Polygonal spline spaces and the numerical solution of the Poisson equation

Michael S. Floater∗, Ming-Jun Lai†

September 10, 2015

Abstract

It is known that generalized barycentric coordinates (GBCs) can be used to form Bernstein polynomial-like functions over a polygon with any number of sides. We propose to use these functions to form a space of continuous polygonal splines (piecewisely defined functions) of order \(d\) over a partition consisting of polygons which is able to reproduce all polynomials of degree \(d\). Locally supported basis functions for the space are constructed for order \(d \geq 2\). The construction for \(d = 2\) is simpler than the ‘serendipity’ quadratic finite elements that have appeared in the recent literature. The number of basis functions is similar to, but fewer than, those of the virtual element method. We use them for the numerical solution of the Poisson equation on two special types of non-triangular partitions to present a proof of concept for solving PDE’s over polygonal partitions. Numerical solutions based on quadrangulations and pentagonal partitions are demonstrated to show the efficiency of these polygonal spline functions. They can lead to a more accurate solution by using less degrees of freedom than the traditional continuous polynomial finite element method if the solutions are smooth although assembling the mass and stiffness matrices takes more time.

Key Words: Polygonal Finite Elements, Generalized Barycentric Coordinates, Numerical Solution of PDE

1 Introduction

In this paper, we are interested in constructing spline functions (piecewisely defined functions) over an arbitrary polygonal partition \(\Delta\) of a domain \(\Omega \subset \mathbb{R}^2\) and present an application. It is known that one can define bivariate spline functions over a triangulation \(\Delta\) of \(\Omega\) consisting of triangles and use them for the numerical solution of partial differential equations (PDEs), scattered data fitting, etc., see, e.g. [2], [15], [16], and [17]. If a triangulation \(\Delta\) is replaced by a collection \(\Delta\) of polygons, say convex quadrilaterals or pentagons or even a mix of arbitrary polygons such as a Voronoi diagram of a bounded domain, can we extend the bivariate splines to \(\Delta\)? The answer is yes. The method of virtual elements of [6] is one approach, based on solutions to the Laplace equation in each polygon; see also [5] and [19]. The construction of serendipity quadratic finite elements using GBCs in [23] provides another approach. Motivated by these studies we use GBCs to define polygonal splines of order \(d \geq 2\) on a partition \(\Delta\) of convex polygons. Some advantages of these splines are

1. we can partition a polygonal domain \(\Omega\) of interest using various \(n\)-sided polygons including triangles, even though triangles are very convenient for partitioning \(\Omega\);

2. as there are various GBCs which include polynomial, rational, trigonometric, logarithmic functions, etc. (see a summary of various GBCs in a recent survey article [10]), we are allowed to use more general functions than polynomials to approximate known and unknown functions, e.g. solutions of PDEs;

3. there are redundant functions over each polygon when the side number \(n > 3\) which may be useful for certain applications; and

∗Department of Mathematics, University of Oslo, PO Box 1053, Blindern, 0316 Oslo, Norway, email: michaelf@math.uio.no
†Department of Mathematics, University of Georgia, Athens, GA email: mjla@math.uga.edu This author’s research was partially supported by the Simons Foundation under a collaboration grant and the National Science Foundation under grant #DMS–1521537.
our numerical solutions of the Poisson equation indicate that they have some advantages over the standard finite element method. See details in Section 5.4.

The potential advantages of this method come, however, at a price because

(i) the number of GBC-based functions on each polygon grows rapidly as the number of sides of the polygon increases. For example, the number of quadratic functions generated by GBCs in an \( n \)-gon is \( n(n+1)/2 \), which grows quadratically with \( n \).

(ii) the functions are not in general linearly independent, making it difficult to choose suitable degrees of freedom; and

(iii) the evaluation of functions based on GBCs is not as efficient as the evaluation of polynomials. In particular, it is not as efficient as the finite element method for assembling the mass and stiffness matrices associated with weak solution of Poisson equations.

The polygonal splines can be attractive for computation when the overhead for generating the mass and stiffness matrices is not a major component of the computation, e.g. solving a time-dependent diffusion equation over a spatial polygon \( \Omega \subset \mathbb{R}^2 \) (IBVP) or when solving a nonlinear Poisson equation, e.g. \(-\Delta u + f(u) = 0\), where \( f(u) \) is a nonlinear function of \( u \).

One way to curtail the redundancy of the polygonal GBC-based functions is to use a reduced space of functions which still forms a continuous polygonal spline when pieced together, and still reproduces all polynomials of degree \( \leq d \). This is what we do in this paper. Let us first emphasize that this is not a brand new idea and such reduced elements were constructed in the quadratic case in [23], based on the observation that many of the quadratic functions are zero on the boundary of the polygon, and thus do not contribute to inter-element continuity. By forming various linear combinations of the quadratic functions they obtained ‘reduced’ quadratic elements, having only \( 2n \) degrees of freedom, but retaining inter-element continuity. Their construction includes the usual 8-node serendipity element on squares and parallelograms as a special case [24] and [1]. However, their construction is not trivial. One of the purposes of this paper is to offer a simpler way of constructing quadratic elements of this kind. Our main idea is to use not only GBCs but also the local barycentric coordinates defined by triples of consecutive vertices of the polygon. Similar to [23], this generates \( 2n \) functions in the quadratic case. From these functions we also recover the quadratic rational interpolatory element constructed by Wachspress [25], but in explicit form.

To obtain polygonal functions of higher order we specialize to the rational GBCs of Wachspress and use additional degrees of freedom in the interior of the polygon. The dimension of the space of order \( d \) over an \( n \)-gon is then

\[
dn + (d - 1)(d - 2)/2.
\]

The space is able to reproduce all polynomials of degree \( d \). This dimension is smaller than that of the virtual element method of [6] to be explained in Section 4.1, Remark 1. Once we have these functions over each polygon, we can piece them together to make a polygonal spline space \( S_d(\Delta) \) over a collection \( \Delta \) of mixed polygons \( P_n \) with various \( n \geq 3 \). Hence, the dimension of the polygonal spline space of degree \( d \) over a mixed partition \( \Delta \) can be established as in Theorem 4 when \( d = 2 \) for any generalized barycentric coordinates and when \( d \geq 3 \) for Wachspress coordinates.

When \( P_n \) is a triangle \( (n = 3) \), the number in (1) is \( (d + 2)(d + 1)/2 \), which is the number of Bernstein polynomials of degree \( d \) in \( P_n \). When \( \Delta \) is a triangulation, the dimension in (30) in Theorem 4 agrees with the dimension of the continuous spline space of degree \( d \) explained in [15].

In addition to the construction of a locally supported basis, we study the standard interpolation problem using these polygonal spline functions. As one might expect, we can use the space \( S_d(\Delta) \) to do interpolation which will be useful to approximate the functions on the right-hand side of the Poisson equation for numerical solution.

Finally, we shall describe how to implement these functions for the numerical solution of the Poisson equation in the last section. Our numerical results demonstrate the efficiency of this approach. In addition, we introduce a novel refinement scheme for pentagons and show that the root mean square error (RMSE) of polygonal spline solutions over uniformly refined pentagonal partitions decreases faster than that of the standard finite element method over uniformly refined triangulations; see the tables of numerical results in Section 5.4.

The major contributions of this paper are as follows:
a. We provide a simpler construction of serendipity quadratic finite elements than the one in [23] and a construction of higher order finite elements using Wachspress coordinates.

b. The number of basis functions per polygon is smaller than that of the virtual elements of [6] for order $d \geq 2$.

c. We have used our elements to solve the Poisson equation numerically, which extends the study in [19] where polygonal finite elements of order $d = 1$ were used.

d. We introduce a refinement scheme for pentagons which is new to the best of our knowledge.

However, the generation of the mass and stiffness matrices for these polygonal splines takes much more time than for polynomial finite elements. More research needs to be done to overcome this difficulty if these splines are to be useful in practice.

## 2 Background on GBCs

There are many ways to define barycentric coordinates in a polygon with $n$ sides, $n \geq 3$; see [10]. We will consider only convex polygons in this paper. Let $P_n = (v_1, \ldots, v_n)$ be a convex polygon. Any functions $\phi_i : P_n \to \mathbb{R}$, $i = 1, \ldots, n$, will be called generalized barycentric coordinates (GBCs) if, for all $x \in P_n$, $\phi_i(x) \geq 0$ and

$$\sum_{i=1}^{n} \phi_i(x) = 1, \quad \text{and} \quad \sum_{i=1}^{n} \phi_i(x)v_i = x. \quad (2)$$

When $P_n$ is a triangle, the coordinates $\phi_1$, $\phi_2$, $\phi_3$ are the usual barycentric coordinates. For $n > 3$, the $\phi_i$ are not uniquely determined by (2) but they share the basic property that they are piecewise linear on the boundary of $P_n$: $\phi_i(v_j) = \delta_{ij}$ and

$$\phi_i((1 - \mu)v_j + \mu v_{j+1}) = (1 - \mu)\phi_i(v_j) + \mu \phi_i(v_{j+1}), \quad \mu \in [0, 1]. \quad (3)$$

Here, and throughout the paper, we use the cyclic indexing, $\phi_{n+1} := \phi_1$, $\phi_0 := \phi_n$, and so on.

For convenience, let us describe three well known GBCs, which are all smooth in the interior of $P_n$.

**Example 1** (Inverse bilinear coordinates). For $n = 4$, the quadrilateral $P_n$ is the image of a bilinear map from the unit square $[0, 1]^2$, and for each $x \in P_n$, there exists a unique pair $(\alpha, \beta) \in [0, 1]^2$ such that

$$x = (1 - \alpha)(1 - \beta)v_1 + \alpha(1 - \beta)v_2 + \alpha\beta v_3 + (1 - \alpha)\beta v_4.$$  

Thus the four functions

$$\phi_1(x) = (1 - \alpha)(1 - \beta), \quad \phi_2(x) = \alpha(1 - \beta), \quad \phi_3(x) = \alpha\beta, \quad \phi_4(x) = (1 - \alpha)\beta \quad (4)$$

are GBCs. An explicit formula is given in [10].

**Example 2** (Wachspress (rational) coordinates). For general $n \geq 3$, let $n_i \in \mathbb{R}^2$ be the outward unit normal to the edge $e_i = [v_i, v_{i+1}]$, $i = 1, \ldots, n$, and for any $x \in P_n$ let $h_i(x)$ be the perpendicular distance of $x$ to the edge $e_i$, so that

$$h_i(x) = (v_i - x) \cdot n_i = (v_{i+1} - x) \cdot n_i. \quad (5)$$

Let

$$w_i(x) = d_i \prod_{j=1,...,n, j \neq i-1,i} h_j(x), \quad (6)$$

where $d_i$ is the cross product

$$d_i = n_{i-1} \times n_i = \begin{vmatrix} n_i^x & n_i^y \\ n_{i-1}^x & n_{i-1}^y \end{vmatrix},$$

and $n_j = (n_j^x, n_j^y)$. Then, with

$$W = \sum_{j=1}^{n} w_j, \quad (7)$$

the functions $\phi_i = w_i/W$, $i = 1, \ldots, n$, are GBCs, which are rational of degree $(n-2, n-3)$. See [10] for several other representations of these coordinates.
Example 3 (Mean value coordinates). For general $n \geq 3$, and for $\mathbf{x}$ in the interior of $P_n$, let $\alpha_i$ be the angle at $\mathbf{x}$ of the triangle $(\mathbf{v}_i,\mathbf{x},\mathbf{v}_{i+1})$, $i = 1, \ldots, n$, and define (cf. [9])

$$w_i(\mathbf{x}) = \frac{\tan(\alpha_{i-1}/2) + \tan(\alpha_i/2)}{\|\mathbf{x} - \mathbf{v}_i\|} \quad \text{and} \quad W = \sum_{i=1}^{n} w_i,$$

with $\|\cdot\|$ the Euclidean norm in $\mathbb{R}^2$. Then the functions $\phi_i = w_i/W$, which extend continuously to the boundary of $P_n$, are GBCs. See [10] for an explanation of this and how these coordinates extend (though not necessarily positively) to non-convex polygons.

Various other smooth GBCs have been constructed such as Gordon-Wixom and maximum entropy coordinates. We note also that both Wachspress and mean value coordinates have been generalized to 3D (polyhedra); see [10] for a summary.

3 Polygonal Bernstein-Bézier functions

We now study generalizations of Bernstein-Bézier polynomials to a convex polygon, dealing first with the ‘full’ space of functions that has been studied in the literature, and then construct a reduced space.

3.1 The full space

It is well-known that the standard barycentric coordinates on a triangle in $\mathbb{R}^2$ can be used to generate Bernstein-Bézier polynomials of degree $d$ via the multinomial theorem (cf. [8]). Similarly, on a convex polygon in $\mathbb{R}^2$ one can use GBCs to generate analogous functions, whose span will contain polynomials of degree $\leq d$. Such functions were studied by Loop and DeRose [18] with a view to constructing multi-sided surface patches, ‘S-patches’, for modelling geometry.

For a convex polygon $P_n$ with $n \geq 3$ sides, let $\phi_1, \ldots, \phi_n$ be a set of GBCs. For any $d \geq 0$, and any multi-index $\mathbf{j} = (j_1, \ldots, j_n) \in \mathbb{N}_0^n$, with $|\mathbf{j}| := j_1 + \cdots + j_n = d$, let

$$B^d_{\mathbf{j}}(\mathbf{x}) = \frac{d!}{j_1! \cdots j_n!} \phi_1^{j_1}(\mathbf{x}) \cdots \phi_n^{j_n}(\mathbf{x}), \quad \mathbf{x} \in P_n,$$

which we will call a Bernstein-Bézier function.

In the triangular case, $n = 3$, $B^d_1$ is a Bernstein-Bézier polynomial of degree $d$ (cf. [15]). For $n > 3$, $B^d_1$ is no longer a polynomial in general and its function type will depend on the $\phi_i$. Nevertheless, for any $n$, the linear space $\Phi_d(P_n)$ of functions of the form

$$s(\mathbf{x}) = \sum_{|\mathbf{j}|=d} c_{\mathbf{j}} B^d_{\mathbf{j}}(\mathbf{x}), \quad \mathbf{x} \in P_n,$$

with $c_{\mathbf{j}} \in \mathbb{R}$, inherits two important properties from the $\phi_i$.

(P1) $\Pi_d \subset \Phi_d(P_n)$, where $\Pi_d$ is the space of polynomials of degree $\leq d$. This is because, due to (2), any linear polynomial can be expressed as a linear combination of the $\phi_i$. Then based on (8), polynomials of degree $d$ can be generated.

(P2) Due to (3), the function $s(\mathbf{x})$ in (9) is a univariate polynomial of degree $\leq d$ on each edge of the polygon.

In general, the functions $B^d_{\mathbf{j}}$, $|\mathbf{j}| = d$, are not linearly independent, and, as we will see, the dimension of $\Phi_d(P_n)$ is less than the number of these polygonal Bernstein-Bézier functions. Indeed, let us consider the case $d = 2$. Among the pairwise products $\phi_i \phi_j$, $i, j = 1, \ldots, n$, there are $n$ products of the form $\phi_i^2$, $i = 1, \ldots, n$ and $\binom{n}{2}$ of the form $\phi_i \phi_j$, $i \neq j$. So the total number of $B^2_{\mathbf{j}}$, $|\mathbf{j}| = 2$, is

$$D_n = n + \binom{n}{2} = \binom{n+1}{2}.$$
So,
\[ D_3 = 6, \quad D_4 = 10, \quad D_5 = 15, \quad D_6 = 21, \]
and so on. However, the dimension of \( \Phi_2(P_n) \) is lower than \( D_n \) when \( n \geq 4 \). We propose an open problem:
What is the dimension of \( \Phi_d(P_n) \) for various GBCs and \( d \geq 2 \) and \( n > 3 \)?

For \( d = 2 \) and for Wachspress coordinates, we can show

**Theorem 1.**

\[
\dim(\Phi_2(P_n)) = 2n + \frac{(n-2)(n-3)}{2}.
\]  

**Proof.** The products \( \phi^2, \phi_i\phi_{i+1}, i = 1, \ldots, n \), are linearly independent due to their values at \( v_i, (v_i + v_{i+1})/2 \), \( i = 1, \ldots, n \). Therefore we can express \( \Phi_2(P_n) \) as
\[
\Phi_2(P_n) = B_n \oplus I_n,
\]
where
\[
B_n = \text{span}\{\phi^2, \phi_i\phi_{i+1}, i = 1, \ldots, n\} \quad \text{and} \quad I_n = \text{span}\{\phi_i\phi_j: |i - j| \geq 2\}.
\]
The sum is direct because all the functions in \( I_n \) are zero on the entire boundary \( \partial P_n \) while the only function in \( B_n \) with this property is 0.

Since \( \dim(B_n) = 2n \), it remains to determine the dimension of \( I_n \). Recalling \( h_j \) in (6), let \( b \) be the bubble function
\[
b(x) = \prod_{k=1}^{n} h_k(x).
\]  

When \( |i - j| \geq 2 \), the four indices \( i - 1, i, j - 1, j \) are distinct and so from (6),
\[
\phi_i\phi_j = \frac{w_i w_j}{W^2} = \frac{bd_i d_j \nu_{ij}}{W^2},
\]
where
\[
\nu_{ij} = \prod_{k \neq i-1, i, j, j-1} h_k.
\]  

Therefore, \( \dim(I_n) = \dim(H_n) \), where
\[
H_n = \text{span}\{\nu_{ij}: |i - j| \geq 2\}.
\]
Observe that \( \nu_{ij} \in \Pi_{n-4} \) since \( h_k \) is a linear polynomial and so \( H_n \subset \Pi_{n-4} \). In fact we will show that
\[
H_n = \Pi_{n-4},
\]  

(14)
from which we deduce that \( \dim(H_n) = (n-2)(n-3)/2 \) which yields (11). We now present a proof of (14).

For \( n = 4 \) it is easy as (13) is an empty product, i.e., \( \nu_{13} = \nu_{24} = 1 \). For \( n \geq 5 \), notice that each edge of the polygon can be parallel with at most one other edge. Therefore, any three lines \( h_k = 0 \) intersect in at least two points from which it follows that any three \( h_k \) are linearly independent, and thus form a basis for \( \Pi_1 \). In the case \( n = 5 \), the \( \nu_{ij} \) are linear polynomials, which are \( \nu_{i,i+2} = h_{i+3}, i = 1, \ldots, 5 \). Since any three of them span \( \Pi_1 \) (14) follows.

Finally, for \( n \geq 6 \), consider the monomial \( x^{n-4} \) where \( x = (x, y) \). For any distinct \( r, s, t \) in \( \{1, \ldots, n\} \), we can write it as
\[
x^{n-4} = x^{n-5} = (c_r h_r + c_s h_s + c_t h_t)x^{n-5} = c_r h_r x^{n-5} + c_s h_s x^{n-5} + c_t h_t x^{n-5},
\]
for coefficients \( c_r, c_s, c_t \in \mathbb{R} \), and a similar decomposition can be applied to any other monomial of degree \( n - 4 \), and so if, for any \( r = 1, \ldots, n \), we can express \( x^{n-5} \) as a linear combination of the polynomials
\[
\mu_{i,j} = \prod_{k \neq r, t-1, u, j-1, j} h_k,
\]

5
where \( r, i - 1, i, j - 1, j \) are distinct, equation (14) will follow. Thus it is sufficient to show that the polynomials \( \mu_{ij} \) span \( \Pi_{n-5} \), and we may assume that \( r \) is fixed and \( r = n \). The possible indices of the \( \mu_{ij} \), with \( i < j \), are \( i = 2, \ldots, n - 3 \) and \( j = i + 2, \ldots, n - 1 \), and thus the number of \( \mu_{ij} \) is \((n - 4)(n - 3)/2\), which is precisely the dimension of \( \Pi_{n-5} \). So, showing that the \( \mu_{ij} \) span \( \Pi_{n-5} \) is equivalent to showing that they are linearly independent.

We thus complete the proof by showing that the \( \mu_{ij} \) are linearly independent. We prove this by induction on \( n, n \geq 5 \). It is true for \( n = 5 \) since there is only one \( \mu_{ij} \), namely \( \mu_{2,4} \). For \( n > 5 \) suppose that

\[
\sum_{i=2}^{n-3} \sum_{j=i+2}^{n-1} c_{ij} \mu_{ij}(x) = 0, \quad x \in \mathbb{R}^2. \tag{15}
\]

Let \( l_k \) denote the line \( h_k = 0 \) for each \( k = 1, \ldots, n - 1 \). Suppose first that \( l_1 \) and \( l_{n-1} \) are not parallel, and let \( x = l_1 \cap l_{n-1} \). Then all the \( \mu_{ij} \) are zero at \( x \) except \( \mu_{2,n-1} \), and so \( c_{2,n-1} = 0 \) in (15). Otherwise, \( l_1 \) and \( l_{n-1} \) are parallel, in which case let \( x_0 = (v_1 + v_n)/2 \) and let \( d \) be the unit vector parallel to \( l_1 \) and \( l_{n-1} \) in the direction away from the polygon, and consider the point \( x = x_0 + \lambda d \) as \( \lambda \to \infty \). From (5), \( h_1(x) \) and \( h_{n-1}(x) \) are constant in \( \lambda \), while

\[ h_k(x) \sim -\lambda d \cdot n_k, \quad k = 2, \ldots, n - 2. \]

Therefore, since

\[ \frac{\mu_{i,j}}{\mu_{2,n-1}} = \frac{h_1 h_2 h_{n-2} h_{n-1}}{h_{i-1} h_{i-1} h_{j-1} h_j}, \]

we see that if \((i, j) \neq (2, n - 1)\),

\[ \frac{\mu_{i,j}(x)}{\mu_{2,n-1}(x)} \to 0 \quad \text{as} \quad \lambda \to \infty, \]

and dividing (15) by \( \mu_{2,n-1}(x) \) and letting \( \lambda \to \infty \) shows again that \( c_{2,n-1} = 0 \).

Next, let \( x = l_2 \cap l_{n-1} \) if \( l_2 \) and \( l_{n-1} \) are not parallel. Then all the \( \mu_{ij} \) are zero at \( x \) except \( \mu_{2,n-1} \) and \( \mu_{3,n-1} \). Since \( c_{2,n-1} = 0 \) this means that \( c_{3,n-1} = 0 \) as well. The previous argument for parallel lines also applies to show that \( c_{3,n-1} = 0 \) also if \( l_2 \) and \( l_{n-1} \) are parallel. We now continue in this way with the pairs of lines \( l_i \) and \( l_{n-1} \) for \( i = 3, \ldots, n - 4 \), to show sequentially that \( c_{i,n-1} = 0 \) for \( i = 4, \ldots, n - 3 \). This reduces (15) to

\[ h_{n-1}(x) \sum_{i=2}^{n-4} \sum_{j=i+2}^{n-2} c_{ij} \mu_{i,j}(x) = 0, \quad x \in \mathbb{R}^2, \tag{16} \]

where the \( \mu_{ij} \) are the analogs of \( \mu_{ij} \) for the convex polygon with vertices \( v_1, \ldots, v_{n-1} \). By the induction hypothesis, the \( \mu_{ij} \) are linearly independent, and therefore the \( c_{ij} \) in the sum are zero. \( \square \)

Thus at least for Wachspress coordinates, the dimension of \( \Phi_2(P_n) \), while lower than the number of functions \( B_j^2, |j| = 2 \), nevertheless grows quadratically with \( n \). In fact, in this case, the redundancy of the \( B_j^2, |j| = 2 \), is

\[ D_n - \dim(\Phi_2(P_n)) = n - 3. \]

We still do not know the dimension of \( \Phi_2(P_n) \) for other GBCs.

### 3.2 A reduced space

Since the dimension of the whole space \( \Phi_d(P_n) \) is high for \( d \geq 2 \), we would like to construct a subspace \( \Psi_d(P_n) \) that satisfies the same two properties (P1) and (P2): a ‘reduced’ space. Our idea for this is to expand any polynomial of degree \( \leq d \) from its polar form. Recall that any \( p \in \Pi_d \) has a \( d \)-variate polar form (or blossom) \( P(x_1, \ldots, x_d) \) for points \( x_1, \ldots, x_d \in \mathbb{R}^2 \) (see [21], [20]), which is uniquely defined by the three properties: (i) it is symmetric in the variables \( x_1, \ldots, x_d \); (ii) it is multi-affine, i.e., affine in each variable while the others are fixed; and (iii) it has the diagonal property,

\[ P(x^{[d]}):= \left.P(x, \ldots, x) \right|_{x_d} = p(x). \tag{17} \]
Given a polynomial \( p \in \Pi_d \), we use (17) and expand one of the \( x \) variables using the GBCs \( \phi_i \) and the remaining \( d-1 \) variables using the local barycentric coordinates, \( \lambda_{i,j}, j = -1, 0, 1 \), defined by

\[
x = \sum_{j=-1}^{1} \lambda_{i,j}(x)v_{i+j}, \quad 1 = \sum_{j=-1}^{1} \lambda_{i,j}(x).
\] (18)

Consider first the quadratic case.

**Theorem 2.** Let

\[
F_i = \phi_i \lambda_{i,0}, \quad F_{i,1} = \phi_i \lambda_{i,1} + \phi_{i+1} \lambda_{i+1,-1}, \quad i = 1, \ldots, n,
\]

and

\[
\Psi_2(P_n) = \text{span}\{F_i, F_{i,1}, i = 1, \ldots, n\},
\] (19)

Then \( \Pi_2 \subset \Psi_2(P_n) \subset \Phi_2(P_n) \).

**Proof.** For any \( p \in \Pi_2 \), expanding one of the two \( x \) variables in (17) using the GBCs \( \phi_i \) gives

\[
p(x) = P(x, x) = \sum_{i=1}^{n} \phi_i(x)P(v_i, x).
\] (20)

Then expanding the remaining \( x \) variable using the coordinates \( \lambda_{i,j} \) gives

\[
p(x) = \sum_{i=1}^{n} \phi_i(x) \sum_{j=-1}^{1} \lambda_{i,j}(x)P(v_i, v_{i+j}).
\] (21)

Considering the two cases \( i = j \) and \( i \neq j \), and since \( P(v_i, v_i) = P(v_i) \), we thus obtain

\[
p(x) = \sum_{i=1}^{n} p(v_i)F_i(x) + \sum_{i=1}^{n} p(v_i, v_{i+1})F_{i,1}(x),
\]

with the \( F_i \) and \( F_{i,1} \) as in (19). This shows that \( \Pi_2 \subset \Psi_2(P_n) \). Since each \( \lambda_{i,j} \) can be expressed as a linear combination of the \( \phi_k, k = 1, \ldots, n \), it also follows that \( F_i, F_{i,1} \in \Phi_2(P_n) \). \( \square \)

**Example 4.** In order to understand these functions \( F_i \) and \( F_{i,1} \) better, we plot them in Figures 1 and 2. These are \( F_i \) and \( F_{i,1} \) based on Wachspress coordinates. The graphs of \( F_i \) and \( F_{i,1} \) using mean value coordinates are very similar.

![Figure 1: Some F_i’s based on Wachspress coordinates](image)

On the edge \([v_i, v_{i+1}]\), the only \( F \)-functions that are non-zero are \( F_i, F_{i,1}, \) and \( F_{i+1}, \) and if

\[
x = (1 - \mu)v_i + \mu v_{i+1}, \quad 0 \leq \mu \leq 1,
\] (22)

then

\[
F_i(x) = (1 - \mu)^2, \quad F_{i,1}(x) = 2\mu(1 - \mu), \quad F_{i+1}(x) = \mu^2.
\] (23)

Thus the \( 2n \) functions \( F_i \) and \( F_{i,1} \) are linearly independent.

In constructing a reduced space for order \( d \geq 3 \) the polar form expansion involves some functions that are zero on the boundary of \( P_n \), and some of these could be linearly dependent. However, by specializing the coordinates \( \phi_k \) to be Wachspress coordinates, we are able to make a precise statement about these dependencies and ascertain the function space and its dimension.
Theorem 3. For \(d \geq 3\) and with Wachspress coordinates \(\phi_i\), let

\[
F_i = \phi_i \lambda_{i,0}^{d-1}, \quad i = 1, \ldots, n,
\]
\[
F_{i,k} = \binom{d-1}{k} \phi_i \lambda_{i,1}^{d-1-k} + \binom{d-1}{k-1} \phi_{i+1} \lambda_{i+1,0}^{d-k} \lambda_{i+1,1}, \quad i = 1, \ldots, n, \quad k = 1, \ldots, d-1,
\]

and

\[
\Psi_d(P_n) := \text{span}\{F_i\} \oplus \text{span}\{F_{i,k}\} \oplus \frac{b}{W} \Pi_{d-3},
\]

with \(W\) as in (7) and \(b\) in (12). Then \(\Pi_d \subset \Psi_d(P_n) \subset \Phi_d(P_n)\).

Proof. For any \(p \in \Pi_d\) with \(d \geq 3\), expanding one of the \(x\) variables in (17) using any GBCs \(\phi_i\) gives

\[
p(x) = \sum_{i=1}^{n} \phi_i(x)q_i(x),
\]

where

\[
q_i(x) = P(v_i, x^{d-1}).
\]

Then expanding the remaining \(x\)'s using the \(\lambda_{ij}\) gives

\[
q_i(x) = \sum_{j_1, \ldots, j_{d-1}=-1}^{1} \lambda_{i,j_1}(x) \cdots \lambda_{i,j_{d-1}}(x)P(v_i, v_{i+j_1}, \ldots, v_{i+j_{d-1}}),
\]

and so \(q_i\) is the Bernstein-Bézier polynomial

\[
q_i(x) = \sum_{|j|=d-1} P(v_{i-j_1}^{[j_1]}, v_{i+1}^{[j_2]}, v_{i+j_1}^{[j_3]})B_j^{d-1}(x),
\]

where

\[
B_j^{d-1} = \frac{(d-1)!}{j_1!j_2!j_3!} \lambda_{i-1,j_1}^{j_1} \lambda_{i,j_2}^{j_2} \lambda_{i+1,j_3}^{j_3}.
\]

By grouping together the cases \(j = (0, d-1, 0), j = (k, d-1-k, 0)\) and \(j = (0, d-1-k, k), k = 1, \ldots, d-1,\) we obtain

\[
q_i(x) = P(v_i^{[d]}, \lambda_{i,0}^{d-1}(x))
\]
\[+ \sum_{j=-1,1}^{d-1} \sum_{k=1}^{1} P(v_i^{[d-k]}, v_{i+j}^{[k]}) \binom{d-1}{k} \lambda_{i,0}^{d-1-k}(x) \lambda_{i,j}^{k}(x)
\]
\[+ \lambda_{i,-1}(x) \lambda_{i,1}(x) r_i(x)
\]

for some polynomial \(r_i \in \Pi_{d-3}\). 

Figure 2: Some \(F_{i,1}\)'s based on Wachspress coordinates
Now, for Wachspress coordinates \( \phi_i \), the polynomial weight function \( w_i \) in (6) contains the factors \( h_j \), \( j \neq i-1, i \), and \( \lambda_i-1 \) contains the factor \( h_i \) and \( \lambda_{i+1} \) contains the factor \( h_{i+1} \), and so there is some constant \( K_i \) such that

\[
\phi_i(x)\lambda_{i-1}(x)\lambda_{i+1}(x) = K_i \frac{b(x)}{W(x)},
\]

with \( b \) the bubble function in (12), and \( W \) as in (7). Therefore, substituting (26) into (25) gives

\[
p = \sum_{i=1}^{n} p(v_i)F_i + \sum_{i=1}^{n} \sum_{k=1}^{d-1} P(v_i^{[d-k]}, v_{i+1}^{[k]})F_{i,k} + \frac{b}{W}r,
\]

(27)

with the \( F_i \) and \( F_{i,k} \) as in (24) and \( r \) some polynomial in \( \Pi_{d-3} \). This shows that \( \Pi_d \subset \Psi_d(P_n) \). The sum in (24) is direct due to \( b \) being zero on the edges of \( P_n \) and the \( F_{i,k} \) being zero at the vertices of \( P_n \).

**Example 5.** To help understand these \( F_i, F_{i,k} \) better, we plot these \( F_i \) (the left one on the top row) and \( F_{i,k}, k = 1, 2 \) of order 3 as well as a \( \psi = b/W \) (the right one on the second row) in Figure 3.

![Figure 3: Some \( F_{i,k} \)'s of order 3](image)

In analogy to the quadratic case, if \( x \) is the boundary point in (22), then

\[
F_i(x) = b_0^d(\mu), \quad F_{i,k}(x) = b_k^d(\mu), \quad k = 1, \ldots, d-1, \quad F_{i+1}(x) = b_{d}^d(\mu),
\]

where

\[
b_k^d(\mu) = \binom{d}{k} \mu^k (1 - \mu)^{d-k}, \quad k = 0, 1, \ldots, d,
\]

(28)

the univariate Bernstein polynomials of degree \( d \) in \( \mu \), and all remaining \( F_i \)'s are zero on \( e_i \).

By the linear independence of the \( b_k^d \), the \( n \) functions \( F_i \) and the \( (d-1)n \) functions \( F_{i,k} \) are linearly independent. By the dimension of \( \Pi_{d-3} \), we have

\[
\dim \Psi_d(P_n) = dn + (d-1)(d-2)/2.
\]

**Remark 1.** This space dimension is similar to, but less than, that of the virtual element space of [6] which is

\[
dn + d(d-1)/2.
\]

The functions in [6, Sec. 4] are defined by the properties: (i) they are continuous on the boundary of \( P_n \), (ii) their restriction to an edge of \( P_n \) is a univariate polynomial of degree \( \leq d \), and (iii) their Laplacian is a polynomial of degree \( \leq d-2 \) (interpreted in the case \( d = 1 \) to mean they are harmonic). From (iii) it trivially follows that this space, like \( \Psi_d(P_n) \), contains \( \Pi_d(P_n) \).
4 Polygonal spline spaces

Recall that Bernstein-Bézier polynomials on triangles can be pieced together (continuously and even smoothly) over a collection of triangles to form a bivariate spline function (cf. e.g. [8, 15]). Similarly, polygonal Bernstein-Bézier functions can be pieced together continuously over a collection of convex polygons to form a bivariate polygonal spline function.

Let $\Delta$ be a collection of convex polygons and let $\Omega = \bigcup_{P \in \Delta} P$ be the domain consisting of these polygons in $\Delta$. We assume that the interiors of any two polygons $P$ and $\tilde{P}$ do not intersect and the intersection of $P$ and $\tilde{P}$ is either their common edge or common vertex if the intersection is not empty. Then let

$$ S_d(\Delta) = \{ s \in C(\Omega), \ s|_{P_n} \in \Psi_d(P_n), \forall P_n \in \Delta \}, $$

(29)

where, for general GBCs,

$$ \Psi_1(P_n) := \Phi_1(P_n) = \text{span}\{\phi_i, i = 1, \ldots, n\}, $$

and $\Psi_2(P_n)$ is defined by (19), and for Wachspress coordinates and $d \geq 3$, $\Psi_d(P_n)$ is defined in (24).

4.1 Space dimension

By the results of the previous section, we have

**Theorem 4.** For any GBCs,

$$ \dim(S_1(\Delta)) = \#(V), \quad \dim(S_2(\Delta)) = \#(V) + \#(E), $$

and for Wachspress coordinates and $d \geq 3$,

$$ \dim(S_d(\Delta)) = \#(V) + (d-1)\#(E) + \frac{(d-1)(d-2)}{2} \#(\triangle), $$

(30)

where $V$ and $E$ are the sets of vertices and edges of $\Delta$, respectively.

**Proof.** We prove this only for the quadratic case, $d = 2$. The other cases are similar. For each vertex $v \in \Delta$, let $\mathcal{P}(v)$ be the set of polygons in $\Delta$ sharing the vertex $v$. For each $P \in \mathcal{P}(v)$, recall the functions $F_i$ and $F_{i,1}$ of (19). Since $v$ is a vertex of $P$, we let $F_{v,P}$ be the function $F_i$ associated with vertex $v$. Then we define the vertex spline

$$ S_v(x) = \begin{cases} 
F_{v,P}(x) & \text{if } x \in P \in \mathcal{P}_v \\
0 & \text{otherwise} 
\end{cases} $$

(31)

Its support is the union of the polygons in $\mathcal{P}_v$. We now show that $S_v$ is continuous on $\Omega$. For any two polygons $P$ and $\tilde{P}$ in $\mathcal{P}_v$ which share a common edge $[v, u]$, we see from (23) that at the point $x = (1-\mu)v + \mu u$, with $\mu \in [0, 1]$,

$$ F_{v,P}(x) = (1 - \mu)^2 = F_{v,\tilde{P}}(x). $$

As all $F_{v,P}$ are zero on the edges of $P$ which do not connect to $v$, we know that $S_v$ is zero on the boundary of the union of the polygons in $\mathcal{P}_v$, and thus $S_v$ is continuous.

Figure 4: Vertex and edge splines
Next for each interior edge $e = [v, u]$ in $\Delta$, there are two polygons $P$ and $\tilde{P}$ sharing $e$. Let $F_{e,P}$ be the function $F_{i,1}$ of (19) on $P$ associated with $e$, and $F_{e,\tilde{P}}$ the one on $\tilde{P}$ associated with $e$. Then we define the edge spline

$$S_e(x) = \begin{cases} 
F_{e,P}(x), & \text{if } x \in P \\
F_{e,\tilde{P}}(x), & \text{if } x \in \tilde{P} \\
0, & \text{otherwise},
\end{cases}$$

(32)

whose support is $P \cup \tilde{P}$. Since $S_e$ is zero on the boundary of $P \cup \tilde{P}$, and since at the point $x = (1 - \mu)v + \mu u$, $\mu \in [0, 1]$, $F_{e,P}(x) = 2\mu(1 - \mu) = F_{e,\tilde{P}}(x)$, it follows that $S_e$ is continuous on $\Omega$. Fig. 4 shows examples of $S_v$ and $S_e$.

For a boundary edge $e$, we define $S_e$ similarly. Since the functions $S_v$ and $S_e$ form a basis for $S_2(\Delta)$, the dimension of $S_2(\Delta)$ follows.

4.2 Interpolation

We now explain how to use the functions in $S_2(\Delta)$ for interpolation. Given the values $f(v)$ of a function $f$ defined at the vertices $v \in V$ of $\Delta$ there is a unique function $S_f \in S_1(\Delta)$ satisfying

$$S_f(v) = f(v), \quad v \in V,$$

(33)

namely

$$S_f = \sum_{v \in V} f(v) S_v,$$

(34)

where $S_v$ is a vertex spline in $S_1(\Delta)$, analogous to (31). From the definition of GBCs, $S_f$ reproduces linear polynomials. We now extend the study to find Lagrange functions for interpolation for degree $d \geq 2$.

Consider $d = 2$. Let us start with a single polygon $P_n$. Following an observation of [23] we can transform the functions $F_i$ and $F_{i,1}$ into an interpolatory basis, interpolating at the vertices $v_j$ and the midpoints $v_{j,1} := (v_j + v_{j+1})/2$.

Indeed, since

$$F_i(v_j) = \delta_{ij}, \quad F_i(v_{i-1,1}) = F_i(v_{i,1}) = 1/4, \quad F_i(v_{j,1}) = 0, \quad j \neq i - 1, i,$$

and

$$F_{i,1}(v_j) = 0, \quad F_{i,1}(v_{j,1}) = \delta_{ij}/2,$$

we obtain

**Theorem 5.** For a fixed polygon $P_n$ in $\Delta$, the functions defined by

$$L_i = -\frac{1}{2}F_{i-1,1} + F_i - \frac{1}{2}F_{i,1}, \quad i = 1, \ldots, n,$$

$$L_{i,1} = 2F_{i,1}, \quad i = 1, \ldots, n,$$

(35)

are an interpolatory basis for $\Psi_2(P_n)$, i.e., for any function $f : P_n \to \mathbb{R}$, if $s_f \in \Psi_2(P_n)$ is defined by

$$s_f(x) = \sum_{i=1}^n f(v_i)L_i(x) + \sum_{i=1}^n f(v_{i,1})L_{i,1}(x), \quad x \in P_n,$$

(36)

then $s_f = f$ for all $f \in \Psi_2(P_n)$.  

11
We are now ready to consider \( d \geq 3 \) with Wachspress coordinates. Let \( \ell_i^d, i = 0, 1, \ldots, d \), be the univariate Lagrange fundamental polynomials of degree \( \leq d \) w.r.t. the points \( j/d, j = 0, 1, \ldots, d \), satisfying \( \ell_i^d(j/d) = \delta_{ij} \). Then the inner polynomials in (28) can be expressed as

\[
b_i^d = \sum_{j=1}^{d-1} b_i^d(j/d) \ell_j^d, \quad i = 1, \ldots, d - 1.
\]

Then if \( A = [a_{ij}] \) is the inverse of the matrix \([b_i^d(j/d)]_{i,j=1,\ldots,d}\), we know that

\[
\ell_i^d = \sum_{j=1}^{d-1} a_{ij} b_j^d, \quad i = 1, \ldots, d - 1.
\]

It follows that the functions

\[
L_{i,k} = \sum_{r=1}^{d-1} a_{kr} F_{i,r}, \quad i = 1, \ldots, n, \quad k = 1, \ldots, d - 1,
\]

\[
L_i = F_i - \sum_{k=1}^{d-1} (1 - k/d)^d (L_{i,k} + L_{i-1,d-k}), \quad i = 1, \ldots, n,
\]

are an interpolatory basis for the ‘boundary’ part

\[
\Psi_{d,B}(P_n) := \text{span}\{F_i\} \oplus \text{span}\{F_{i,k}\}
\]

of \( \Psi_d(P_n) \), with respect to the side nodes,

\[
v_{i,k} := \frac{(d - k)v_i + kv_{i+1}}{d}, \quad k = 1, \ldots, d - 1.
\]

In other words

\[
I_B f(x) := \sum_{i=1}^{n} f(v_i) L_i(x) + \sum_{i=1}^{n} \sum_{k=1}^{d-1} f(v_{i,k}) L_{i,k}(x), \quad x \in P_n,
\]

is the unique interpolant to \( f \) in \( \Psi_{d,B}(P_n) \). For example, when \( d = 3 \),

\[
\begin{bmatrix}
  b_1^3 \\
  b_2^3
\end{bmatrix} = \frac{2}{9} \begin{bmatrix}
  2 & 1 \\
  1 & 2
\end{bmatrix} \begin{bmatrix}
  \ell_1^2 \\
  \ell_2^2
\end{bmatrix},
\]

and then

\[
\begin{bmatrix}
  \ell_1^3 \\
  \ell_2^3
\end{bmatrix} = \frac{3}{2} \begin{bmatrix}
  2 & -1 \\
  -1 & 2
\end{bmatrix} \begin{bmatrix}
  b_1^2 \\
  b_2^2
\end{bmatrix},
\]

and so the interpolatory basis is

\[
L_{i,1} = \frac{3}{2} (2 F_{i,1} - F_{i,2}), \quad L_{i,2} = \frac{3}{2} (-F_{i,1} + 2 F_{i,2}), \quad i = 1, \ldots, n,
\]

\[
L_i = F_i - \frac{8}{27} (L_{i,1} + L_{i-1,2}) - \frac{1}{27} (L_{i,2} + L_{i-1,1}), \quad i = 1, \ldots, n.
\]

**Example 6.** We plot the \( L_i \) (on the top row) and \( L_{i,k} \), \( k = 1, 2 \) of order 3 (on the second row) in Figure 5.

Interpolation in the whole space \( \Psi_d(P_n) \) can now be carried out in two steps: by first interpolating \( f \) using \( \Psi_{d,B}(P_n) \) and then interpolating the error on a set of unisolvent points in the interior of \( P_n \) and combining the two terms. Let \( \xi_j \in P_n, j \in \mathbb{N}_0 \), \( |j| = d - 3 \), be a set of distinct points which are unisolvent for polynomial interpolation in \( \Pi_{d-3} \). Let \( \tilde{L}_j \in \Pi_{d-3}, |j| = d - 3 \), be the Lagrange fundamental polynomials for these points, i.e., \( \tilde{L}_j(\xi_k) = \delta_{j,k} \). Then

\[
s_f := I_B f + \frac{b}{W} \sum_{|j|=d-3} W(\xi_j) \frac{b(\xi_j)}{b(\xi_j)} (f(\xi_j) - I_B f(\xi_j)) \tilde{L}_j.
\]
is an interpolant to $f$ in $\Psi_d(P_n)$.

For $d = 3$ we could choose any single point $\xi_{000}$ in $P_n$. For $d \geq 4$ we could choose a triangular grid of points

$$\xi_j := \frac{j_1 p_1 + j_2 p_2 + j_3 p_3}{d - 3}, \quad |j| = d - 3,$$

where $T = [p_1, p_2, p_3]$ is a triangle inside $P_n$. Figure 6 shows possible configurations of boundary and interior points for interpolation.

This explains how to use polygonal Bernstein-Bézier functions to do interpolation on a single polygon. Clearly, we can extend the interpolation scheme to a polygonal partition $\Delta$ so that we can use the spline functions in $S_d(\Delta)$ to interpolate any continuous function $f$ at the vertices of a polygonal partition.

**Remark 2.** Certainly, it is interesting to study a smooth polygonal spline space, e.g.,

$$S^r_d(\Delta) = \{ s \in C^r(\Omega), s|_{P_n} \in \Psi_d(P_n), \forall P_n \in \Delta \}$$

for $r \geq 1$. This is one of our future research topics.

## 5 Numerical Solution of Poisson Equation

We will use the polygonal splines of order $d \geq 2$ to solve the Dirichlet boundary value problem:

$$\begin{cases} -\Delta u = f, & x \in \Omega \subset \mathbb{R}^2 \\ u = g, & x \in \partial \Omega, \end{cases}$$

Figure 5: Some $L_i$ and $L_{i,k}$ of order 3

Figure 6: Linear, quadratic, cubic, and quartic elements
where $\Omega$ is a polygonal domain and $\Delta$ is the Laplace operator. Our main point is to provide a proof of concept that these splines can be used to solve PDEs numerically in analogy to any standard finite element method, macro element method, virtual element method, multivariate splines, and polygonal finite elements of order 1, etc., see for example, [7], [4], [3], [2], [5], [6], [19]. Although we employ the standard weak formulation of the Poisson equation, we shall use the constraint minimization approach to find a numerical solution. This approach is different to the traditional finite element method and thus we give an outline of it below.

### 5.1 Basics on Weak Solution

For simplicity, let us illustrate the approach with some simple partitions. We will take $\Delta$ to consist either of quadrilaterals or of pentagons. A method of generating a quadrangulation of $\Omega$ can be found in, e.g., [15]. For a pentagonal partition we will propose a refinement scheme later in this section.

Now let $S_d(\Delta)$ be the collection of polygonal spline functions of degree $d \geq 1$ over $\Delta$. Let $h = |\Delta|$ be the size of $\Delta$. That is, $h$ is the longest of the length of edges of $\Delta$. Let $B(u, v) = \int_\Omega \nabla u \cdot \nabla v dx$ be the standard bilinear form associated with the Poisson equation. Our numerical method is to solve the following weak solution: for $u_h \in S_d(\Delta) \cap H^1_0(\Omega)$ such that

$$B(u_h, v_h) = (f, v_h), \forall v_h \in S_d(\Delta) \cap H^1_0(\Omega). \tag{41}$$

Using the standard argument in the finite element method (see the proof of the C´ ea Lemma, cf. [7] and [3]), we have

**Lemma 1.** Let $u_h$ be approximate weak solution satisfying (41) in $V_h = S_d(\Delta) \cap H^1_0(\Omega)$. Then

$$\|u - u_h\|_{H^1_0(\Omega)} \leq \frac{1}{m} \min_{v \in V_h} \|u - v\|_{H^1_0(\Omega)},$$

where $m > 0$ is a constant dependent on the domain $\Omega$ (see p. 104 in [7] or p. 62 in [3].)

As $S_d(\Delta)$ is a space containing all piecewise polynomials of degree $d$ over $\Delta$. It is known, i.e. by Bramble-Hilbert Lemma that if $u \in H^{d+1}(\Omega)$, there exists an interpolant $S_u \in S_d(\Delta)$ such that

$$\|u - S_u\|_{L^2(\Omega)} \leq C_1 h^{d+1} \text{ and } \|
abla (u - S_u)\|_{L^2(\Omega)} \leq C_2 h^d \tag{42}$$

for some positive constants $C_1$ and $C_2$ which are dependent on the chunkiness of the underlying polygons $\Delta$ (cf. [3], [15]) as well as the semi-norm of $u$ in $H^{d+1}(\Omega)$. Thus, combining Lemma 1 and the second inequality in (42), one has

**Theorem 6.** *(The serendipity finite element approximation of the weak solution)* Suppose that the weak solution $u \in H^{d+1}(\Omega)$. Then

$$\|u - u_h\|_{H^1_0(\Omega)} \leq \frac{C_2}{m} |u|_{H^{d+1}(\Omega)} h^d,$$

where $C_2$ is a positive constant dependent on the chunkiness of the underlying polygons $\Delta$.

### 5.2 Numerical Solution of the Poisson Equation

We have implemented these polygonal spline functions of order $d$ with $d = 2, d = 3$ over arbitrary partition $\Delta$ in MATLAB for numerical solution of Poisson equations. Our implementation follows the approach discussed in [2] based on multivariate splines for numerical solutions of many linear and nonlinear PDE. See similar approaches in [17] and [13].

As the approach is not a standard one in finite element literature, we summarize the steps as follows. For convenience, let us focus on $d = 2$. A Matlab program is attached in Appendix.

1. First, we use a coefficient vector $c$ ($2n$ per polygon with $n$ sides over the list of polygons in $\Delta$) to represent a polygonal spline function in $S_d(\Delta)$. Note that the vector $c$ is redundant which is different from traditional finite element method. An advantage of such vector $c$ is that it allows us to compute the mass and stiffness matrix in parallel as well as graph display can be done in parallel, e.g. using GPUs.
(2) Secondly, when two polygons share a common edge $e$, there are 3 pairs of the polygonal spline functions supported on these two polygons and each pair have the same value on the common edge $e$ due to the continuity of our polygonal splines. For each pair, they are belong to the same locally supported polygonal spline function and thus, the coefficients associated with each pair should be same. In this way we build a continuity condition matrix $H$ such that $Hc = 0$ represents all these continuity conditions among the coefficients in $c$.

(3) Thirdly we form the mass and stiffness matrices by using tensor product of Gauss quadrature formula with order 5 or larger. Note that our mass and stiffness matrices are blockly diagonal due to the redundancy of vector $c$. For example, the mass matrix $M = \text{diag}(M_p, p \in \Delta)$ with $M_p = [M_{p,i,j}]_{i,j=1,\ldots,2n}$ with $n = n(p)$ being the number of sides of $p$, where

$$M_{p,i,j} = \int_p L_iL_j\,dxdy$$

with $L_i = L_{i,p}$ being interpolatory polygonal spline functions in (35) which are defined on $p$. When $p = P_n$ is a quadrilateral with $n = 4$, we simply use the tensor product of Gauss quadrature formula with order 5 or larger. When $n > 4$, we first divide $p$ into a few (strictly convex or degenerated) quadrilaterals and then apply for the tensor product of Gauss quadrature formula to each quadrilateral and add these numerical values together. For example, when $p$ is a pentagon, we first choose the middle point from the longest edge of $p$ and divide $p$ into two quadrilaterals and then apply the Gauss quadrature of order 5 over each of two quadrilaterals and add two numerical values together. Similar for the stiffness matrix $K$.

(4) Then we use the interpolatory scheme discussed in a previous section to find polygonal interpolant $I_f$ and then use the mass matrix to generate an approximation to the right-hand side $(f, L_{i,p})$ in (41) for $v = L_{i,p}$. That is, we approximate $f$ by $I_f = \sum_{p\in\Delta} \sum_{i=1}^{n(p)} f(v_i(p))L_{2i-1} + f((v_i(p) + v_{i+1}(p))/2)L_{2i}$ and then compute

$$\langle f, L_{i,p} \rangle = \sum_{j=1}^{n(p)} f(v_j(p)) \int_p L_{2j-1}L_{i,p}\,dxdy + f((v_i(p) + v_{i+1}(p))/2) \int_p L_{2j}L_{i,p}\,dxdy,$$

where the two integrals are already calculated in the mass matrix. Here $v_i(p)$, $i = 1, \ldots, n$ are vertices of $p$. These are entries for the right-hand side vector $F$.

(5) Fifthly, boundary conditions are also generated by using the interpolation scheme. We can write them together in terms of a system of linear equations $Bc = G$.

(6) Finally, it is known that the Poisson equation (40) is the Euler-Lagrange equation of the following minimization

$$\min_{u \in H_0^1(\Omega)} \frac{1}{2} \int_{\Omega} |\nabla u|^2\,dxdy - \int_{\Omega} f(x,y)u(x,y)\,dxdy. \quad (43)$$

In the finite dimensional setting, the weak solution (41) is the Euler-Lagrange equation of the minimization

$$\min_{u_h \in S_h(\Delta) | u_h \in H_0^1(\Omega)} \frac{1}{2} \int_{\Omega} |\nabla u_h|^2\,dxdy - \int_{\Omega} f(x,y)u(x,y)\,dxdy. \quad (44)$$

In terms of the coefficient vector $c$ of polygonal spline functions, we numerically solve the following constrained minimization problem:

$$\min_c \frac{1}{2} c^T Kc - F^T Mc, \quad Hc = 0, Bc = G, \quad (45)$$

where $Hc = 0$ represents the all continuity conditions and $Bc = G$ all the boundary conditions, $M$ and $K$ are mass and stiffness matrices and $F^T Mc$ for the right-hand side of the weak formulation (41). This minimization problem (45) can be solved using the iterative method discussed in [2] in a few iterative steps. In our computation, we only do 5 iterative steps.
The rest is to evaluate the polygonal spline solution and its gradient over the domain. In our experiments, we evaluate the solution and its gradient at those points from 1,002,001 points equally spaced points over the minimal rectangular containing \( \Omega \) that are located in \( \Omega \) to display the surface of the solution. We also use the values for computing the Rooted Mean Square Error (RMSE) of the solution against the exact solution to see the performance of our polygonal splines. As the gradient of polygonal splines is not continuous, we have to exclude the points located on the edges of the underlying polygonal partition so that we are able to compute the accuracy of the gradients. For the domain \( \Omega \) listed in Fig. 8, there are totally 726,551 points located inside \( \Omega \), but not on the edges of the partition. The RMSE estimates for numerical solutions and their gradients are based on the values over these 726,551 points. In finite element literature, the errors in \( L_2 \) norm and \( H^1 \) norm are usually calculated. The calculation is heavily dependent on the accuracy of the numerical quadrature for the exact solution. That is, the numerical integration of the \( L_2 \) norm of the exact solution has to be more accurate than the \( L_2 \) error of the solution. In particular, when the partition is refined and the solution is very accurate, it is hard to know how accurate the numerical quadrature should be. On the other hand, the RMSE approach does not need such a quadrature formulation and hence, is a much simpler calculation. In addition, the RMSE is widely used in all other areas of computational sciences such as Statistics, Machine Learning, Image Processing and etc..

5.3 A Refinement Scheme of Pentagonal Partitions

Refinements of a triangulation and a quadrangulation are well-known. Let us introduce a scheme to refine a pentagon partition \( \Delta \). We start with a pentagon \( P = (v_1, \ldots, v_5) \). Let \( p = (v_1 + \cdots + v_5)/5 \) be the geometric center of \( P \). Write \( u_i = (v_i + v_{i+1})/2 \) for \( i = 1, \ldots, 5 \) with \( v_6 := v_1 \) and \( w_i = (u_i + p)/2, i = 1, \ldots, 5 \) as shown in Fig. 7. Connect these \( v_i, u_i, w_i, i = 1, \ldots, 5 \) as shown to form a uniform refinement of the pentagon \( P \).

![Figure 7: An illustration of the pentagon refinement with vertices \( v_i, u_i, w_i, i = 1, \ldots, 5 \) ](image)

For a collection \( \Delta \) of pentagons, we apply the above refinement scheme to each pentagon from \( \Delta \). Let us recall that the size \( |\Delta| \) of \( \Delta \) is the longest of the lengths of edges in \( \Delta \). An elementary proof shows

**Lemma 2.** Let \( \Delta_1 \) be the refinement of pentagon partition \( \Delta \). Then

\[
|\Delta_1| \leq \frac{1}{2} |\Delta|.
\]

If we let \( d(P) \) to be the smallest of the diameter of the circle containing \( P \), then we know that \( |P| \to 0 \) is equivalent to \( d(P) \to 0 \).

In addition to refining triangles, quadrilaterals, pentagons, one is also interested how to refine a general convex \( n \)-gon for \( n \geq 6 \). See how to do in [14].

5.4 Numerical Results of Polygonal Splines

We now use our MATLAB code to show some numerical results of Poisson equations based on polygonal spline functions. We have tested that our codes for \( d = 2 \) and \( d = 3 \) can reproduce all polynomials of degree 2 and degree 3, respectively based on Wachspress coordinates. As we explained above, we shall compute the RMSE for the solution and its gradient to show the performance of polygonal spline functions in Tables 1–8. All the numerical results are obtained by using a laptop computer Dell XPS which has 4 CPU cores with 4 GB memory and speed at 2.20GHz. The MATLAB parallel computation toolbox was used.
Example 7. We have done several tests on smooth solutions:
\[ u_1(x, y) = \frac{1}{1 + x^2 + y^2}, \quad u_2(x, y) = x^4 + y^4, \]
\[ u_3(x, y) = \sin(\pi(x^2 + y^2)) + 1, \quad u_4(x, y) = 10\exp(-x^2 - y^2) \]
to check the performance of our polygonal splines of order 2 based on partitions obtained by repeated pentagonal refinements as in Fig. 8. In Tables 1–8, we use \( \text{Etime}, \text{Stime}, \text{MKtime} \) for the times for evaluation, solution of the linear system, and generation of mass and stiffness matrices, respectively. All are counted in second. Let \( s_1 \) be the polygonal spline solution of \( u_1 \). The RMSE \( (u_1) \) is
\[
\text{RMSE}(u_1) = \sqrt{\frac{1}{N} \sum_{i,j=1}^{1001} (s_1(x_i,y_j) - u_1(x_i,y_j))^2},
\]
where \( N = \# \{ (x_i,y_i) \in \Omega \backslash E, i,j = 1, \ldots, 1001 \} \), where \( E \) is the collection of all edges of \( \Delta \). Our reason to exclude the points on \( E \) is because of the calculation of the gradient values of polygonal splines. Similar for the RMSE\( (\nabla u_1) \). That is,
\[
\text{RMSE}(\nabla u_1) = \sqrt{\frac{1}{N} \sum_{i,j=1}^{1001} (D_x s_1(x_i,y_j) - D_x u_1(x_i,y_j))^2 + (D_y s_1(x_i,y_j) - D_y u_1(x_i,y_j))^2}.
\]
It is interesting to know the rate of convergence. By Lemma 2, the size of \( \Delta \) is reduced by \( 1/2 \). The RMSE \( \text{err}_\ell \) of \( u_i \) at refinement level \( \ell > 0 \) is defined by
\[
\text{err}_\ell = O(1/2^\ell)^\alpha \tag{47}
\]
for some \( \alpha > 0 \), where \( i = 1, 2, 3, 4 \). Similar we can compute the rate of convergence of \( \nabla u_i \).

These can be compared with the convergence of the standard quadrature finite elements for these exact solutions. To do so, we include a table for quadrature finite element for solution \( u_i, i = 1, 2, 3, 4 \) in Tables 2, 4, 6 and 8.

These numerical results in Tables 1, 2, \ldots, 7, and 8 show that the polygonal spline of order 2 are able to approximate the solution very well with \( \alpha \approx 3.4 \) for function values and \( \alpha \approx 2.4 \) for gradient values, even though we are not able to prove the boundedness of the chunkiness of refined pentagon partitions.

The computational times for evaluation for totally 726,551 points located inside \( \Omega \), for solution of the associated linear systems and for generation of mass and stiffness matrices show that the polygonal splines are better over evaluation and solution while are significantly worse than the traditional finite element method for the mass and stiffness matrices generation. More study is needed to speed up the generating mass and stiffness matrices. One possible approach is to use GPU.

![Table 1](image)

Table 1: Convergence of polygonal spline of order 2 over repeatedly refined pentagon partitions in Fig. 8, where \( u_1 = 1/(1 + x^2 + y^2) \).

Remark 3. From Tables 1–7, we see that assembling mass and stiffness matrices cost a lot of time. This method may be useful if we use a computer with GPU to solve a nonlinear Poisson equation, e.g.
\[
\begin{cases}
-\Delta u + f(u, x) = 0, & x \in \Omega \subset \mathbb{R}^2 \\
u = g, & x \in \partial \Omega,
\end{cases}
\tag{48}
\]
where \( f(u, x) \) is a nonlinear function of \( u \), say \( f(u, x) = u(1 - u), f(u, x) = u^3, f(u, x) = \exp(u) \) and etc.
<table>
<thead>
<tr>
<th>DoF</th>
<th>RMSE ($u_1$)</th>
<th>rate</th>
<th>RMSE ($\nabla u_1$)</th>
<th>rate</th>
<th>Etime</th>
<th>Stime</th>
<th>MKtime</th>
</tr>
</thead>
<tbody>
<tr>
<td>88</td>
<td>0.00141741</td>
<td></td>
<td>0.017789</td>
<td></td>
<td>3.18</td>
<td>0.04</td>
<td>0.49</td>
</tr>
<tr>
<td>320</td>
<td>0.00016451</td>
<td>3.11</td>
<td>0.004509</td>
<td>1.98</td>
<td>15.64</td>
<td>0.07</td>
<td>0.63</td>
</tr>
<tr>
<td>1216</td>
<td>0.0001996</td>
<td>3.04</td>
<td>0.001131</td>
<td>1.99</td>
<td>58.55</td>
<td>0.19</td>
<td>2.18</td>
</tr>
<tr>
<td>4736</td>
<td>0.0000248</td>
<td>3.01</td>
<td>0.000283</td>
<td>2.00</td>
<td>227.27</td>
<td>0.70</td>
<td>9.01</td>
</tr>
<tr>
<td>18688</td>
<td>0.0000031</td>
<td>3.00</td>
<td>0.000071</td>
<td>2.00</td>
<td>907.29</td>
<td>2.63</td>
<td>42.55</td>
</tr>
</tbody>
</table>

Table 2: Convergence of quadratic finite element over repeatedly refined triangulations as in Fig. 9, where $u_1 = 1/(1 + x^2 + y^2)$.

<table>
<thead>
<tr>
<th>DoF</th>
<th>RMSE ($u_2$)</th>
<th>rate</th>
<th>RMSE ($\nabla u_2$)</th>
<th>rate</th>
<th>Etime</th>
<th>Stime</th>
<th>MKtime</th>
</tr>
</thead>
<tbody>
<tr>
<td>76</td>
<td>0.031270</td>
<td>0.00</td>
<td>0.446502</td>
<td>0.00</td>
<td>5.22</td>
<td>0.002</td>
<td>0.47</td>
</tr>
<tr>
<td>392</td>
<td>0.002379</td>
<td>3.72</td>
<td>0.072348</td>
<td>2.63</td>
<td>4.26</td>
<td>0.003</td>
<td>2.44</td>
</tr>
<tr>
<td>2224</td>
<td>0.000195</td>
<td>3.61</td>
<td>0.013119</td>
<td>2.46</td>
<td>17.68</td>
<td>0.03</td>
<td>13.50</td>
</tr>
<tr>
<td>13088</td>
<td>0.000016</td>
<td>3.57</td>
<td>0.002501</td>
<td>2.39</td>
<td>99.47</td>
<td>0.19</td>
<td>84.26</td>
</tr>
<tr>
<td>78016</td>
<td>0.000002</td>
<td>3.37</td>
<td>0.000550</td>
<td>2.19</td>
<td>585.15</td>
<td>1.38</td>
<td>664.99</td>
</tr>
</tbody>
</table>

Table 3: Convergence of polygonal spline of order 2 over repeatedly refined pentagon partitions in Fig. 8, where $u_2 = x^4 + y^4$.

<table>
<thead>
<tr>
<th>DoF</th>
<th>RMSE ($u_3$)</th>
<th>rate</th>
<th>RMSE ($\nabla u_3$)</th>
<th>rate</th>
<th>Etime</th>
<th>Stime</th>
<th>MKtime</th>
</tr>
</thead>
<tbody>
<tr>
<td>76</td>
<td>1.417091</td>
<td>4.43</td>
<td>8.211055</td>
<td>2.74</td>
<td>4.24</td>
<td>0.003</td>
<td>2.41</td>
</tr>
<tr>
<td>392</td>
<td>0.065557</td>
<td>3.86</td>
<td>1.227940</td>
<td>2.33</td>
<td>17.70</td>
<td>0.024</td>
<td>13.72</td>
</tr>
<tr>
<td>2224</td>
<td>0.004502</td>
<td>3.66</td>
<td>0.243838</td>
<td>2.35</td>
<td>100.36</td>
<td>0.199</td>
<td>86.14</td>
</tr>
<tr>
<td>13088</td>
<td>0.000355</td>
<td>3.59</td>
<td>0.047945</td>
<td>2.37</td>
<td>595.08</td>
<td>1.421</td>
<td>658.04</td>
</tr>
<tr>
<td>78016</td>
<td>0.000029</td>
<td>3.59</td>
<td>0.009245</td>
<td>2.37</td>
<td>898.80</td>
<td>2.57</td>
<td>42.66</td>
</tr>
</tbody>
</table>

Table 4: Convergence of quadratic finite element over repeatedly refined triangulations as in Fig. 9, where $u_2 = x^4 + y^4$.

<table>
<thead>
<tr>
<th>DoF</th>
<th>RMSE ($u_3$)</th>
<th>rate</th>
<th>RMSE ($\nabla u_3$)</th>
<th>rate</th>
<th>Etime</th>
<th>Stime</th>
<th>MKtime</th>
</tr>
</thead>
<tbody>
<tr>
<td>88</td>
<td>0.626154</td>
<td>2.54</td>
<td>5.724203</td>
<td>5.38</td>
<td>0.03</td>
<td>0.03</td>
<td>0.26</td>
</tr>
<tr>
<td>320</td>
<td>0.107462</td>
<td>2.54</td>
<td>2.079933</td>
<td>1.46</td>
<td>15.98</td>
<td>0.06</td>
<td>0.64</td>
</tr>
<tr>
<td>1216</td>
<td>0.012978</td>
<td>3.05</td>
<td>0.608579</td>
<td>1.77</td>
<td>58.13</td>
<td>0.19</td>
<td>2.25</td>
</tr>
<tr>
<td>4736</td>
<td>0.001498</td>
<td>3.11</td>
<td>0.159646</td>
<td>1.93</td>
<td>227.82</td>
<td>0.67</td>
<td>9.20</td>
</tr>
<tr>
<td>18688</td>
<td>0.000182</td>
<td>3.04</td>
<td>0.040416</td>
<td>1.98</td>
<td>898.31</td>
<td>2.59</td>
<td>42.84</td>
</tr>
</tbody>
</table>

Table 5: Convergence of polygonal spline of order 2 over repeatedly refined pentagon partitions in Fig. 8, where $u_3 = \sin((x^2 + y^2)\pi) + 1$.

<table>
<thead>
<tr>
<th>DoF</th>
<th>RMSE ($u_3$)</th>
<th>rate</th>
<th>RMSE ($\nabla u_3$)</th>
<th>rate</th>
<th>Etime</th>
<th>Stime</th>
<th>MKtime</th>
</tr>
</thead>
<tbody>
<tr>
<td>88</td>
<td>0.626154</td>
<td>2.54</td>
<td>5.724203</td>
<td>5.38</td>
<td>0.03</td>
<td>0.03</td>
<td>0.26</td>
</tr>
<tr>
<td>320</td>
<td>0.107462</td>
<td>2.54</td>
<td>2.079933</td>
<td>1.46</td>
<td>15.98</td>
<td>0.06</td>
<td>0.64</td>
</tr>
<tr>
<td>1216</td>
<td>0.012978</td>
<td>3.05</td>
<td>0.608579</td>
<td>1.77</td>
<td>58.13</td>
<td>0.19</td>
<td>2.25</td>
</tr>
<tr>
<td>4736</td>
<td>0.001498</td>
<td>3.11</td>
<td>0.159646</td>
<td>1.93</td>
<td>227.82</td>
<td>0.67</td>
<td>9.20</td>
</tr>
<tr>
<td>18688</td>
<td>0.000182</td>
<td>3.04</td>
<td>0.040416</td>
<td>1.98</td>
<td>898.31</td>
<td>2.59</td>
<td>42.84</td>
</tr>
</tbody>
</table>

Table 6: Convergence of quadratic finite element over repeatedly refined triangulations as in Fig. 9, where $u_3 = \sin((x^2 + y^2)\pi) + 1$.  

18
Table 7: Convergence of polygonal spline of order 2 over repeatedly refined pentagon partitions in Fig. 8, where \( u_4 = 10 \exp(-x^2 - y^2) \).

<table>
<thead>
<tr>
<th>DoF</th>
<th>RMSE ((u_4))</th>
<th>rate</th>
<th>RMSE ((\nabla u_4))</th>
<th>rate</th>
<th>Etime</th>
<th>Stime</th>
<th>MKtime</th>
</tr>
</thead>
<tbody>
<tr>
<td>76</td>
<td>0.026166</td>
<td></td>
<td>0.319205</td>
<td></td>
<td>2.65</td>
<td>0.002</td>
<td>0.52</td>
</tr>
<tr>
<td>396</td>
<td>0.003915</td>
<td>3.77</td>
<td>0.053791</td>
<td>2.57</td>
<td>4.21</td>
<td>0.002</td>
<td>2.30</td>
</tr>
<tr>
<td>2224</td>
<td>0.000161</td>
<td>3.57</td>
<td>0.010215</td>
<td>2.40</td>
<td>17.74</td>
<td>0.02</td>
<td>13.76</td>
</tr>
<tr>
<td>13088</td>
<td>0.000014</td>
<td>3.54</td>
<td>0.001975</td>
<td>2.37</td>
<td>100.19</td>
<td>0.18</td>
<td>86.41</td>
</tr>
<tr>
<td>78016</td>
<td>0.000001</td>
<td>3.53</td>
<td>0.000389</td>
<td>2.34</td>
<td>586.68</td>
<td>1.42</td>
<td>656.69</td>
</tr>
</tbody>
</table>

Table 8: Convergence of quadratic finite element over repeatedly refined triangulations as in Fig. 9, where \( u_4 = 10 \exp(-x^2 - y^2) \).

<table>
<thead>
<tr>
<th>DoF</th>
<th>RMSE ((u_4))</th>
<th>rate</th>
<th>RMSE ((\nabla u_4))</th>
<th>rate</th>
<th>Etime</th>
<th>Stime</th>
<th>MKtime</th>
</tr>
</thead>
<tbody>
<tr>
<td>88</td>
<td>0.018453</td>
<td>0.00</td>
<td>0.235958</td>
<td>0.00</td>
<td>3.59</td>
<td>0.04</td>
<td>0.48</td>
</tr>
<tr>
<td>320</td>
<td>0.002149</td>
<td>3.10</td>
<td>0.059854</td>
<td>1.98</td>
<td>10.65</td>
<td>0.03</td>
<td>0.32</td>
</tr>
<tr>
<td>1216</td>
<td>0.000263</td>
<td>3.03</td>
<td>0.015053</td>
<td>1.99</td>
<td>40.56</td>
<td>0.10</td>
<td>1.13</td>
</tr>
<tr>
<td>4736</td>
<td>0.000032</td>
<td>3.01</td>
<td>0.003770</td>
<td>2.00</td>
<td>159.75</td>
<td>0.34</td>
<td>4.52</td>
</tr>
<tr>
<td>18888</td>
<td>0.000004</td>
<td>3.00</td>
<td>0.000943</td>
<td>2.00</td>
<td>643.77</td>
<td>1.30</td>
<td>21.56</td>
</tr>
</tbody>
</table>

Figure 8: A Pentagon Partition (top-left) and its refinements

Figure 9: A triangulation (top-left) and its refinements

Table 9: Convergence of polygonal spline of order 2 over repeatedly refined pentagon partitions in Fig. 8 based on the mean value coordinates.

<table>
<thead>
<tr>
<th>DoF</th>
<th>RMSE ((u_5))</th>
<th>rate</th>
<th>RMSE ((\nabla u_5))</th>
<th>rate</th>
<th>Etime</th>
<th>Stime</th>
<th>MKtime</th>
</tr>
</thead>
<tbody>
<tr>
<td>76</td>
<td>1.07749006</td>
<td>0.00</td>
<td>7.251159</td>
<td>0.00</td>
<td>2.91</td>
<td>0.003</td>
<td>3.69</td>
</tr>
<tr>
<td>392</td>
<td>0.04905039</td>
<td>4.46</td>
<td>1.007859</td>
<td>2.85</td>
<td>4.39</td>
<td>0.03</td>
<td>19.91</td>
</tr>
<tr>
<td>2224</td>
<td>0.00364113</td>
<td>3.75</td>
<td>0.203799</td>
<td>2.31</td>
<td>17.73</td>
<td>0.024</td>
<td>19.51</td>
</tr>
<tr>
<td>13088</td>
<td>0.00029163</td>
<td>3.64</td>
<td>0.038882</td>
<td>2.39</td>
<td>98.69</td>
<td>0.215</td>
<td>700.64</td>
</tr>
<tr>
<td>78016</td>
<td>0.00002405</td>
<td>3.60</td>
<td>0.007281</td>
<td>2.42</td>
<td>591.48</td>
<td>1.406</td>
<td>4309.75</td>
</tr>
</tbody>
</table>
Table 10: Convergence of continuous finite elements of degree 3 over repeatedly refined triangulations in Fig. 10.

Example 8. In addition to the numerical results based on Wachspress coordinates, let us present some results using the Mean Value coordinates. We shall use $u_3$ and $u_5 = \sin(\pi x)\sin(\pi y)$ as testing solutions. Numerical results are listed in Table 9. It takes much longer than Wachspress coordinates to generate the mass and stiffness matrices as we have to use high order Gaussian quadrature to ensure the sufficient accuracy. This shows the Wachspress coordinates are better than the Mean Value coordinates.

Example 9. In this example, we show polygonal splines of order 3 to solve the Poisson equation. For convenience, we only present numerical results from one testing function $u_3$ based on pentagonal refinements and quadrangulation refinements as shown in Fig. 10. To compare, we also list the results results from continuous finite element of degree 3 over refined triangulations. Our computer allows to refine the L-domain in 10 5 times using pentagonal refinement scheme while 6 times in quadrilateral refinement and triangulation refinement. From the numerical results in Tables 12, 11 and 10, we can see that polygonal splines can achieve more accurate in solution and gradients using less number of degrees of freedom when using quadrilaterals although assembling the mass and stiffness matrices takes much longer time.

Table 11: Convergence of polygonal spline of order 3 over repeatedly refined quadrilateral partitions in Fig. 10.

Table 12: Convergence of polygonal spline of order 3 over repeatedly refined pentagon partitions in Fig. 10.

6 Conclusions and Future Research Problems

We proposed to define a continuous polygonal spline space $S_d(\Delta)$ of order $d \geq 1$ over a mixed partition of convex polygons with arbitrary number of sides which possesses a property of reproducing polynomials of degree $d$. Then we presented a construction of locally supported basis functions for $S_d(\Delta)$ over the mixed partition $\Delta$ of convex polygons for $d \geq 2$. For $d = 2$, our construction is simpler than the one in [23]. For $d \geq 3$, our
construction is only based on Wachspress coordinates. We then explained how to use these spline functions to solve the Poisson equation numerically over a partition of quadrilaterals or pentagons. In particular, we found a refinement scheme for pentagons and showed that this refinement leads to a higher order of convergence rate.

There are many problems remaining open. For example, when \( d \geq 3 \), we are only able to construct polygonal splines of order \( d \) using Wachspress coordinates. It is interesting to see if one can use other generalized barycentric coordinates to construct higher order splines. Also, we would like to know the approximation properties of our interpolation schemes. In particular, is it possible that the polygonal splines studied in this paper have higher approximation power than standard polynomial splines? Our numerical results suggest that the refinement scheme for pentagons may lead to higher accuracy than the uniform refinement of triangulations and quadrangulations. We would like to understand this phenomenon better.

In addition, we would like to extend our study to a higher dimensional setting. Currently we are working on constructing polyhedral splines over a 3D partition of mixed polyhedra and using them for solving partial differential equations.

Finally, we would like to construct polygonal splines which are \( C^1 \) or smoother in the 2D and 3D settings using GBCs in [12], [11], [26], and etc.. These are all under investigation. Of course, we welcome any interested reader to join our efforts and understand polygonal splines better.

References


7 Appendix

This section shows the main driven program to show how we solve Poisson equations numerically.

Algorithm 1. This is a matlab demo program to show how to compute a numerical solution of Poisson equation using quadratic polygonal spline spaces. We use a pentagonal partition and then refine it uniformly several times to show our computational results.

Authors: Michael Floater and Ming-Jun Lai

V=[1 0; 2 0; 2 1; 1 1; 2 0; 2 0; 1 0.1 0.1];
P={[1 2 3 4 8],[8 4 5 6 7]}; % A Pentagonal Partition.
CaseNumber=4; % The number in a list of testing functions.
d=2; % degree is 2.
xmin=min(V(:,1)); xmax=max(V(:,1)); ymin=min(V(:,2)); ymax=max(V(:,2));
xstep=(xmax-xmin)/1000; ystep=(ymax-ymin)/1000;
[xx,yy]=meshgrid(xmin:xstep:xmax,ymin:ystep:ymax); % equally-spaced points
zz=g(xx,yy,CaseNumber); % values of an exact solution
zx=gx(xx,yy,CaseNumber); zy=gy(xx,yy,CaseNumber); % values of the exact gradient
L=5; % level number of refinements
for test=1:L
[V,P]=refine(V,P); % refinement of pentagon partition
[E,PE,EV,BV]=vdata(V,P); % relations among edges, vertices, and etc..
H = c0continuity(P,PE,d); % continuous condition matrix
[M,K]=msGBCvP(V,P); % mass and stiffness matrices generator in parallel computing
F = coeffv(V,P,'laplace',CaseNumber); % the right-hand side vector
[G,B]=dirichlet(V,P,PE,d,'g',CaseNumber); % boundary condition
n = size(H,1); % The side of the smoothness matrix
k = length(G); % Number of boundary conditions
p = size(H,1); ZERO = zeros(p,1);
tol=1.0e-6; eps=1.0e-4; max_it=5; % tolerance, perturbation, and number of iterates
C=CImin(K,[H;B],M*F,[ZERO;G],eps,max_it,tol); % Awanou, Lai and Wenston’s iterative method
[Z,Zx,Zy]=gbcEval(V,P,C,xx1(:),yy1(:)); % Evaluation of the solution
I=isnan(Z); J=find(I==0); % Evaluation of the solution inside the domain.
rms(zz(J)-Z(J)); % the RMSE of the exact solution
[rms(zx(J)-Zx(J)), rms(zy(J)-Zy(J))] % the RMSE of the gradient
end % The end of the main driven code.