

# Web-spline Finite Elements

## IMA Mathematical Modeling Workshop 2006

Mentor: Thomas Grandine  
The Boeing Company, Seattle WA

Yanping Cao  
University of California

Olga Terlyga  
Northern Illinois University

Jon Van Laarhoven  
University of Iowa

Jianbao Wu  
University of Georgia

Guangri Xue  
Pennsylvania State University

Ping Zhang  
University of Kentucky

August 18, 2006

## 1 Introduction

B-splines are an intriguing choice of finite elements in the finite element method. B-splines form a basis for any space of piecewise polynomial functions in one variable, including those which have specified continuity conditions at the junctions between the individual polynomial pieces. However, extensions to more than one variable have been hard to come by.

In our project, we implement a finite element method for an elliptical PDE by using web-splines which are basis functions on regular grids. The weighted extended B-spline approximation takes care of not only the boundary constraints but also the issue of well conditioning of the Galerkin systems. This choice of basis functions provides optimal approximation order with a

minimal number of parameters, thus providing a natural link between geometry description and finite element simulations.

We wrote Python code and tested it on Poisson’s equation over a 1) a circle 2) a piecewise smooth, simply connected region and 3) a region with a hole in it.

## 2 The Formulation

The finite element method (FEM) is a widely accepted method for solving problems that arise in heat transfer, fluid flow, mechanical systems, etc... FEM uses a finite set of basis functions (so-called “elements”) to approximate the solution. Oftentimes the domain of interest is partitioned into a set of regular polygons and piecewise continuous functions are defined each polygon. Generating this mesh, however, is no easy task so mesh-free alternatives become appealing.[1]

One might look to B-splines (a basis of piecewise polynomial functions) as a basis for mesh-free problems considering their flexibility and ubiquity for modeling a variety of functions. Two principle difficulties arise in such a task: modeling essential (Dirchlet) boundary conditions and stability of the Galerkin system. The former problem is addressed by a sufficiently smooth weight function vanishing on the boundary, and the latter is remedied by a pre-processing step wherein basis elements with small support in the domain are appended to more heavily supported basis elements.[1]

The paper will first cover the formulation in the abstract sense and then illuminate the weak-formulation by considering Poisson’s equation. After describing the construction of the tensor product web-splines—including the weight function and stabilization process—we illustrate their utility by numerically solving Poisson’s equation. Finally, we will give numerical results to confirm our theoretical analysis.

### 2.1 Abstract Formulation

We begin with a linear operator equation over a region  $\Omega$ , where  $B$  is an operator describing the boundary conditions:

$$\lambda(u) = f \quad \text{where} \quad u \in \mathbf{H}_0^1(\Omega) \quad \text{and} \quad B(u) = 0 \quad \text{on} \quad \partial\Omega. \quad (1)$$

Examples of such include differential equations and integral equations. The Hilbert space  $\mathbf{H}_0^1(\Omega)$  is infinite dimensional, so we define the finite dimensional subspace  $\tilde{\mathbf{H}}_0^1(\Omega)$ . We would like our approximate solution  $u_h \in \tilde{\mathbf{H}}_0^1(\Omega)$  to be “close” to solving the original equation. That is, we would like:

$$\lambda(u_h) \approx f \quad \text{where} \quad u_h \in \tilde{\mathbf{H}}_0^1(\Omega) \quad \text{and} \quad B(u_h) = 0 \quad \text{on} \quad \partial\Omega. \quad (2)$$

We can not hope for  $u_h$  to be a solution in the classical sense. To that end, we expand our notion of solution by seeking equality in a broader sense. We set the inner product pairing of the lefthand side term equal to the linear operator applied to  $v \in \tilde{\mathbf{H}}_0^1(\Omega)$ . That is,  $a(\cdot, \cdot)$  is a bilinear form and  $l(\cdot)$  is a linear operator so we have:

$$a(u_h, v) = l(v) \quad \text{where} \quad B(u_h) = 0 \quad \text{on} \quad \partial\Omega. \quad (3)$$

If  $u_h$  is taken as a linear combination of linear basis elements (“hat functions”) we have the usual finite element method, and solving the problem amounts to solving a linear system.[2]

## 2.2 The Weak Formulation of Poisson’s Equation

Let us clarify the abstract formulation, as well as the application of web-splines, with a specific boundary value problem. Poisson’s equation occurs in electrostatics, mechanical engineering, and physics:

$$-\Delta u = f \quad \text{in} \quad \Omega \quad \text{and} \quad u = 0 \quad \text{on} \quad \partial\Omega. \quad (4)$$

When  $f$  is not regular enough (if  $f \in \mathbf{H}^{-1}$ , for instance) the existence of classical solution is suspended. So we come up with the weak formulation of the solution.

Multiplying the equation by an arbitrary test function  $v \in \tilde{\mathbf{H}}_0^1(\Omega)$  and integrating the equality we have:

$$-\int_{\Omega} \Delta u v dx = \int_{\Omega} f v dx. \quad (5)$$

If we apply integration by parts on the left term in the above equation (realizing the boundary term vanishes because  $v \in \tilde{\mathbf{H}}_0^1(\Omega)$ ) we are left with:

$$\int_{\Omega} \nabla u \cdot \nabla v dx = \int_{\Omega} f v dx. \quad (6)$$

We then define our basis elements  $B_i \in \tilde{\mathbf{H}}_0^1(\Omega)$  and express  $u_h$  as a linear combination of basis elements:  $u_h = \sum_i \alpha_i B_i$ . Choosing  $v = B_k$  our weak formulation becomes:

$$\int_{\Omega} \left( \nabla \sum_i \alpha_i B_i \right) \cdot \nabla B_k dx = \int_{\Omega} f B_k dx. \quad (7)$$

Using linearity of  $\nabla$  and  $\int_{\Omega}$  we are left with:

$$\sum_i \alpha_i \int_{\Omega} \nabla B_i \cdot \nabla B_k dx = \int_{\Omega} f B_k dx. \quad (8)$$

leaving us with the task of solving the linear system  $GU = F$  in order to find the  $\alpha_i$ , the coefficients of the basis functions, where  $G = (\int_{\Omega} \nabla B_i \cdot \nabla B_k)_{i,k}$ ,  $F = \int_{\Omega} f B_k$ ,  $u = (\alpha_i)_i$ .

### 3 Web-Splines

In our application, the basis functions must yield a stable system and they must be able to accurately reflect the boundary conditions. The weighted extend B-spline (web-spline) method exactly conforms to the boundary conditions and yields approximations of optimal order. In addition, the web-splines basis is uniformly stable with respect to grid size. In the following, we first discuss how to extend one-dimensional B-splines to higher dimensions and follow it by a discussion of the weight function and extension procedure to create web-splines.

We begin with a few definitions about splines.

A *spline* of degree  $\leq n$  and grid width  $h$  is an  $(n - 1)$  times continuously differentiable function and agrees with a polynomial of degree  $\leq n$  on each grid interval  $[i, i + 1]h$  of the parameter interval  $D$ . From splines we can generate *B-splines* which are a combination of splines to form a basis for a polynomial space of degree  $n$ . B-Splines have many nice properties, but the most useful in this context is their local support, differentiability, and the fact that a degree  $n$  B-spline is a translation of any other degree  $n$  B-spline. See figure 1 for an illustration B-splines of varying degree.

The *spline-based finite element subspace*  $\mathbb{B}_h^n(\Omega)$  on a bounded domain  $\Omega \in \mathbf{R}^m$  consists of all linear combinations

$$\sum_{k \in K} c_k b_{k,h}^n \quad (9)$$

of relevant B-splines; i.e., the set  $K$  of relevant indices contains all  $k$  with  $b_{k,h}^n(x) \neq 0$  for some  $x \in \Omega$ .

The *tensor product B-spline* is the extension of the B-spline to higher dimensions; it can be defined the following way. The  $m$ -variate B-spline  $b_{k,h}^n$  with degree  $n_{\nu}$  in the  $\nu$ th variable, index  $k = k_1, \dots, k_m$  and grid width  $h$  is defined as

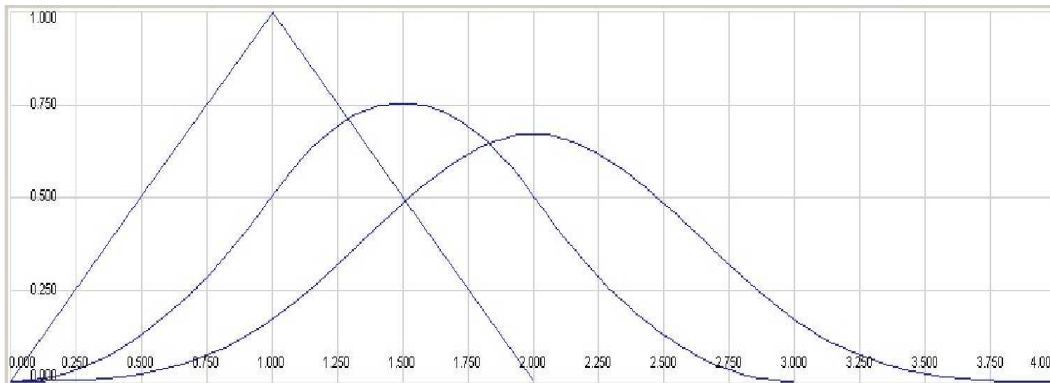


Figure 1: One-dimensional B-splines of degrees 1, 2, and 3.

$$b_{k,h}^n(x) = \prod_{\nu=1}^m b_{k_\nu,h}^{n_\nu}(x_\nu), \quad (10)$$

with convention that  $n_1 = \dots = n_m$  unless otherwise stated. For our purposes  $n$  can be taken to be an integer, rather than an integer-vector.

The construction of our web-space begins with a space of biquadratic B-splines (quadratic in each variable)  $\mathbb{B}_h^2$ . Each spline element of this space is supported on a square  $3h$  by  $3h$  wide. See figure 2 for a picture of one biquadratic B-spline.

### 3.1 Weight functions

B-splines are able to model essential boundary conditions only for the one-dimensional problem. To remedy this in higher dimensions, we define a weight function which takes value 1 in the interior of  $\Omega$  and 0 on  $\partial\Omega$ . Given the region  $\Omega$ , we need a measure of how large our “buffer-zone” of our weight function can be (recall larger is better since it will make  $\nabla w$  small). We define

$$\delta = \frac{.95}{\kappa} \quad (11)$$

where  $\kappa$  is the max radius of curvature of our region. Having computed delta, we define a cubic spline orthogonal to the boundary which takes the values  $s(0) = 0$  and  $s(\delta) = 1$  where we have 3rd order derivatives at  $\delta$ :

$$w(x) = s(\text{dist}(x, \partial\Omega)) \quad (12)$$

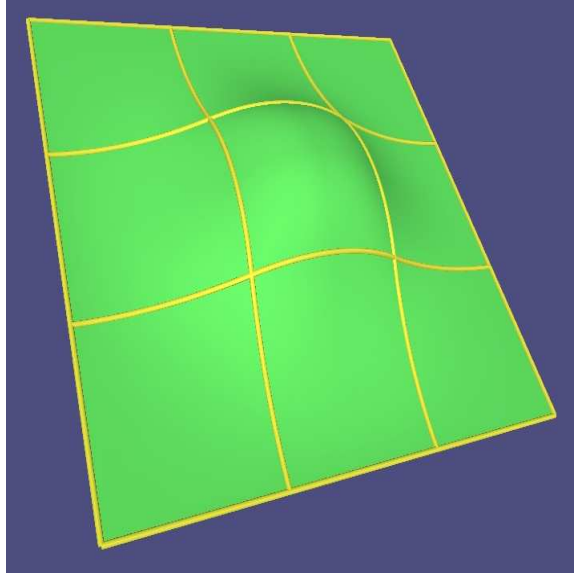


Figure 2: A two-dimensional biquadratic B-spline.

See Figure 3 for a picture of this function over the domain of test problem two.

### 3.2 B-spline Extension

While the spaces  $\mathbb{B}$  and  $w\mathbb{B}$  provide optimal approximation order, the traditional  $B$ -spline basis is not uniformly stable with respect to the grid width  $h$ . This instability, due to  $B$ -splines which have only very little support in the domain  $\Omega$ , causes severe numerical problems as  $h \rightarrow 0$ . For example, the Ritz-Galerkin systems become ill conditioned, which affects the convergence of iterative schemes and the accuracy of numerical solutions. We resolve the stability problem by suitably adjoining  $B$ -splines with small support to a stable subset of the basis for  $\mathbb{B}$ . To this end we follow [1] and make the following classification.

#### Inner and Outer B – Splines

We partition the grid cells  $Q = lh + [0, 1]^m h$  into interior, boundary, and exterior cells, depending on whether  $Q \subseteq \bar{\Omega}$ , the interior of  $Q$  intersects  $\partial\Omega$ , or  $Q \cap \Omega = \emptyset$ . Among the relevant  $B$ -splines  $b_k, k \in K$ , we distinguish between inner  $B$ -splines

$$b_i, i \in I,$$

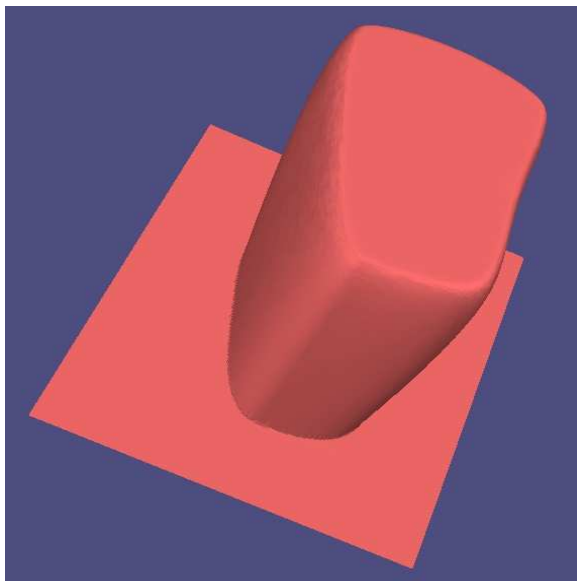


Figure 3: Weight function shown over a smooth boundary.

which have at least one interior cell  $Q_i$  in their support, and outer B-splines

$$b_j, j \in J = K \setminus I,$$

for which  $\text{supp}(b_j)$  consists entirely of boundary and exterior cells.

Figure 4 shows for a two-dimensional domain the interior (light gray) and boundary (dark gray) cells, as well as the classification of biquadratic B-splines. The inner and outer B-splines are distinguished with black and white dots placed at the lower left corner of their supports, respectively. We highlight the support of one B-spline  $b_i$ .

### 3.3 Web-splines

To connect the outer to the inner B-splines without affecting the approximation power of the spline space  $\mathbb{B}$ , we introduce the following definition:

For an outer index  $j \in J$  let  $I(j) = l + \{0, \dots, n\}^m \subset I$  be an  $m$ -dimensional array of inner indices closest to  $j$  assuming that  $h$  is small enough so that such an array exists. Moreover, denote by

$$e_{i,j} = \prod_{\nu=1}^m \prod_{\mu=0, l_\nu+\mu \neq i_\nu}^n \frac{j_\nu - l_\nu - \mu}{i_\nu - l_\nu - \mu}$$

the values of the Lagrange polynomials associated with  $I(j)$  and by  $J(i)$  the set of all  $j$  with  $i \in I(j)$ . Then, the web-splines

$$B_i = w \left[ b_i + \sum_{j \in J(i)} e_{i,j} b_j \right], \quad i \in I,$$

form a basis for the web-space  $w^e \mathbb{B}_h^n(D)$ .

We now guide the curious reader through the construction of one bi-quadratic web-spline which has been illustrated in 2. On the left, the figure shows for an outer index  $j$  the array  $I(j)$  with the values  $e_{i,j}$  of the Lagrange polynomials. For example, for  $i = j - (0, 2)$ ,

$$e_{i,j} = \left( \frac{2-0}{2-0} \frac{2-1}{2-1} \right) \left( \frac{3-0}{1-0} \frac{3-2}{1-2} \right) = -3$$

since  $j-l = (2, 3)$  and  $i-l = (2, 1)$ . The appearance of many zeros is typical since the Lagrange polynomials vanish whenever  $j_\nu \in (l_\nu + \{0, \dots, n\} \setminus \{i_\nu\})$  for some  $\nu$ . On the right of figure we see the support of a web-spline and the set  $J(i)$  with coefficients of the adjoined outer  $b$ -splines. We have also marked the point  $x_i$  at the center of the grid cell  $Q_i$ . Since  $e_{i,j} = 0$  for  $j = i + (1, -1)$ , the linear combination  $\sum_{j \in J(i)} e_{i,j} b_j$  consists only of three terms, corresponding to  $j - i = (1, 0), (0, 1), (1, 1)$ . As in this example, the union of the supports of the  $B$ -splines, involved in the definition of  $B_i$  (highlighted in the figure), is usually only slightly larger than the support of the inner  $B$ -spline  $b_i$ .

Except for positivity, web-splines inherit all essential properties of  $B$ -splines. Some facts are particularly relevant for finite element approximation: extension coefficients, local support, stability, approximation order. For details see [1].

To recap, web-splines are an enhancement of  $B$ -splines which remain uniformly stable with respect to grid size—without affecting approximation order of the spline space—and are able to model essential boundary conditions.

## 4 Application and Results

Grid cells were flagged and integrated over as indicated in Table 1 and figure 7. Recall there were two curves: an inner curve offset from the original boundary by width  $\delta$  in all directions.:

We ran our code on three different cases, reporting RMS error, as well as convergence order. After realizing the error was being dominated by error on the boundary, for the first case we looked at the error just on the interior

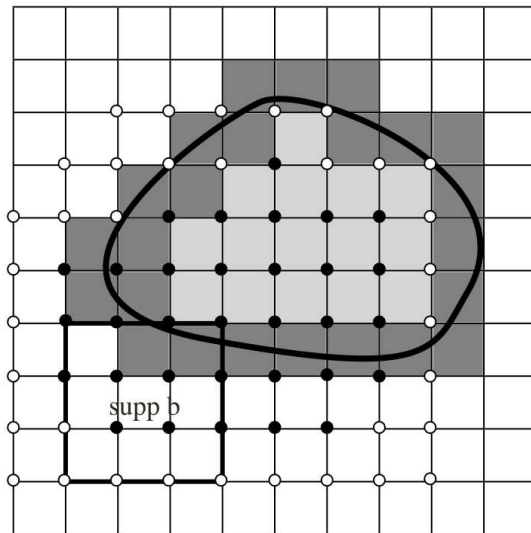


Figure 4: Inner and outer B-splines.

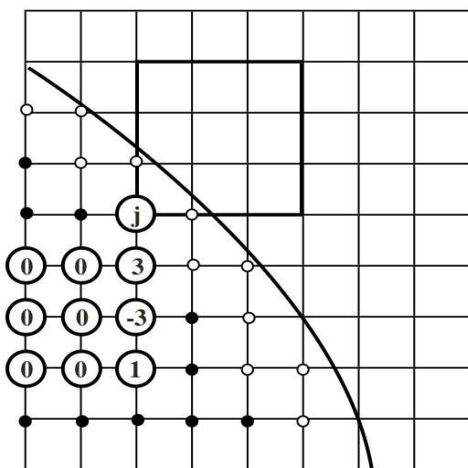


Figure 5: B-spline  $j$ 's contribution to other splines.

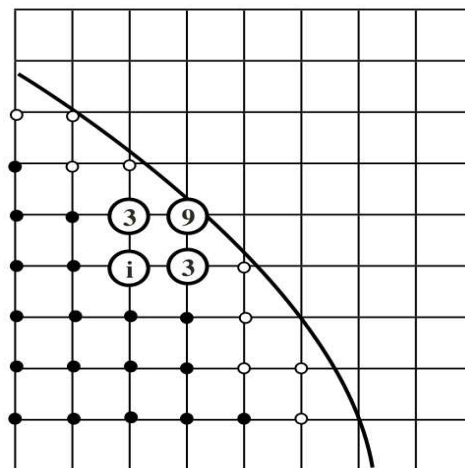


Figure 6: B-spline  $i$ 's adoption of surrounding splines.

Table 1: Flagging Method and Integration Method for various cells

Flag	Cell Type	Integration Method
0	cell outside of region	integral returns 0
1	cell intersects either curve	20 by 20 midpoint method
2	cell lies strictly between two curves	5 point Gauss quadrature
3	cell lies inside the offset curve	3 point Gauss quadrature

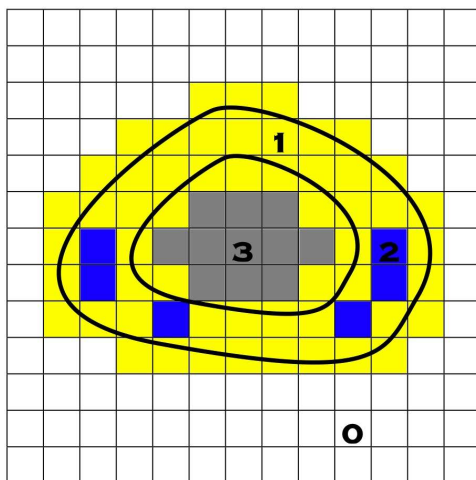


Figure 7: How cells are flagged.

to demonstrate we had a good result.

*Test problem one* is Poisson's equation over a circle:

$$\Omega = \{(x, y) : (x - 0.5)^2 + (y - 0.5)^2 \leq 0.3^2\} \text{ and } f = -100e^{-(x-0.5)^2-(y-0.5)^2} \quad (13)$$

See Table 2 for numerical solutions of test problem one and a graphical solution in figure 13. See figures 8, 9, 10, 11, and 12 for a graphical depiction of the RMS error function. After looking at the graphical depiction of the error function, we realized the error was being dominated by boundary terms. To that end, we produced Table 4 which only considers error on the interior.

In an effort to deduce the importance of extending the basis to better condition the stiffness matrix we found the rather startling result in Table 3. This problem would have essentially been unsolvable without extending the basis.

Table 2: Numerical data for test problem one

Poisson's over the circle			
$h$	$m$	RMS error	Order
.1	10	.007780	
.066	15	.001737	$O(3.698)$
.05	20	.001207	$O(1.264)$
.04	25	.0008288	$O(1.686)$
.01	100	.0001863	$O(1.077)$

Table 3: Conditioning number improvements for test problem one

Matrix	Corresponding Basis Type	Calculated conditioning number
Stiffness Matrix	B-spline basis	$\infty$
Reduced Stiffness Matrix	Extended B-spline basis	775

Table 4: Numerical data for test problem one *excluding* boundary error

Poisson's over piecewise smooth region			
$h$	$m$	RMS error	Order
0.1	10	0.006785	
0.066	15	0.001234	$O(4.198)$
0.05	20	0.0006721	$O(2.120)$
0.04	25	0.0003018	$O(3.589)$
0.01	100	.0001314	$O(0.600)$

*Test problem two* is exactly the same as test problem one, but over the the piecewise smooth region depicted in figure 3: See figure 14 for a graphical representation of the solution.

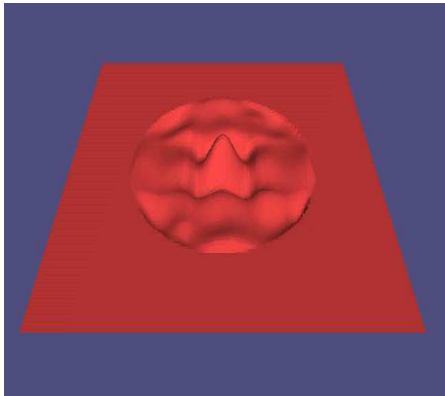


Figure 8: Error for test problem one:  $h=0.1$ .

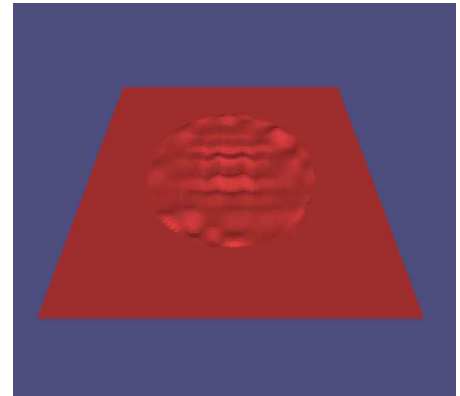


Figure 9: Error for test problem one:  $h=0.0667$ .

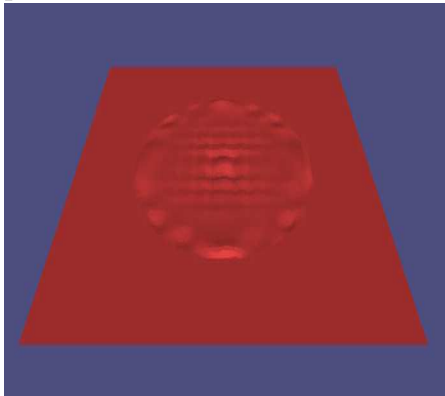


Figure 10: Error for test problem one:  $h=0.05$ .

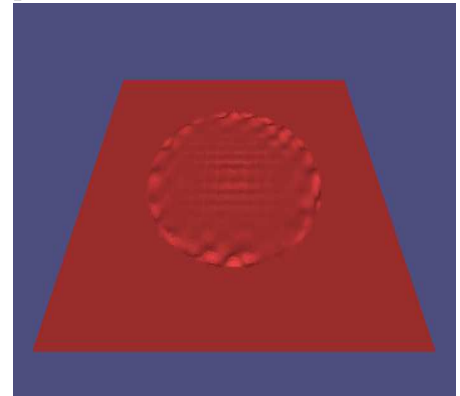


Figure 11: Error for test problem one:  $h=0.04$ .

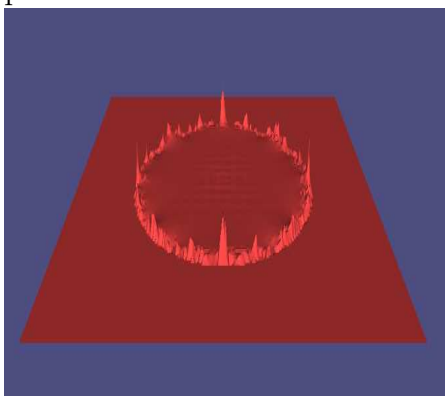


Figure 12: Error (multiplied by 100) for test problem one:  $h=0.01$ .

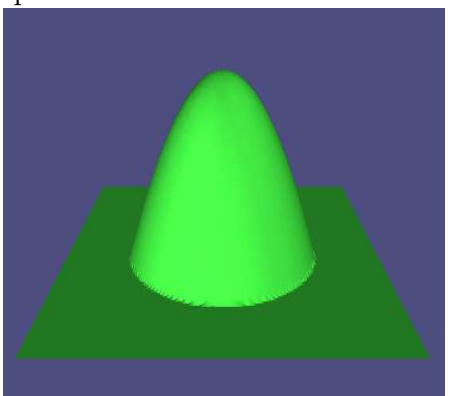


Figure 13: Solution for test problem one:  $h=0.01$ .

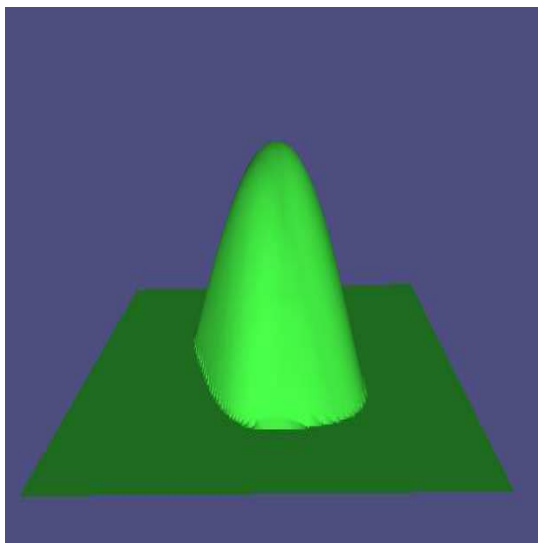


Figure 14: Solution for problem two with 30 by 30 grid

*Test problem three* is Laplace's equation over the the same region as that depicted in figure 3:

$$-\Delta u = 0 \quad \text{in } \Omega \quad \text{and} \quad u = x^2 + y^2 \quad \text{on } \partial\Omega. \quad (14)$$

See figure 15 for a graphical solution of the this problem.

*Test problem four* is the same as the previous problem, but with a hole in  $\Omega$ , see figure 16.

## 5 Conclusion

We have implemented and tested WEB-spline finite elements method for elliptic PDEs. Comparing conditioning numbers of stiffness and reduced stiffness matrixes shows that B-spline extension process is vital for producing a reliable numerical solution.

We have tested the order of convergence of this method against known analytical solution. Based on methods used to compute surface integrals we expect order of convergence  $O(h)$  in regions intersecting the boundary or offsetted boundary. In majority of the region we expect convergence order  $O(h^3)$ . The results of our test runs show expected order of convergence in most of the cases. For finer grid we see reduction in the order of convergence, which we attribute to insufficient accuracy of midpoint rule for calculating

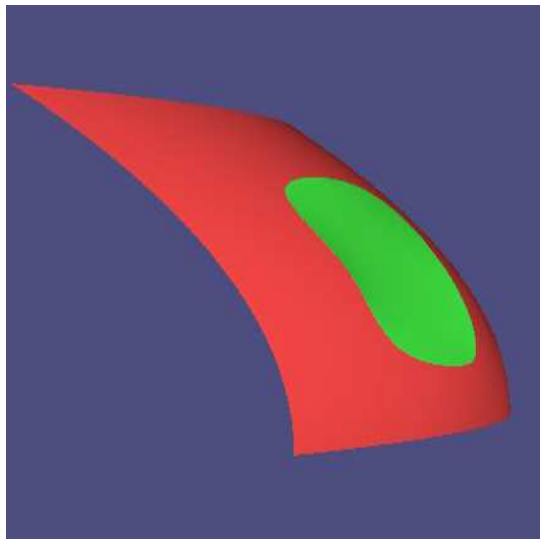


Figure 15: Solution for problem three with 30 by 30 grid

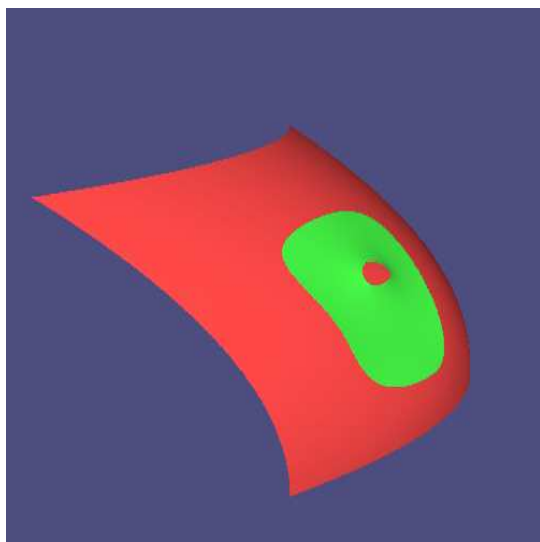


Figure 16: Solution for problem four with 30 by 30 grid

surface integrals in the boundary regions. This result brings us to the conclusion that it is important to take into account properties of forcing function as well as topology of the boundary in order to compute stiffness matrix and right hand side with desired accuracy. Hollig suggests subdividing boundary cells into two regions and integrating over smoothly deformed rectangles using appropriate quadrature rule.

Overall we conclude that web-splines finite element method is a good choice for solving PDEs numerically. Extension routine only needs to be programmed once and can be used for any domain, even if boundary is made up of several curves. This eliminates the expensive operation creating a mesh for every domain. Weight function makes it easy to satisfy boundary condition. And higher order of convergence reduces the time needed to get a solution of desired accuracy.

## 6 Acknowledgements

Team Web-spline would like to thank IMA for all its support, encouragement, and use of the deluxe coffee making machine. A huge shout out to the administrative assistants who keep things running smoothly and arrange delicious snacks for tea time. In addition, the students would like to thank Mentor Thomas Grandine whose boundless enthusiasm inspired us all to explore new applications of mathematics regardless of the time of day or night.

## References

- [1] Klaus Höllig: *Finite Element Methods with B-Splines*, Siam, Philadelphia, 2003.
- [2] Kendall Atkinson, Weimin Han: *Theoretical Numerical Analysis: A Functional Analysis Framework*, Second Edition, Springer, 2005.