SIAM J. SCI. COMPUT.
Vol. 29, No. 3, pp. 1338–1354

BIVARIATE SPLINES OF VARIOUS DEGREES FOR NUMERICAL SOLUTION OF PARTIAL DIFFERENTIAL EQUATIONS*

XIAN-LIANG HU†, DAN-FU HAN†, AND MING-JUN LAI‡

Abstract. Bivariate splines with various degrees are considered in this paper. A matrix form of the extended smoothness conditions for these splines is presented. Upon this form, the multivariate spline method for numerical solution of partial differential equations (PDEs) proposed by Awanou, Lai, and Wenston in [*The multivariate spline method for scattered data fitting and numerical solutions of partial differential equations*, in Wavelets and Splines, G. Chen and M. J. Lai, eds., Nashboro Press, Brentwood, TN, 2006, pp. 24–76] is generalized to obtain a new spline method. It is observed that, combined with prelocal refinement of triangulation and automatic degree raising over triangles of interest, the new spline method of bivariate splines of various degrees is able to solve linear PDEs very effectively and efficiently.

Key words. bivariate splines, smoothness condition, splines of various degrees, numerical solution of Poisson and biharmonic equations

AMS subject classifications. 65D07, 65D15, 35Q30, 75D05

DOI. 10.1137/060667207

1. Introduction.

Let $\Omega \subset \mathbf{R}^2$ be a polygonal domain and $\triangle$ be a given triangulation of Ω. A standard spline space $S_d^r(\triangle)$ is the space of all smooth piecewise polynomial functions of degree d and smoothness r over $\triangle$ with $r < d$. That is,

$$S_d^r(\triangle) = \{s \in C^r(\Omega),\, s|_t \in \mathbf{P}_d, \quad t \in \triangle\},$$

where $\mathbf{P}_d$ denotes the space of polynomials of degree d, t denotes a triangle, and $\Omega = \bigcup\{t,\, t \in \triangle\}$. Such a spline space has been studied for a long time in the literature and in practice. (See [6] and references therein.)

In this paper we would like to consider bivariate splines of various degrees d_t, $t \in \triangle$, where d_t is a positive integer for each t. That is, for a given triangulation $\triangle$ and a degree vector $\mathbf{d} = \{d_t,\, t \in \triangle\}$, we let

$$\mathcal{S}_{\mathbf{d}}^r(\triangle) = \{s \in C^r(\Omega),\, s|_t \in \mathbf{P}_{d_t}, \quad t \in \triangle\}.$$

Our study of such a spline space $\mathcal{S}_{\mathbf{d}}^r(\triangle)$ is motivated by our interest in numerically solving partial differential equations (PDEs) more effectively and efficiently. For a PDE with a singularity at the boundary of the domain Ω, a finite element method requires a much finer mesh at the singularity. Otherwise, one has to increase the degrees of finite elements. Instead of increasing degrees globally, we would like to increase the degrees locally and combine this with local triangulation refinement near the singularity to efficiently solve the PDE. For continuous finite elements, there are several constructions of finite elements of various degrees available in the literature, e.g., [15] and [16]. More recently, optimal meshes of p- and hp-finite element methods

*Received by the editors August 9, 2006; accepted for publication (in revised form) January 11, 2007; published electronically June 7, 2007.
http://www.siam.org/journals/sisc/29-3/66720.html

†Department of Mathematics, Zhejiang University, Hangzhou, 310027, China (huxl98@yahoo.com.cn, mhdf2@zju.edu.cn). The work of the second author was partially supported by the National Basic Research Program under grant 2005CB32170X.

‡Department of Mathematics, The University of Georgia, Athens, GA 30602 (mjlai@math.uga.edu).

for PDEs with singularity are discussed in [13], and a constructive scheme for p-finite elements is explained in [14]. However, the implementation of such p-finite elements is very complicated. Recently, in [1], Awanou, Lai, and Wenston introduced a multivariate spline method for numerical solution of PDEs which overcomes the difficulty of the implementation. The researchers in [1] streamlined the process by skipping the construction of smooth finite elements. Instead, they simply used discontinuous piecewise polynomial functions over a triangulation and treated any desired smoothness conditions across edges of triangles as side constraints. Together with the side constraints of boundary conditions, they applied a Lagrange multiplier method to solve the system associated with the PDE of interest. Another key ingredient was to use a special iterative algorithm to solve the associated linear system. (See [1] for more details.) Smooth spline spaces with various degrees were also considered in [1], where a degree reduction condition is introduced and incorporated with standard smoothness conditions acting as constraints. In other words, let $d = \max\{d_t,\, t \in \triangle\}$, $\mathcal{S}^r_{\mathbf{d}}(\triangle) \subset S^r_d(\triangle)$. As the largest degree d among d_t, $t \in \triangle$, becomes large, the size of the linear system arising from the spline space $S^r_d(\triangle)$ grows rapidly. This will not lead to an efficient algorithm. Hence in this paper we propose a different approach. To keep the given degrees over various triangles, we need a new smoothness condition for polynomial pieces over two adjacent triangles with different degrees. Then we use the new smoothness conditions in the place of the original smoothness conditions in the multivariate spline method for numerical solution of PDEs proposed by Awanou, Lai, and Wenston in [1]. For simplicity, we shall refer to the original multivariate spline method for PDEs as the ALW spline method. To solve a PDE with singularity on the boundary of the domain Ω, we do not know in advance which degrees we should use to achieve the given accuracy of the solution. We adopt a strategy of combining local refinement and local degree raising to effectively solve PDEs with singularity on the boundary.

The present paper is organized as follows: A matrix form of the new smoothness conditions for splines of various degrees is considered in section 2. In section 3, we shall explain how to use bivariate splines of various degrees for numerical solution of PDEs. Numerical examples are shown in section 4 to illustrate that our new spline method is more efficient and effective. Finally we make some remarks about our spline method in section 5.

2. A matrix form of the new smoothness condition. Let T be a fixed triangle in $\triangle$. It is known that any polynomials of degree d can be expressed as the following so-called B-form:

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

where $c_{i,j,k}$ are Bézier coefficients of p and $B^d_{i,j,k}(x,y)$ are Bernstein polynomials of degree d, which are both sorted in lexicographical order with respect to their Bézier ordinate (i,j,k). That is, the order of indices (i,j,k) is actually arranged by the following ordering function:

$$q(i,j,k) = \binom{j+k+1}{2} + k + 1. \tag{2.2}$$

For example, when $d = 3$, the linear order of Bézier coefficients is as follows:

$$c_{3,0,0}, c_{2,1,0}, c_{2,0,1}, c_{1,2,0}, c_{1,1,1}, c_{1,0,2}, c_{0,3,0}, c_{0,2,1}, c_{0,1,2}, c_{0,0,3}.$$

Note that all the Bézier coefficients are sorted by this ordering function in the remainder of this paper.

The smoothness condition of polynomial patches over two adjacent triangular domains is an important and useful tool in the study of multivariate splines. It can be found in the literature, e.g., [3] in the bivariate setting and [2] in the multivariate setting. A geometric interpretation of bivariate C^r smoothness conditions for general $r \geq 0$ is given in [4]. For convenience, we include the smoothness conditions in this paper.

LEMMA 2.1. *Let $T := \langle v_1, v_2, v_3\rangle$ and $\tilde{T} := \langle v_4, v_2, v_3\rangle$ be triangles sharing the edge $e := \langle v_2, v_3\rangle$. Let*

$$p_d(v) := \sum_{i+j+k=d} c_{ijk} B^d_{ijk}(v) \qquad \text{and} \qquad \widetilde{p}_d(v) := \sum_{i+j+k=d} \tilde{c}_{ijk} \tilde{B}^d_{ijk}(v),$$

where $B^d_{ijk}(v)$ and $\tilde{B}^d_{ijk}(v)$ are the Bernstein–Bézier basis functions associated with T and $\tilde{T}$, and D^n_u is the nth order differential operator with respect to direction u. Then

$$D^n_u p_d(v) = D^n_u \widetilde{p}_d(v) \qquad \forall v \in \langle v_2, v_3\rangle$$

for $n = 0, \ldots, r$ if and only if

$$\widetilde{c}_{n,j,k} = \sum_{\mu+\nu+\kappa=n} c_{\mu,j+\nu,k+\kappa} B^n_{\mu\nu\kappa}(\lambda), \qquad j+k = d-n, \qquad n = 0, \ldots, r, \tag{2.3}$$

where λ is the barycentric coordinates of vertex v_4 respective to triangle T.

In practice, when applying the spline space $S^r_d(\triangle)$ for numerical solution of PDEs as in [1] and [7], the smoothness conditions (2.3) are assembled into matrix form for all interior edges of $\triangle$:

$$H\mathbf{c} = 0, \tag{2.4}$$

where $\mathbf{c}$ is a vector of Bézier coefficients of a polynomial of degree d over triangle T for all $T \in \triangle$. To extend the smoothness conditions to the case of different degrees on adjacent patches, we first need the following matrix form of the de Casteljau algorithm.

LEMMA 2.2. *Let $\lambda = (\lambda_1, \lambda_2, \lambda_3)$ be the barycentric coordinate of point (x, y) with respect to T; then the B-form in (2.1) can be evaluated by*

$$P_d(\lambda) = \sum_{i+j+k=d-r} c^{(r)}_{i,j,k}(\lambda) B^{d-r}_{i,j,k}(\lambda) \quad \forall r = 0, 1, \ldots, d, \tag{2.5}$$

with $c^{(0)}_{i,j,k} = c_{i,j,k}$ and the following recurrence:

$$c^{(r+1)}_{i,j,k}(\lambda) = \lambda_1 c^{(r)}_{i+1,j,k}(\lambda) + \lambda_2 c^{(r)}_{i,j+1,k}(\lambda) + \lambda_3 c^{(r)}_{i,j,k+1}(\lambda), \tag{2.6}$$

which can be expressed in matrix form

$$\mathbf{c}^{(r+1)}(\lambda) = \mathcal{D}_{\lambda,d-r} \mathbf{c}^{(r)}(\lambda) \tag{2.7}$$

with notation $\mathbf{c}^{(r)}(\lambda) = \{c^{(r)}_{i,j,k}(\lambda)\}_{i+j+k=d-r}$, $r = 1, \ldots, d$. Furthermore, if the Bézier ordinate indices are sorted by the ordering function q defined in (2.2), the matrix $D_{\lambda,d}$ is of size $d(d+1)/2 \times (d+1)(d+2)/2$.

$$
\begin{array}{ccc|ccc|cccc|cccc|c}
\lambda_1 & \lambda_2 & \lambda_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \lambda_1 & 0 & \lambda_2 & \lambda_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \lambda_1 & 0 & \lambda_2 & \lambda_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & \lambda_1 & 0 & 0 & \lambda_2 & \lambda_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \lambda_1 & 0 & 0 & \lambda_2 & \lambda_3 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \lambda_1 & 0 & 0 & \lambda_2 & \lambda_3 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & 0 & 0 & 0 & \lambda_2 & \lambda_3 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & 0 & 0 & 0 & \lambda_2 & \lambda_3 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & 0 & 0 & 0 & \lambda_2 & \lambda_3 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda_1 & 0 & 0 & 0 & \lambda_2 & \lambda_3
\end{array}
$$

FIG. 2.1. *The hierarchical structure of matrix* $\mathcal{D}_{\lambda,d}$.

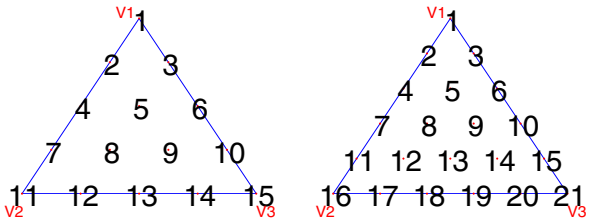

FIG. 2.2. *The hierarchical structure of Bézier ordinates.*

Proof. The first part of this lemma is the standard de Casteljau algorithm of triangular B-form, and (2.7) is easy to verify by writing (2.6) in its matrix form. ∎

Let us point out an interesting property of the matrix $\mathcal{D}_{\lambda,d}$. First we present the matrix $\mathcal{D}_{\lambda,d}$ with degrees d from 1 to 4 as shown in Figure 2.1. The top-left 1×3 submatrix represents $\mathcal{D}_{\lambda,1}$, the top-left 3×6 matrix represents $\mathcal{D}_{\lambda,2}$, etc. Observe that the matrix has a nice pattern. This is because of the definition of the ordering function q; i.e., the Bézier ordinates are sorted line by line incrementally. In Figure 2.2, the ordinate ordering for Bézier ordinates of $d = 4$ indexed by 1–15 is a part of the ordinate ordering for $d = 5$ indexed by 1–21. Then it is easy to see that $\mathcal{D}_{\lambda,d}$ is a block of $\mathcal{D}_{\lambda,d+1}$ when (2.6) is written into (2.7). This property is referred to as a hierarchical property.

On the other hand, if the Bézier ordinate indices are ordered in another form as shown in Figure 2.3 for $d = 4$ and $d = 5$, we find that this new ordering does not share the hierarchical property as in Figure 2.2.

Based on the above matrix form of the de Casteljau algorithm, we can express the standard smoothness conditions in the following matrix form version.

LEMMA 2.3. *Let* $\tilde{\mathbf{c}}$ *and* $\mathbf{c}$ *be the vector forms of* $\{\tilde{c}_{ijk}\}_{i+j+k=d}$ *and* $\{c_{ijk}\}_{i+j+k=d}$

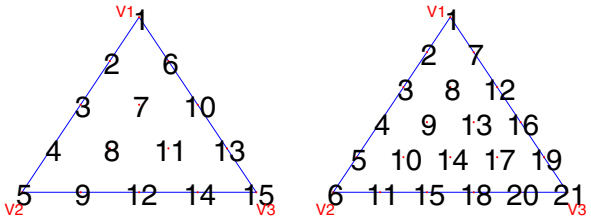

FIG. 2.3. *A new ordering of Bézier ordinates.*

sorted by the ordering function q in (2.2). *Then*

$$D_u^n p_d(v) = D_u^n \widetilde{p}_d(v) \quad \forall v \text{ in } \langle v_2, v_3 \rangle$$

if and only if

$$P_{n,d}\tilde{\mathbf{c}} = P_{0,d-n}\left\{\prod_{l=1}^{n} \mathcal{D}_{\lambda,d+1-l}\right\}\mathbf{c} \qquad \forall n = 0,\dots,r, \tag{2.8}$$

where $P_{n,d}$ is a matrix of size $(d-n+1)\times(d+1)(d+2)/2$ and its entry is

$$P_{n,d}(i,j) = \begin{cases} 1, & j = q(n, i-1, d+1-n-i), \quad i = 1,\dots,d+1-n, \\ 0 & \textit{otherwise} \end{cases}$$

for all $n = 0,\dots,r$.

Proof. We need to prove that (2.8) is equivalent to (2.3). By the definition of $P_{n,d}$, it has only one nonzero entry in each row. So we name it the picking matrix, whose geometric meaning is to pick out the Bézier coefficients on the nth line parallel to common edge e from all the Bézier coefficients of degree d. Then we have

$$\{\tilde{c}_{n,j,k}\}_{j+k=d-n} = P_{n,d}\tilde{\mathbf{c}} \qquad \forall n = 0,\dots,r. \tag{2.9}$$

By (2.5), for all $n = 0,\dots,r$, it is true that

$$\sum_{\mu+\nu+\kappa=n} c_{\mu,j+\nu,k+\kappa} B_{\mu\nu\kappa}^n(\lambda) = \sum_{\mu+\nu+\kappa=n} c_{\mu,j+\nu,k+\kappa}^{(0)} B_{\mu\nu\kappa}^n(\lambda) = c_{0,j,k}^{(n)}(\lambda), \quad j+k = d-n.$$

It can also be written in vector form. That is, applying (2.7) for n times, we get

$$\begin{aligned} \{c_{0,j,k}^{(n)}(\lambda)\}_{j+k=d-n} &= P_{0,d-n}\mathbf{c}^{(\mathbf{n})} \\ &= P_{0,d-n}\mathcal{D}_{\lambda,d+1-n}\mathbf{c}^{(\mathbf{n-1})} \\ &= \cdots \\ &= P_{0,d-n}\left\{\prod_{l=1}^{n} \mathcal{D}_{\lambda,d+1-l}\right\}\mathbf{c}. \end{aligned} \tag{2.10}$$

Combining (2.3), (2.9), and (2.10), we have (2.8). ∎

Next we express the well-known degree raising algorithm for the B-form in matrix form as in Lemma 2.2.

LEMMA 2.4. *Let p be a polynomial of degree d defined on a triangle T written in the B-form* (2.1)*, and let $c_{ijk}^{[d]} = c_{ijk}$ be its coefficients. Then it can also be evaluated by*

$$p = \sum_{i+j+k=d+1} c_{ijk}^{[d+1]} B_{ijk}^{d+1},$$

where

$$c_{ijk}^{[d+1]} = (ic_{i-1,j,k}^{[d]} + jc_{i,j-1,k}^{[d]} + kc_{i,j,k-1}^{[d]})/(d+1) \tag{2.11}$$

for $i+j+k = d+1$. Here coefficients with negative subscripts are assumed to be zero. Denoting $\mathbf{c}^{[d+1]} = \{c_{ijk}^{[d+1]}\}_{i+j+k=d+1}$ and $\mathbf{c}^{[d]} = \{c_{ijk}^{[d]}\}_{i+j+k=d}$ sorted by the ordering function (2.2)*, it follows that*

$$\mathbf{c}^{[d+1]} = \frac{1}{d+1} A_d \mathbf{c}^{[d]}, \tag{2.12}$$

where A_d is a matrix of size $(d+2)(d+3)/2 \times (d+1)(d+2)/2$, whose pattern is hierarchical.

Proof. It is easy to see that the degree raising algorithm (2.11) can be expressed in a matrix form (2.12). It is trivial to prove that the pattern of A_d is hierarchical. ∎

We now extend (2.8) to the case of different degrees on the neighboring triangles and get the desired explicit smoothness condition.

THEOREM 2.5. *Let $d_1 \geq d_2$ be two integers and let*

$$p_{d_1}(v) := \sum_{i+j+k=d_1} c_{ijk} B_{ijk}^{d_1}(v) \qquad \text{and} \qquad \widetilde{p}_{d_2}(v) := \sum_{i+j+k=d_2} \widetilde{c}_{ijk} \tilde{B}_{ijk}^{d_2}(v).$$

Let $\mathbf{c}$ and $\tilde{\mathbf{c}}$ be the vectors of $\{c_{ijk}\}_{i+j+k=d_1}$ and $\{\widetilde{c}_{ijk}\}_{i+j+k=d_2}$ sorted by the ordering function q. Then

$$D_u^n p_{d_1}(v) = D_u^n \widetilde{p}_{d_2}(v) \qquad \forall v \in \langle v_2, v_3 \rangle$$

if and only if

$$P_{n,d_1} \left\{ \frac{d_2!}{d_1!} \prod_{d=d_2}^{d_1-1} A_d \right\} \tilde{\mathbf{c}} = P_{0,d_1-n} \left\{ \prod_{l=1}^{n} \mathcal{D}_{\lambda,d_1+1-l} \right\} \mathbf{c} \qquad \forall n = 0, \ldots, r. \tag{2.13}$$

Proof. We separate the proof into two cases: $d_1 = d_2$ and $d_1 > d_2$. For the first case, it is easy to see that (2.13) is reduced to (2.8).

For the second case, we raise the degree d_2 of polynomial to d_1 by applying (2.11) for $d_1 - d_2$ times to get

$$\tilde{\mathbf{c}}^{[d_1]} = \frac{d_2!}{d_1!} \left\{ \prod_{d=d_2}^{d_1-1} A_d \right\} \tilde{\mathbf{c}}.$$

Note that Bézier coefficients $\tilde{\mathbf{c}}^{[d_1]}$ have the same degree as d_1. Then we can apply the usual smoothness conditions, i.e., (2.8). Thus the smoothness conditions for various degrees follow. ∎

They are the new smoothness conditions which will be used in the rest of the paper.

3. A new spline method for numerical solution of PDEs. In [1], Awanou, Lai, and Wenston proposed a multivariate spline method (i.e., the ALW method) for numerical solution of Poisson and biharmonic equations. The most striking advantage of the ALW method is that one can directly use piecewise polynomial functions for solving Poisson and biharmonic equations without constructing finite elements, macroelements, and locally supported basis functions. The ALW spline method can be explained briefly as follows: It uses piecewise polynomials over a triangulation and treats smoothness conditions as side constraints together with boundary conditions to minimize the Euler–Lagrange functionals associated with Poisson or biharmonic equations. An iterative method is introduced to solve the special linear systems arising from the discretization using bivariate spline functions. The method is shown to be very convenient for numerical solution of Poisson and biharmonic equations. In this section, our main purpose is to improve the ALW spline method by using bivariate splines of various degrees through adaptive procedure.

3.1. A unified model for Poisson and biharmonic equations. First let us introduce some necessary notation. Let L_k be a symmetric partial differential operator of order $2k$ with $k \geq 1$. In the bivariate setting, let

$$D^\alpha = \left(\frac{\partial}{\partial x}\right)^{\alpha_1} \left(\frac{\partial}{\partial y}\right)^{\alpha_2} \quad \forall \alpha = (\alpha_1, \alpha_2)$$

and let $|\alpha| = \alpha_1 + \alpha_2$. We intend to find approximate solutions of the following boundary value problem:

$$\left\{ \begin{array}{lcll} L_k u & = & f & \text{in } \Omega, \\ D^\alpha u & = & D^\alpha g & \text{on } \partial\Omega, \ |\alpha| \leq k-1, \end{array} \right. \tag{3.1}$$

where Ω is a polygonal domain in $\mathbf{R}^2$ and $f \in L^2(\Omega)$, $g \in C^{k-1}(\Omega)$. Note that model problem (3.1) includes the Poisson equation in the case $L_1 = -\Delta$ and the biharmonic equation in the case $L_2 = \Delta^2$.

The weak formulation for model problem (3.1) reads as follows: Find $u \in H^k(\Omega)$ which satisfies its boundary condition such that

$$a(u, v) = \langle f, v \rangle \quad \forall v \in H_0^r(\Omega), \tag{3.2}$$

where $a(u, v)$ is the bilinear form defined by

$$a(u,v) = \left\{ \begin{array}{ll} \displaystyle\int_\Omega \nabla u \cdot \nabla v dx dy, & k = 1, \\ \displaystyle\int_\Omega \triangle u \triangle v dx dy, & k = 2, \end{array} \right.$$

and $\langle f, v \rangle = \int_\Omega f(x,y)v(x,y)dxdy$ is the L_2 inner product of f and v. Here $H^k(\Omega)$ and $H_0^k(\Omega)$ are standard Sobolev spaces.

It is known (cf. [8]) that model problem (3.2) has a unique solution u in $H^k(\Omega)$. The proof can be adapted to show that there is a unique solution s_u in spline space $S_{\mathbf{d}}^r(\triangle) \subset H^k(\Omega)$, where $r = k-1$ and s_u satisfies

$$a(S_u, v) = \langle f, v \rangle \quad \forall v \in S_{\mathbf{d}}^r(\triangle) \tag{3.3}$$

and the boundary conditions.

Based on the standard calculus of variations, model problem (3.1) is the Euler–Lagrange equation of the following energy functional:

$$E_r(s) = \frac{1}{2}a(s,s) - \langle f, s\rangle.$$

It is known that the weak solution of the model problem is the minimizer of the energy functional $E(s)$ in the following spline space (cf. [8, section 8.2.3]):

$$V = \{s \in S^r_{\mathbf{d}}(\triangle),\, D^\alpha s = D^\alpha g_h \quad \text{on } \partial\Omega,\, |\alpha| \le k-1\},$$

where $g_h \in S^r_{\mathbf{d}}(\triangle)$ is a spline such that $g_h|_{\partial\Omega}$ is a good approximation of the Dirichlet boundary condition g. For simplicity let us assume that g is a function of piecewise polynomial of degree $\le d$. Then the weak solution u satisfies

$$E_r(u) = \min_{s\in V} E_r(s). \tag{3.4}$$

Also any minimizer satisfying (3.4) is the weak solution.

3.2. Spline solution for (3.4). Next we discuss how to approximate weak solutions using spline spaces $S^r_{\mathbf{d}}(\triangle)$, which is also denoted as $S^r_{\mathbf{d}}$ if no confusion arises. We shall compute the approximation $s_u \in S^r_{\mathbf{d}}$ satisfying

$$E_r(s_u) = \min_{s\in S^r_{\mathbf{d}}\cap V} E_r(s).$$

Let us write a spline function $s \in S^r_{\mathbf{d}}$ in the following piecewise polynomial format:

$$s(x,y)|_t = \sum_{i+j+k=d_t} c^t_{i,j,k} B^t_{i,j,k}(x,y), \quad (x,y) \in t \in \triangle.$$

Let $\mathbf{c} = (c^t_{i,j,k},\, i+j+k = d_t,\, t \in \triangle)$ be the B-coefficient vector associated with s. The above energy functional can be written as

$$\tilde{E}_r(\mathbf{c}) = \frac{1}{2}\mathbf{c}^T K_r \mathbf{c} - \mathbf{c}^T M \mathbf{f}$$

with respect to B-coefficient vector $\mathbf{c}$, subject to the smoothness conditions $H\mathbf{c} = 0$ according to (2.13) and boundary condition $B\mathbf{c} = G$, as treated in [1]. Here, $\mathbf{K}_r = \mathrm{diag}(K^t_r,\, t \in \triangle)$ is the stiffness matrix with blocks

$$K^t_1 = \left[\int_t \nabla B^t_{ijk} \nabla B^t_{lmn} dxdy\right]^{l+m+n=d_t}_{i+j+k=d_t} \qquad \forall t \in \triangle$$

when $r = 1$, while K_r is the bending matrix in case $r = 2$, with

$$K^t_2 = \left[\int_t \Delta B^t_{ijk} \Delta B^t_{lmn} dxdy\right]^{l+m+n=d_t}_{i+j+k=d_t} \qquad \forall t \in \triangle$$

as its blocks. Similarly, let $\mathbf{M} = \mathrm{diag}(M_t,\, t \in \triangle)$ be the mass matrix with blocks

$$M_t = \left[\int_t B^t_{ijk} B^t_{lmn} dxdy\right]^{l+m+n=d_t}_{i+j+k=d_t} \qquad \forall t \in \triangle$$

and let $\mathbf{f}$ be the B-coefficient vector of the spline S_f interpolating f at the domain points of t of degree d_t for $t \in \triangle$. By using a Lagrange multiplier method, we let

$$\tilde{L}_r(\mathbf{c}, \alpha, \beta) = \frac{1}{2}\mathbf{c}^T\mathbf{K}_r\mathbf{c} - \mathbf{c}^T\mathbf{M}\mathbf{f} + \alpha H\mathbf{c} + \beta(B\mathbf{c} - G)$$

and compute local minimizers. It is easy to see that we need to solve the following system:

$$\begin{bmatrix} B^T & H^T & \mathbf{K}_r \\ 0 & 0 & B \\ 0 & 0 & H \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \mathbf{c} \end{bmatrix} = \begin{bmatrix} \mathbf{M}\mathbf{f} \\ G \\ 0 \end{bmatrix}. \tag{3.5}$$

Note that the existence and uniqueness of s_u imply that there exists a unique $\mathbf{c}$ satisfying the above linear system. Thus the iterative method in [1] can be applied and is effective in solving this linear system. If we denote $L^T = [B^T H^T]$, $\lambda = [\lambda_1^T \lambda_2^T]^T$, $\mathbf{F} = \mathbf{M}\mathbf{f}$ and the vector G absorbs the zero vector, then system (3.5) becomes

$$\begin{bmatrix} L^T & \mathbf{K}_r \\ 0 & L \end{bmatrix} \begin{bmatrix} \lambda \\ \mathbf{c} \end{bmatrix} = \begin{bmatrix} \mathbf{F} \\ G \end{bmatrix}.$$

For any fixed $\epsilon > 0$, given an initial guess $\lambda^{(0)}$, e.g., $\lambda^{(0)} = 0$, we first compute

$$c^{(1)} = \left(\mathbf{K_r} + \frac{1}{\epsilon}L^T L\right)^{-1} \left(F + \frac{1}{\epsilon}L^T b - L^T\lambda^{(0)}\right),$$

and for $k = 1, 2, \ldots,$ we iteratively compute

$$c^{(k+1)} = \left(\mathbf{K_r} + \frac{1}{\epsilon}L^T L\right)^{-1} \left(\mathbf{K_r}c^{(k)} + \frac{1}{\epsilon}L^T b\right). \tag{3.6}$$

The matrix $(\mathbf{K_r} + \frac{1}{\epsilon}L^T L)$ can be proved to be invertible, and the iterative algorithm (3.6) is convergent (cf. [1] for more details). For large-scale problems, the conjugate gradient method for the above iteration is preferred.

3.3. Adaptive algorithms. For the case $r = 0$ or $r = 1$, it is known that when $d_t = d$ for all $t \in \triangle$ with $d \geq 3r + 2$, the spline space S_d^r possesses the optimal approximation order (cf. [5]). So we require $\min_{t\in\triangle} d_t \geq 3r + 2$. What we need to do now is decide the degree vector $\mathbf{d}$ triangle by triangle. It depends on an error indicator and an adaptive procedure. For most problems, an a posteriori error estimate combined with a robust adaptive strategy is necessary as in adaptive finite element methods. So robust adaptive procedures are introduced here, and we leave the a posteriori error estimate to the numerical examples. For a better understanding of the performance of spline spaces with various degrees, we pay attention only to the performances of the local mesh refinement and the local degree raising. Let us present these two adaptive procedures in the following pseudocodes.

ALGORITHM 3.1 (hVersion).

```
mesh = initialize the Mesh;
TOL = 1e-5;
STEP = 1;
MAXSTEP = 30;
refine_percent = 0.8;
```

```
while STEP <MAXSTEP
    x = solve PDE with fixed degree method on mesh;
    error = posteriori_estimate(mesh,x);
    if sum(error) < TOL
        STEP = MAXSTEP + 1;
        break;
    end
    tris = indicate_by_fraction(error,refine_percent);
    mesh = refine_mesh(mesh,tris);
    STEP = STEP + 1;
end
```

In the above algorithm, the PDE is solved by any fixed degree algorithms such as the finite element method in Example 4.1 and the ALW spline method in Example 4.2. We use the fixed fraction strategy for mesh refinement, in which some fraction of elements with highest error are marked for refinement. It appears to be the most robust and convenient strategy among error equidistribution, fixed threshold, error density equidistribution, and fixed fraction strategies (cf. [11]), and has been proved to be convergent in energy norm for the Poisson equation in [12]. The marked elements are refined by Rivara's algorithm, which bisects the prescribed triangle. It was originally developed in [10] to produce a conforming triangulation. Rivara's algorithm is given as follows:

```
find the midpoint P of the longest side in T;
Bisect T with P;
while P is a nonconforming point for T' different to T
    find the midpoint Q of the longest side in T';
    bisect T' with Q;
    if P != Q
        join P and Q;
        P = Q
    end
end,
```

which can be illustrated as in Figure 3.1.

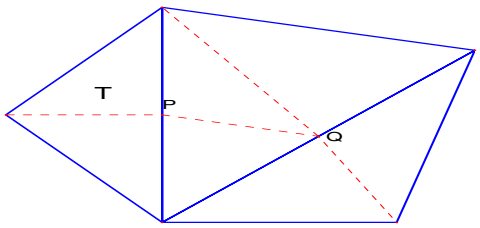

FIG. 3.1. *Conforming refinement using Rivara's algorithm.*

ALGORITHM 3.2 (pVersion).

```
mesh = initMesh;
TOL = 1e-5;
STEP = 1;
MAXSTEP = 50;
refine_percent = 0.8;
```

```
while STEP < MAXSTEP
    x = solve (3.5) with extended smoothness condition (2.13) on mesh;
    error = posteriori_estimate(mesh,x);
    error = smooth_error(mesh,error);
    if max(error) < TOL
        STEP = MAXSTEP + 1;
        break;
    end
    tris = refine_by_fraction(error,refine_percent);
    degree(tris) = degree(tris) + 1;
    STEP = STEP + 1;
end
```

In the above algorithms, we have adopted an error smoothing scheme which is as follows:

```
step 1: compute the mean error on each node according to its
        adjacent elements.
step 2: recompute the mean error on each element
        according to its adjacent nodes.
```

It is important for the convergence of our p-version algorithm to have such an error smoothing scheme because it is not desired that the degree vary too rapidly for adjacent triangles.

We shall use these two adaptive procedures in the next section where numerical results are presented.

4. Numerical examples. We first present our numerical experiments for the Poisson equation.

Example 4.1. Consider the Poisson equation on an L-shaped domain:

$$\begin{cases} -\triangle u &= f & \text{in } \Omega, \\ u &= g & \text{on } \partial\Omega, \end{cases}$$

where $\Omega = \{(x,y),\, (x,y) \in (-1,1)\times(-1,1)\backslash(-1,0)\times(-1,0)\}$. Note that our spline method does not require the information of an analytic solution. For a clear comparison, let

$$u(x,y) = r^{\frac{2}{3}} \sin\left(\frac{2\theta+\pi}{3}\right)$$

be an analytic function, where $r(x,y) = \sqrt{(x^2+y^2)}$ and θ is the angle between vector (x,y) and the position direction of axis x with $\tan(\theta) = y/x$. We use the standard finite element method and our spline method to approximate $u(x,y)$ by solving the Poisson equation with $f = -\Delta u$ inside Ω and $g = u$ on the boundary.

Since we do not use any information about the analytic solution during the adaptive procedure, it is standard to introduce an a posteriori error estimator. We prefer the explicit residual a posteriori error esitmator given in [9] for the Poisson equation. That is, for any triangle T, its L_2 error indicator is

$$\eta_T := \left\{ |T|^4 ||f + \triangle u_h||^2_{0,2;T} + \frac{1}{2} \sum_{e\in\partial T} |e|^3 ||R_e||^2_{0,2;e} \right\}^{\frac{1}{2}}, \tag{4.1}$$

where $|T|$ is the minimal diameter of circles that contain triangle T, $|e|$ is the length of edge e, and R_e stands for the jump of $\frac{\partial u_h}{\partial n}$ on the internal edges or the jump of $\frac{\partial u_h}{\partial n} - g$ on the boundary edges.

The properties of solution for this problem have been discussed theoretically in [15], and its p/hp finite element solution is presented in [14]. For both methods we begin with a special triangulation as shown in Figure 4.1.

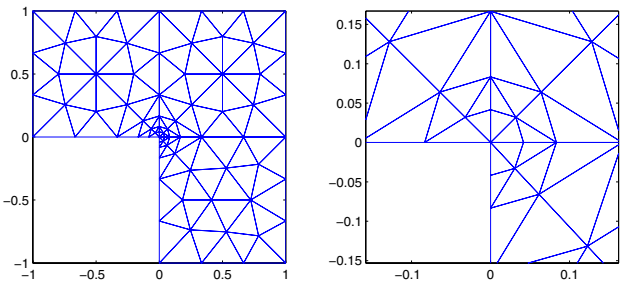

FIG. 4.1. *At the left is an original triangulation, which is amplified at the right.*

We use such a specially designed mesh, which is generated by hand, to capture the corner singularity. The full automatic hp adaptive procedure proposed in [14] required a more accurate reference solution and hence is too expensive to use. Since we pay attention only to the performance of our p-version spline method, a prebuilt mesh is enough.

In Table 4.1 we list the performance of a continuous fixed degree finite element method based on Algorithm 3.1 (hVersion) as well as the performance of our spline method with local degree raising (Algorithm 3.2) starting with continuous bivariate splines of degree 3, referred to as $S_3^0(\triangle)$. We show the number of triangles and the number of unknown coefficients of the finite element solutions and spline solutions which achieve a given tolerance $5e-3$, $5e-4$, or $5e-5$, respectively. We use cubic, quartic, and quintic finite element methods according to different tolerances because the low order finite element method yields too many elements, which is beyond the ability of our machine when the tolerance is small. In addition, we show the total CPU time for computing the finite element solution and spline solution for each of the tolerances. Our machine is equipped with Intel Celeron 2.0GHz and 512M memory.

TABLE 4.1
The performance of fixed order finite element methods and our spline method.

	$5e-3$		$5e-4$		$5e-5$	
	Cubic	Spline	Quartic	Spline	Quintic	Spline
Number of elements	872	116	953	180	1078	216
Number of unknowns	8720	2942	14295	4361	22638	6681
CPU time	29.21s	9.83s	45.98s	16.71s	72.89s	32.46s

From Table 4.1, it is clear to see that our spline method with local degree raising yields higher efficiency due to the smaller size of its system matrix. We also plot in Figure 4.2 the approximated solutions of both approaches whose tolerance is $5e-3$ and in Figure 4.3 the differences of both solutions against the exact analytic solution.

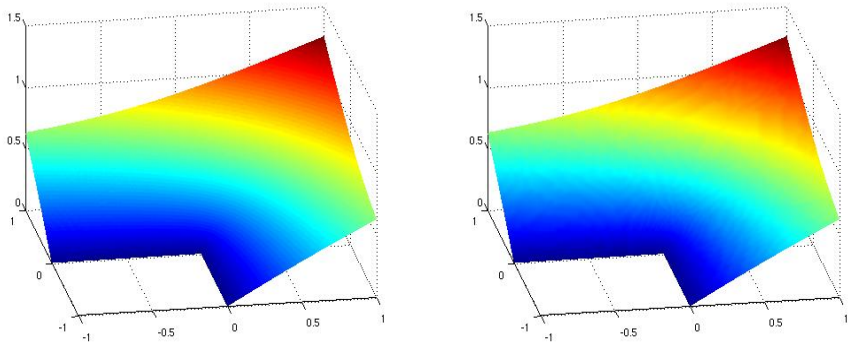

FIG. 4.2. *The approximate solution of the cubic finite element (left) and our spline method (right) under tolerance* $5e-3$.

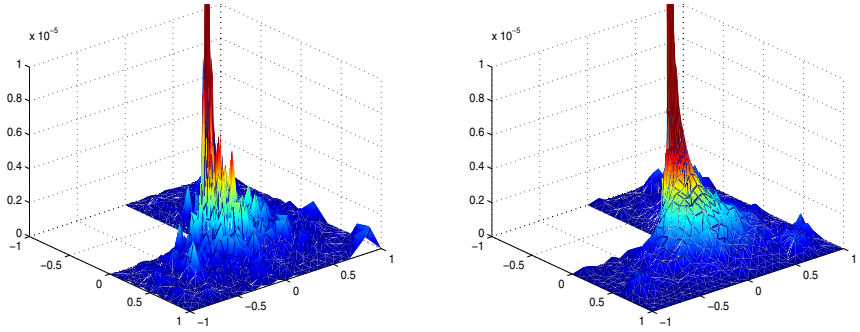

FIG. 4.3. *The error of approximate solution of the cubic finite element (left) and our spline method (right) under tolerance* $5e-3$.

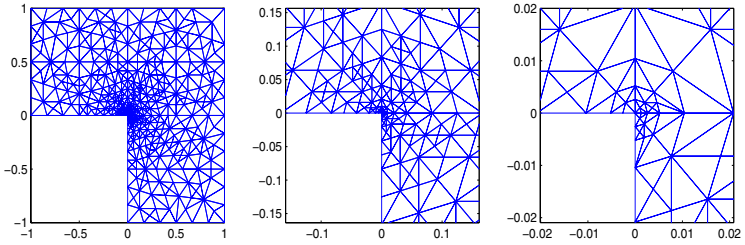

FIG. 4.4. *The resulting triangulation (left) for finite element solution and corner amplification (middle and right).*

From the figures in Figure 4.3, we can see that the error plot of our spline solution is less bumpy than that of the finite element solution and hence, that the surface of our spline solution is closer to the surface of the exact solution that of the finite element solution.

In addition, we present in Figure 4.4 the triangulation of the finite element solution

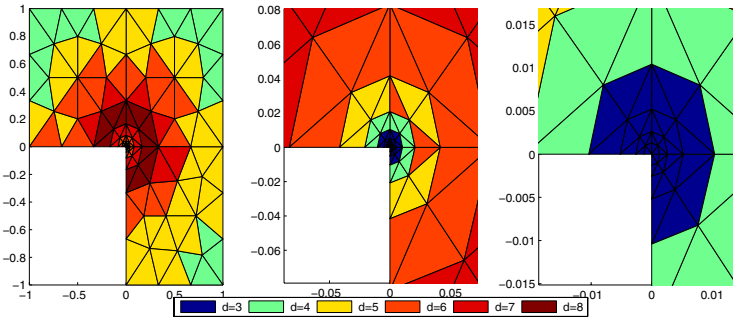

FIG. 4.5. *The triangulation and degrees for our spline solution and two corner amplifications.*

when the given tolerance is $5e-3$, and in Figure 4.5 the degrees as well as the triangulation of the spline solution when the given tolerance is $5e-3$, where we can see that the highest degree in our spline solution is 8, which happens in a few triangles around the singularity.

Next we present our numerical results for biharmonic equations.

Example 4.2. We consider a biharmonic problem on an L-shaped domain:

$$\begin{cases} \Delta^2 u &= f & \text{in } \Omega, \\ u &= g & \text{on } \partial\Omega, \\ \frac{\partial}{\partial \mathbf{n}} u &= h & \text{on } \partial\Omega, \end{cases}$$

where Ω is the same L-shaped domain as in the previous example. We test the analytic solution

$$u(x,y) = r^{\frac{5}{2}} \sin\left(\frac{5\theta}{2}\right),$$

where $r(x,y) = \sqrt{(x^2+y^2)}$ and θ is the angle between vector (x,y) and the position direction of axis x with $\tan(\theta) = y/x$. Then f, g, h can be computed directly from u.

If we denote $[\cdot]_e$ as the jump of a function from inside T to outside through edge e, then the L_2 error indicator for any triangle T is the following explicit residual a posteriori estimator, as proposed in [9]:

(4.2)

$$\eta_T := \left\{ |T|^4 ||\triangle^2 u_h - f||^2_{0,2;T} + \sum_{e\in\partial T} [|e|\,||[\triangle u_h]_e||^2_{2;e} + |e|^3 ||[n_e \cdot \nabla \triangle u_h]_e||^2_{2;e}] \right\}^{\frac{1}{2}},$$

where the means of $|T|$ and $|e|$ are the same as above and n_e is the unit outward normal to edge $e \in \partial T$. The behavior of the true solution is very similar to the one in Example 4.1. We first use $S^1_5(\triangle)$ based on the special triangulation as in Figure 4.1 to solve this biharmonic problem by using the ALW spline method in [1] together with Algorithm 3.1. Then we start with $S^1_5(\triangle)$ using Algorithm 3.2 to solve this problem again. For each tolerance $1e-3$, $1e-4$, $1e-5$, we show in Table 4.2 the number of triangles, the number of unknown coefficients, and the total computational times of two spline solutions.

TABLE 4.2
The performance of the ALW spline method and our spline method.

	$1e-3$		$1e-4$		$1e-5$	
	ALW	New	ALW	New	ALW	New
Number of elements	210	116	370	122	716	128
Number of unknowns	4410	3766	7770	4484	15036	5192
Total CPU time	13.45s	15.93s	31.52s	27.61s	75.73s	39.10s

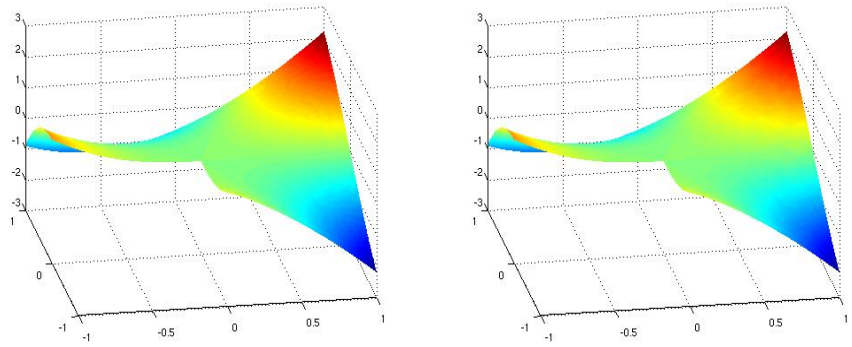

FIG. 4.6. *The approximate solution of the ALW spline method (left) and our spline method (right) under tolerance* $1e-5$.

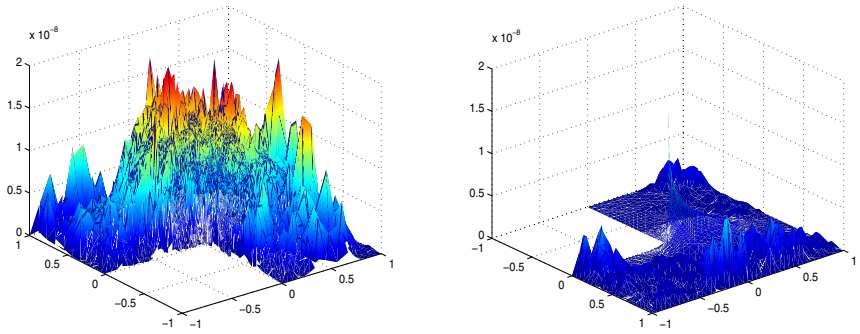

FIG. 4.7. *The error of approximate solution of the ALW spline method (left) and our spline method (right) under tolerance* $1e-5$.

Table 4.2 clearly shows that our spline method is much more efficient than the ALW spline method using h-adaptivity when the tolerance is small. We also plot in Figure 4.6 the approximated solutions of both approaches whose tolerance is $1e-5$ and in Figure 4.7 the differences of both solutions against the exact analytic solution. Comparing the figures in Figure 4.7, we can see that our spline solution is closer to the exact solution than the ALW solution.

In addition, we present in Figure 4.8 the triangulation of the ALW spline solution when the given tolerance is $1e-5$ and in Figure 4.9 the degrees as well as the triangulation of the spline solution with the same tolerance, where we can see that the highest degree in our spline solution is 12, which happens in a few triangles close to the singularity.

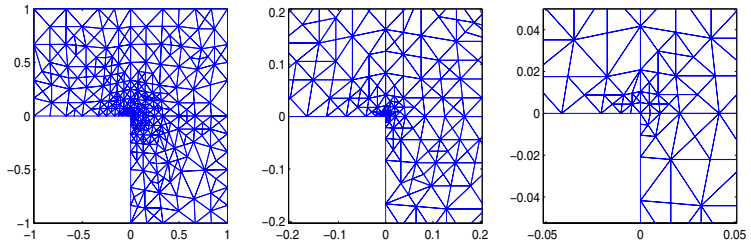

FIG. 4.8. *The resulting triangulation (left) for the ALW spline method and corner amplifications (middle and right).*

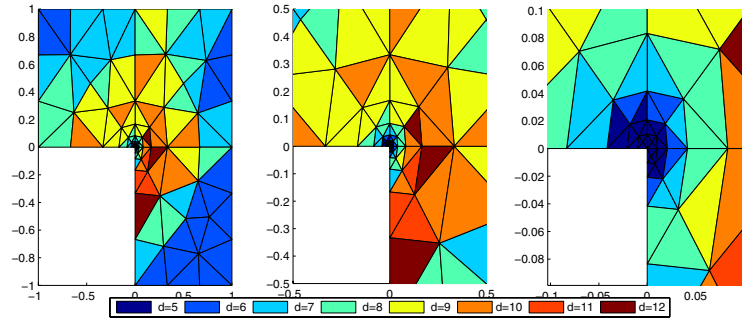

FIG. 4.9. *The triangulation and degrees for our spline solution and two corner amplifications.*

In general, we do not have an analytic solution available to guide our adaptive procedures. Our adaptive algorithms work for these cases. On the other hand, more accurate a posteriori error estimators are preferred for deciding whether to raise degrees or refine the triangle. In fact, a full automatic strategy that combines Algorithms 3.1 and 3.2 is favorable. It is left for future research.

5. Conclusion and remarks. In this paper, we give a new smoothness condition for polynomials with difference degrees over two adjacent triangular *Bézier* patches. Based on the new smoothness conditions we implement bivariate splines of various degrees for numerical solution of PDEs. The p-version adaptive strategy enables us to efficiently solve Poisson and biharmonic equations with singularity on the boundary.

We offer the following remarks:

Remark 5.1. It is easy to see that the new smoothness conditions can be generalized to deal with polynomials over adjacent simplices in $\mathbf{R}^s$, $s \geq 3$.

Remark 5.2. It is also easy to see that our spline method can be generalized to numerically solve trivariate partial differential equations, e.g., three-dimensional Poisson equations and three-dimensional biharmonic equations.

Remark 5.3. It is possible to use our spline method together with local adaptive procedures (h-version) to solve nonlinear PDEs such as two-dimensional Navier–Stokes equations. It will be useful to capture the details for eddies of fluid flows. We are currently working on this research problem.

REFERENCES

[1] G. Awanou, M. J. Lai, and P. Wenston, *The multivariate spline method for scattered data fitting and numerical solutions of partial differential equations*, in Wavelets and Splines, G. Chen and M. J. Lai, eds., Nashboro Press, Brentwood, TN, 2006, pp. 24–76.

[2] C. de Boor, *B-form basics*, in Geometric Modeling: Algorithms and New Trends, G. Garin, ed., SIAM, Philadelphia, 1987, pp. 131–148.

[3] G. Farin, *Triangular Bernstein-Bézier patches*, Comput. Aided Geom. Design, 3 (1986), pp. 83–127.

[4] M. J. Lai, *Geometric interpretation of smoothness conditions of triangular polynomial patches*, Comput. Aided Geom. Design, 14 (1997), pp. 191–199.

[5] M. J. Lai and L. L. Schumaker, *On the approximation power of bivariate splines*, Adv. Comput. Math., 9 (1998), pp. 251–279.

[6] M. J. Lai and L. L. Schumaker, *Spline Functions over Triangulations*, Cambridge University Press, Cambridge, UK, 2007.

[7] M. J. Lai and P. Wenston, *Bivariate splines for fluid flows*, Comput. & Fluids, 33 (2004), pp. 1047–1073.

[8] L. Evans, *Partial Differential Equations*, AMS, Providence, RI, 1998.

[9] R. Verfürth, *A Review of A Posteriori Error Estimation and Adaptive Mesh-Refinement Techniques*, Wiley, Teubner, Stuttgart, Germany, 1996.

[10] M. C. Rivara, *Algorithms for refining triangular grids suitable for adaptive and multigrid techniques*, Internat. J. Numer. Methods Engrg., 20 (1984), pp. 745–756.

[11] R. Becker and R. Rannacher, *A general concept of adaptivity in finite element methods with applications to problems in fluid and structural mechanics*, in Grid Generation and Adaptive Algorithms, IMA Vol. Math. Appl. 113, Springer-Verlag, New York, 1999, pp. 51–75.

[12] W. Dörfler, *A convergent adaptive algorithm for Poisson's equation*, SIAM J. Numer. Anal., 33 (1996), pp. 1106–1124.

[13] J. M. Melenk, *hp-Finite Element Methods for Singular Perturbations*, Lecture Notes in Math. 1796, Springer-Verlag, Berlin, 2002.

[14] P. Solín, K. Segeth, and I. Dolezel, *High-Order Finite Element Methods*, Chapman & Hall/ CRC Press, Boca Raton, FL, 2003.

[15] G. Strang and F. Fix, *An Analysis of the Finite Element Method*, Prentice–Hall, Englewood Cliffs, NJ, 1973.

[16] P. G. Ciarlet, *The Finite Element Method for Elliptic Problems*, North–Holland, Amsterdam, 1978.