

## The Stable Evaluation of Multivariate Simplex Splines

By Thomas A. Grandine\*

**Abstract.** This paper gives a general method for the stable evaluation of multivariate simplex splines, based on the well-known recurrence relation of Micchelli [12]. This paper deals with two problems which arise in the implementation of the recurrence relation. First, the coefficients in the recurrence are shown to be efficiently computable via the dual simplex method of linear programming. Secondly, the problem of evaluation along mesh boundaries is discussed in detail.

**1. Introduction.** In de Boor [2], the multivariate simplex spline  $M(\cdot | t_0, \dots, t_n)$  was defined to be

$$(1) \quad M(x | t_0, \dots, t_n) := \frac{\text{vol}_{n-m}\{(P^{-1}x) \cap [t_0, \dots, t_n]\}}{\text{vol}_n[t_0, \dots, t_n]}, \quad x \in \mathbf{R}^m.$$

In this definition,  $t_0, \dots, t_n$  are points in  $\mathbf{R}^n$ ,  $[A]$  is the convex hull of  $A$ , and  $P$  is the canonical projector

$$P: \mathbf{R}^n \rightarrow \mathbf{R}^m: x \mapsto (x(i))_1^m.$$

In addition,  $\text{vol}_k(A)$  is the  $k$ -dimensional volume of the set  $A$ .

The simplex spline is, in general, a nonnegative piecewise polynomial function of degree  $\leq n - m$ , supported in  $[Pt_0, \dots, Pt_n]$ , and it is in  $C^{n-m-1}$  provided that the points  $Pt_0, \dots, Pt_n$  are in "general position." "General position" means that any  $m + 1$  of the points  $Pt_0, \dots, Pt_n$  are in general position, i.e., they are affinely independent. For a proof of this result, see, for example, [4].

To compute the value of a simplex spline at a point, all that is required is to know which polynomial coincides with the spline at the given point. As the following example illustrates, this is far from trivial.

*Example 1.* Consider the knots  $(1, 0)$ ,  $(1, 1)$ ,  $(0, 1)$ ,  $(-1, 0)$ ,  $(-1, -1)$ ,  $(0, -1)$  for a bivariate cubic simplex spline. Then the spline will be made up, in its region of support, of 24 separate cubic polynomials pieced together as in Figure 1. Given an arbitrary point  $x$  in the plane, the mere determination of which of the 24 regions in which  $x$  resides is a very difficult problem, even for this specific collection of knots.

---

Received January 30, 1986; revised July 30, 1986 and October 27, 1986.

1980 *Mathematics Subject Classification* (1985 Revision). Primary 41A15, 65D07.

*Key words and phrases.* B-spline, simplex spline, multivariate, recurrence relation, linear programming, simplex method.

\* Sponsored by the United States Army under Contract No. DAAG29-80-C-0041.

©1988 American Mathematical Society  
0025-5718/88 \$1.00 + \$.25 per page

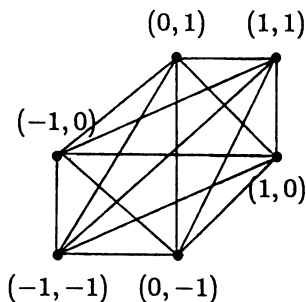


FIGURE 1

It is important to realize that humans, gifted with eyes, are able to view Figure 1 and infer all kinds of spatial and geometric information about the problem which a computer can only determine by performing various algebraic computations. Even determining which of the vertices are extreme points of the region is not a simple undertaking (in Figure 1, all points are extreme). Furthermore, once the location of  $x$  is actually determined, there still remains the problem of knowing just what polynomial to evaluate.

A better approach might involve the use of recurrence relations, the multivariate analog of what de Boor does in his routine BSPLVB, in [3]. In [13], C. A. Micchelli proposed a more convenient definition of a multivariate simplex spline as the *distribution* on  $C_0^\infty(\mathbf{R}^m)$  given by

$$(2) \quad \int_{\mathbf{R}^m} M(\cdot | t_0, \dots, t_n) f := n! \int_{[t_0, \dots, t_n]} f \circ P.$$

This definition makes sense even if  $t_0, \dots, t_n$  are not in general position. Using this definition, Micchelli was able to prove *recurrence relations* for these multivariate simplex splines. These relations are given in Theorem 1.

**THEOREM 1.** *If  $x = \sum_{i=0}^n \alpha_i P t_i$ , with  $\sum_{i=0}^n \alpha_i = 1$ , then*

$$(3) \quad M(x | t_0, \dots, t_n) = \frac{n}{n - m} \sum_{i=0}^n \alpha_i M(x | t_0, \dots, t_{i-1}, t_{i+1}, \dots, t_n).$$

*Furthermore, if  $y = \sum_{i=0}^n \alpha_i P t_i$ , with  $\sum_{i=0}^n \alpha_i = 0$ , then*

$$(4) \quad D_y M(\cdot | t_0, \dots, t_n) = n \sum_{i=0}^n \alpha_i M(\cdot | t_0, \dots, t_{i-1}, t_{i+1}, \dots, t_n).$$

*Proof.* The proof of this theorem has been given in various forms in a number of places, including [12], [9], [8], and [5].

Since all the simplex splines appearing on the right-hand side of the recurrence relation (3) are of degree one less than the simplex spline appearing on the left-hand side, the value of any simplex spline at a point can be written, using (3) recursively, as a linear combination of piecewise constant simplex splines. The evaluation of a piecewise constant simplex spline involves nothing more than computing the  $m$ -dimensional volume of a certain simplex.

On the face of it, this is a straightforward task. It becomes somewhat more difficult when the question arises of just how the recurrence relation should be implemented. One thing worth noting is that the recurrence relations do not depend upon the actual location of the points  $t_0, \dots, t_n$ , but only upon the location of the points  $Pt_0, \dots, Pt_n$ . A simplex spline is therefore determined uniquely by considering just the knots. Hence, if the points  $t_0, \dots, t_n$  are taken to be in  $\mathbf{R}^m$  instead of  $\mathbf{R}^n$ , then Theorem 1 is still true if the projector  $P$  is omitted. Hence, unless stated otherwise, the points  $t_0, \dots, t_n$  will henceforth be assumed to be in  $\mathbf{R}^m$ .

This remark, however, does not simplify the actual implementation of the recurrence relation. The real problem is this: Suppose that an arbitrary collection of knots, say  $t_0, \dots, t_n$  and a point  $x$ , have been given. How should the  $\alpha_i$  be chosen so that  $\sum_{i=0}^n \alpha_i t_i = x$  and  $\sum_{i=0}^n \alpha_i = 1$ ? Efficiency will demand that as many of the  $\alpha_i$  as possible be zero, for each nonzero  $\alpha_i$  means the evaluation of yet another lower-degree simplex spline to contribute to the quantity sought. Accuracy will demand that  $\alpha_i \geq 0$  for each  $i$ , since each simplex spline is nonnegative on its region of support, and it is desirable to eliminate the effects of cancellation when accumulating quantities of opposite sign. Essentially, a solution to the following problem is sought: Find  $\alpha_i, i = 0, \dots, n$ , such that

$$(5) \quad \sum_{i=0}^n \alpha_i t_i = x, \quad \sum_{i=0}^n \alpha_i = 1, \quad \alpha_i \geq 0, \quad i = 0, \dots, n.$$

The difficulty here lies in satisfying the nonnegativity constraints. Arbitrarily setting certain  $\alpha_i$  to zero, and then solving for the remaining  $\alpha_i$  does not guarantee that a solution to (5) will, in fact, be obtained.

**2. Computing the Recurrence Relation Coefficients.** Fortunately, there is a procedure for handling problems of this sort. More precisely, (5) is the description of a feasible region of a typical *linear programming problem*. Locating a point in such a region is a problem which has been studied in great detail for many years, and many ways of computing its solution are known. Among the most effective of these for small problems is the dual simplex method, and it will be the method of choice here.

The dual simplex method can be implemented via the Tucker tableau, which is described in detail in [10]. (Further information regarding the simplex method and linear programming can be found in [7].) The Tucker tableau for finding the feasible point described by (5) is the following:

$$(6) \quad \begin{matrix} & \alpha_0 & \alpha_1 & \cdots & \alpha_n & 1 \\ r_0 & \left( \begin{matrix} t_0(1) & t_1(1) & \cdots & t_n(1) & x(1) \\ t_0(2) & t_1(2) & \cdots & t_n(2) & x(2) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ t_0(m) & t_1(m) & \cdots & t_n(m) & x(m) \\ 1 & 1 & \cdots & 1 & 1 \end{matrix} \right) & & & & \end{matrix},$$

where the  $r_i$  represent the residuals in the  $m + 1$  equalities in (5) if the  $\alpha_i$  are set to zero. In general, the variables along the left side of the tableau are the *basic* or *bound* variables, while the variables along the top side of the tableau are the *nonbasic* or *free* variables for the problem. In any given state, the nonbasic variables are

assumed to have zero value, while the basic variables take on the appropriate residual value.

The dual simplex method is performed by exchanging basic and nonbasic variables. If the entry in the  $i$ th row and  $j$ th column of the tableau is given by  $t_{ij}$ , then the pivot rules for exchanging the  $k$ th basic variable with the  $l$ th nonbasic variable are:

$$\begin{aligned} t_{kl} &\rightarrow 1/t_{kl}, \\ t_{il} &\rightarrow -t_{il}/t_{kl}, \\ t_{kj} &\rightarrow t_{kj}/t_{kl}, \\ t_{ij} &\rightarrow t_{ij} - t_{il}t_{kj}/t_{kl}, \end{aligned} \quad i \neq k, j \neq l.$$

Once the  $r_i$  have been set to zero (by making them nonbasic), the equalities in (5) are satisfied. However, the inequality constraints may not be satisfied, and it may be necessary to exchange basic and nonbasic variables in order to make all the basic variables nonnegative. This is done by selecting a row whose basic variable has a negative value, selecting a column whose tableau entry in the selected row is also negative, and then exchanging the two variables. (Ties in the selection process may be broken by Bland's rule [1] or by lexicographic ordering [10]. Both methods are guaranteed to avoid cycling.) If such an exchange cannot be accomplished (because all entries in the selected row are nonnegative), an infeasible problem is indicated. This means that  $x$  lies outside the convex hull of the knots, and hence outside the region of support of the simplex spline. This condition can usually be detected fairly early in the computation.

*Example 2.* Consider once again the knots  $(1, 0)$ ,  $(1, 1)$ ,  $(0, 1)$ ,  $(-1, 0)$ ,  $(-1, -1)$ ,  $(0, -1)$  for a bivariate cubic simplex spline and the point  $x = (\frac{1}{2}, \frac{1}{4})$ . Setting up the Tucker Tableau for problem (5) results in

$$\begin{array}{c} \alpha_0 \quad \alpha_1 \quad \alpha_2 \quad \alpha_3 \quad \alpha_4 \quad \alpha_5 \quad 1 \\ r_0 \left( \begin{array}{ccccccc} 1 & 1 & 0 & -1 & -1 & 0 & \frac{1}{2} \\ 0 & 1 & 1 & 0 & -1 & -1 & \frac{1}{4} \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array} \right). \end{array}$$

Initially  $r_0$ ,  $r_1$ , and  $r_2$  must be set to zero. This is done first for  $r_0$ , by exchanging it with  $\alpha_0$ :

$$\begin{array}{c} r_0 \quad \alpha_1 \quad \alpha_2 \quad \alpha_3 \quad \alpha_4 \quad \alpha_5 \quad 1 \\ \alpha_0 \left( \begin{array}{ccccccc} 1 & 1 & 0 & -1 & -1 & 0 & \frac{1}{2} \\ 0 & 1 & 1 & 0 & -1 & -1 & \frac{1}{4} \\ -1 & 0 & 1 & 2 & 2 & 1 & \frac{1}{2} \end{array} \right). \end{array}$$

This tableau now corresponds to a solution in which  $r_0 = 0$ . Since this should be a permanent condition throughout the computation, it suffices to make  $r_0$  permanently nonbasic. This can be easily accomplished by merely deleting the first column of the tableau entirely, leaving

$$\begin{array}{c} \alpha_1 \quad \alpha_2 \quad \alpha_3 \quad \alpha_4 \quad \alpha_5 \quad 1 \\ \alpha_0 \left( \begin{array}{cccccc} 1 & 0 & -1 & -1 & 0 & \frac{1}{2} \\ 1 & 1 & 0 & -1 & -1 & \frac{1}{4} \\ 0 & 1 & 2 & 2 & 1 & \frac{1}{2} \end{array} \right). \end{array}$$

Now, the same thing can be done with  $r_1$ , and this yields

$$\begin{matrix} & \alpha_2 & \alpha_3 & \alpha_4 & \alpha_5 & 1 \\ \alpha_0 & \left( \begin{matrix} -1 & -1 & 0 & 1 & \frac{1}{4} \end{matrix} \right) \\ \alpha_1 & \left( \begin{matrix} 1 & 0 & -1 & -1 & \frac{1}{4} \end{matrix} \right) \\ r_2 & \left( \begin{matrix} 1 & 2 & 2 & 1 & \frac{1}{2} \end{matrix} \right) \end{matrix}$$

Again, but this time with  $r_2$ ,

$$\begin{matrix} & \alpha_3 & \alpha_4 & \alpha_5 & 1 \\ \alpha_0 & \left( \begin{matrix} 1 & 2 & 2 & \frac{3}{4} \end{matrix} \right) \\ \alpha_1 & \left( \begin{matrix} -2 & -3 & -2 & -\frac{1}{4} \end{matrix} \right) \\ \alpha_2 & \left( \begin{matrix} 2 & 2 & 1 & \frac{1}{2} \end{matrix} \right) \end{matrix}$$

However,  $\alpha_0 = \frac{3}{4}$ ,  $\alpha_1 = -\frac{1}{4}$ ,  $\alpha_2 = \frac{1}{2}$  does not satisfy the nonnegativity constraint, so  $\alpha_1$  and  $\alpha_3$  are exchanged:

$$\begin{matrix} & \alpha_1 & \alpha_4 & \alpha_5 & 1 \\ \alpha_0 & \left( \begin{matrix} \frac{1}{2} & \frac{1}{2} & 1 & \frac{5}{8} \end{matrix} \right) \\ \alpha_3 & \left( \begin{matrix} -\frac{1}{2} & \frac{3}{2} & 1 & \frac{1}{8} \end{matrix} \right) \\ \alpha_2 & \left( \begin{matrix} 1 & -1 & -1 & \frac{1}{4} \end{matrix} \right) \end{matrix}$$

$\alpha_0 = \frac{5}{8}$ ,  $\alpha_2 = \frac{1}{4}$ ,  $\alpha_3 = \frac{1}{8}$ , with  $\alpha_1 = \alpha_4 = \alpha_5 = 0$  is one of the many solutions to (5). Fortunately, the stability of the recurrence relation is unaffected by the choice of solution. It is sufficient to use whatever solution is first obtained by the simplex method. In addition, note that the simplex method will always produce only solutions which have at most  $m + 1$  nonzero components, a fact made possible by Carathéodory's Theorem. Hence, efficiency is optimized, in the sense of evaluating as few lower-degree simplex splines as possible, by this technique.

Of course, this is but one of many such problems that might have to be solved in the evaluation of a single spline. This is because each of the splines occurring on the right-hand side of the recurrence relation has, in general, an unknown value. Therefore, each of them must somehow be evaluated.

Fortunately, the Tucker tableau is well suited for this. It is possible to start with the tableau which solves (5) initially, and use it to obtain solutions to the subproblems. This is easy, for each subproblem can be viewed as being the original problem with the additional constraint  $\alpha_i = 0$ . All that is necessary to impose this constraint is to pivot  $\alpha_i$  to the top of the tableau, and then delete the corresponding column. This new tableau is then optimized. This produces a solution to the subproblem very quickly, often in just one pivot. Obviously, each subproblem may have its own associated subproblems, and these may be solved in the same way. By storing the tableaus on a stack, all of the subproblems down to the lowest levels can be solved in a highly efficient manner.

**3. Evaluation Along Grid Lines.** It would seem, at this point, that the difficulties involved in evaluating a simplex spline have been completely overcome. This is, quite unfortunately, not the case. The simplex spline is, after all, by definition (2) a distribution, and the recurrence relations treat it as such. Yet, here is an attempt to

compute a pointwise value for this distribution using a relation which may be true only in a distributional sense. It should not come as a surprise, then, when things go awry. Certainly, nothing can go wrong as long as only continuous splines are dealt with, but, unfortunately, the piecewise constant splines which are ultimately obtained on the right-hand side of the recurrence relations are discontinuous along the *grid lines*. A grid line is defined to be a set consisting of the convex hull of  $m$  or fewer points taken from the set of knots. A point  $x$  is said to lie on a grid line if it is a member of some such set. It is on the grid lines that one must be wary.

*Example 3.* Suppose one wishes to evaluate  $M(0, 0 | t_0, t_1, t_2, t_3)$ , where  $t_0 = (1, 1)$ ,  $t_1 = (-1, 1)$ ,  $t_2 = (-1, -1)$ , and  $t_3 = (1, -1)$ . Then  $(0, 0) = \frac{1}{2}t_0 + \frac{1}{2}t_2$ , and therefore

$$M(0, 0 | t_0, t_1, t_2, t_3) = \frac{3}{2}M(0, 0 | t_1, t_2, t_3) + \frac{3}{2}M(0, 0 | t_0, t_1, t_3).$$

But  $M(\cdot | t_1, t_2, t_3)$  and  $M(\cdot | t_0, t_1, t_3)$  are discontinuous at  $(0, 0)$ , so it is unclear whether to choose the interior or exterior limits as values for these splines. If interior limits are chosen, the computed value of the spline will be twice as large as the actual value. If exterior limits are chosen, the value will be zero, and this is obviously also incorrect.

In the papers where actual computations of simplex splines are performed ([6], [11], and [12]), this problem is not discussed. Somehow the numbers come out right in the end, but the method by which they were obtained is never actually described. There are many ways of attempting to circumvent this problem, but nearly all fail in some way, especially when the inexact nature of the arithmetic performed by the computer is taken into account. In general, there seems to me to be no reasonable way to handle this problem, but for smooth splines, there are a few things that might be tried.

The obvious approach is to prohibit the evaluation of splines on the grid lines. This is certainly the most sure-fire answer, and it is also a simple enough scheme to be easily implemented. All that needs to be done is to move the point at which to evaluate by some small amount, say  $\epsilon$ , whenever that point happens to be located on a grid line and try again. Higher-order splines are, in general, continuous, so this small change in the location of the point will make a very small change in the value of the spline. Because of roundoff error, however, it is impossible to tell when a point is actually "on" a grid line or merely "near" the grid line, for as the number of variables increases, the structure of the grid lines becomes increasingly complex. For example, in one variable, the grid lines consist only of the knots, while in two variables, the grid lines consist of the knots as well as the line segments joining the knots (see Figure 1). This increasingly complex structure not only makes the actual grid line checking more difficult, but it also necessarily complicates the strategy of trying to avoid the grid lines altogether.

In one variable, the problem is not so bad. Typically, the decision is made *a priori* to make all the piecewise constant splines either continuous from the left or continuous from the right. Then, when evaluation at a knot is desired, the value of the spline is set to zero if it is the left knot, for example, and nonzero if it is the right knot. Thus, for one variable at least, little needs to be done to clear up this nuisance.

Conveniently, such a plan of attack generalizes to more than one variable. A direction in  $\mathbf{R}^m$  is chosen (somewhat arbitrarily), and piecewise constant splines are evaluated according to the following rule: If  $x$  is in the interior of the region of

support, or if  $x$  is on the boundary of the region of support and the arbitrarily chosen direction points into the interior, the value of the piecewise constant spline shall be used; in all other cases the value of the spline shall be 0. In the univariate case, the grid lines consist only of the knots, so a simple check for equality suffices to determine whether or not a point lies on the boundary of the region of support. In theory, this rule eliminates the difficulty in the multivariate case, too. However, in the scheme discussed above, one never actually deals with the coordinates of a point, but only with its *barycentric coordinates* with respect to various collections of  $n - m + 1$  knots from the set of all  $n + 1$  knots. Because of roundoff error and the multiplicity of representations for any given point, the situation where a single point lies both on the exterior and the interior of a single region of support can exist, so the ambiguity about what to do persists. Thus, for multivariate splines an alternate approach must be taken. Its successful implementation depends on the following theorem.

**THEOREM 2.** *Let  $t_0, \dots, t_{m+1}$  be a collection of  $m + 2$  points in general position in  $\mathbf{R}^m$ . Let  $A := [t_0, \dots, t_{m+1}]$ , and let  $A_i := [t_0, \dots, t_{i-1}, t_{i+1}, \dots, t_{m+1}]$ . Then for all  $x \in A$ , with  $x$  not on any of the grid lines,  $x$  lies in exactly two of the  $A_i$ .*

*Proof.* The statement  $x \in A$  is equivalent to the statement that there exists a solution to the problem:

$$(7) \quad \sum_{i=0}^{m+1} \alpha_i t_i = x, \quad \sum_{i=0}^{m+1} \alpha_i = 1, \quad \alpha_i \geq 0, \quad i = 0, \dots, m + 1.$$

Ignoring for the moment the inequality constraint, (7) can be viewed as a linear system, namely

$$T\alpha = \beta,$$

where  $T$  is an  $(m + 1) \times (m + 2)$  matrix,  $\alpha$  is the vector whose individual components are the  $\alpha_i$ , and  $\beta$  is the vector obtained by adding the component 1 to the end of  $x$ .  $T$  clearly has rank  $m + 1$ , since the  $t_i$  are in general position. Thus, this linear system has a one-parameter family of solutions, say  $\alpha(s) := y + sz$ , where  $y, z \in \mathbf{R}^{m+1}$  and  $s \in \mathbf{R}$ . Now consider the inequality constraints,  $\alpha_i \geq 0, i = 0, \dots, m + 1$ . This is equivalent to  $y_i + sz_i \geq 0, i = 0, \dots, m + 1$ . Taken together, all these conditions define some interval  $S := [s_-, s_+]$  in which  $s$  must lie in order for  $\alpha(s)$  to satisfy the inequality constraints. Since  $x \in A$ , it is clear that  $S$  is nonempty. Furthermore, it is clear that  $\sum_{i=0}^{m+1} y_i = 1$  and  $\sum_{i=0}^{m+1} z_i = 0$ , for  $\sum_{i=0}^{m+1} \alpha_i = 1$ , independent of  $s$ . Since the solution cannot be unique, at least one of the  $z_i$  is nonzero. But  $\sum_{i=0}^{m+1} z_i = 0$ , so there must be at least two of the  $z_i$  nonzero and of opposite sign. When  $z_i$  is positive, a lower bound for  $s$  is obtained, while  $z_i$  negative gives an upper bound for  $s$ . Hence,  $S$  is a finite interval. Since  $\alpha(s)$  is a continuous function of  $s$ , no components of  $\alpha$  can have sign changes in  $S$ . Furthermore,  $s$  outside of  $S$  means that one or more components of  $\alpha(s)$  are negative there, hence must change sign on the boundary of  $S$ . Suppose  $\alpha_i(s_-) = 0$  and  $\alpha_j(s_+) = 0$ . Since  $x$  does not lie on a grid line,  $\alpha_i(s_-)$  is the only zero component of  $\alpha$  at  $s_-$ . Similarly,  $\alpha_j(s_+)$  is the only zero component of  $\alpha$  at  $s_+$ . But this says that  $x \in A_i$  and  $x \in A_j$ . Since  $\alpha(s)$  is affine, no other solutions with a zero component are possible. Thus,  $x$  lies in exactly two of the  $A_i$ . This proves the theorem.

With this theorem, continuous piecewise linear simplex splines, i.e., those splines whose set of knots is in general position, can now be correctly evaluated. Suppose one wishes to evaluate  $M(s|t_0, \dots, t_{m+1})$ , where  $t_0, \dots, t_{m+1}$  are points in general position in  $\mathbf{R}^m$ , at a point  $x$  which does not lie on a grid line. If, as in Theorem 1,  $A := [t_0, \dots, t_{m+1}]$  and  $A_i := [t_0, \dots, t_{i-1}, t_{i+1}, \dots, t_{m+1}]$ , then  $x$  lies in only two of the  $A_i$ , say  $A_j$  and  $A_k$ . In order to apply the recurrence relations, (5) is solved to obtain the barycentric coordinates with respect to (without loss of generality)  $A_j$ . These barycentric coordinates then appear as the  $\alpha_i$  in the equation

$$(8) \quad M(x|t_0, \dots, t_{m+1}) = (m+1) \sum_{i=0}^{m+1} \alpha_i M(x|t_0, \dots, t_{i-1}, t_{i+1}, \dots, t_{m+1}),$$

and, in particular,  $\alpha_j = 0$ . On the other hand, of all of the possible other terms, only  $M(x|t_0, \dots, t_{k-1}, t_{k+1}, \dots, t_{m+1})$  will be nonzero, since  $x$  lies in the span of only  $A_j$  and  $A_k$ . Thus, if  $x$  is not on a grid line, then only one term on the right-hand side of (8) is nonzero.

This observation leads to a stable method of handling the problem of grid lines, for if  $x$  is on a grid line, this condition can still be imposed. This leads to the following rule: If the value of a linear simplex spline is desired at any point, then only one term on the right-hand side of the recurrence relations is allowed to have a nonzero value. This trick forces evaluation on the grid lines to behave just like evaluation off the grid lines for linear simplex splines. Thus, continuous linear simplex splines are evaluated properly. But this means that all simplex splines of a higher degree which have maximum nontrivial smoothness are evaluated properly as well, since they are ultimately expressed as a linear combination of continuous linear splines, each of which can be evaluated properly. The only difficulty arises in the selection of the correct piecewise constant spline to be nonzero.

*Example 4.* Consider again the setup of Example 3. The condition that exactly one of  $M(\cdot|t_1, t_2, t_3)$  and  $M(\cdot|t_0, t_2, t_3)$  can be nonzero at  $(0, 0)$  is imposed. In this case, it does not matter what is chosen. However, this problem might be attempted numerically with the following result:

$$M(0, 0|t_0, t_1, t_2, t_3) = \frac{3}{2}M(0, 0|t_1, t_2, t_3) + 10^{-6}M(0, 0|t_0, t_2, t_3) + \frac{3}{2}M(0, 0|t_0, t_1, t_3).$$

Clearly,  $M(\cdot|t_0, t_2, t_3)$  cannot be chosen as the only nonzero spline if reasonable results are the desired goal.

This indicates that the nonzero spline must be chosen with care. A good method is to choose from all possible splines the one which has the largest coefficient. This will eliminate the kind of numeric nuisance which occurs in Example 4.

However, it must be pointed out that this approach only works if the linear spline is continuous. If it is discontinuous, the spline can be evaluated stably everywhere except along the discontinuity, where numeric noise makes the exact location of the discontinuity impossible to calculate. This means, necessarily, that higher-degree splines which are not as smooth as possible (whose knot sets are not in general position) may not be evaluated properly along the higher-degree discontinuities. Thus, there is more to smoothness than just aesthetics; it is necessary in order for this method to properly produce function values.



Using this technique, a code has been constructed for the bivariate case. However, it could be easily modified to handle the general case. The actual CPU time (in seconds) used to compute typical simplex splines is given in the following table. The computations were performed on the Boeing Computer Services Cyber 876. Each number is the total CPU time required to perform 2601 function evaluations over a square region surrounding the simplex spline (a  $51 \times 51$  grid), so the average computational cost per function evaluation can be computed by dividing each number by 2601. Obviously, some of the evaluations required substantially more time than others, since the value of the functions are zero near the boundary, a fact which is quickly discovered when an infeasible tableau occurs at the first step.

Degree	CPU Time
1	0.595
2	0.717
3	1.134
4	1.526
5	3.059
6	5.584
7	13.053
8	30.345

Boeing Computer Services  
 Engineering Technology Applications Division  
 P.O. Box 24346, MS 7L-21  
 Seattle, Washington 98124-0346

1. ROBERT G. BLAND, "New finite pivoting rules for the simplex method," *Math. Oper. Res.*, v. 2, 1977, pp. 103-107.
2. CARL DE BOOR, "Splines as linear combinations of  $B$ -splines," in *Approximation Theory II* (G.G. Lorentz, C. K. Chui, and L. L. Schumaker, eds.), Academic Press, New York, 1976, pp. 1-47.
3. CARL DE BOOR, *A Practical Guide to Splines*, Springer-Verlag, Berlin and New York, 1978.
4. CARL DE BOOR, "Topics in multivariate approximation theory," in *Topics in Numerical Analysis* (P. Turner, ed.), Lecture Notes in Math., vol. 965, Springer-Verlag, Berlin and New York, 1982, pp. 39-78.
5. CARL DE BOOR & KLAUS HÖLLIG, " $B$ -splines from parallelepipeds," *J. Analyse Math.*, v. 42, 1982/83, pp. 99-115.
6. W. DAHMEN & C. A. MICCHELLI, "Numerical algorithms for least squares approximation by multivariate  $B$ -splines," in *Numerical Methods of Approximation Theory*, vol. 6 (Collatz, Meinardus, and Werner, eds.), Birkhäuser, Basel, 1981.
7. G. B. DANTZIG, *Linear Programming and Extensions*, Princeton Univ. Press, Princeton, N. J., 1963.
8. H. HAKOPIAN, "Multivariate spline functions.  $B$ -spline basis and polynomial interpolations," *SIAM J. Numer. Anal.*, v. 19, 1982, pp. 510-517.
9. KLAUS HÖLLIG, "A remark on multivariate  $B$ -splines," *J. Approx. Theory*, v. 33, 1981, pp. 119-125.
10. OLVİ MANGASARIAN, *Linear Programming Lecture Notes*, Manuscript, 1978.
11. R. H. J. GMELIG MEYLING, *An Algorithm for Constructing Configurations of Knots for Bivariate  $B$ -splines*, Mathematisch Instituut, Universiteit van Amsterdam, Report 85-06, 1985.
12. C. A. MICCHELLI, "On a numerically efficient method for computing multivariate  $B$ -splines," in *Multivariate Approximation Theory* (W. Schempp and K. Zeller, eds.), ISNM 51, Birkhäuser, Basel, 1979.
13. C. A. MICCHELLI, "A constructive approach to Kergin interpolation in  $\mathbf{R}^k$ : Multivariate  $B$ -splines and Lagrange interpolation," *Rocky Mountain J. Math.*, v. 10, 1980, pp. 485-497.