A New Approach for Solving Partial Differential Equations Based on B-splines and Transfinite Interpolation

Yuanjie Liu
Key Laboratory of Mathematics Mechanization AMSS
Chinese Academy of Sciences,
100190, Beijing, China
liuyj@mmrc.iss.ac.cn

Hongbo Li
Key Laboratory of Mathematics Mechanization AMSS
Chinese Academy of Sciences,
100190, Beijing, China
hli@mmrc.iss.ac.cn

ABSTRACT

The objective of this paper is to discuss an approach which combines B-spline patches and transfinite interpolation to establish a linear algebraic system for solving partial differential equations. We modified the WEB-spline method developed by Klaus Hollig to derive this new idea. First of all, we replace the R-function method with transfinite interpolation to build a function which vanishes on boundaries. Secondly, we simulate the partial differential equation by directly applying differential operators to basis functions, which is similar to the RBF method rather than Hollig's method. These new strategies then make the constructing of bases and the linear system much more straightforward. And as the interpolation is brought in, the design of schemes for solving practical PDEs can be more flexible.

Keywords: Finite Element Method, B-spline representation, transfinite interpolation, WEB-spline, RBF.

1 Introduction

While the widely used standard finite element method is powerful, there are still several defects: *The basis functions are based on meshes, but the generating of meshes is usually an expensive process, especially for domains with complex boundaries. Since the basis functions are built according to the tessellation of the domain, it is difficult to construct basis functions with high order continuity.* To avoid these problems, massive efforts have been made to develop meshless methods. We refer the reader to [1] and the references therein for a survey.

In this paper, we are particularly interested in two methods: the WEB-spline method [8] and the Radial Basis Function method [12]. WEB-spline method uses regular grids and R-functions to establish basis functions, then build a linear system through Galerkin's Method. The Radial Basis Function method, which is usually called RBF for short, establishes basis functions based on scattered points and directly differentiates these bases to build the linear system.

Our work is mainly inspired by these two schemes. We developed a meshless method which combines B-spline and transfinite interpolation techniques to offer a more convenient way of constructing basis functions with arbitrary order of continuity, like what the WEB-spline method does. Since we have replaced the R-functions method with the transfinite interpolation method, it is appropriate to simulate the partial differential equation through direct differentiation as what the RBF method does.

In this paper, we will take the Poisson equation on a bounded domain with homogeneous Dirichlet boundary conditions to illustrate the basic steps. Section 2 briefly introduces the idea of FEM and explains what a B-spline patch is. Section 3 shows the previous work on the WEB-spline method. Section 4 introduces radial basis function method. Then Section 5 exhibits the method we develop in this paper which employs both B-spline patches and transfinite interpolation to attain an approximation of finite elements. We also discuss about the main difference

between the old and new results in this section. Section 6 presents the solution for a simple situation to illustrate the result of our implementation. Section 7 concludes the main ideas and speculates on possible future directions.

2 Finite Element Method and B-spline patches

2.1 Finite Element Method

The finite element method is one of the most widely used techniques in computational mechanics. The mathematical origin of the method can be traced to a paper by Courant in 1943 [2]. We refer the readers to the article by Oden [3] for the history of the finite element method and [4] [5] for further understanding. Here we give a short introduction for sketch.

The basic principle of finite element method can be illustrated by solving the Poisson equation on a bounded domain with homogeneous Dirichlet boundary conditions:

$$-\Delta u = f \text{ in } D \subset \mathbb{R}^m, u = 0 \text{ on } \partial D \qquad (1)$$

$u(x)$ can be considered as the displacement of an elastic membrane fixed at the boundary under a transverse load $f(x)$, as shown in Figure 1 (Left).

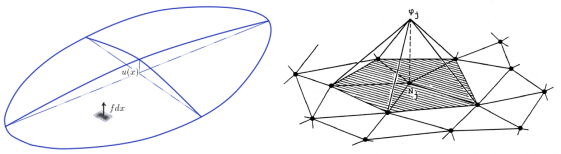

Figure 1: *Left: $u(x)$ is the displacement, $f(x)$ is the load. Right: φ_j is the hat-function.*

We briefly describe FEM for the model problem with piecewise linear functions (this part of contents mainly cited from [6]).

Construct a finite-dimensional subspace $V_h \in V$ as follows, and in what follows, let us assume that ∂D is a polygonal curve. Make a triangulation of D by subdividing D into a set $T_h = \{K_1, K_2, ..., K_m\}$ of non-overlapping triangles, such that no vertex of one triangle lies within the edge of another triangle,

$$D = \bigcup_{K \in T_h} K = K_1 \bigcup K_2 \cdots .$$

T_h is known as a tessellation of domain D. Let N_i denote the vertices of these triangles.

Based on this tessellation, we construct a group of piecewise linear functions $F = \{\varphi_j\}$ so that F satisfies

1. $\varphi_i(N_i) = 1$.
2. φ_i is linear on each K_l which has N_i as one of its vertex.
3. φ_i is zero outside the area $\bigcup\{K_l$, where N_i is one vertex of $K_l\}$.

Figure 1 (Right) shows the definition of φ_j clearly. φ_j is usually called a hat-function and is considered as the basis function.

Then, let us define: $V_h = \{v : v$ is continuous on D, $v|_K$ is linear for K $\in T_h$, $v = 0$ on $\partial D\}$ where $v|_K$ denotes the restriction of v to K (i.e. the function defined on K agreeing with v on K).

To describe $v \in V_h$, we choose the values $v(N_i)$ at the nodes N_i, of T_h, but exclude the nodes on the boundary for $v = 0$ on ∂D.

The space V_h is a linear space of dimension M with basis $\{\varphi_i\}_{i=1}^M$.

Then v has the representation: $v(x) = \sum \eta_i \phi_i(x),\ x \in int(D) \bigcup \partial D$.

A weak solution of this problem can be characterized as the minimum of the functional

$$\frac{1}{2}\int_D \nabla v \nabla v - \int_D fv\ , v \in V, \qquad (2)$$

or equivalently by the equations

$$\int_D \nabla v \nabla v = \int_D fv\ , \forall v \in V, \qquad (3)$$

where, $V = {H_0}^1$ is the Sobolev space of functions which vanishes on the boundary and has square integrable first derivatives. Replace $V = {H_0}^1$ by finite dimensional spaces V_h, which contains closer approximations to u as their dimensions increase. This leads to special cases of the methods of Ritz and Galerkin, respectively.

Take the Galerkin's method for example. Let $a(\cdot,\cdot)$ denote the operator $\int_D(\nabla\cdot, \nabla\cdot)$. To find $u_h \in V_h$ such that $a(u_h, \varphi_i) = (f, \varphi_i), i = 1, 2, \ldots, M$. Assume $u_h{=}\Sigma\eta_j\varphi_j$. Since $a(\cdot,\cdot)$ is a linear functional, $a(u_h, \varphi_i){=}a(\Sigma\eta_j\varphi_j, \varphi_i){=}\Sigma\eta_j a(\varphi_j, \varphi_i)$. We finally obtain the following linear system of equations:

$$\sum \eta_j a(\varphi_j, \varphi_i) = (f, \varphi_i), i = 1, 2, \ldots, M \qquad (4)$$

where $A = (a_{ij})$ is the $M \times M$ stiffness matrix with $a_{ij} = a(\varphi_j, \varphi_i)$, and $b = (b_i)$ is the force vector

where $b_i = (f, \varphi_i)$, and $\xi = (\xi_i)$ is the solution vector where $\xi_i = u_h(N_i), i = 1, 2, \ldots, M$.

2.2 B-spline patches

The B-spline basis is widely used in CAGD [7], and its rational form, known as NURBS, is the industrial standard for geometric modeling nowadays [11]. B-splines are defined in terms of a knot sequence $t := (t_j)$,

$$\cdots \leq t_j \leq t_{j+1} \leq \cdots .$$

The jth B-spline of order 1 for the knot sequence t is the characteristic function of the half-open interval $[t_j,\ t_j + 1)$, i.e., the function given by the rule

$$B_{j,1,t}(x) = \begin{cases} 1 & : \quad t_j \leq x < t_{j+1}; \\ 0 & : \quad otherwise \end{cases}$$

A B-spline basis is often defined in its iteration form:

$$B_{j,k,t}(x) = \frac{x - t_j}{t_{j+k-1}} B_{j,k-1,t} + \frac{t_{j+k} - x}{t_{j+k} - t_j} B_{j+1,k-1,t}$$

The subscript j is the sequence number, k is the order and t indicates the corresponding partition sequence. When the order k and partition t are fixed, we often omit them and only write B_j for short.

Actually, the B-spline basis can be treated as a piecewise continuous function and its order minus 1 is equal to its order of continuity in the global view. This can be seen in Figure 2.

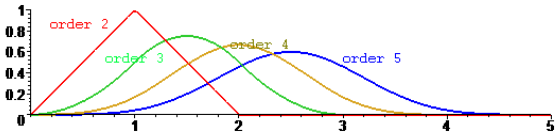

Figure 2: *B-spline bases with order 2, 3, 4, 5.*

Because of this property, it is extraordinarily appropriate to use B-spline as a basis function to construct a solution with any order continuity.

Extending the B-spline basis to dimension 2 directly, we get the B-spline patch which is the tensor product of B-spline bases in two directions. It can be defined as:

$$\mathrm{B_{ij}}(x, y) = B_i(x) B_j(y) \tag{5}$$

3 WEB-spline method

Weighted extended B-spline method (WEB-spline method) for boundary value problems are proposed as a natural generalization of standard B-splines to address two important issues of exact fulfillment of the boundary conditions and well conditioning of the Galerkin system [11]. The development of this new strategy for the Laplace operator problem is credited to Hollig et al. in [8].

To approximate the finite element with splines, one problem must be fixed: the boundary condition. The basic idea is to construct an auxiliary function which is zero on the boundary and nonzero in the interior of the domain, then multiply this function with B-spline basis. The new basis we get is called a weighted B-spline basis. We use R-functions to build the auxiliary function we desire.

3.1 R-functions

First of all, we define the weight function of a domain mathematically.

Definition 1. *A function w is a weight function of the domain D, if $w(x) > 0$ when $x \in int(D)$, $w(x) = 0$ when $x \in \partial D$, and $w(x) < 0$ otherwise.*

We want to establish a weight function of the designated domain through the weight functions of its subsets. R-function is a method to achieve this goal.

Definition 2. *A function $r : \mathbb{R}^k \to \mathbb{R}$ is an R-function, if its sign depends only on the sign of its arguments.*

R-functions are closely related to boolean functions. In fact, for any Boolean set operation $\circ$ there exist associated R-functions $r_\circ$, which define the corresponding operation on weight functions. In other words, if w_v is a weight function for a set D_v, then

$$(w_1 \circ_R w_2)(x) := r_\circ(w_1(x), w_2(x))$$

is a weight function for $D_1 \circ D_2$. It is not difficult to construct these R-functions. And the following table is a good choice [8].

Set operation	Corresponding R-function
$D_1 \cap D_2$	$r_\cap = x_1 + x_2 - \sqrt{{x_1}^2 + {x_2}^2}$
$D_1 \cup D_2$	$r_\cup = x_1 + x_2 + \sqrt{{x_1}^2 + {x_2}^2}$
D^c	$r_c(X) = -x$

To build weight functions on arbitrary domain with complex boundaries, it will be necessary to approximate the domain with several kinds of primitives through boolean operations, i.e., union, intersection or minus. Such primitives include disk, polygon and other simple elements whose weight functions are not too difficult to identify. Then, we need to get these functions together through R-functions of the corresponding boolean operations between these elementary domains. The weight function we get through the method of R-functions may be complex.

3.2 WEB-spline method

With the help of R-functions, we get the weight function w on a particular domain which will be zero on the boundary. This fact enables us to multiply B-spline patches with this w and obtain a set of bases which is always zero on boundary of the same domain. This set $\{wB_i\}$ is called weighted B-spline bases and is already suitable to play the role of finite elements to constitute the solution of PDE. However, according to Hollig [8], only using weighted B-spline bases is not a good choice for solving because it often leads to a large condition number of the stiffness matrix. To conquer this problem, they invented the extended B-spline; the concept can be explained by Figure 3.

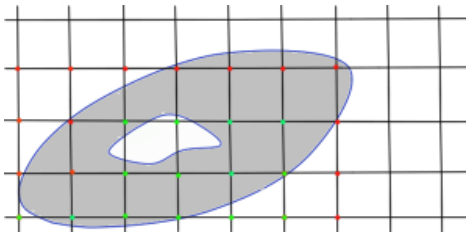

Figure 3: *Red dot labels outer basis, green dot labels inner basis, blue dot means the basis has even more than one cell totally contained in the domain.*

In this picture, we take B-spline patches with degree 3 for illustration. We use the low left point to represent the support of each patch. If there is at least one lattice (or called cell) of the support totally contained in the domain, we call it inner B-spline, like the green ones in the picture. Those patches whose support intersects with but has no lattice totally contained in the domain are called outer B-splines, like the red ones in the picture. Let us consider the B-spline representation $p(x) = \sum q(k)B_k(x), x \in D$. We can represent the outer ones' coefficients $q(k)$ through the inner ones' (see [8] for detail). Then rearrange the summation and get the new expression $p(x) = \Sigma q(i)\left[B_i(x) + \Sigma e_{i,j}B_j(x)\right],\ x \in D$ where B_i represents the inner splines and B_j represents the outer ones. The $B_i(x) + \Sigma e_{i,j}B_j(x)$ is treated as the new basis, known as WEB-splines which means weighted extended B-splines. Using the WEB-spline basis to build linear systems can avoid an ill coefficient matrix in most cases.

4 RBF method

4.1 Radial Basis Functions

A function $\phi(r_k)$,where r_k=$\sqrt{(x-x_k)^2+(y-y_k)^2}$, is referred to as a radial function, because it depends only upon the Euclidean distances between the points (x, y) and (x_k, y_k). The points (x_k, y_k) are referred to as centers or knots. In particular, the function $\phi(r_k)$ is radially symmetric around the center (x_k, y_k) [12].

Initially, RBF is introduced for scattered data interpolation. The solution to the scattered data interpolation problem is obtained by considering a linear combination of the translates of a suitably chosen radial basis function. Sometimes a polynomial term is added to the solution, when ϕ_k is conditionally positive definite, or in order to achieve polynomial precision [12].

4.2 RBF for solving partial differential equation

In this frame, we can consider a form $F(x, y) = \sum A_k\phi(r_k)$ to represent a solution of a differential equation. Apply the differential operator σ directly to $F(x, y)$ and solve for A_k. That is:

$$
\begin{aligned}
&A_1\sigma(\Phi r_1)(x_1) + \cdots + A_n\sigma(\Phi r_n)(x_1) = b_1\\
&A_1\sigma(\Phi r_1)(x_2) + \cdots + A_n\sigma(\Phi r_n)(x_2) = b_2\\
&\cdots\cdots\\
&A_1\sigma(\Phi r_1)(x_n) + \cdots + A_n\sigma(\Phi r_n)(x_n) = b_n
\end{aligned}
$$

In another approach, radial basis functions with compact support were introduced by several researchers such as Schaback [13]. However, radial basis functions with compact support seem to exhibit inferior convergence properties in comparison to radial basis functions with global support [14].

5 B-spline with Transfinite Interpolation method

Both the WEB-splines method and the RBF method are effective meshless schemes for solving elliptic equations. Inevitably, there are also disadvantages for both of them. The R-function method used in WEB-spline scheme may need huge computation and the integration is also expensive when assembling the stiffness matrix. The RBF method does not need to take the computation for integration since it directly applies the operators to simulate the differential equation. However, the RBF method is built on isolated points and often requires a large number of these points. The representation and evaluation of RBF finite space is generally difficult.

Compared with the previous methods, we have conceived a new approach which employs transfinite interpolation on the boundary to make sure that the solution will fit the boundary constraints. Like the B-spline patches, Coons patches are also convenient to manipulate. We could easily design the order of continuity and set the values on the boundary. We prefer to name our approach **BTI** for short, which actually means B-spline and Transfinite Interpolation method.

Besides, we are very pleased to see that our idea has some relationship with the novel area Isogeometric Analysis (IGA) which is gaining more and more attention. The IGA method employs NURBS directly as finite elements and develops refinement schemes to approximate the solution of the PDE. We refer the readers to [15] for a survey.

5.1 Transfinite interpolation on the boundary

We use the expression Transfinite Interpolation to indicate the situation that when continuous curves are given, we construct a continuous surface to pass through them. This idea is also known as Coons patches which was developed by Coons himself [16]. The simplest case is the bilinear Coons patch. It can be explained as follows.

We are given two pairs of curves $(X(0,u), X(1,u))$, $(X(v,0), X(v,1))$ which are all continuous and intersect in four points $X(0,0), X(0,1), X(1,0), X(1,1)$. The Coons patch is defined as:

$$
\begin{aligned}
&X(u,v) = \\
&(1-u)X(0,v) + uX(1,v) + (1-v)X(u,0) + vX(u,1) \\
&- \begin{pmatrix} 1-u & u \end{pmatrix} \begin{pmatrix} X(0,0) & X(0,1) \\ X(1,0) & X(1,1) \end{pmatrix} \begin{pmatrix} 1-v \\ v \end{pmatrix}
\end{aligned}
$$

This definition can be comprehended as that we compute linear interpolation on the two pairs of curves and bilinear interpolation on the four corner points, and then combine them to construct the desired surface.

Further more, the coons patch can be generalized if we replace lines with curves of higher order for interpolation. For example, we can use bi-cubic coons patches to carry out the cubic Hermite interpolation [17]. The coons patches can also be generalized, resulting in a new method based on a blend of variational principles [18].

In our method, when the ellipse differential equation with boundary constraints is given(here we assume that the domain is convex and the boundary curve is C^2 for simple), we first convert the initial problem into a homogeneous form. Then, according to the boundary which is treated as a pair of curves, we employ the following idea slightly modified from Coons' to construct a patch which passes through them.

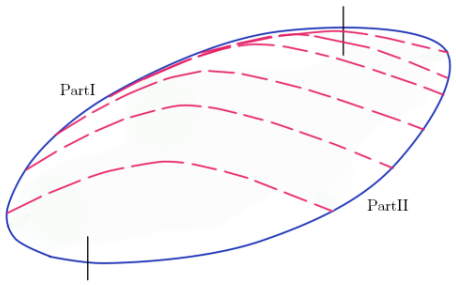

Figure 4: *The boundary is cut into two parts, red curves in perpendicular planes connect corresponding points of the each part.*

Figure 4 is an illustration for the idea. In this sketch map, we cut the boundary into PartI and PartII. The red curve, which is called *bridge*, means the interpolating curve that connects points of these two parts.

We just consider a simple connected area for short. Use a line parallel with x-axis, scan it from the top to the bottom. Label the first point this line touches the domain as θ_1 and the last one as θ_2.

So the boundary of our domain is splitted into two parts: γ_1, γ_2 which are connected with each other through θ_1 and θ_2. Then the two curves are the data we need to interpolate.

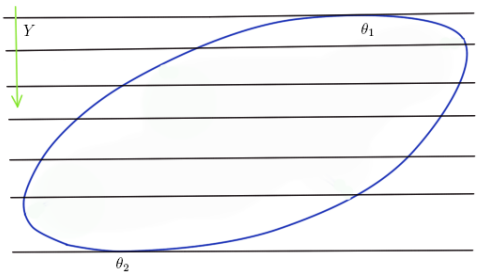

Figure 5: *Scan from the top to the bottom, the first and last points touched by the scan line are labeled θ_1 and θ_2 respectively, which are border points of the two parts.*

In the very simple illustration of this paper, we simply use circular arc as the bridge for interpolation. If more flexible curves are needed, it is not difficult to extend the bridge curves to Bezier, NURBS or any other format.

Theorem 1. *Let $x = \gamma_1(y), x = \gamma_2(y)$ be the two parts of designated boundary obtained by the scheme referred to above. Apply transfinite interpolation to them using the square of the semi-circle bridge, i.e. $z = r^2 - (O_x - x)^2$, for fixed y and radius r, center O; and we get a patch expressed as:*

$$(\ x, \ y, \ (\gamma_1(y) + \gamma_2(y))x - x^2 - \gamma_1(y)\gamma_2(y) \ ) \quad (6)$$

so it is as smooth as the boundary.

Proof. Scan along the direction of y, for a fixed $\bar{y}$, we get x_1 on left part of the boundary and x_2 on the right. An arbitrary point on the square-arc bridge connecting the two points $(x_1, \bar{y})$ and $(x_2, \bar{y})$ is (x, y, z), then $y = \bar{y}$. The radius of this semi-circle arc r is $\frac{x_2 - x_1}{2}$ and the center O is $(\frac{x_1 + x_2}{2}, \bar{y}, 0)$, so

$$z = r^2 - (\frac{x_1 + x_2}{2} - x)^2 = (\frac{x_2 - x_1}{2})^2 - (\frac{x_1 + x_2}{2} - x)^2 = (x_1 + x_2)x - x^2 - x_1 x_2$$

Therefore, the coordinates (x, y, z) of the surface we constructed can be expressed as $(\ x, \ y, \ (x_1 + x_2)x_1 - {x_1}^2 - x_1 x_2 \ )$, i.e. $(\ x, \ y, \ (\gamma_1(y) + \gamma_2(y))x - x^2 - \gamma_1(y)\gamma_2(y) \ )$. □

5.2 Establishing the linear system

Now, it's time to construct algebraic equations. Assume u is the solution of the PDE. The basis functions are B-spline patches (see eq.(5)) multiplied with the smooth function obtained by applying the transfinite interpolation technique on the boundary:

$$(\gamma_1(y) + \gamma_2(y))x - x^2 - \gamma_1(y)\gamma_2(y)$$

we write u as the summation of these basis functions with coefficients c_{ij}:

$$u = \sum c_{ij} B_i(x) B_j(y) \cdot ((\gamma_1(y) + \gamma_2(y))x - x^2 - \gamma_1(y)\gamma_2(y))$$

Then apply differential operator δ to u, and get an expression:

$$\delta u = \sum c_{ij} \delta(B_i(x) B_j(y) \cdot ((\gamma_1(y) + \gamma_2(y))x - x^2 - \gamma_1(y)\gamma_2(y))) \quad (7)$$

$B_i(x)B_j(y)((\gamma_1(y)+\gamma_2(y))x - x^2 - \gamma_1(y)\gamma_2(y))$ is the BTI basis and we label it T_i. Finally, to get equations, we need to choose some points in the domain and evaluate the expression on these points, and set them equal to their corresponding value determined by the function on the right side of the differential equation. The choosing of these data points for testing may be flexible and it can influence the solving process and result. To avoid the shake of solution relative to the choice of data points, it is necessary to construct a coefficient matrix several times and take their average. Then, we get the linear system:

$$\begin{aligned} &a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ &a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ &\cdots\cdots \\ &a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = b_n \end{aligned}$$

In the system, $a_{ls} = (\delta T_s)(p_l)$ and x_s is the coefficient of T_s, p_l is the data point where we evaluate u.

5.3 Solving the equations

The numeric method for solving the linear system can be chosen freely. However, since the B-spline basis has finite supports, it is more likely to get a sparse matrix when the number of basis functions is large. When the order of the coefficient matrix is not very high, direct methods, such as the simple Gauss diminishing, are convenient. For a big

matrix, we suggest GMRES [10] as a way to find a solution although the speed of convergence is not guaranteed. In our experiment, we adopted a preconditioning method which was developed by several Chinese mathematicians [9] and successfully accelerated the speed of convergence of GMRES.

5.4 Implementation

The whole process of finite element simulation with B-splines and Transfinite Interpolation consists of four steps:
1.The preconditioning procedure.
2.Construction of the basis.
3.Assembly of the finite element system;
4.Direct or iterative solving.

In the preconditioning step, we need to transform the differential equation into its homogeneous form first. Then, divide the boundary for interpolation. The algorithm for partition is just scanning to find the highest and lowest points and make them the ends of each part. If the designated domain is not trivially simple connected, partition it into simple parts, set the value on additional boundaries to nonzero and then implement the interpolation.

In the second step, to construct the basis, we cover the whole domain with regular grids. Define B-spline patches on these grids. Multiply these tensor B-spline bases with the function which was obtained in the previous step to get our new bases T_i.

Next, apply the differential operators δ to the summation $\Sigma x_i T_i$ where x_i is the coefficient of corresponding basis T_i. Choose appropriate data points and evaluate the expression $\Sigma x_i \delta T_i$ on these points. We get the algebraic linear system $\Sigma x_i \delta T_i(p_i) = f(p_i)$ to solve.

Finally, choose an appropriate method to solve this system: the direct or iterative method.

6 Example and Results

We just take a most simple Poisson equation for illustration. In

$$-\Delta u = f \text{ in } D \subset \mathbb{R}^m, u = 0 \text{ on } \quad \partial D \qquad (8)$$

Let $f \equiv -1$ and choose a regular Domain $\{x^2+y^2 < 1\}$ as D, so that the solution for equation (7) can be easily written out. It is $u = \frac{1}{4}(x^2+y^2-1)$. The diagram of this function is shown in Figure 6.

Use the **BTI** method to solve this problem; cover D with the square $[-1,1]\times[-1,1]$, divide it into $28\times$

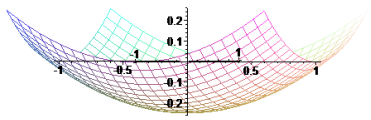

Figure 6: *The solution of the equation (8).*

28 lattices. Take the same number of data points on the square uniformly. In order to avoid the twist of solution caused by the choice of sample points, we take an average value. To each data point, take 9 points in the neighborhood with radius one third of the lattice's width and evaluate the expression (7) on them. Average these evaluations as one equation in the linear system. Then, solve this system, we can get a perfect solution as shown in Figure 7.

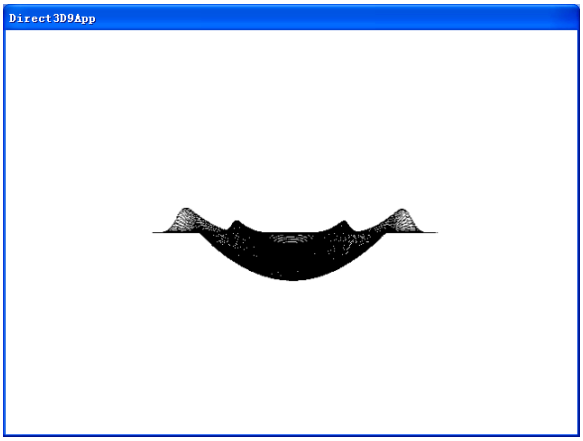

Figure 7: *The solution resulted from our method.*

7 Conclusion and further work

The idea of our **BTI** method is basically similar to the WEB-spline method, both of which consist of two principal steps to construct the basis. We use transfinite interpolation to construct the aid function instead of using the R-function method. We also modified the process of assembling the linear system by applying the operator to the summation directly rather than trying to minimize a functional. In this way, the computation of integration can be avoided and the whole process becomes more convenient and intuitional.

Obviously, further work needs to be done about the **BTI** method. The convergence of this algorithm should be verified when we increase the number of bases in a fixed domain. We will also analyze the stability of this method in the next step, and particularly focus on the choice of data points for getting equations and how it influences the final result.

Furthermore, from my point of view, this **BTI** approach is likely to be paralleled. The GMRES method can be designed to run under CUDA. If we are able to design a proper scheme to divide our domain to parallel the interpolation, the entire **BTI** approach will become a parallel algorithm.

References

[1] T. Belyschko, Y. Krongauz, D. Organ, M. Fleming, and P. Krysl, Meshless methods: An overview and recent developments, *Comput. Methods Appl. Mech. Enrg.*, pp.3-47, 139(1996).

[2] Giuseppe Pelosi, The finite-element method, Part I: R. L. Courant: Historical Corner, *Antennas and Propagation Magazine, IEEE*, pp.180-182, April 2007.

[3] J. Oden, Finite elements: an introduction, *Handbook of Numerical Analysis, vol. II, Ciarlet PG and Lions JL (eds)*, North Holland: Amsterdam, 3-15, 1991.

[4] Philippe G. Ciarlet, The finite element method for elliptic problems, *North-Holland Pub.Co., Amsterdam*, 1978.

[5] Douglas H. Norrie and Gerard de Vries, The finite element method, *Academic Press, New York*, 1973.

[6] N. Zabaras, Introduction to the finite element method for elliptic problems, `http://mpdc.mae.cornell.edu/Courses/MAEFEM/Lecture1.pdf`

[7] C. de Boor, A practical Guide to Splines. *Springer-Verlag, New York*, 1978.

[8] Klaus Hollig, Ulrich Reif, and Joachim Wipper, Weighted extended b-spline approximation of Dirichlet problems, *Siam J. Numer. Anal.*, Vol.39, No.2, pp.442-462, 2002.

[9] Quan Zhong, Xiang Shuhuang, A GMRES Based Polynomial Preconditioning Algorithm, *Mathematica Numerica Sinica*, Vol.28, No.4. Nov. 2006.

[10] Richard Barrett, Michael Berry, Tony F. Chan, James Demmel, June M. Donato, Jack Dongarra, Victor Eijkhout, Roldan Pozo, Charles Romine, and Henk Van der Vorst, Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, `(http://www.siam.org/books)`

[11] Gerald Farin, Josef Hoschek, Myung-Soo Kim, Handbook of Computer Aided Geometric Design. *Elsevier*, 2002.

[12] Suresh K. Lodha and Richard Franke, Scattered Data Interpolation: Radial Basis and Other Methods, *Handbook of Computer Aided Geometric Design. Elsevier*, Elsevier, pp.389-404. 2002.

[13] R. Schaback and H. Wendland, Characterization and construction of radial basis functions, *Multivariate Approximation and Applications*, pp.1-16, ISBN-13: 9780521800235 — ISBN-10: 0521800234.

[14] M. D. Buhmann, Radial Basis Functions, *Acta Numerica*, pp.1-38, 2000.

[15] T. J. R. Hughes, J. A. Cottrell, Y. Bazilevs, Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement, *Comput. Methods Appl. Mech. Engrg.*, pp.4135-4195, 194 (2005).

[16] S. A. Coons, Surfaces for computer-aided design of space forms, *Technical Report: TR-41*, MIT, 1967.

[17] Farin Gerald, Curves and surfaces for computer-aided geometric design :a practical guide, *Academic Press, San Diego*, c1997

[18] Gerald Farin and Dianne Hansford, Discrete Coons patches, *Computer Aided Geometric Design* 691C700, 16 (1999).