Hybrid Bézier Patches on Sphere-like Surfaces

by

Xiaoyu Liu [1] and Larry L. Schumaker [2]

Abstract. We develop a method for interpolating scattered data on sphere-like surfaces based on a local triangular patch which is constructed from a blend of certain spherical Bernstein-Bézier polynomials introduced recently by Alfeld, Neamtu & Schumaker [2]. The method produces a C^1 interpolant which matches values and derivatives, and is a natural analog of a planar method of Foley & Opitz [6] and Goodman & Said [7]. We also show how the same approach can be used to construct a C^2 quintic hybrid patch matching second derivative information as an analog of a planar method of Chang & Said [5].

1. Introduction

We begin by stating the interpolation problem of interest. Suppose S is the unit sphere (or a sphere-like surface, see Remark 8.1), and that $\{v_i\}_{i=1}^n$ is a set of scattered points located on S. Given real numbers $\{f_i\}_{i=1}^n$, we seek a function s defined on S which interpolates the given data in the sense that

$$s(v_i) = f_i, \qquad i = 1, \ldots, n. \tag{1.1}$$

This problem arises in many applications, and a variety of methods for solving it have been proposed, see [4] and references therein. Our approach is to construct an interpolant defined piecewise over a spherical triangulation with vertices at the data points. Each piece of the interpolant is defined locally over a single triangle

[1] Department of Mathematics, Vanderbilt University, Nashville, TN 37240, xlliu@athena.cas.vanderbilt.edu.

[2] Department of Mathematics, Vanderbilt University, Nashville, TN 37240, s@mars.cas.vanderbilt.edu. Supported in part by National Science Foundation Grant 9500643.

using spherical analogs of the classical Bernstein-Bézier polynomials recently introduced by Alfeld, Neamtu, & Schumaker [2, 3, 4]. To avoid having to subdivide triangles, we employ the blending idea which Foley & Opitz [6] and Goodman & Said [7] used to create rational (hybrid) patches on planar triangles.

In this paper our aim is to construct a C^1 cubic hybrid patch on a spherical triangle. The approach can easily be adapted to produce a C^2 quintic hybrid patch, see Remark 8.3. We proceed as follows. First we create a (spherical) triangulation of S with vertices at the data points. Then on each triangle T we construct a function which interpolates the given values and prescribed derivatives at each of the three vertices of T. Each piece of the interpolant is a blended version of the spherical Bernstein-Bézier polynomials introduced in [2], and is constructed in such a way that the pieces fit together to form a globally C^1 surface. The method retains virtually all of the advantages of the spherical Clough-Tocher method discussed in [4], but without the need to split triangles.

The paper is organized as follows. In Sect. 2 we introduce some basic notation related to spherical barycentric coordinates and spherical Bernstein Bézier polynomials. Our new hybrid cubic patch is introduced in Sect. 3, and in Sect. 4 we show how it can be used to solve the interpolation problem. In Sect. 5 we present the results of some numerical experiments and a comparison with the Clough-Tocher method discussed in [4]. The problem of how to estimate derivative information at the data points is examined in Sect. 6. In Sect. 7 we discuss some numerical experiments using estimated derivatives. Finally, we conclude the paper with several remarks. For simplicity, we assume throughout the paper that S is the sphere, but everything carries over immediately to sphere-like surfaces, see Remark 8.1.

2. Spherical Barycentric Coordinates and SBB-Polynomials

The following notation is taken from [2]. Suppose $\{v_i\}_{i=1}^3$ is a set of linearly independent unit vectors (points on the unit sphere S). Then

$$T = \{v \in S \ : \ v = b_1 v_1 + b_2 v_2 + b_3 v_3, \quad b_i \geq 0\}$$

is a spherical triangle with vertices v_1, v_2, v_3. For each v on S, the unique real numbers b_1, b_2, b_3 are called the *the spherical barycentric coordinates of v with respect to T*.

Given an integer d, the set of functions

$$B^d_{ijk}(v) := \frac{d!}{i!j!k!} b_1^i b_2^j b_3^k, \qquad i+j+k = d,$$

are called the *spherical Bernstein basis polynomials* associated with T. As shown in [2], they can be viewed as homogeneous functions of degree d whose extensions

to $\mathbb{R}^3$ form a basis for all homogeneous polynomials of degree d. A function of the form

$$p(v) = \sum_{i+j+k=d} c_{ijk} B^d_{ijk}(v), \tag{2.1}$$

with $c_{ijk} \in \mathbb{R}$ is called a *spherical Bernstein-Bézier (SBB)-polynomial* or *SBB-patch.*

It is shown in [2] that SBB-polynomials possess almost all of the properties of the classical planar Bernstein-Bézier polynomials. In particular, they are easy to store, can be evaluated by a stable and efficient version of the de Casteljau algorithm, have derivatives which are again SBB-polynomials, and can be easily joined together to construct convenient classes of splines on the sphere S and on general sphere-like surfaces.

For later use, we recall that the coefficients c_{ijk} of an SBB-patch can be associated with the *domain points*

$$\xi_{ijk} := \frac{(iv_1 + jv_2 + kv_3)}{\|iv_1 + jv_2 + kv_3\|}, \qquad i + j + k = d.$$

Here $\| \ \|$ denotes the usual Euclidean norm in $\mathbb{R}^3$. We are particularly interested in $d = 3$, in which case we can refer to the coefficient c_{111} as an *interior coefficient*, and the other nine coefficients as *boundary coefficients.*

Suppose p is an SBB-polynomial in the form (2.1), and that g is an arbitrary vector in $\mathbb{R}^3$. Then (see Theorem 3.3 of [4]), the directional derivative of p (considered as a trivariate homogeneous function)

$$D_g p(v) := \frac{d}{d\theta}\ p(v + \theta g)\Big|_{\theta=0} = g^T \nabla p(v) \tag{2.2}$$

is given by

$$D_g p(v) = d \sum_{i+j+k=d-1} c^1_{ijk}(g) B^{d-1}_{ijk}(v), \tag{2.3}$$

where

$$c^1_{ijk}(g) = b_1(g)c_{i+1,j,k} + b_2(g)c_{i,j+1,k} + b_3(g)c_{i,j,k+1}, \quad i + j + k = d - 1, \tag{2.4}$$

and $(b_1(g), b_2(g), b_3(g))$ are the spherical barycentric coordinates of g relative to T.

We close this section with an interpolation result from [4]. For any two vertices of T, let D_{ij} denote the directional derivative corresponding to the direction $v_j - v_i$, where we interpret subscripts mod 3.

Lemma 2.1. *The nine boundary coefficients of a cubic SBB-polynomial p are uniquely determined by the values $p(v_i)$, $D_{i,i+1}p(v_i)$, and $D_{i,i+2}p(v_i)$, $i = 1, 2, 3$.*

Proof: See Lemma 5.3 of [4]. The coefficients at the vertices match the given values, *i.e.*,

$$c_{300} = p(v_1), \qquad c_{030} = p(v_2), \qquad c_{003} = p(v_3).$$

The remaining boundary coefficients of p are determined by the derivatives at the vertices. For example,

$$c_{210} = c_{300} + D_{12}p(v_1)/3. \quad \blacksquare$$

3. A Hybrid Cubic SBB-Patch

Given a spherical triangle $T = \langle v_1, v_2, v_3 \rangle$, let $\mathring{T}$ be its interior, and ∂T its boundary. For each $i = 1, 2, 3$, let e_i be the side opposite v_i corresponding to $b_i = 0$. Motivated by the planar case [6, 7], we define a *hybrid cubic SBB-patch* by

$$P(v) = \sum_{i+j+k=3} c_{ijk} B^3_{ijk}(v), \tag{3.1}$$

with

$$c_{111} = c_{111}(v) = \sum_{\ell=1}^{3} \alpha_\ell A_\ell(b_1, b_2, b_3), \tag{3.2}$$

where A_ℓ are prescribed functions. Here $\alpha_1, \alpha_2, \alpha_3$ and the c_{ijk} except for c_{111} are given real numbers, and b_1, b_2, b_3 are the barycentric coordinates of v with respect to T.

Substituting (3.2) in (3.1) and rearranging the sums, it follows immediately that

$$P(v) = \sum_{\ell=1}^{3} A_\ell(v) p_\ell(v),$$

where $p_\ell(v)$ is the ordinary SBB-patch which has the interior coefficient $c_{111}(v) = \alpha_\ell$ and the same boundary coefficients as P. This means that P is just a blend of three SBB-patches.

In order to assure that P has certain desirable properties, throughout the remainder of the paper we assume that the blending functions satisfy the following hypotheses:

H1) $A_\ell(v) := A_\ell(b_1(v), b_2(v), b_3(v))$ is continuous on $\mathring{T}$ and bounded on T,

H2) $\sum_{\ell=1}^{3} A_\ell(v) \equiv 1$ for $v \in \mathring{T}$,

H3) $A_\ell(v) = 1$ on the interior of e_ℓ, and is zero on the rest of ∂T,

H4) A_ℓ is positively homogeneous, *i.e.*, $A_\ell(av) = a^\rho A_\ell(v)$ for some real ρ and all positive real numbers a,

H5) $\frac{\partial A_\ell}{\partial b_\nu}$ is continuous on $\mathring{T}$, and

$$\lim_{v \to \partial T} \frac{\partial A_\ell(v)}{\partial b_\nu} B^3_{111}(v) \to 0. \tag{3.3}$$

Some specific choices for the A_ℓ are given in Sect. 4.2 below. In the remainder of this section we show that under these hypotheses, the hybrid patch is well-defined and is a C^1 function on T. We begin with continuity.

Theorem 3.1. *The hybrid patch P is a continuous function on T.*

Proof: First we observe that for v on the boundary ∂T of T, P reduces to the ordinary SBB-polynomial p_0 whose boundary coefficients are the same as those of P, and whose interior coefficient is 0. Since P is clearly continuous on $\mathring{T}$, it suffices to consider what happens as we approach ∂T. Now in view of H1), it is clear that if $v^* \in \partial T$, then $P(v) \to P(v^*) = p_0(v^*)$ as $v \to v^*$. ■

For later use we note that the values of P at the vertices are given by

$$P(v_1) = c_{300}, \qquad P(v_2) = c_{030}, \qquad P(v_3) = c_{003}. \tag{3.4}$$

We now turn to directional derivatives of P. In view of our assumption H4) on the A_ℓ, a hybrid SBB-patch has a natural extension to $\mathbb{R}^3$ as a homogeneous function. This allows us to compute its directional derivatives by the formula (2.2).

Theorem 3.2. *For any vector g in $\mathbb{R}^3$, $D_g P$ is a continuous function on T. For $v \in \mathring{T}$,*

$$D_g P(v) = 3 \sum_{i+j+k=2} c^1_{ijk}(g) B^2_{ijk}(v) + [D_g c_{111}(v)] B^3_{111}(v), \tag{3.5}$$

where c^1_{ijk} are defined by (2.4), and

$$D_g c_{111}(v) = \sum_{\ell=1}^{3} \alpha_\ell \sum_{\nu=1}^{3} b_\nu(g) \frac{\partial A_\ell}{\partial b_\nu}. \tag{3.6}$$

Moreover, for points v on the edge e_ℓ, we have $D_g P(v) = D_g p_\ell$, where p_ℓ is the ordinary SBB-polynomial whose boundary coefficients are the same as those of P, and whose interior coefficient is α_ℓ.

Proof: We first examine $D_g P(v)$ for $v \in \mathring{T}$. Applying D_g to (3.1) and using the chain rule, we immediately get (3.5). Using the chain rule again, we have

$$D_g c_{111}(v) = \sum_{\nu=1}^{3} \frac{\partial c_{111}(v)}{\partial b_\nu} D_g b_\nu. \tag{3.7}$$

It was shown in [2] that $D_g b_\nu = b_\nu(g)$ for $\nu = 1,2,3$, and (3.6) follows. Now hypothesis H5) implies $D_g P$ is continuous on $\overset{\circ}{T}$.

As v approaches a point v^* on the boundary, (3.5) and (3.3) imply that $D_g P(v)$ approaches $Q(v^*)$, where

$$Q(v) := 3 \sum_{i+j+k=2} c^1_{ijk}(g) B^2_{ijk}(v).$$

Note that in (3.5) and in this formula for $Q(v)$, the coefficient $c^1_{111}(g)$ also depends on v. Clearly Q is a continuous function on ∂T.

We now examine the derivative $D_g P$ on the boundary. Clearly, at the vertices of T,

$$D_g P(v_1) = c^1_{200}(g), \qquad D_g P(v_2) = c^1_{020}(g), \qquad D_g P(v_3) = c^1_{002}(g). \tag{3.8}$$

Hypothesis H3) implies $c^3_{111}(v) \equiv \alpha_\ell$ for all v in the interior of the edge e_ℓ. Thus, for such v, $Q(v)$ reduces to the derivative at v of the ordinary SBB-polynomial whose boundary coefficients are the same as those of P, and whose interior coefficient is α_ℓ. ■

4. A Hermite Interpolation Method

In this section we show how to use the hybrid patch introduced in the previous section to solve a Hermite version of the original interpolation problem (1.1), where we now specify derivative information at each data point v_i. Suppose that for each $i = 1, \ldots, n$, g^1_i and g^2_i are noncollinear vectors, and that (z^1_i, z^2_i) are given real numbers. Then we seek a function s such that

$$\begin{aligned} s(v_i) &= f_i, \\ D_{g^1_i} s(v_i) &= z^1_i, \\ D_{g^2_i} s(v_i) &= z^2_i, \end{aligned} \tag{4.1}$$

for $i = 1, \ldots, n$. We propose the following algorithm for solving this interpolation problem:

Algorithm 4.1.

1) *find a triangulation $\Delta = \{T_j\}_{j=1}^N$ of S with vertices at the data points $\{v_i\}_{i=1}^n$,*

2) *for each triangle $T = \langle v_{m_1}, v_{m_2}, v_{m_3} \rangle$ in Δ, choose the boundary coefficients of a cubic hybrid SBB-patch P_T so that for $j = 1,2,3$,*

$$\begin{aligned} P_T(v_{m_j}) &= f_{m_j} \\ D_{g^1_i} P_T(v_{m_j}) &= z^1_{m_j} \\ D_{g^2_i} P_T(v_{m_j}) &= z^2_{m_j}, \end{aligned}$$

3) for each triangle T choose the three additional parameters $\alpha_1^T, \alpha_2^T, \alpha_3^T$ defining P_T to make the following global function C^1 continuous on S:

$$s(v) = \begin{cases} P_{T_1}(v), & v \in T_1 \\ \cdots & \\ P_{T_N}(v), & v \in T_N. \end{cases} \tag{4.2}$$

Given a prescribed set of data points $\{v_i\}_{i=1}^n$ on the sphere, there are many possible triangulations using these points as vertices. One reasonable choice would be the Delaunay triangulation which can be computed using the code of Renka [9].

Step 2) can be carried out using Lemma 2.1. Indeed, given the values for $D_{g^1} s(v_i)$ and $D_{g^2} s(v_i)$, we can compute all of the derivatives needed in the lemma to get the boundary coefficients of P_T. We now discuss how to do step 3).

4.1. Computing the Interior Coefficients

In this section we present three methods for computing the parameters defining the interior coefficients of the interpolating hybrid patches. The goal is to choose these coefficients so that the overall interpolant (4.2) is C^1. We consider only local methods, where the choice of the parameters for a patch P_T depends only on information in the triangle T (or at most in a neighboring triangle). It suffices to explain how to compute one parameter, as the others can be done in a similar way.

The first method is based on setting a cross-boundary derivative at the center of each edge of the triangulation.

Method I.

Given a triangle $T = \langle v_1, v_2, v_3 \rangle$, let w be the midpoint of the edge $e := \langle v_2, v_3 \rangle$, and let $g = v_3 \times v_2$. This is a vector perpendicular to the plane spanned by v_2 and v_3, and defines a cross derivative D_g along e. Given a prescribed real number z_w, by (3.5) we can make $D_g P(w) = z_w$ by choosing

$$\begin{aligned} z_w/3 =& B^2_{020}(w)[b_1(g)c_{120} + b_2(g)c_{030} + b_3(g)c_{021}] \\ &+ B^2_{011}(w)[b_1(g)\alpha_1 + b_2(g)c_{021} + b_3(g)c_{012}] \\ &+ B^2_{002}[b_1(g)c_{102} + b_2(g)c_{012} + b_3(g)c_{003}]. \end{aligned}$$

Since $b_1(g)$ and $B^2_{011}(w)$ are both nonzero, this equation uniquely determines α_1.

It is easy to see that this construction leads to patches which join together to form a C^1 surface on all of S. Indeed, for $v \in e$, $D_g P(v) = c^1_{020} B^2_{020}(v) + c^1_{011} B^2_{011}(v) + c^1_{002} B^2_{002}(v)$, which is uniquely determined by the values of $D_g P(v_2)$, $D_g P(w)$, and $D_g P(v_3)$.

This method requires a value for $z_w = D_g f(w)$. In practice it may be necessary to estimate this quantity, see Sect. 6. Alternatively, α_1 can be determined by forcing the cross-boundary derivative to be linear, see Remark 8.2.

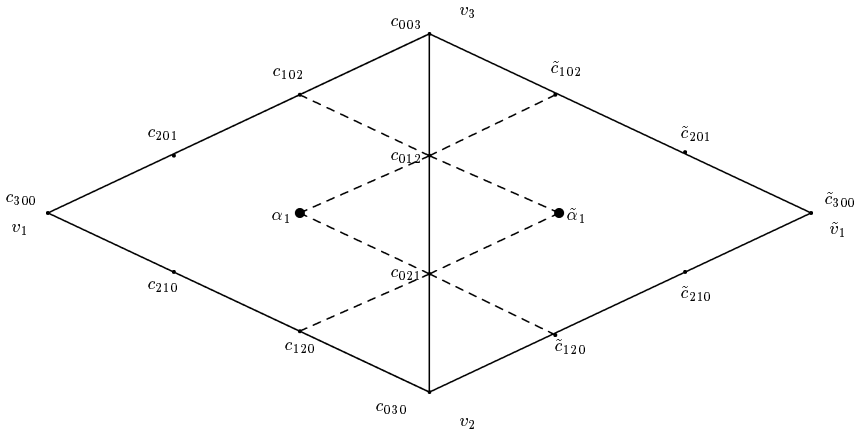

Fig. 1. A pair of neighboring triangles.

Method II.

For completeness, we now discuss a method suggested in the planar case by Foley & Opitz [6], although since it fails for certain triangulations (which are hard to identify in advance), we do not recommend using it in practice. It involves working on a pair of neighboring triangles. Suppose T and $\tilde{T}$ are two adjoining triangles as shown in Fig. 1. Let P and $\tilde{P}$ be the associated hybrid patches. We denote their boundary coefficients by c_{ijk} and $\tilde{c}_{ijk}$, respectively. For a C^0 join, we must have $\tilde{c}_{030} = c_{030}$, $\tilde{c}_{021} = c_{021}$, $\tilde{c}_{012} = c_{012}$, and $\tilde{c}_{003} = c_{003}$. We now show how to compute the parameters α_1 and $\tilde{\alpha}_1$ for these two patches.

Suppose

$$\begin{aligned}\tilde{v}_1 =& r v_1 + s v_2 + t v_3\\ v_1 =& \tilde{r}\tilde{v}_1 + \tilde{s} v_2 + \tilde{t} v_3.\end{aligned}$$

Let p and $\tilde{p}$ be the ordinary SBB-polynomials with the same boundary coefficients as P and $\tilde{P}$, and with interior coefficients α_1 and $\tilde{\alpha}_1$, respectively. Then as shown in [2], the following two conditions are necessary for a C^2 join between the two patches p and $\tilde{p}$:

$$\begin{aligned}L_1 :=& -\tilde{c}_{210} + r^2 c_{210} + 2rsc_{120} + 2rt\alpha_1 + s^2 c_{030} + 2stc_{021} + t^2 c_{012} = 0,\\ L_2 :=& -\tilde{c}_{201} + r^2 c_{201} + 2rs\alpha_1 + 2rtc_{102} + s^2 c_{021} + 2stc_{012} + t^2 c_{003} = 0.\end{aligned} \tag{4.3}$$

We now add these equations together, and if the resulting coefficient $r(s+t)$ of α_1 is nonzero, solve to get

$$\begin{aligned}\alpha_1 =& [(\tilde{c}_{210} + \tilde{c}_{201}) - r^2(c_{210} + c_{201}) - s^2(c_{030} + c_{021}) - t^2(c_{012} + c_{003})\\ &- 2st(c_{021} + c_{012}) - 2rsc_{120} - 2rtc_{102}]/2r(s+t).\end{aligned} \tag{4.4}$$

A similar computation with the other two C^2 conditions

$$\begin{aligned} L_3 &:= -c_{210} + \tilde{r}^2\tilde{c}_{210} + 2\tilde{r}\tilde{s}\tilde{c}_{120} + 2\tilde{r}\tilde{t}\tilde{\alpha}_1 + \tilde{s}^2 c_{030} + 2\tilde{s}\tilde{t}c_{021} + \tilde{t}^2 c_{012} = 0, \\ L_4 &:= -c_{201} + \tilde{r}^2\tilde{c}_{201} + 2\tilde{r}\tilde{s}\tilde{\alpha}_1 + 2\tilde{r}\tilde{t}\tilde{c}_{102} + \tilde{s}^2 c_{021} + 2\tilde{s}\tilde{t}c_{012} + \tilde{t}^2 c_{003} = 0, \end{aligned} \tag{4.5}$$

yields

$$\begin{aligned} \tilde{\alpha}_1 =&[(c_{210} + c_{201}) - \tilde{r}^2(\tilde{c}_{210} + \tilde{c}_{201}) - \tilde{s}^2(c_{030} + c_{021}) - \tilde{t}^2(c_{012} + c_{003}) \\ &- 2\tilde{s}\tilde{t}(c_{021} + c_{012}) - 2\tilde{r}\tilde{s}\tilde{c}_{120} - 2\tilde{r}\tilde{t}\tilde{c}_{102}]/2\tilde{r}(\tilde{s} + \tilde{t}). \end{aligned} \tag{4.6}$$

This method for computing α_1 and $\tilde{\alpha}_1$ can only be applied when the factors $r(s+t)$ and $\tilde{r}(\tilde{s}+\tilde{t})$ are nonzero. It is easy to see that for all configurations of the triangles T and $\tilde{T}$, $r < 0$ and $\tilde{r} < 0$. However (in contrast to the planar case), $s+t$ and $\tilde{s}+\tilde{t}$ can be zero for certain configurations. This happens, for example, when the points v_2 and $\tilde{v}_2$ are antipodal.

We now show that if α_1 and $\tilde{\alpha}$ can be computed by the above formulae, then the two patches automatically join together with C^1 continuity across their common edge $\langle v_2, v_3\rangle$.

Theorem 4.2. *Suppose P and $\tilde{P}$ are two cubic hybrid SBB-patches on adjoining triangles with coefficients determined as above. Then P and $\tilde{P}$ join with C^1 continuity across the common edge $\langle v_2, v_3\rangle$.*

Proof: It is easy to check that

$$\tilde{r} = 1/r, \qquad \tilde{s} = -s/r, \qquad \tilde{t} = -t/r. \tag{4.7}$$

As shown in [2], there are three C^1 conditions to establish:

$$\begin{aligned} M_1 &:= -\tilde{c}_{120} + rc_{120} + sc_{030} + tc_{021} = 0 \\ M_2 &:= -\tilde{\alpha}_1 + r\alpha_1 + sc_{021} + tc_{012} = 0 \\ M_3 &:= -\tilde{c}_{102} + rc_{102} + sc_{012} + tc_{003} = 0. \end{aligned}$$

By the choice of the boundary coefficients, the first and third conditions are automatically satisfied. Inserting (4.4) and (4.6) in M_2 and using (4.7) along with the fact that $M_1 = M_3 = 0$, it is easy to check that $M_2 = 0$. ■

Method III.

In Method II we have solved for the parameters α_1 and $\tilde{\alpha}_1$ by combining certain C^2 continuity conditions. However, in general, none of these four C^2 conditions will actually be satisfied. This suggests that we try to make them be satisfied in a least-squares sense, subject to side conditions which insure that the two adjacent patches

P and $\tilde{P}$ are C^1 continuous across their common edge. We consider the following problem:

$$\text{minimize} \sum_{i=1}^{4} L_i^2, \text{ subject to } M_2 = 0.$$

This is a standard constrained least squares problem, and can be solved for example by introducing a Lagrange multiplier λ to reduce it to a 3×3 linear system for α_1, $\tilde{\alpha}_1$, and the λ. We get

$$\begin{aligned}
8r^2(s^2+t^2)\alpha_1 + r\lambda =& 4rt[\tilde{c}_{210} - r^2 c_{210} - 2rsc_{120} - s^2 c_{030} - 2stc_{021} - t^2 c_{012}] \\
&+ 4rs[\tilde{c}_{201} - r^2 c_{201} - 2rtc_{102} - s^2 c_{021} - 2stc_{012} - t^2 c_{003}] \\
8\tilde{r}^2(\tilde{s}^2+\tilde{t}^2)\tilde{\alpha}_1 - \lambda =& 4\tilde{r}\tilde{t}[c_{210} - \tilde{r}^2 \tilde{c}_{210} - 2\tilde{r}\tilde{s}\tilde{c}_{120} - \tilde{s}^2 c_{030} - 2\tilde{s}\tilde{t}c_{021} - \tilde{t}^2 c_{012}] \\
&+ 4\tilde{r}\tilde{s}[c_{201} - \tilde{r}^2 \tilde{c}_{201} - 2\tilde{r}\tilde{t}\tilde{c}_{102} - \tilde{s}^2 c_{021} - 2\tilde{s}\tilde{t}c_{012} - \tilde{t}^2 c_{003}] \\
r\alpha_1 - \tilde{\alpha}_1 =& - sc_{021} - tc_{012}.
\end{aligned}$$

The determinant of this system is

$$-8r^2[\tilde{r}^2(\tilde{s}^2+\tilde{t}^2) + (s^2+t^2)] = \frac{-8r^6(s^2+t^2)}{(1+r^4)}.$$

Since $r < 0$, this determinant can only be zero if $s = t = 0$. This happens if and only if v_2 and $\tilde{v}_2$ are antipodal.

4.2. Choice of the Blending Functions A_ℓ

There is a a fair amount of leeway in selecting blending functions A_ℓ satisfying properties H1) – H5) of Sect. 3. Here we discuss a choice which has been used in the planar case [6, 7]. We give only the definition of A_1 as the definitions of A_2 and A_3 are similar and are obtained by simply shifting indices.

Example 4.3. *Let m be a positive integer, and let*

$$A_1(v) := \begin{cases} 0, & v = v_1, v_2, v_3, \\ \dfrac{b_2^m b_3^m}{b_1^m b_2^m + b_2^m b_3^m + b_1^m b_3^m}, & \textit{otherwise.} \end{cases} \tag{4.8}$$

Discussion: For v not at a vertex of T,

$$\begin{aligned}
\frac{\partial A_1}{\partial b_1} =& \frac{-mb_1^{m-1} b_2^m b_3^m (b_2^m + b_3^m)}{D^2}, \\
\frac{\partial A_1}{\partial b_2} =& \frac{mb_1^m b_2^{m-1} b_3^{2m}}{D^2}, \\
\frac{\partial A_1}{\partial b_3} =& \frac{mb_1^m b_2^{2m} b_3^{m-1}}{D^2},
\end{aligned}$$

where $D = b_1^m b_2^m + b_2^m b_3^m + b_1^m b_3^m$. It is now easy to check that all of the properties H1) – H5) hold. In particular, $0 \leq A_1(v) \leq 1$ for all $v \in T$. ■

This type of blending function was used in the planar case by Foley & Opitz [6] with $m = 1$, and by Goodman & Said [7] with $m = 2$. These kinds of rational functions have also been used to construct certain *bubble functions* for use in the finite-element method, see [1, 8].

4.3. Properties of the Hybrid Interpolant

Before giving some numerical experiments to illustrate the performance of our hybrid interpolation method, we make some general remarks about its properties, and compare it to the Clough-Tocher macro element method discussed in [4].

Storage: To store a hybrid interpolant, we have to store one coefficient for each vertex of the triangulation, 2 coefficients for each edge, and 3 coefficients for each triangle. Assuming there are V vertices and using the formula $E = 3V - 6$ for the number of edges and $N = 2V - 4$ for the number of triangles, we see that the total storage is $13V - 24$. Referring to Table 1 of [4], we see that this is less than one-half the amount required for the Clough-Tocher method.

Evaluation: To evaluate a hybrid interpolant at a vertex of a triangle T, we simply use the formulae (3.4). To evaluate it at any other point v in T, we first compute the value of $c_{111}(v)$, and then use the de Casteljau algorithm [2]. For the kinds of weight functions suggested in Sect. 4.2, computing $c_{111}(v)$ involves only a few operations. Thus, the total cost of a hybrid evaluation is essentially the same as an evaluation of a Clough-Tocher interpolant, since the latter also requires testing to find which of the three subtriangles of T contains v.

Precision: It was shown in [4] that the Clough-Tocher method gives exact results when applied to data taken from a cubic SBB-polynomial. The following theorem shows that if the parameters of our hybrid interpolant are chosen by any of the methods in Sect. 4.1, then it is also exact for cubic SBB-polynomials.

Theorem 4.4. *Suppose f is a cubic SBB-polynomial, and that s is the piecewise hybrid cubic patch which interpolates f and its derivatives at the vertices of a triangulation Δ, where the interior Bézier coefficients are computed by any of the methods in Sect. 4.1. Then $s \equiv f$.*

Proof: Fix a triangle T in the triangulation Δ, and suppose we write f in the form (2.1). Consider the interpolating patch P_T associated with a triangle T. In view of Lemma 2.1, the boundary coefficients of P_T must agree with those of f. Moreover, it is easy to check that for all three methods the three parameters definining $c_{111}^T(v)$ are equal to the interior coefficient c_{111} of f. Now in view of our hypothesis H2) on

the weight functions, it follows that $c^T_{111}(v) = c_{111}$ for all $v \in \mathring{T}$, and so $P_T \equiv f$ on T. ■

5. Numerical Results with Exact Derivatives

We have conducted a number of experiments to compare the hybrid method with the methods discussed in [4], using the same setup as used there. In particular, we suppose that exact data is taken from the test function

$$f(x,y,z) = 1 + x^8 + e^{2y^3} + e^{2z^2} + 10xyz \tag{5.1}$$

at the vertices of the sequence of regular triangulations $\Delta_1, \ldots, \Delta_7$ used in [4]. Δ_1 has 6 vertices, one at each of the intersections of the x, y, and z axes with the unit sphere. For each $\ell \geq 2$, Δ_ℓ is obtained from $\Delta_{\ell-1}$ by subdividing each triangle in $\Delta_{\ell-1}$ into four subtriangles. These triangulations have 6, 18, 66, 258, 1026, 4098, and 16386 vertices, respectively.

To show the performance of the hybrid method, we compute the maximum relative error between the hybrid interpolant and the true function on certain discrete sets of points on the sphere. In particular, when working with Δ_ℓ, the error is measured on the discrete grid V_ℓ defined in [4]. Each such grid contains approximately 1 million points. All computations were done in double precision.

Table 1 shows the results obtained using Methods I – III to compute interior coefficients, and the blending functions given in Example 4.3 with $m = 1$. For comparison purposes, the last column shows the errors obtained with the Clough-Tocher interpolant described in [4]. The NC in the row corresponding to Δ_1 stands for "not computable". It occurs because for the triangulation Δ_1, for some pairs of triangles the vertices v_1 and $\tilde{v}_1$ of Fig. 1 are antipodal, and as discussed in Sect. 4.1, Methods II and III cannot be used in this case.

	Method I	Method II	Method III	CT Method
Δ_1	1.06 (-1)	NC	NC	1.06 (-1)
Δ_2	4.39 (-2)	7.05 (-2)	7.05 (-2)	4.42 (-2)
Δ_3	1.05 (-2)	1.05 (-2)	1.05 (-2)	1.05 (-2)
Δ_4	1.07 (-3)	1.07 (-3)	1.07 (-3)	1.07 (-3)
Δ_5	7.62 (-5)	7.62 (-5)	7.62 (-5)	7.62 (-5)
Δ_6	4.89 (-6)	4.89 (-6)	4.89 (-6)	4.89 (-6)
Δ_7	3.03 (-7)	3.03 (-7)	3.03 (-7)	3.03 (-7)

Table 1. Error for Hybrid and CT Methods.

Except for Δ_2 (which has only 18 vertices), the errors for all three hybrid methods are almost exactly the same as those for the Clough-Tocher method. We also tried other triangulations and other test functions, and in all cases got comparable results for the three methods and the Clough-Tocher method.

To explore the effect of choosing different values of m in the blending functions (4.8), we repeated the above computations with $m = 2, 5, 50, 10000$. We found that there is not much difference in the results.

6. Estimating Derivatives

In the original interpolation problem (1.1) we are given only function values at scattered data points. But the hybrid method requires three pieces of information at each data point: a value, and two directional derivatives. Moreover, if we use Method I to compute the interior coefficients, we also need cross-boundary information at the center of each edge. Thus, to construct a hybrid interpolant, we first have to estimate the missing derivative information.

6.1. A Derivative Estimation Algorithm

In this subsection we consider the following problem:

> Let f be a smooth function defined on the sphere S and let $v^* \in S$. Given a vector g, find an estimate for the directional derivative $D_g f(v^*)$ based on the values of f at points $W := \{w_\nu\}_{\nu=1}^N$ on S.

This is the usual numerical differentiation problem, except that here it is posed on the sphere. The standard approach is to use the data to construct an approximation p to f, and then use $D_g p(v^*)$ as an estimate of $D_g f(v^*)$. Since we are working on the sphere, it is natural to choose an SBB-polynomial for the approximant. We are led to the following algorithm.

Algorithm 6.1.

1) *choose a triangle T, and let $\{B^d_{ijk}\}_{i+j+k=d}$ be the associated spherical Bernstein basis polynomials,*

2) *find c_{ijk} so that*

$$p(v) = \sum_{i+j+k=d} c_{ijk} B^d_{ijk}(v)$$

minimizes

$$\sum_{\nu=1}^{N} [p(w_\nu) - f(w_\nu)]^2,$$

3) *compute $D_g p(v^*)$ as an estimate of $D_g f(v^*)$.*

Step 2) involves solving a standard discrete least squares problem, and reduces to solving a linear system for the coefficients c_{ijk}. In Step 3) we use the formula (3.5) to compute $D_g p(v^*)$.

6.2. Choice of T

Theoretically, in Step 1) of Algorithm 6.1 it doesn't matter which triangle we choose to define the Bernstein basis functions B^d_{ijk}. However, in practice the choice of T does have a considerable effect on the condition of the system of equations for the c_{ijk}, and on the accuracy of the estimate. To illustrate the effect of varying the size and location of T, we present some numerical results using cubic SBB-polynomials ($d = 3$) to estimate the derivative $D_g f(v^*)$ for the function $f = x + y + z$, where $v^* = (1, 0, 0)$ and $g = (0, 0, 1)$. We choose the data set W to consist of 15 random data points with spherical coordinates (θ, ϕ) in the interval $[-10°, 10°]$.

Table 2 shows the effect of varying the *size of* T. The quantity β in the table controls the size of the triangle $T = \langle v_1, v_2, v_3 \rangle$ on which the basis functions are defined. For each β, T is a symmetric triangle centered about v^*, such that the angles between the v^* and the vectors v_i are equal to β, measured in degrees. The first vertex of T is located at the point with spherical coordinates $(\beta, 0)$. The second column of the table gives the condition numbers of the corresponding linear systems. The third column of the table lists the errors $|D_g f(v^*) - D_g p(v^*)|$.

Table 3 illlustrates the effect of varying the *location of* T. Each row in the table corresponds to using the same triangle used in the third row of Table 2, but translated so that its center lies at the point with spherical coordinates $(\gamma, 0)$ instead of at the point v^* with spherical coordinates $(0, 0)$. As in Table 2, we list the condition numbers of the corresponding linear systems, and the errors $|D_g f(v^*) - D_g p(v^*)|$.

Since the function $f = x + y + z$ is itself a cubic, with 15 data points the least squares SBB-polynomial should fit exactly, and we should get an exact estimate for the derivative. Thus, the errors shown in the two tables are due solely to numerical errors arising in setting up and solving the least squares problem. The tables clearly show that both the condition number and accuracy are affected by the size of β and γ. Triangles which are too small, too large, or too far way from the data set result in higher condition numbers and loss of accuracy.

In this section we have illustrated the effect of varying the size and location of T. However, for a given set W of data, both the condition number and error also depend on the *shape* and *orientation* of T. Thus, in general, finding the best possible T would be computationally very expensive.

6.3. Automatic Selection of T

For practical applications, it is important to have an automatic process for selecting a good triangle T. Our numerical experience suggests that it is best to choose T

β	Condition	Error
1	6.9 (5)	3.3 (-10)
5	5.4 (2)	1.3 (-14)
10	3.1 (3)	1.0 (-13)
20	3.2 (5)	7.9 (-13)
40	5.4 (7)	1.2 (-11)

Table 2. Effect of Triangle Size on Derivative Estimation.

γ	Condition	Error
0	3.1 (3)	1.0 (-13)
5	5.6 (4)	5.4 (-13)
10	3.6 (6)	4.9 (-12)
20	1.3 (9)	2.1 (-10)
45	2.5 (12)	6.2 (-8)
90	1.0 (14)	1.1 (-6)

Table 3. Effect of Triangle Location on Derivative Estimation.

just large enough to contain most of the data points in W. We propose the following procedure:

1) compute the unit vector $v_c = w_c/\|w_c\|$, where $w_c = (w_1 + \cdots + w_N)$.
2) choose T to be a symmetric triangle centered at v_c which is as small as possible so that the inscribed disk contains all of the points in W.

For the data in Table 2, this procedure leads to a triangle $T = \langle v_1, v_2, v_3 \rangle$ such that the angles between v_c and the v_i are all equal to 13°. The corresponding condition number is 3.3 (3) and the associated error is 9.8 (-14). These numbers compare quite favorably with the choice $\beta = 5$ which gave the best error in the table.

6.4. Performance of Numerical Differentiation

The overall performance of the numerical differentiation procedure depends on the smoothness of f, the number of data points in W, how close they are to the point v^* of interest, and how they are distributed in a disk around v^*.

To get some feeling for how well numerical differentiation performs in a typical situation, we computed estimates for the derivative $D_g f(v^*)$ of the function f defined in (5.1) and compared them with the true values. The basis triangles were computed using the automatic method of the previous section.

Table 4 shows the results for $v^* = (1, 0, 0)$ and $g = (0, 0, 1)$. It is based on a collection of data sets $W_{k,\beta}$ with $k = 12, 15, 30$ and $\beta = 1, 5, 10, 20, 40$, where

the set $W_{k,\beta}$ contains k random points whose polar coordinates (θ, ϕ) in degrees lie in the interval $[-\beta, \beta]$. For each choice of k and β, the table lists the error $|D_g f(v^*) - D_g p(v^*)|$.

β	12 data	15 data	30 data
1	1.410 (-6)	6.493 (-7)	8.809 (-7)
5	1.751 (-5)	8.047 (-5)	1.096 (-4)
10	1.373 (-3)	6.289 (-4)	8.639 (-4)
20	1.014 (-2)	4.504 (-3)	6.525 (-3)
40	5.892 (-2)	2.289 (-2)	4.255 (-2)

Table 4. Effect of Size and Spread of Data Sets.

As expected, for all choices the k, the error increases monotonically as we increase the spread of the data points. However, the behavior as we varied k is a little unexpected. In most cases $k = 15$ gave better results than $k = 12$, but also better results than $k = 30$. We observed the same behavior in tests on a number of other functions.

6.5. Application to Hybrid Interpolation

We return now to the problem of solving the interpolation problem (1.1) using the hybrid interpolation method based on a triangulation Δ with vertices at the data points $V := \{v_i\}_{i=1}^n$. As pointed out above, to construct the hybrid interpolant, we need to estimate two derivatives at each vertex (and if Method I is used to compute interior coefficients, also one derivative at the center of each edge of the triangulation).

Suppose v^* is the point where we need to estimate a derivative corresponding to the direction g. Since we are working with cubic SBB-polynomials, it is natural to use a cubic SBB-polynomial to compute the estimated derivative $D_g f(v^*)$. Since p has 10 coefficients, in order to apply Algorithm 6.1, we have to choose a set W_{v^*} of at least 10 data points near v^*. Assuming that we do not have additional data at our disposal, we have to choose these points from V.

The simplest way to choose W_{v^*} is to take the N points in V which are closest to v^*, based on their geodesic distances from v^*. However, generally we will get better estimates if the points are somewhat uniformly distributed around v^*. As in the planar case, it is possible to design algorithms to achieve this by looking at more than N points in choosing W_{v^*}.

7. Numerical Results with Estimated Derivatives

To illustrate the effect of using estimated derivatives, we reran the experiments of Sect. 5 for the same function (5.1), using Method I for calculating interior coefficients. This time, however, we do not use exact derivatives, but instead compute estimates by the method of Sect. 6, using the triangle constructed by the automatic procedure described there.

Table 5 shows the results for the hybrid interpolant based on the triangulations Δ_3, Δ_4, Δ_5. For each Δ_ℓ, the needed derivatives were estimated based on values of f at the vertices of Δ_ℓ itself, using only the 15 closest points for each estimate. The columns marked E_∞, E_2, E_1 give the relative errors measured by $E_p^\ell = e_p^\ell / \|f\|_\infty$, where

$$e_\infty^\ell := \max_\nu |f(\eta_\nu^\ell) - s(\eta_\nu^\ell)|$$

$$e_2^\ell := \Big(\frac{\sum_{\nu=1}^N [f(\eta_\nu^\ell) - s(\eta_\nu^\ell)]^2}{N} \Big)^{1/2}$$

$$e_1^\ell := \sum_{\nu=1}^N |f(\eta_\nu^\ell) - s(\eta_\nu^\ell)|/N,$$

and $V_\ell = \{\eta_\nu^\ell\}_{\nu=1}^N$ are the sets of $N = 1,048,578$ points on the sphere defined in [4]. The column marked *Exact* gives the values of E_∞ when exact derivatives are used. We see that in all cases, the errors with estimated derivatives are only a few times larger than with exact derivative data.

	Exact	E_∞	E_2	E_1
Δ_3	1.049 (-2)	3.112 (-2)	6.018 (-3)	4.023 (-3)
Δ_4	1.073 (-3)	3.057 (-3)	3.649 (-4)	2.306 (-4)
Δ_5	7.631 (-5)	1.831 (-4)	1.651 (-5)	1.041 (-5)

Table 5. Effect of Estimated Derivatives on Hybrid Interpolation.

In some applications we may have a large set of data points U, but we want to construct an interpolant based on a triangulation with fewer vertices. In this case we can select a subset V of U to use as vertices for the triangulation Δ on which the hybrid interpolant is based, but continue to use all of the points of U in estimating derivatives. To get an idea of how this works, we again consider interpolating the function f in (5.1). We choose V to be the set of 66 vertices of the triangulation Δ_3 in Sect. 5. To see what happens when more data is available to estimate derivatives, we constructed sets U_n from V by adding an addition $n - 66$ data points, chosen randomly on the sphere. Table 6 shows the results for $n = 66, 100, 150, 200$. For comparison purposes, we also list the errors obtained using exact derivatives in the row labelled $n = \infty$.

n	E_∞	E_2	E_1
66	3.112 (-2)	6.018 (-3)	4.023 (-3)
100	2.078 (-2)	4.498 (-3)	3.390 (-3)
150	2.570 (-2)	3.708 (-3)	2.421 (-3)
200	2.446 (-2)	2.966 (-3)	1.852 (-3)
∞	1.049 (-2)	1.533 (-3)	9.853 (-4)

Table 6. Error for the hybrid interpolant on Δ_3 using n data points.

As the table shows, the relative maximum error using estimated derivatives is only a few times worse than the relative maximum error using exact derivatives. It also shows that using more data points to estimate derivatives does not necessarily improve E_∞, but both E_1 and E_2, which measure average error, do decrease as n increases.

8. Remarks

Remark 8.1. A surface $\mathcal{S}$ in $\mathbb{R}^3$ of the form

$$\mathcal{S} := \{\sigma(v) := \rho(v)v \ : \ v \in S\}$$

where ρ is a continuous, positive, real-valued function defined on the unit sphere S is called a *sphere-like surface*. The entire Bernstein-Bézier theory developed in [2, 3, 4] applies to general sphere-like surfaces, and thus our hybrid method also extends immediately to such surfaces.

Remark 8.2. In the planar case, Goodman & Said [7] suggested a scheme similar to our Method I for choosing the parameters defining the interior coefficients c_{111}. In particular, to determine the parameter α_1 associated with a triangle $T = \langle v_1, v_2, v_3\rangle$, they forced the cross-boundary derivative along the edge $\langle v_2, v_3\rangle$ to be a linear polynomial rather than a quadratic one. This approach eliminates the need to provide a value for the cross-boundary derivative at the center of the edge, and also works in the spherical case. However, as pointed out in [6], piecewise cubic surfaces whose cross-boundary derivatives are only linear are visibly less smooth than those without the restriction. In practice we have to compute estimates for derivatives at the vertices, and so it is no additional burden to also compute them at the midpoints of edges.

Remark 8.3. The blending idea presented here can also be used to create a C^2 quintic hybrid patch as was done in the planar case in Chang & Said [5]. Now we take

$$P(v) = \sum_{i+j+k=5} c_{ijk} B^5_{ijk}(v), \tag{8.1}$$

with

$$c_{ijk} = c_{ijk}(v) = \sum_{\ell=1}^{3} c_{ijk,\ell} A_\ell(b_1, b_2, b_3), \tag{8.2}$$

for $(i,j,k) \in \mathcal{C} := \{(2,2,1),(2,1,2),(1,2,2)\}$. The blending functions A_ℓ can be chosen as in Example 4.3, but here we need $m \geq 2$ to insure C^2 continuity between adjoining patches. In this case we can use derivative information up to order 2 at each vertex to determine all coefficients of the patch except for those with subscripts in $\mathcal{C}$. For each $\ell = 1,2,3$, the three coefficients $c_{ijk,\ell}$ with $(i,j,k) \in \mathcal{C}$ can be determined from the value of the perpendicular cross derivative at a point in the interior of the edge e_ℓ, and the values of the second order perpendicular cross derivative at two other points in the interior of the edge e_ℓ. This is the analog of our Method I above.

References

1. Agbeve, K. N., Elements finis triangulaires rationnels de classe C^k, PhD thesis, Univ. de Nantes, 1993.
2. Alfeld, P., M. Neamtu, and L. L. Schumaker, Bernstein-Bézier polynomials on spheres and sphere-like surfaces, Comput. Aided Geom. Design, to appear.
3. Alfeld, P., M. Neamtu, and L. L. Schumaker, Dimension and local bases of homogeneous spline spaces, SIAM J. Math. Anal., to appear.
4. Alfeld, P., M. Neamtu, and L. L. Schumaker, Fitting scattered data on sphere-like surfaces using spherical splines, J. Comp. Appl. Math., this volume.
5. Chang, L. H. T., and H. B. Said, A C^2 triangular patch for scattered data interpolation, Rpt. M8/95, University Sains Malaysia, Penang, 1995.
6. Foley, T., and K. Opitz, Hybrid cubic Bézier triangle patches, in *Mathematical Methods in Computer Aided Geometric Design II* (T. Lyche and L. Schumaker, eds), Academic Press (New York), 1992, 275–286.
7. Goodman, T. N. T., and H. B. Said, A C^1 triangular interpolant suitable for scattered data interpolation, Comm. Appl. Numer. Meth. **7** (1991), 479–485.
8. Le Méhauté, A., and K. N. Agbeve, Rational C^{k+1} finite elements, in *Wavelets, Images, and Surface Fitting* (P.-J. Laurent, A. Le Méhauté, and L. L. Schumaker, eds), A. K. Peters (Wellesley MA), 1994, 335–342.
9. Renka, R. J., Algorithm 623: Interpolation on the surface of a sphere, ACM Trans. Math. Software **10** (1984), 437–439.