Solution of PDEs on Manifolds: The Heat Equation on a Circle

Parousia Rockstroh

May 1, 2008

1 Introduction

The basic techniques of solving elementary partial differential equations (PDEs) in $\mathbb{R}^n$ are readily available in the literature, and are even taught in undergraduate courses. Often times, however, complex PDEs have no known analytic solution. Hence, a great deal of numerical techniques have been developed to deal with solving complex problems involving PDEs. The majority of these techniques apply to PDEs that are posed in a Cartesian coordinate system. Often times in math, science, and engineering, however, it is necessary to solve PDEs on surfaces other than the typical $\mathbb{R}^n$ domain. This area is relatively less studied, and until recently, even the simplest examples have required very complex techniques.

In this paper we will numerically solve the heat equation:

$$u_t = k\Delta u \tag{1}$$

on the unit circle. We will use the circle as a simple example of some of the complications that arise from solving PDEs defined on non-Cartesian domains. We will be particularly interested in using the closest point method for solving the heat equation on the circle. The closest point method is a technique that allows one to solve PDEs on surfaces that are embedded in $\mathbb{R}^n$ by making use of the Cartesian structure of the ambient Euclidean space. Although we are using this technique exclusively for the heat equation on the circle, the closest point method can also be used with more sophisticated PDEs and on more complex surfaces (see for example: [1], [4], [5], [7], or [11]).

2 Posing the Problem: Initial Data on the Unit Circle

As mentioned above, the primary focus of this paper is to numerically solve the heat equation on the circle. In particular, we will focus our attention on

the unit circle. In terms of polar coordinates, we specify the following initial condition on the unit circle:

$$u(\theta, 0) = \sin\theta \qquad\qquad 0 \leq \theta \leq 2\pi. \tag{2}$$

In order to get a better feel for what the initial condition looks like, we have included a 3-D plot of it below. We are interested in how the given heat distribution diffuses through time. That is, we interested in finding a solution to the heat equation with this given initial distribution. We denote such a solution by $u(\theta, t)$. In the section that follows we will derive an analytic solution of $u(\theta, t)$ which governs how the distribution $u(\theta, 0) = \sin\theta$ diffuses over the unit circle through time. Following this, we will develop numerical techniques for solving the same problem, and compare our results with the analytic solution.

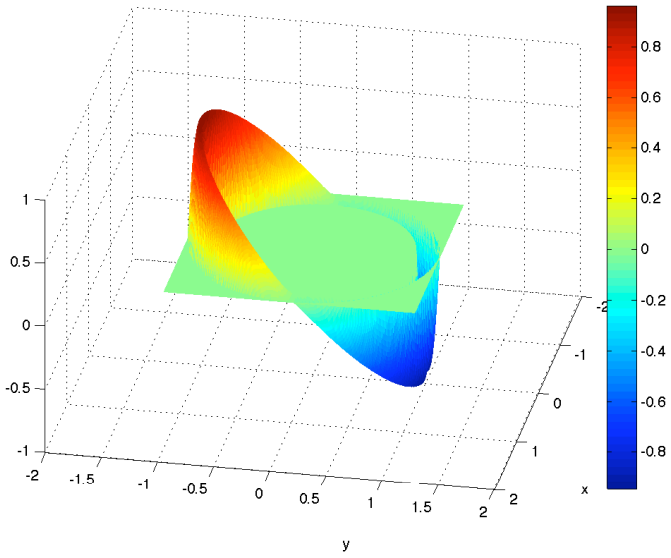

Figure 1: Initial condition for the heat equation on the unit circle. The initial condition, $u(\theta, 0) = \sin\theta$, is used throughout the paper. In the plot above, θ is measured with respect to the center of the blue region. Blue represent an elevation of -1, while red represents an elevation of 1.

3 Analytic Solution of the Heat Equation on a Circle

We will now present an analytic solution of (1) subject to (2) so that we may compare our numerical results to an actual solution. Moreover, this will help us understand the problem at hand more concretely. We begin by stating the heat equation in polar coordinates by making the substitution:

$$\begin{aligned} x &= r\cos\theta \\ y &= r\sin\theta \end{aligned}$$

where r is the radius of the circle and $0 \leq \theta \leq 2\pi$. Under this transformation, the Laplace operator is expressed as:

$$\Delta u = u_{rr} + \frac{1}{r}u_r + \frac{1}{r^2}u_{\theta\theta}. \tag{3}$$

In our case, we are dealing with a circle of constant radius so that (3) becomes $\Delta u = \frac{1}{r^2}u_{\theta\theta}$. Moreover, we are concerned with the unit circle, thus in polar form we are interested in solving:

$$\begin{aligned} u_t &= u_{\theta\theta}, & t \geq 0, 0 \leq \theta \leq 2\pi & \quad (4) \\ u(\theta, 0) &= \sin\theta & 0 \leq \theta \leq 2\pi, & \quad (5) \end{aligned}$$

By applying separation of variables, we find that the solution of (4)-(5) is given by:

$$u(\theta, t) = e^{-t}\sin\theta.$$

Rather than giving a detailed derivation of this formula here, the reader is encouraged to see [4] or [6] for a derivation. It is straightforward to verify that $u(\theta, t)$ satisfies (4)-(5). A graphical representation of $u(\theta, t)$ is shown in Figure 2 below. Notice, in particular, that the solution converges to $u(\theta, t) = 0$ as t increases.

It is useful to have an analytic solution of the (4)-(5) so that we can compare our numerical calculations to the exact solution. We now turn our attention to solving this problem numerically, for which we develop some theory.

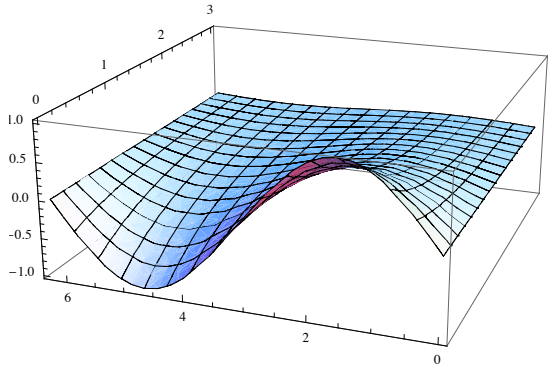

Figure 2: Graphical representation of the solution $u(\theta, t) = e^{-t} \sin \theta$, where $\theta \in [0, 2\pi]$ and time $t \in [0, 3]$. Notice that as time progresses, $u(\theta, t)$ is damped and eventually approaches $u(\theta, t) = 0$.

4 The Closest Point Method

We will now present the closest point method which we use to solve the heat equation on the circle. This is a relatively new method for solving PDEs on surfaces which are embedded in $\mathbb{R}^n$. It is often used as an alternative to solving PDEs via triangulation of the given surface. The closest point method works by creating a "computational band" around the surface that is one dimension higher than the surface in consideration. We are then able to apply classical numerical techniques that hold in a Cartesian coordinate system.

4.1 Outline of the Method

We will now outline the closest point method and explain the importance of each component of the process. We will assume that we are given a PDE with initial data, u_0, that is defined on a closed surface, S, in $\mathbb{R}^n$ which is of co-dimension one. In particular, we require that S be closed so that it partitions $\mathbb{R}^n$ into an interior and an exterior. We will now proceed to present the closest point method.

1. Begin by constructing a function that differentiates between the interior

and exterior of S. Define a function, $\phi(x)$, on S with the following properties:

$$\phi(x) = \begin{cases} d(x, \partial S) & \text{if } x \text{ is in the interior of } S \\ 0 & \text{if } x \text{ is on } S \\ -d(x, \partial S) & \text{if } x \text{ is in the exterior of } S \end{cases}$$

where $d(x, \partial S)$[1] denotes the distance between x and the closest point on the surface S. We call the function $\phi(x)$ a signed distance function. Intuitively, the signed distance function determines how close a point in $\mathbb{R}^n$ is to the surface S, along with whether the point is inside, outside, or on the surface. An important point to note is that S is the zero level set of the signed distance function.

2. Next, construct a "computational band", Ω_c, around S which is defined as follows:

$$\Omega_c = \{x \in \mathbb{R}^n : |\phi(x)| \leq c\}.$$

Geometrically, Ω_c is an n–dimensional band around S of width $2c$. This will be the band that we do our computations in. Notice that the computational band extends the surface to a region that is one dimension higher than S.

3. Now we proceed by extending the initial data off the surface S in the computational domain Ω_c. We extend the initial data, u_0, off the surface by requiring that it be constant normal to S. In practice, this is done by requiring:

$$\nabla u_0 \cdot \nabla \phi = 0$$

for all points in the domain Ω_c.

4. Now, every point of the computational domain has an associated value. Proceed by using the typical numerical techniques for solving the PDE in question. This can be done since the problem is now in n–dimensional Euclidean space where there is a Cartesian coordinate system. Make sure to re-extend the data off the surface S at every time step.

The closest point method can be applied in a multitude of circumstances. Indeed, it has been demonstrated to work in cases where the PDE in consideration has high-order derivatives as in [5], and even when a closed-form

[1]The symbol ∂ means boundary.

signed distance function is not known as in [7]. The appeal of the method is that PDEs which are posed on surfaces can be solved using typical numerical techniques in a Cartesian coordinate system. We are now ready to apply the closest point method to the heat equation on the unit circle.

4.2 The Heat Equation on the Unit Circle

We will apply the closest point method to the heat equation on the unit circle so that we can establish the computational problem that will be solved. The details of the computational techniques that we use are left for section 5.

We are interested in solving (4)-(5), hence we will use $u(\theta, 0) = \sin\theta$ as our initial data. We will now proceed to apply each step of the closest point method to the problem at hand.

1. Define the function, $\psi(x, y)$, as:

$$\psi(x, y) = \sqrt{x^2 + y^2} - 1. \tag{6}$$

   It is clear that $\psi(x, y)$ satisfies the properties of a signed distance function for the unit circle. It is interesting to note that $\psi(x, y)$ is very similar in nature to the Euclidean metric on $\mathbb{R}^n$ – this, however, is not always the case. For surfaces more complex than the circle, the corresponding signed distance functions are also more complex than $\psi(x, y)$.

2. We will use the following computational domain:

$$\Omega_{\frac{1}{10}} = \{x \in \mathbb{R}^2 : |\psi(x, y)| < 1/10\}. \tag{7}$$

   This is the same computational domain that was used in [1], [4], and [5]. The author of [4] justifies in great detail the choice of this domain based on curvature properties of the circle. We will not reproduce those arguments here.

3. Now we will explain how to use the geometry of the circle to extend the initial function, $u(\theta, 0) = \sin\theta$, off the surface. Observe that given a point $(x_0, y_0) \in \mathbb{R}^2$, it lies on the line

$$y = \frac{y_0}{x_0} x.$$

Moreover, since the line prescribed above passes through the origin, it is perpendicular to the unit circle at the point of their intersection. We will now analytically solve for the intersection of the line $y = \frac{y_0}{x_0}x$ with the circle $\sqrt{x^2+y^2} = 1$. Let (x_c, y_c) denote the point of intersection, then:

$$x_c = \sqrt{\frac{1}{1+(\frac{y_0}{x_0})^2}}, \tag{8}$$

and

$$y_c = \sqrt{\frac{1}{1+(\frac{x_0}{y_0})^2}}. \tag{9}$$

In practice, we use equations (8)–(9) to extend the initial data off the unit circle. We also use (8)–(9) in our numerical calculations when the numerical scheme requires points outside of the computational domain.

4. We will leave our discussion of the numerical techniques that we use for section 5.

Now, the problem of solving the heat equation on the unit circle with $u(\theta, 0) = \sin\theta$ is posed in terms of subset of $\mathbb{R}^2$ on which we can apply typical numerical techniques. Next, we discuss the assumptions of the closest point method.

4.3 Assumptions

In using the Closest Point Method, there are several assumptions that are made. We list them explicitly below.

- We assume that the surface that we are dealing with is smooth and closed in $\mathbb{R}^n$ for some $n \in \mathbb{Z}^+$. As discussed in section 4.1, we require that our given surface be closed so that it partitions $\mathbb{R}^n$ into an interior and exterior. In our case we are dealing with the unit circle which is clearly both smooth and closed in $\mathbb{R}^2$.

- We assume that there exists a function[2], $\phi(x)$, as defined in step 1 of section 4.1. As we have shown, $\psi(x, y)$ is such a function for the unit circle.

[2]It has recently been shown that this function need not have a closed form, see **??**.

- When extending initial data off of the zero level set of $\phi(x)$ we assume that the initial conditions are constant along normals to the surface. This ensures that the initial data is defined in the entire computational band while keeping the data on the surface unchanged. In our case, re-extending the data off of the unit circle at each step causes the heat equation to disperse along each circular level set in Ω_c. Thus, the assumption that data is constant normal to the surface enables us to solve the heat equation in a circular region.

- We will also assume that

$$|\phi(x)| \leq c < \frac{1}{\kappa_{max}}, \tag{10}$$

  where κ_{max} is the maximum of the principle curvatures along the surface. This restriction ensures that $\nabla\phi$ is well-defined, for a proof of this fact see [4].

Next, we will present the numerical methods that we use to solve the heat equation on the domain $\Omega_{\frac{1}{10}}$. We now turn our attention to the computational implementation of the heat equation on the unit circle.

5 Computational Implementation

As mentioned in step (4) of the closest point method presented in section 4.1, after posing our initial value problem on the computational domain $\Omega_{\frac{1}{10}}$, we can apply classical numerical techniques that apply in the Cartesian coordinate system. This is done in section 5.1. It should also be noted that the problem that we wish to solve is posed in terms of polar coordinates, whereas the numerical method that we develop is one that applies in Cartesian coordinates. In practice, we resolve this problem computationally which we discuss in section 5.2, along with the implementation of our numerical techniques in MATLAB.

5.1 Numerical Solution of the Heat Equation in $\Omega_{\frac{1}{10}}$

We begin by discretizing (4)–(5) in terms of familiar finite difference formulas. For simplicity, we will avoid using techniques that involve multiple spacial steps since the geometry of the domain makes it cumbersome to apply

boundary conditions in these cases. We use a first order forward time difference for the time derivative, and a second order centered difference for both spacial variables. For the purposes of analysis, we will employ the notation $u_{i,j}^n = u(x_i, y_j, t_n)$. In terms of this notation, we discretize the time derivative using a first order forward time difference as follows:

$$u_t \approx \frac{u_{i,j}^{n+1} - u_{i,j}^n}{\Delta t} \tag{11}$$

Next, we discretize each of the spacial derivatives using a second order centered difference which yields:

$$u_{xx} \approx \frac{u_{i+1,j} - 2u_{i,j}^n + u_{i-1,j}^n}{(\Delta x)^2}$$

and

$$u_{yy} \approx \frac{u_{i,j+1} - 2u_{i,j}^n + u_{i,j-1}^n}{(\Delta y)^2}.$$

Combining the discretizations of u_{xx} and u_{yy} presented above, we get the following second order centered difference formula for the Laplacian in $\mathbb{R}^2$:

$$\Delta u \approx \frac{u_{i+1,j} - 2u_{i,j}^n + u_{i-1,j}^n}{(\Delta x)^2} + \frac{u_{i,j+1} - 2u_{i,j}^n + u_{i,j-1}^n}{(\Delta y)^2}. \tag{12}$$

Combining (11) and (12), we arrive at the following discretization of the heat equation:

$$\frac{u_{i,j}^{n+1} - u_{i,j}^n}{\Delta t} = \kappa \left[\frac{u_{i+1,j} - 2u_{i,j}^n + u_{i-1,j}^n}{(\Delta x)^2} + \frac{u_{i,j+1} - 2u_{i,j}^n + u_{i,j-1}^n}{(\Delta y)^2} \right], \tag{13}$$

where κ is the heat diffusion constant which measures how fast heat dissipates. We now isolate $u_{i,j}^{n+1}$ in (13) which yields:

$$u_{i,j}^{n+1} = u_{i,j}^n - \kappa \Delta t \left[\frac{u_{i+1,j} - 2u_{i,j}^n + u_{i-1,j}^n}{(\Delta x)^2} + \frac{u_{i,j+1} - 2u_{i,j}^n + u_{i,j-1}^n}{(\Delta y)^2} \right]. \tag{14}$$

Equation (14) gives us an explicit method for approximating $u(x, y, t)$. We now turn our attention to explaining the implementation of this method in MATLAB.

5.2 Computer Implementation

We will now discuss the implementation of the numerical scheme stated above in MATLAB. We use an $n \times n$ dimensional matrix array to represent a discretization of the square in which our computational domain $\Omega_{\frac{1}{10}}$ is inscribed. Next, we define a an $n \times n$ matrix R which contains the values $\sqrt{x^2 + y^2}$, where x and y represent the vertical and horizontal components of data points in the computational domain. Thus, R defines a discretized metric on the computational domain. The signed distance function for the circle is readily obtained from R by simply subtracting 1 from each entry.

Next, we define a matrix $Z1$ in which we store the initial values of the function $u(\theta, 0) = \sin\theta$. Notice that our domain has a Cartesian structure, whereas our initial condition is in terms of polar coordinates. We resolve this by noting $\theta = \arctan(y/x)$. Thus, we evaluate $\sin[\arctan(y/x)]$ in place of $\sin\theta$. In practice, this method of evaluation in MATLAB leads to a singularity at $x = 0$ despite the fact that $\arctan(\infty) = \pi/2$. As we will see later, this leads to problems in the numerical implementation of our method.

After defining the initial conditions in $Z1$, we evaluate 14 using the values of $Z1$. We apply the boundary conditions by finding the point in $Z1$ that has the closest x value to (8) and the closest y value to (9). The results are then stored in a second matrix. With every addition time step, we repeat this process, extending the data off the unit circle to calculate boundary conditions.

We present our data in two ways: firstly as a 3-D plot of the heat equation on the computational band, and secondly as a plot of the average relative point-wise errors at each time step. The point-wise error is computed with respect to the analytic solution given by equation (3). By means of these two methods, we analyze our data in the next section. The 3-D plot allows us to heuristically understand the solution, while the average relative point-wise error gives us a sense of how accurate our solution is.

6 Analysis of Results

We will begin this section by presenting a result other than the case $u(\theta, 0) = \sin\theta$. Unfortunately, in the case that we will present, we do not have an analytic solution, hence we will analyze the result heuristically. Despite the fact that our analysis of the figures will be heuristic, it will become important

when it comes time to examine the case $u(\theta, 0) = \sin\theta$ on the unit circle. Section 6.1 will demonstrate that our method produces results that agree with our intuition. Following this, in section 6.2 we will investigate the numerical solution of (4)–(5). We will then compare the behavior of solutions found in section 6.1 to those found in section 6.2.

6.1 Heuristic Results for Heat Diffusion on a Circle

In addition to giving us an intuition for how the heat equation behaves on the unit circle, considering the example that follows will aide us in isolating a problem that arose in the numerical solution of (4)–(5).

During the course of this project, we tested our diffusion method with several different initial functions in addition to $u_2(\theta, 0) = \sin(\theta)$. Most notably, we also considered the initial distribution $u_2(\theta, 0) = |\sin\theta|$ which is shown in figure 3.

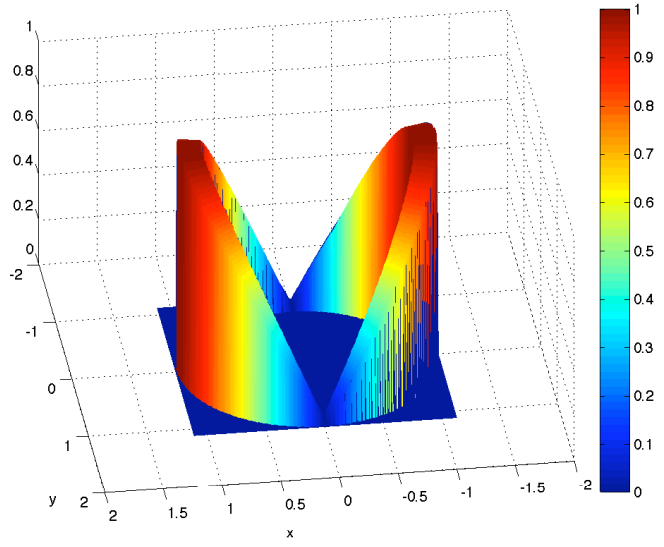

Figure 3: Initial distribution of $u_2(\theta, 0) = |\sin\theta|$

We observe that the numerical solution of $u_2(\theta, 0) = |\sin\theta|$ as presented in figure 4 is well behaved under the given conditions. Indeed, $u_2(\theta, t)$ seems to converge to some value in the range $(.4, .6)$ for large values of t - the

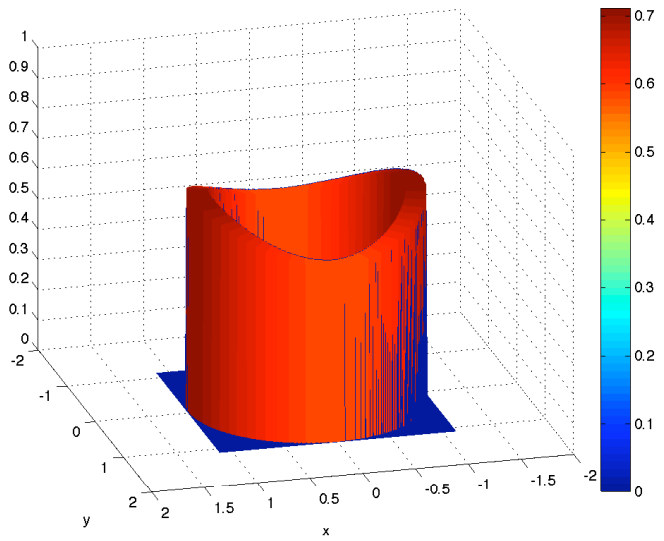

Figure 4: Distribution of heat at $t \approx 1.5$ where $u_2(\theta, 0) = |\sin\theta|$.

picture presented here is $t \approx 1.5$. Since $|\sin\theta|$ is a symmetric and continuous function, we expect that the restriction of the heat equation will cause the heat distribution to converge to some value near the the midpoint of $[min|\sin\theta|, max|\sin\theta|] = [0, 1]$. Also notice that the solution at $t \approx 1.5$ is quite smooth with no irregular dips.

6.2 Numerical Solution of $u(\theta, 0) = \sin\theta$

Now we will turn our attention to analyzing the case $u(\theta, 0) = \sin\theta$. Recall that the initial distribution $u(\theta, 0) = \sin\theta$ is represented by the plot in figure 1.

Using our program, we subjected the above heat distribution to the heat equation by means of the closest point method. Figure 5 shows our numerical solution of $u(\theta, t)$ at $t \approx .1$. Notice that there is a slight dip that forms where the maxima and minima of the function should occur. Figure 7, which is a plot of $u(\theta, t)$ at time $t \approx .3$, demonstrates the dip becomes more pronounced with each time step.

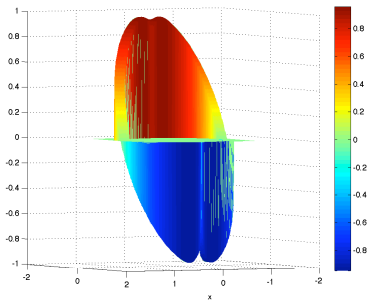

Figure 5: The solution $u(\theta, t)$ at $t \approx .01$.

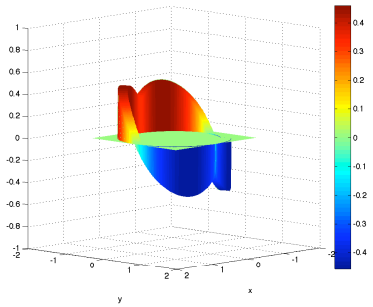

Figure 6: The solution $u(\theta, t)$ at $t \approx .3$.

In order to analyze this problem more carefully, we construct a plot of the average point–wise relative error for each time step up to $t \approx .3$. We do so

by comparing our numerical results to the actual solution $u(\theta, t) = e^{-t} \sin \theta$. This will give us a sense for how far off our solution is on average with each time step. In addition, we will plot the maximum point-wise relative error of each time step. This will tell us the greatest magnitude of our deviation from the exact solution.

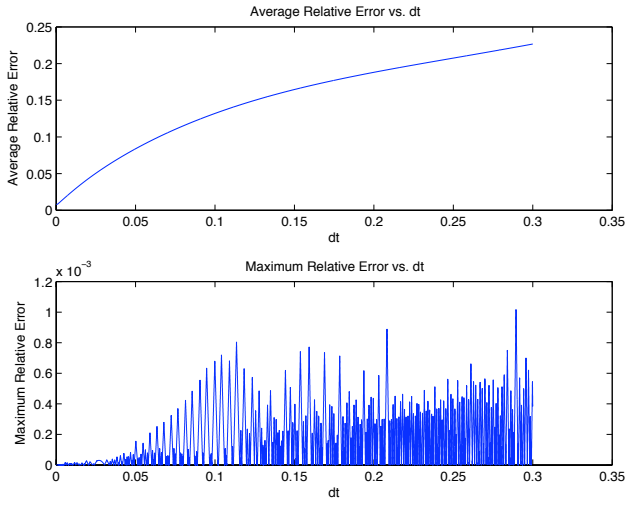

Figure 7: The solution $u(\theta, t)$ at $t \approx .3$.

Notice that the average point-wise error grows quite rapidly, reaching a value of .25 at $t \approx .3$. This tells us that either our method is not accurate, or the dip in the solution is causing a catastrophic error, or perhaps both. Given our visual representation of the solution in figure 7 it is clear that the dip in the solution contributes significantly to the error. The plot of maximum relative error, which we assume to be occurring at $x = 0$, tends to indicate that there is an initial transient period during which there is a large gap between the actual solution and the numerical solution. The maximum relative error tends to level off after the initial transient period, with a few occasional spikes. This indicates that the parts of the solution bordering the gap do not diffuse into the gap over time. This indicates that there may be a problem in the implementation of the algorithm. We showed, however, in the previous section that our algorithm works for the case $u(\theta, 0) = |\sin \theta|$. Which is very similar in nature. Another possibility is that

there is a problem with the initiation of the initial conditions. In particular, the use of arctan[(y/x)] may create a singularity at $x = 0$. This is the region that seems to be effected the most, so this is a plausible explanation.

7 Conclusion

In this paper we explored the closest point method for numerically solving PDEs on various closed surfaces. In particular we were interested in solving:

$$\begin{aligned} u_t &= u_{\theta\theta}, & t \geq 0, 0 \leq \theta \leq 2\pi \\ u(\theta, 0) &= \sin\theta & 0 \leq \theta \leq 2\pi \end{aligned}$$

on the unit circle. We did this by extending the initial data into a computational domain where we in turn solved the problem using a forward time difference method along with a second order centered difference method for both spacial variables. This resulted in an explicit scheme which we solved in MATLAB.

Our approach seemed to work quite well for evolving the initial condition $u_2(\theta, 0) = |\sin\theta|$, but did not work for numerically solving the the problem of interest – evolving $u(\theta, 0) = \sin\theta$. We came to the conclusion that this was either because of an error setting the initial condition - possibly due to problems with arctan(y/x) at $x = 0$, or due to a problem in the implementation of the method. It should be noted, however, that our implementation worked well for the $u_2(\theta, 0)$.

7.1 Future Work

For future work, we would like to pinpoint the error that caused the dip in our numerical simulations. In addition, we would like to investigate using the closest point method to solve the heat/diffusion equation on domains other than the circle. It is, for example, feasible to solve the diffusion equation on a surface such as torus.

7.2 Acknowledgements

I would like to thank Professor Steven Ruuth at Simon Fraser University for several interesting conversations about applications of the closest point

method. In addition, I would like to thank Professor Yong for giving me an extension on this paper.

References

[1] M. Bertalmio, L.T. Cheng, S. Osher, and G. Sapiro. Variational problems and partial differential equations on implicit surfaces. J. Comput. Phys., 174(2):759-780, 2001.

[2] M. Burger. *Finite element approximation of elliptic partial differential equations on implicit surfaces*, UCLA CAM preprint, 05-46.

[3] M.P. doCarmo. *Differential Geometry of Curves and Surfaces*. Prentice Hall Inc., Englewood Cliffs, NJ, 1976. Translated from the Portuguese.

[4] J.B. Greer. *An improvement of recent Eulerian method for solving PDEs on general geometries*, UCLA CAM preprint, 05-41.

[5] J.B. Greer, A.L. Bertozzi, and G. Sapiro. Fourth order partial differential equations on general geometries. UCLA CAM preprint, 05-17.

[6] J. David Logan. Applied Partial Differential Equations. Springer-Verlag, second edition, 2004.

[7] C.B. Macdonald and S. Ruuth. *Level set equations on surfaces via the closest point method*, preprint.

[8] F. Memoli and G. Sapiro. Fast computation of weighted

[9] S. Osher and R. Fedkiw. *Level set methods and dynamic implicit surfaces*, volume 153 of *Applied Mathematical Sciences*. Springer-Verlag, New York, 2003.

[10] D. Peng, B. Merriman, S. Osher, H. Zhao, and M. Kang. A PDE-based fast local level set method. J. Comput. Phys., 155(2):410-438, 1999.

[11] S. Ruuth and Merriman. *A simple embedding method for solving partial differential equations on implicit surfaces*, UCLA CAM preprint, 06-54.

[12] J.A. Sethian. *Level set methods and fast marching method. Evolving interfaces in computational geometry, fluid mechanics, computer vision, and material sciences*, volume 3 of *Cambridge Monographs on Applied and Computational Mathematics*. Cambridge University Press, Cambridge, second edition, 1999.