to O$ and $G\colon (s,t) \to O$ have a C^k joint at s. The parameter value s at which the joint occurs is called a ***knot***, and the number $m = n - k$ of derivatives that are allowed to be discontinuous as we pass through that knot is called the ***multiplicity*** of the knot. This convention works out well because a knot of multiplicity m behaves very much like the limit of a cluster of m closely spaced simple knots that have coalesced. Rephrasing Proposition 1, we observe that the multiplicity of a knot is precisely the number of blossom arguments that must be kept fixed at that knot in order to guarantee that the blossoms of the two joining curves will agree.

These conventions about knots and multiplicities allow us to specify all of the boundary conditions of a spline curve by means of one sequence of points $\{t_i\}$, called the ***knot sequence***, which consists of all of the knots repeated according to their multiplicities and sorted into nondecreasing order. If the common value $t_{i+1} = t_{i+2} = \cdots = t_{i+m}$ is a knot of multiplicity m, then the curve $F_i\colon (t_i, t_{i+1}) \to O$ joins the curve $F_{i+m}\colon (t_{i+m}, t_{i+m+1}) \to O$ at that knot with C^{n-m} continuity. The knot sequence is just what we need to understand the extent to which the blossoms of two segments of a spline curve are guaranteed to agree. In particular, we can show that the argument bags on which lots of blossoms must agree are the bags that contain substrings of the knot sequence.

Let $\{t_i\}$ be the knot sequence of an n-ic spline curve $F(u)$. For each i such that $t_i < t_{i+1}$, the behavior of $F(u)$ for u in (t_i, t_{i+1}) is determined by the n-ic polynomial curve $F_i(u)$, and that curve has an n-blossom $f_i(u_1, \ldots, u_n)$. For simplicity at first, let use suppose that there are no multiple knots. Consider the values of the various blossoms when applied to an argument bag of the form $B = \{t_{i+1}, \ldots, t_{i+m}, u_{m+1}, \ldots, u_n\}$. Since the bag B includes the simple knot t_{i+1}, the blossoms f_i and f_{i+1} must agree on B; since B also includes t_{i+2}, the blossoms f_{i+1} and f_{i+2} must also agree on B. We deduce that, in fact, all of the blossoms from f_i through f_{i+m} must agree on B. Furthermore, this conclusion still holds even if we allow some or all of the m simple knots t_{i+1} through t_{i+m} to coalesce into knots of higher multiplicity. For example, if $t_{i+1} = t_{i+2}$ is a double knot, then there is no segment F_{i+1} and no associated blossom f_{i+1}. But the adjacent blossoms f_i and f_{i+2} must still agree on the argument bag B because it includes two copies of the double knot $t_{i+1} = t_{i+2}$. This sort of reasoning demonstrates the following.

Proposition 2. *Let the nonempty interval (t_i, t_{i+1}) be the domain of one segment of a spline curve, and let the nonempty interval (t_j, t_{j+1}) be the domain of a later segment. Then, the blossoms f_i and f_j of these two segments must agree on any argument bag that includes all of the intervening knots as often as their multiplicity, that is, on any bag that includes $\{t_{i+1}, \ldots, t_j\}$ as a sub-bag. All of the blossoms of intermediate segments, if any, will also agree with f_i and f_j on such bags.*

To take full advantage of this proposition, let us extend our notation by allowing ourselves to write sets of indices as well as single indices as subscripts on f: if we know that all of the blossoms f_i for i in the non-empty set S must return the same value on the argument bag $\{u_1, \ldots, u_n\}$, we shall denote that common value by the term $f_S(u_1, \ldots, u_n)$. In particular, a common value of f_i and f_j as guaranteed by Proposition 2 would be denoted $f_{\{i,j\}}(t_{i+1}, \ldots, t_j, u_{j-i+1}, \ldots, u_n)$. The rules for manipulating with these new f_S terms are straightforward. For example, if we know the points $f_S(x, u_2, \ldots, u_n)$ and $f_T(y, u_2, \ldots, u_n)$ and if the sets S and T are not disjoint, we can linearly interpolate (or extrapolate, as appropriate) to compute the point $f_{S \cap T}(z, u_2, \ldots, u_n)$ for any z.

One problem with the f_S notation is that our indexing scheme for the segments of a spline is rather complex. The curve segment F_i is the one that the spline curve F follows from knot t_i until knot t_{i+1}, if those two knots are distinct. If $t_i = t_{i+1}$, there is no segment F_i, and hence we really shouldn't include i in the set S of an f_S term. To avoid having to worry about which indices are valid, we shall extend our notation once again by allowing ourselves to write f_I, where I is an interval in the domain space $\mathbf{R}$. In particular, we define the term $f_I(u_1, \ldots, u_n)$ to denote $f_S(u_1, \ldots, u_n)$ where $S = \{i \mid (t_i, t_{i+1}) \cap I \neq \emptyset\}$. In order to write down a term of the form $f_I(u_1, \ldots, u_n)$, we must show that the various f_i for i in S will agree on the argument bag $\{u_1, \ldots, u_n\}$, of course. In a term of the form f_I, we shall refer to I as the *validity interval.*

The result of Proposition 2 looks particularly simple when written in terms of validity intervals: it tells us that the term $f_{(t_i, t_{j+1})}(t_{i+1}, \ldots, t_j, u_{j-i+1}, \ldots, u_n)$ is well-defined whenever its validity interval is nonempty, that is, whenever $t_i < t_{j+1}$. This result sets the stage for a multiaffine attack on B-spline curves.

10. The blossoms of B-spline curves

When is Proposition 2 is pushed as far as it can go? That is, what are the argument bags on which the widest range of segment blossoms must agree? The extreme case occurs when $j = i + n$, in the terms $f_{(t_i,t_{i+n+1})}(t_{i+1}, \ldots, t_{i+n})$. Note that there are no free arguments u_j left in such a term. What we have is a single point on which a string of $n + 1$ consecutive blossoms are known to agree; let's call it an *extremal point.* (If multiple knots are involved, some of the $n+1$ blossoms that agree on an extremal point may have degenerated into nonexistence.) An amazingly simple thing happens at this point in the theory of B-spline curves—be warned that this doesn't work for spline surfaces: the extremal points form a basis for the space of all spline curves with $\{t_i\}$ as their knot sequence. That is, not only can we compute the extremal points from the spline curve, we can also go backward and compute a unique corresponding spline curve from arbitrarily specified values for the extremal points.

Proposition 3. *Let n be nonnegative and let $\{t_i\}$ be a knot sequence that doesn't include any knots of multiplicity greater than $n + 1$. Spline curves $F\colon \mathbf{R} \to O$ of degree n with the knot sequence $\{t_i\}$ are in one-to-one correspondence with sequences of points $\{\mathbf{x}_i\}$ in O by means of the formula $\mathbf{x}_i = f_{(t_i,t_{i+n+1})}(t_{i+1}, \ldots, t_{i+n})$.*

Proposition 4. *In fact, the points $\mathbf{x}_i = f_{(t_i,t_{i+n+1})}(t_{i+1}, \ldots, t_{i+n})$ are precisely the de Boor points by which a B-spline curve is controlled in the standard theory. Furthermore, the validity interval of each $\mathbf{x}_i$ is precisely the region of parameter space over which that de Boor point influences the value of the spline curve.*

Proposition 4 is proved by tracing through the de Boor Algorithm and expressing all of the intermediate results as blossom values, hence verifying that the standard theory and the multiaffine approach do indeed compute the same answers. Once Proposition 4 is verified, the cowardly way to prove Proposition 3 is to appeal to the corresponding result in the standard theory. It is also possible to prove Proposition 3 from scratch by reasoning about blossoms. Unfortunately, we don't have space for any of these proofs.

11. Overloading the notation for a spline blossom

When working with the blossom f of a spline curve F, it is irritating to have to keep writing the validity intervals as subscripts on f all the time. Indeed, the labels in Figs. 2 and 3 were spline blossom values without validity intervals. The way to avoid writing the subscripts is to invent a scheme for deducing the proper subscript from the points in the bag of arguments. Note that the notation for the spline curve F itself involves just such a deduction scheme. To interpret the term $F(u)$, we first deduce the proper i by determining which domain interval (t_i, t_{i+1}) contains u; then we evaluate $F_i(u)$. We shall refer to the adoption of a scheme for deducing the proper subscript as *overloading.* Note that the non-overloaded notation $F_i(u)$ is more powerful, since it allows us to evaluate any segment of the spline at any point, even points where $u < t_i$ or $u > t_{i+1}$. But the overloaded notation $F(u)$ is more convenient. Achieving the same convenience by overloading the spline blossom f is rather delicate, because there are n arguments instead of just one.

Starting at the beginning, note that we can't possibly make overloading work when $n = 0$, since the blossom doesn't have any arguments to examine. Indeed, in the case $n = 0$, the validity intervals convey just the right information. A de Boor point of a zeroth degree spline curve (a "constant spline") $F(u)$ has a label of the form $f_{(t_i,t_{i+1})}()$, and we have $F(u) = f_{(t_i,t_{i+1})}()$ whenever u is in (t_i, t_{i+1}).

When $n = 1$, each affine segment $F_i(u)$ of the spline is equal to its blossom $f_i(u)$, so we can overload the spline blossom in the same way that we overload the spline itself. The only problems arise at knots of multiplicity at least 2.

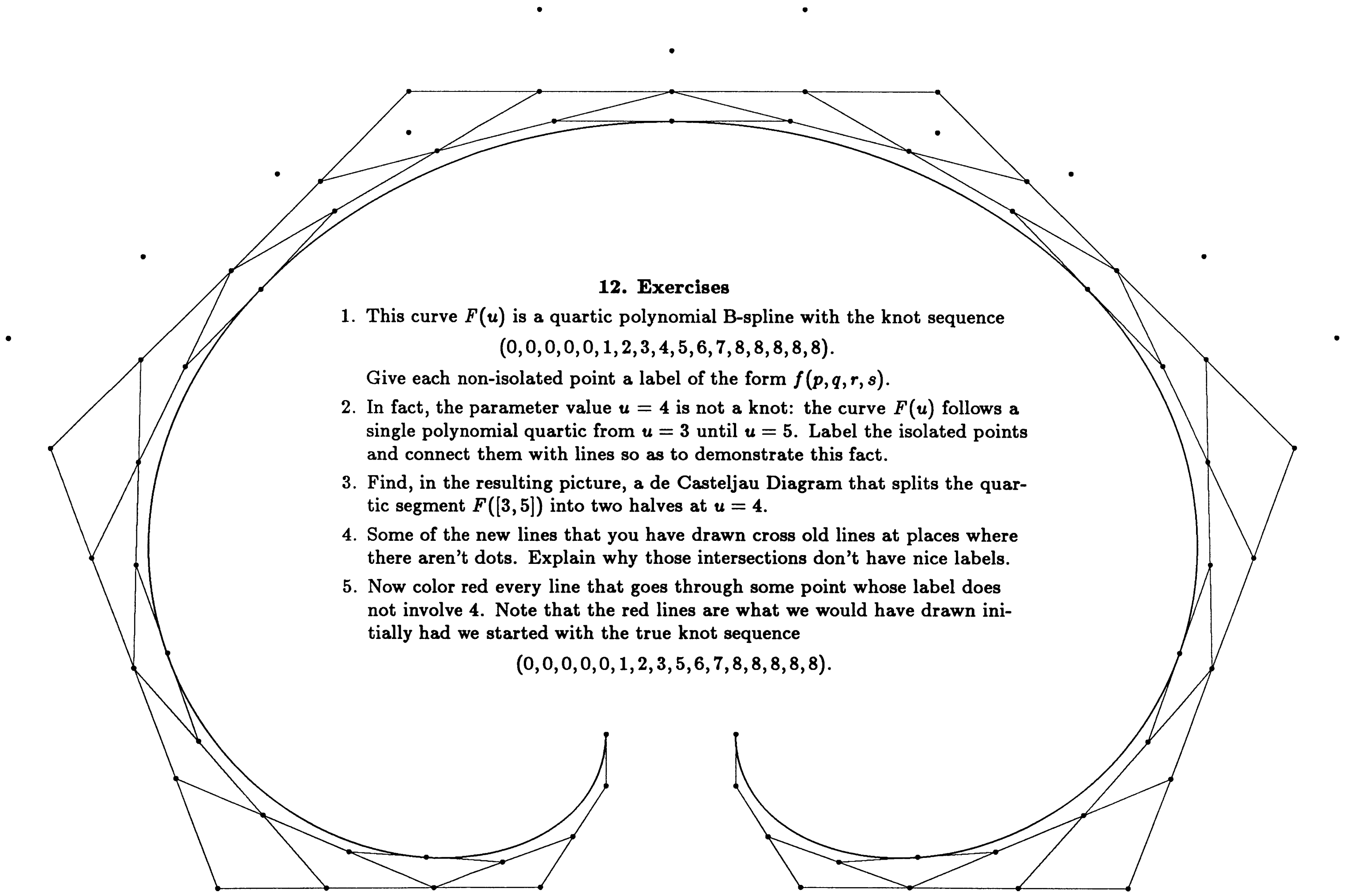
When $n \geq 2$, there are two reasonable conventions that one could adopt, which we shall refer to as *tame* and *wild* overloading. In each case, the heart of the convention is a rule for deciding, given a bag of arguments $\{u_j\}$, which parameter intervals (t_i, t_{i+1}) should be *candidates* for use when assigning a value to the blossom expression $f(u_1, \ldots, u_n)$. Given a rule for candidacy, a blossom expression $f(u_1, \ldots, u_n)$ is well-defined if and only if there is at least one candidate interval and all of the values $f_{(t_i,t_{i+1})}(u_1, \ldots, u_n)$ for candidate intervals (t_i, t_{i+1}) are the same.

In *tame overloading*, the interval (t_i, t_{i+1}) is a candidate for use when defining $f(u_1, \ldots, u_n)$ whenever the closed intervals $[t_i, t_{i+1}]$ and $[\min\{u_j\}, \max\{u_j\}]$ intersect. Appealing to Proposition 2, we can deduce that the expression $f(u_1, \ldots, u_n)$ is well-defined under tame overloading if and only if every knot in the closed interval $[\min\{u_j\}, \max\{u_j\}]$ is included in the bag $\{u_j\}$ at least as often as it is a knot.

In *wild overloading*, there are two cases. If the blossom arguments u_j are not all equal, then the interval (t_i, t_{i+1}) is a candidate for defining the term $f(u_1, \ldots, u_n)$ only when the open intervals (t_i, t_{i+1}) and $(\min\{u_j\}, \max\{u_j\})$ intersect. If all of the u_j are equal, the interval (t_i, t_{i+1}) is a candidate whenever the common value of the u_j's lies in $[t_i, t_{i+1}]$. Appealing once again to Proposition 2, we deduce that the expression $f(u_1, \ldots, u_n)$ is well-defined under wild overloading if and only if either $\min\{u_j\} < \max\{u_j\}$ and every knot in the interval $(\min\{u_j\}, \max\{u_j\})$ is included among the u_j at least as often as it is a knot; or $\min\{u_j\} = \max\{u_j\}$ and the multiplicity of that common value as a knot does not exceed n.

Wild overloading is more powerful, as we can see by considering the labels in Fig. 2. Note that the label $g(0,1,1)$ in that figure is not well-defined under tame overloading, since 0 appears only once as an argument, while 0 appears four times as a knot. Indeed, the presence of one 0 among the arguments of g tells us that $g(0,1,1)$ should lie in the osculating plane to $G(u)$ at $u = 0$. But $G(u)$ is not even C^0 at $u = 0$; hence, it has two different osculating planes there, one associated with small positive u and the other with small negative u. The wild overloading convention makes $g(0,1,1)$ well-defined by choosing the positive plane, on the grounds that the other argument values are all nonnegative and at least one of them is positive. (Note that the label $g(0,0,0)$ in Fig. 2 isn't well-defined even under wild overloading; this is as it should be, since, strictly speaking, $G(0)$ isn't well-defined either.)

But the power of wild overloading can lead to confusion in some cases. Consider a quadratic spline curve $F(u)$ with knot sequence $(\ldots, 0, 1, 2, 2, 3, 4, \ldots)$. The term $f(1,2)$ is well-defined under wild overloading, with the value $f_{(0,2)}(1,2)$. Similarly, the term $f(2,3)$ is well-defined, with the value $f_{(2,4)}(2,3)$. Furthermore, the terms $f(1,2)$ and $f(2,3)$ have all but one argument in common. We might be tempted to compute $f(x,2)$ for $1 < x < 3$ by interpolating between $f(1,2)$ and $(2,3)$. But the validity intervals of $f(1,2)$ and $f(2,3)$ are disjoint; hence we have no right to do so. Such apparent failures of multiaffineness can't happen under tame overloading. In particular, suppose that $f(x, u_2, \ldots, u_n)$ and $f(y, u_2, \ldots, u_n)$ are both well-defined under tame overloading. Then, for any z between x and y, the term $f(z, u_2, \ldots, u_n)$ will also be well-defined under tame overloading, and the corresponding point can be found by interpolation.



12. Exercises

1. This curve $F(u)$ is a quartic polynomial B-spline with the knot sequence
$$(0,0,0,0,0,1,2,3,4,5,6,7,8,8,8,8,8).$$
Give each non-isolated point a label of the form $f(p,q,r,s)$.
2. In fact, the parameter value $u = 4$ is not a knot: the curve $F(u)$ follows a single polynomial quartic from $u = 3$ until $u = 5$. Label the isolated points and connect them with lines so as to demonstrate this fact.
3. Find, in the resulting picture, a de Casteljau Diagram that splits the quartic segment $F([3,5])$ into two halves at $u = 4$.
4. Some of the new lines that you have drawn cross old lines at places where there aren't dots. Explain why those intersections don't have nice labels.
5. Now color red every line that goes through some point whose label does not involve 4. Note that the red lines are what we would have drawn initially had we started with the true knot sequence
$$(0,0,0,0,0,1,2,3,5,6,7,8,8,8,8,8).$$

13. Open questions

This paper has explored the application of blossoming to individual polynomial curves and surfaces and to polynomial spline curves with parametric continuity. In those simple situations, we found that blossoms and the multiaffine point of view helped to clarify and simplify the existing theory. It will be interesting to learn if blossoming can be extended into more subtle areas. First, can blossoming be usefully applied to spline curves with geometric, rather than parametric, continuity? Second, what about spline surfaces? Of course, a tensor product spline surface is just a spline curve of spline curves; it is straightforward to blossom such a thing. But a spline surface built by assembling triangular patches of polynomial surfaces is a subtler beast. Third, the projection of a polynomial curve or surface F down onto a hyperplane in O is a rational curve or surface; the projection of the multiaffine blossom f of F would be a multiprojective function. What can be said about the multiprojective blossoms of rational curves and surfaces?

Acknowledgments

I want to thank Greg Nelson, Jim Saxe, and Jorge Stolfi for many helpful conversations during the gestation of these ideas.

References

[1] Wolfgang Böhm. "Generating the Bézier points of triangular splines." In Robert E. Barnhill and Wolfgang Böhm, editors, ***Surfaces in Computer Aided Geometric Design***, pages 77–91. North-Holland, 1983.

[2] Wolfgang Böhm, Gerald Farin, and Jürgen Kahmann. "A survey of curve and surface methods in CAGD." ***Computer Aided Geometric Design*** **1**, 1 (July 1984), 1–60.

[3] Carl de Boor. ***A Practical Guide to Splines.*** Springer-Verlag, 1978.

[4] Anthony D. DeRose. ***Geometric Continuity: A Parametrization Independent Measure of Continuity for Computer Aided Geometric Design.*** PhD thesis, Univeristy of California at Berkeley, 1985. Also available from Berkeley as technical report UCB/CSD 86/255.

[5] Ronald N. Goldman. "Subdivision algorithms for Bézier triangles." ***Computer-Aided Design*** **15**, 3 (May 1983), 159–166. See also the letter to the editor from Wolfgang Böhm and Gerald Farin in **15**, 5 (September 1983), 260–261.

[6] P. Sablonniére. "Spline and Bézier polygons associated with a polynomial spline curve." ***Computer-Aided Design*** **10**, 4 (July 1978), 257–261.

[7] E. C. Zeeman. "The umbilic bracelet and the double-cusp catastrophe." In P. Hilton, editor, ***Structural Stability, the Theory of Catastrophes, and Applications in the Sciences***, published as volume 525 of ***Lecture Notes in Mathematics***, pages 328–366. Springer-Verlag, 1976.

Author's biography

Lyle Ramshaw is a Principal Software Engineer at the Systems Research Center of the Digital Equipment Corporation in Palo Alto, California. Before moving to DEC in 1984, he worked at the Palo Alto Research Center of the Xerox Corporation. He received his PhD from Stanford University in 1979, working under Prof. Donald E. Knuth on formalizing the analysis of algorithms.