A Study of Two Implicit Data Interpolation Schemes

University of Waterloo Technical Report CS-2013-09

Stephen Mann

September 25, 2013

Abstract

Implicit surfaces are one technique for surface modeling in computer graphics. The are several implicit surface schemes to interpolate point data. In this report, I study two schemes, one by Turk and O'Brien, the other by Shen, and test how well they work. Further, although Shen derives a linear version of his scheme, he only implemented a constant version. In this report, I implemented and tested the constant and linear variations of Shen's scheme.

1 Introduction

In Spring 2012, a seminar course in the David R. Cheriton School of Computer Science at the University of Waterloo studied implicit surfaces. In a separate report [1], we summarized our findings. However, in running the seminar, I looked closely at two schemes, one by Turk and O'Brien [12, 13] and one by Shen [9]. In this report I summarize my findings.[1]

For this work, I implemented 2D versions of their methods in Octave, and created a variety of data sets to test their method. Further, a few variations on both schemes were tested. In particular, I tested both methods using "pseudo-normals" and using "true-normals".[2] I also ran some tests using interior constraints with Turk-O'Brien's scheme, but these constraints appear to be too difficult to be practical, so for the most part I used the pseudo-normal constraints to test the Turk-O'Brien scheme. Also, while deriving his method, Shen's derivation used linear polynomials; however, the method he implemented only used constant polynomials. In this report, I give the derivation of the constant scheme, and implemented both constant and linear versions of Shen's method.

I first give a set of tests that I ran on the Turk-O'Brien scheme. I used some obvious data sets (such as co-circular points), as well as devising a data set to test locality. For larger data sets, I used the Koch snowflake, which, while convenient, suffers from two problems: first, the Koch snowflake data sets have a lot of symmetry in them, and second, the geometric variation in the data is fairly high (i.e., no three consecutive points are close to co-linear). I also hand digitized a point set from Shen's thesis, which I used to test both methods.

In the tests, I was looking for several things including the following:

- Are there any shape artifacts in the curves?
- Is the curve a single, connected component?
- Does the curve follow the connectivity of the polyline defining the data?
- Are there extra sheets that do not interpolate the data?

While the tests focus on the interpolating variations of these schemes, a few tests investigate the approximating variation of Shen's scheme. The initial tests were a check of various parameters in the Turk-O'Brien scheme; only one test was devised to check if the Turk-O'Brien scheme would **always** produce a single, connected component (it does not, but the point of the data set was to demonstrate that their method could produce disconnected components), in several of the other tests for both schemes, extra sheets appeared and/or the curve broke into multiple, disconnected components.

Most of the evaluation of the curves is just a subjective, visual check of the shape. This is mostly sufficient in checking the last three questions above (although potentially there are disconnected components far from the data that were not detected). For one data set, one of the curves constructed with Turk-O'Brien's scheme appeared to have flat spots, so I made curvature

[1]Note that Shen developed two methods in his dissertation: one to interpolate or approximate a point set, the other to interpolate or approximate a triangle mesh. In this report, I am studying Shen's method for interpolating point sets.

[2]Turk and O'Brien proposed some normal constraints; later Shen, calling the Turk-O'Brien constraints "pseudo-normal" constraints, proposed a different normal constraint that he refers to as "true-normal" constraints. While it is clear that neither scheme interpolates the given normals using the pseudo-normal constraints, it is not clear whether or not either scheme interpolates the normals when using the true-normal constraints: while both schemes appear to interpolate the normals, Shen did not prove that the normals would be interpolated in his dissertation.

plots to try to decide whether or not it did have flat spots (it did not). However, none of the other curves needed such additional tests: for example, the shape artifacts that occurred with Shen's scheme were severe enough as to require no other tests.

A summary of my results is that while Turk-O'Brien's scheme may have an issue with the generation of extra sheets, overall the curves it produces seem reasonable when using pseudo-normals, but the interior/exterior constraints of Turk-O'Brien were too difficult to use in practice, and I had mixed results with the Turk-O'Brien scheme using true-normals: for some data sets, the resulting curves were far better than the ones produced using true-normals, but for other data sets, the curve broke into disconnected components when using true-normals (where pseudo-normals on the same data sets gave reasonable results).

The results for the variations of Shen's scheme were disappointing: the constant variation of his scheme using either pseudo-normals or true-normals gave curves with gross curvature concentration. And while the linear variation of his scheme appeared to give much better shaped curves when looking at one scale, magnifying the curves by a factor of 100 or 1000 revealed serious shape artifacts in the regions near the data points. These small scale artifacts appear to be a result of Shen's choice of radial basis function. Using Shen's scheme to approximate data gave better results (it uses a different radial basis function), but two unintuitive shape parameters need to be set to get a reasonable approximation.

The schemes were implemented in Octave [4][3], and a version of Marching Squares was used to generate the figures. The density of the squares was chosen to accurately represent the curve, with an occasional Marching Squares artifact pointed out in the text of the paper. The Marching Squares grid size varied from 50×50 to 200×200. The sampling region is at times larger than the region shown in the figure due to the way that Octave draws the figures.

In Section 2, I show the results of my tests on the Turk-O'Brien scheme. In the much longer Section 3, I give the results of my tests on the four variations of Shen's schemes, as well as the development of some details of the schemes that did not appear in Shen's dissertation.

2 Turk-O'Brien

Turk and O'Brien developed a method for fitting an implicit surface to a set of points and normals [12, 13]. Given a set of k points $\mathbf{x_i}$, they wish to find an implicit function f such that $f(\mathbf{x_i}) = v_i$. Their method is based on the radial basis function $w(\vec{v}) = ||\vec{v}||^3$, with their interpolant written as

$$f(\mathbf{x}) = \sum_{j=1}^{k} w_j r(\mathbf{x} - \mathbf{x_i}) + P(\mathbf{x}), \tag{1}$$

where P is a linear polynomial. The equations $f(\mathbf{x}_i) = v_i$ together with constraints on P form a system of $k+d+1$ equations in $k+d+1$ unknowns (the w_j plus the coefficients of P), where d is the dimension of our space. If all the $v_i = 0$, then the system has the trivial solution of all coefficients being 0, so at least one non-zero constraint is required.

Turk and O'Brien mention two types of constraints: interior and exterior constraints (with a third type, normal constraints, that is a variation on the first two). An interior constraint is just a point on the "inside" of the implicit surface that has non-zero value, while an exterior constraint is a point on the exterior of the implicit surface that has non-zero value (but of opposite sign to the interior constraints). Normal constraints are points that lie close to one of the $\mathbf{x}_i$ to interpolate in the direction of a normal associated with $\mathbf{x}_i$ and are given a non-zero v_i. Turk-O'Brien seem to set all the normal constraint $v_i = 1$.

In Section 2.1, I give the results of the initial tests I ran on their scheme using interior/exterior constraints. In particular, in addition to gaining a feel for using the constraints, I also tried changing some of the points to see how local the changes were, and I tested how far apart the points could be and still have a single connected component. In Section 2.2, I used the Koch snowflake to generate larger data sets, which still using interior/exterior constraints. The results of the tests in these two sections are that the scheme works okay, but it is difficult to set the interior/exterior constraints.

In Section 2.3, I tried using the Turk-O'Brien pseudo-normal constraints on the Koch snowflake data sets. The results here were better: it is easy to use the pseudo-normal constraints and the resulting curves were somewhat better in quality than the ones I generated with interior/exterior constraints. In all tests, the only significant issue that arose with the Turk-O'Brien scheme was the appearance of unwanted disconnected components.

2.1 Initial Tests

As a first test, three points were interpolated with a fourth point to have a value 1 (Figure 1, upper left). The resulting curve appears to be unbounded, suggesting (quite reasonably) that interior point should be "surrounded" by the other points. Fitting a curve to four points at the corners of a unit square, with a fifth point at the centroid to be interpolated with value 1 gives the result shown in Figure 1, top row, center. Translating the points and refitting appears to give the same curve (Figure 1, top right), supporting the notion that the method is translation invariant.

[3]http://www.octave.org

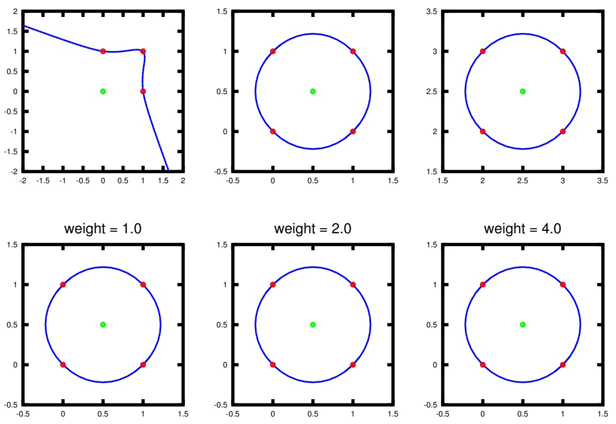


Figure 1: Some simple tests of the Turk-O'Brien scheme using interior constraints. $v = 0$ at red points; $v = 1$ at green points unless otherwise noted.

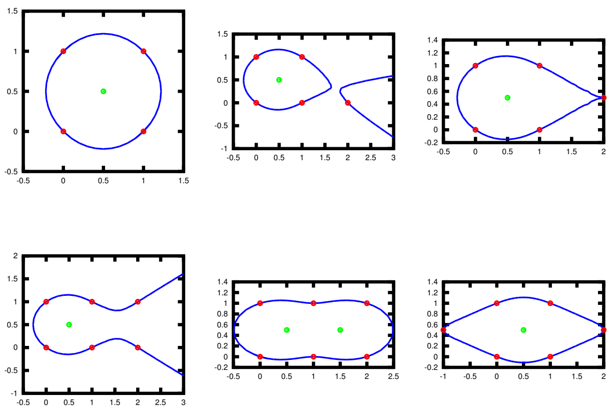


Figure 2: Tests of the Turk-O'Brien scheme using interior constraints. $v = 0$ at red points; $v = 1$ at green points.

On the bottom row of Figure 1, the value at the center point (called "weight" in the figure) was set to different values. This appears to have no affect on the resulting curve, which should be expected (i.e., scaling an implicit formula by a constant factor does not change the zero set).

In Figure 2, a few more points were added to the initial four points in the previous example. Figure 2, upper left, shows the initial four points interpolated, with the value of the implicit function forced to 1 at the center of the square. In Figure 2, top row, center, one additional point has been added, resulting in a curve with at least two disconnected components—a fairly bad result. This data set is discussed further in later in this section; see Figure 6.

In Figure 2, upper right, the fifth point has been moved up. The result is a tear drop like curve. A few notes: First, the bumps near the right side of the curve are Marching Cubes artifacts; with a denser Marching Cubes sampling, these bumps disappear. Second, the curve is C^∞ at the right where in the figure it only appears C^0. And third, the Marching Cubes grid used extended to 3 in x, but Octave only displayed to $x = 2$; i.e., the curve does not continue to the right of what is shown in the diagram.

In Figure 2, bottom right, a sixth point has been added, symmetric to the one that was added in Figure 2, upper right. As can be seen in the figure, both ends of the curve in this six point example are smoother than the right end of the curve in the five point example.

In Figure 2, bottom left and center, a total of six points to be interpolated were placed on two grid cells. On the bottom left, there is only one interior point, and the result appears to be unbounded. In the bottom center, there were two interior points (both with weight 1), resulting in a closed curve. This suggests that if we are to construct curves with this method using only interior weights, then the polyline of points needs to be triangulated with an interior point place within each triangle with an appropriate weight to be interpolated (see, however, Section 2.3, where normal constraints are used, which appears to be a much better approach).

It is unclear how local the Turk-O'Brien method is. I.e., the blending functions for a point's weight increases as the cube of the distance. For a large data set, if we move one point, we would like the curve/surface to change only near the changed point, and not have the entire curve/surface wiggle. As a test of this, 20 points were sampled on a sine curve, then replicated and offset in y by 1. This gave 40 points to be interpolated. Nine points were located at the centers of the obvious rectangles and given a weight of 1. This data was then interpolated with the Turk-O'Brien method, giving the curve shown at the top of Figure 3. One point was then moved and the data reinterpolated as shown on the second row of Figure 3. The superposition of the two curves is shown on the third row of Figure 3, where we see that the change appears to be local. The bottom row shows more modifications on one end of the curve, with the result superimposed with the original curve, and again we see that the changes appear to be local.

As a further test of local control, I moved two of the end data points at one end of the points in Figure 3, top, to give non-uniform spacings of the data points, and made the same modifications that were made in the other parts of Figure 3. The test curves are seen in Figure 4, with Figure 4 top being the same as Figure 3. While there was some noticeable non-local effects in these examples, they were fairly minor (i.e., when the two curves were overlayed, while a separation between the curves was visible in the sparse portion of the data, the separation was no more than 1 pixel on a 1680×1050 display).

Of a bit more interest is that Figure 4 shows that large gaps in the data set can lead to disconnected components. This effect is expected, and at least in this example, the gap that created disconnected components was unrealistically large (i.e., the Turk-O'Brien method seems to handle moderately large gaps fairly well).

Jin et al. [6] note that the linear term P in (1) is needed for affine invariance; Turk and O'Brien only claim that it covers

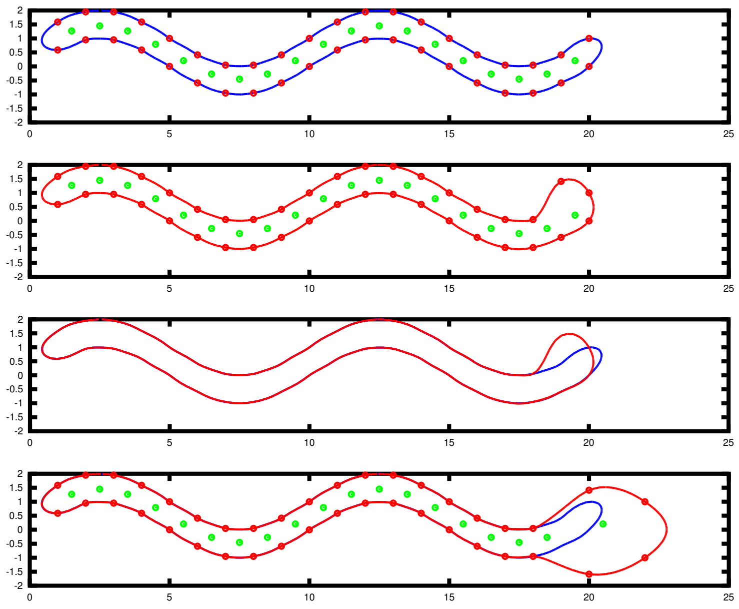


Figure 3: Turk-O'Brien with interior constraints: testing local control.

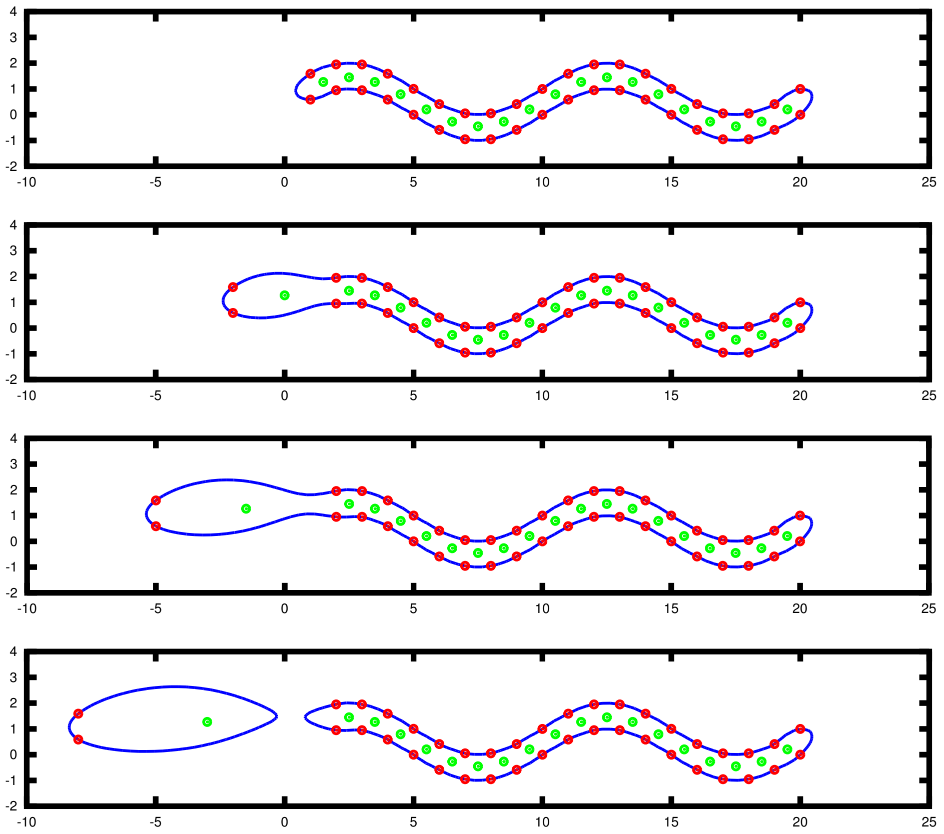


Figure 4: Turk-O'Brien with interior constraints: non-uniform spacing of points.

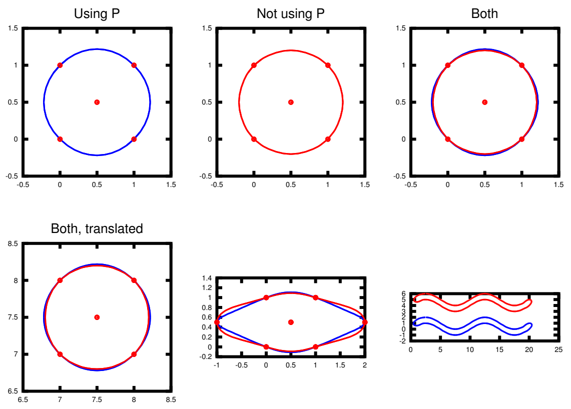


Figure 5: Turk-O'Brien with interior constraints: the effects of P.

the linear terms that are missing in the radial basis function used, with Dinh et al. [3] noting that P spans the null space of ϕ. Also, it can be shown that P is required to guarantee that the linear system is solvable [14]. As a simple test of the affect of P, I interpolated four points on a square as well as tested two other data sets. Figure 5 shows the results of these tests. On the data set consisting of four points on the corners of a square (to be interpolated) and a fifth point in the center of the square (to have value 1), we see that the affect of P is minimal, even when translating. However, Figure 5, bottom center, shows that the effect is larger on other data sets. In Figure 5, bottom right, we have interpolated the data in Figure 3, top, both with (blue) and without (red) P (the resulting curve constructed without P has been translated up so that both are visible in the figure). The affects in this case are small. In a larger image, it is possible to see that both ends of the two curves vary slightly.

Returning to the five point example of Figure 2, various adjustments were made to try to obtain a reasonable curve for the points to be interpolated. The results of these experiments appear in Figure 6. The the upper left is the original curve constructed for these five points, with an interior point (shown in green) having weight 1. As a first attempt to improve the interpolatory curve, a second interior point was added at the center of the triangle consisting of the right three points to be interpolated. This point was given weights 1/2, 1/4, and 1 (with the other interior point having its weight fixed at 1), as shown in the top row of Figure 6. None of the results were satisfactory.

In Figure 6 second row, left, the second interior point was moved down and right (compared to Figure 6, top row, second from left), giving a slight improvement in the curve. Returning this second interior point to its original position, an exterior point was added (Figure 6, center row, second and third from the left). Both of these are much closer to what one would expect when interpolating the red points.

Rather than have to set exterior points manually, we would prefer to have something more automatic. Turk and O'Brien also used normals to specify the interior. Four tests of using normals on this five point data set are shown on the bottom row of Figure 6. Adjusting weights and directions (the directions of the normals are indicated by lines through the points to be interpolated) failed to yield satisfactory results.

2.2 Koch Snowflake

For larger data sets, I used the Koch snowflake (Figure 7) [8]. The initial triangle will be referred to as K_0, the first refinement as K_1 and so on. Two methods for using Turk-O'Brien to fit curves to this data were tried, one using interior points and weights, the other using normals.

Figure 8 shows initial attempts to fit surfaces to the Koch snowflake data using interior points. On the top left, we see the curve fit to K_0. On the top center is the curve fit to K_1 with a single interior point having weight 1. As can be seen in the figure, the result is two separate curves that interpolate the data. Clearly, additional interior points/weights are needed. The obvious choice to interior points is to place one additional interior point at each triangle added since the previous level. The question is: what weights to use. Figure 8, top row right, shows the result of setting all the weights to 1. While the resulting curve appears to not interpolate all the data points, using a denser Marching Cubes grid shows a small loop passing through each of the interior red points that appear to not be interpolated.

Reducing the weights on the second layer of interior points to 1/2 gives a much more reasonable curve (Figure 8, bottom row, left); using the weight of 1/3 gives the curve shown in Figure 8, bottom row, center, with 1/3 being used since the edge length of the triangles used in K_1 is 1/3 that of the length of those used in K_0. If we reduce the weights to 1/9 (the ratio of the areas of the triangles used in successive levels), we get the curve appearing in Figure 8, bottom right. However, using weights of 1/9 for these points results in an unwanted second sheet, as illustrated in Figure 9. The upper right of this figure show the

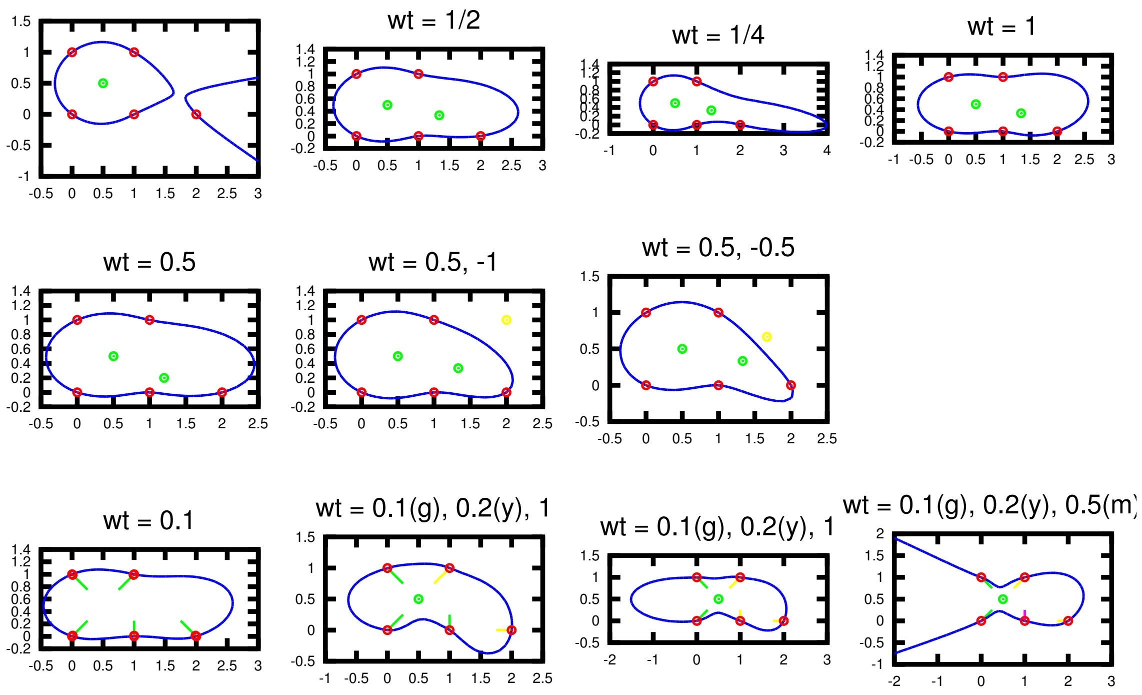


Figure 6: Turk-O'Brien with interior/exterior constraints: a five point example.

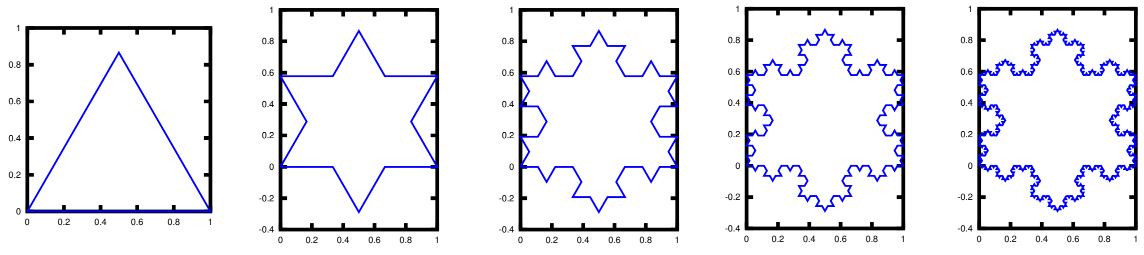


Figure 7: Koch Snowflake

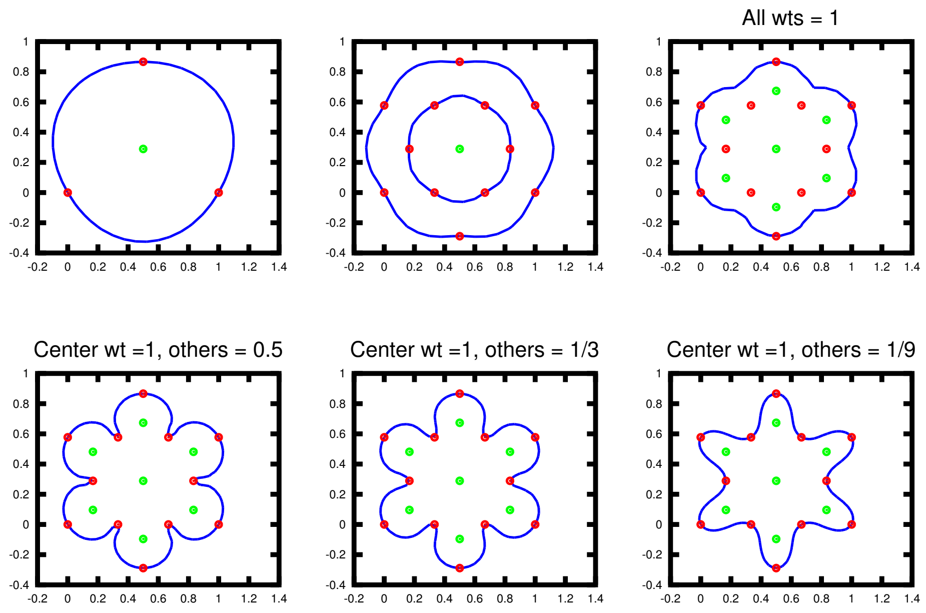


Figure 8: Turk-O'Brien with interior constraints: the Koch snowflake.

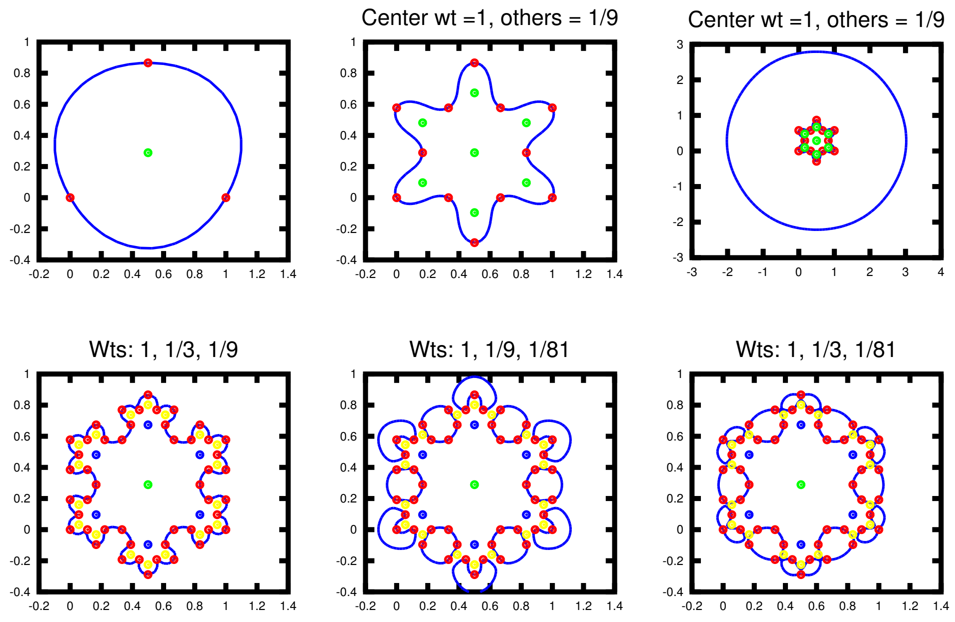


Figure 9: Turk-O'Brien with interior constraints: various weights on the Koch snowflake.

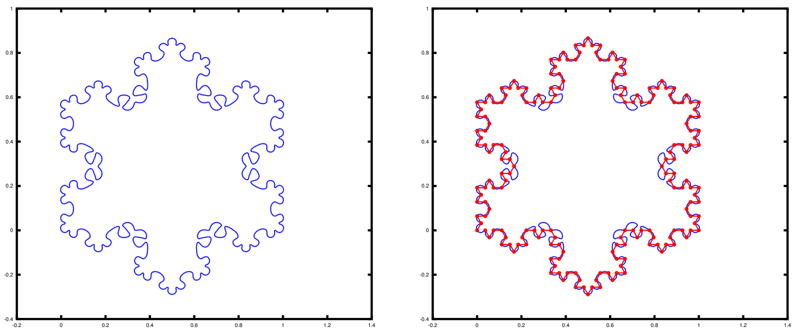


Figure 10: Turk-O'Brien with interior constraints on K_3. Weights: 1, 1/3, 1/9, 1/27

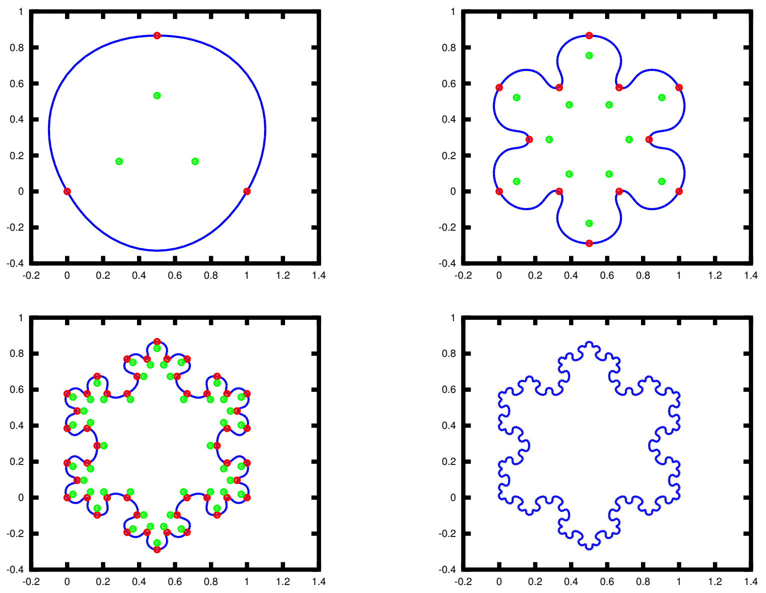


Figure 11: Turk-O'Brien on K_0 to K_3 using pseudo-normal constraints.

curve fit to K_0; the top center show the curve fit to K_1 with weights 1/9. Expanding the Marching Cube grid on the curve fit to K_1 with weights 1/9 (Figure 9, top right), shows this unwanted second sheet.

For K_2, using weights of 1, 1/3, and 1/9 for the centers of the successive layers of triangles, Turk-O'Brien's method gives the curve in Figure 9, bottom left. Using weights of 1, 1/9, 1/81 or 1, 1/3, 1/81, however, gives multiple disconnected components (Figure 9, bottom center, right). Clearly, good settings of the interior weights is critical for this method.

Applying this method to K_3 with weights 1, 1/3, 1/9, 1/27, we get the curve shown in Figure 10, left. While this is a reasonable curve, superimposing the Koch snowflake (Figure 10, right) shows that there are unwanted lobes in the curve.

2.3 Using normal constraints

Instead of using interior points and weights, we could instead specify normals to the curve at each point to interpolate and use the normal constraints proposed by Turk and O'Brien (and called pseudo-normal constraints by Shen). Using the bisector of the edges on either side of each point in the Koch snowflake for the normal directions, and fitting curves to K_0, K_1, K_2 and K_3, we get the curves shown in Figure 11 (the distance of the normal point to the corresponding point of interpolating was set to 1/3 the edge length of K_i).

Applying the same method to K_4 also gives a reasonable curve (Figure 12). In this example, the matrix used by Turk-O'Brien was 1538×1538 in size and took 143 seconds to build and invert in Octave on a 2.2GHz Intel Core 2 Duo CPU with 2GB of RAM.

In Figure 13, left, the points on K_2 to interpolate are in red, and the normals are in green. However, looking at the figure,

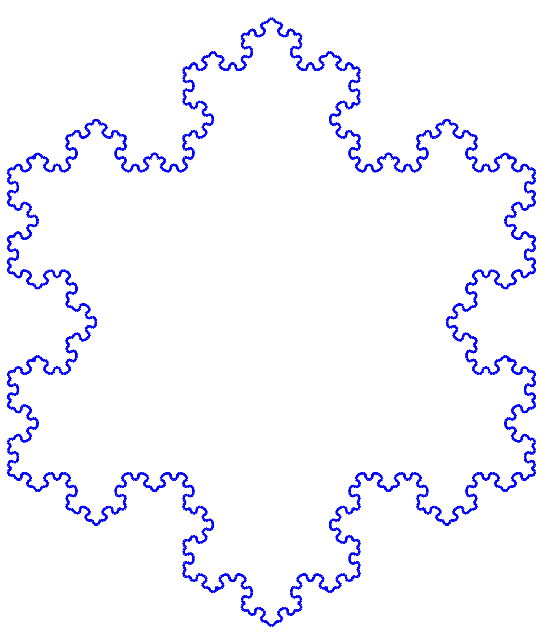


Figure 12: Turk-O'Brien on K_4 using pseudo-normal constraints.

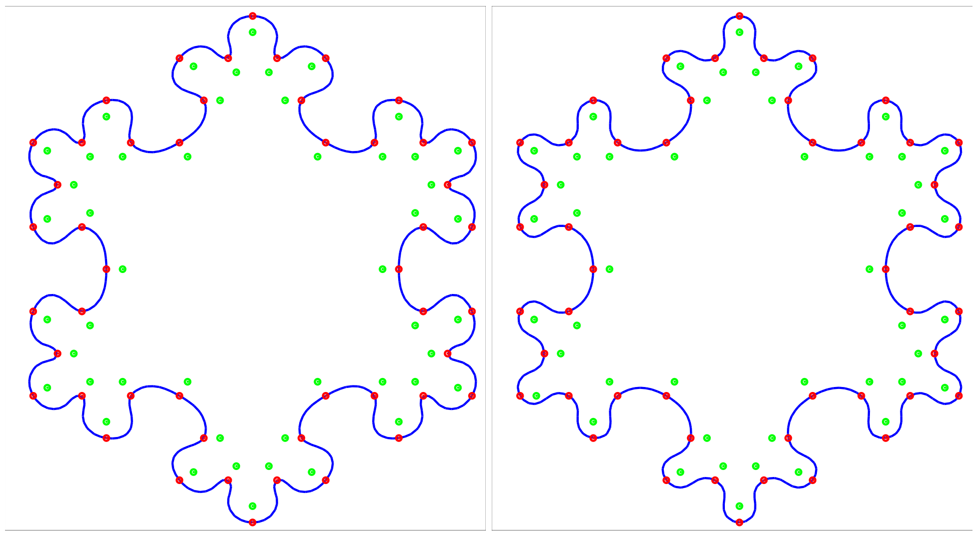


Figure 13: Turk-O'Brien on K_2 using pseudo-normal constraints. Left: normals at green points; right, normals at .01 from red points.

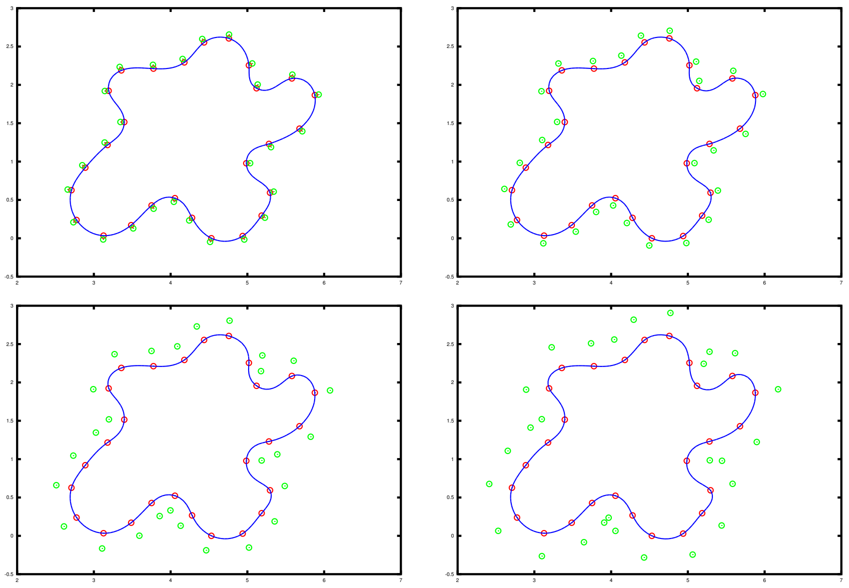


Figure 14: Test of effect of varying the distance of pseudo-normal points on Turk-O'Brien's method.

we see that some of the "normals" are not perpendicular to the curve. I.e., the "normals" appear to be more hints to the system rather than a strong constraint, as noted previously by Shen et al. [10].

In their 2002 paper, Turk and O'Brien recommend placing the normals a distance of 0.01 away from the points, assuming that the object is of about unit size. Rescaling the normals and refitting a curve to the K_2 data gives us the curve that appears in Figure 13, right. We note two things: first, while the normals still are not perpendicular to the curve, they are closer to being perpendicular. And second, the curve (to my eyes) appears to have better shape than the one in Figure 13, left. Unfortunately, when constructing a curve for K_4 data with normals a distance of 0.01 away from the interpolation points, an unwanted sheet surrounding the desired curve appears.

As a test of the effect of the distance of the pseudo-normals from the corresponding interpolation point, I used a data set of Shen et al. [10] (this dataset is also used for a similar test of Shen's scheme in Section 3). The results appear in Figure 14. While there is some change based on the distance of the pseudo-normal points, the effect seems reasonable. Placing a pair of pseudo-normal points on either side of each interpolation point (with opposite sign) had only a minimal effect on the curve, although the effect was larger for pseudo-normal points that were a larger distance away from the interpolation points.

To test a combination of the center weight on a point set with normals, the points to interpolate were located at the corners of a square, with normals perpendicular to a circle passing through these points, and where the interior point was chosen as the center of the square. The weight on the interior point was increased from 1, 10, 100, 500, 1000. The results are shown in Figure 15. While the interior weight seems to have only a small effect on the curve through the four points, for some value of the weight between 100 and 500, a second sheet appears (possibly the second sheet appears for a smaller value; however, a marching cubes grid ranging over $[-2000, 2000]$ was used for the weight of 100 without seeing a second sheet).

As a final test, a dataset from Shen et al. [10] was used. Shen et al. note that their method has "oscillatory behaviour away from the constraints" when using pseudo-normal constraints. I hand digitized this data set in Xfig, giving coordinates on the order of magnitude of 1000. When fitting a Turk-O'Brien curve to this data, Octave gave the following warning:

```
warning: inverse: matrix singular to machine precision, rcond = 1.43097e-23
```

After normalizing the coordinates by dividing by 1000, the condition number of the matrix became `rcond = 9.1460e-06`.

The results of several tests on this Shen et al. dataset appear in Figure 16. The difference between the first two columns is that in the leftmost column uses the points whose coordinates are in the 1000's, while the middle column uses points with normalized coordinates. Despite the poor condition number of the matrices used in the first column, the solution is a scaled/translated version of the one in the second column.[4]

The first two images in the first row show the results of placing a single point in the interior—clearly, this single interior point is insufficient to reproduce the curve. The third image on the first row has five more manually placed points, with the initial center point getting weight 1 and the five new points getting weight 0.5.

[4]The poor condition number is likely due to all the data being scaled; regardless of the scale, the matrix entries have full floating point precision, and since all are of similar magnitude, the scale has little effect. Scaling the data Figure 16, top left, up by a factor of 10^{12}, giving an `rcond` value of 10^{-95}, the shape of the solution curve was indistinguishable from that of the original. Conclusion: don't put blind faith in condition numbers.

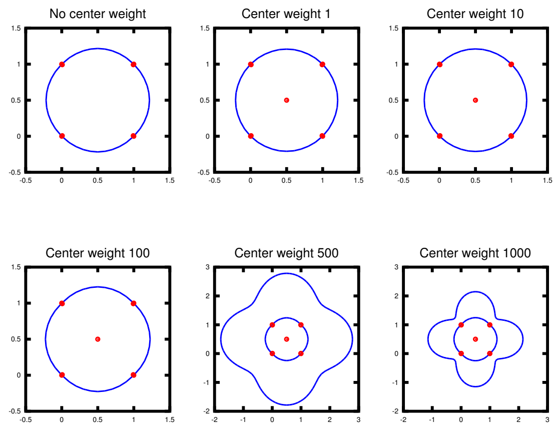


Figure 15: Turk-O'Brien: Testing center weight with normals.

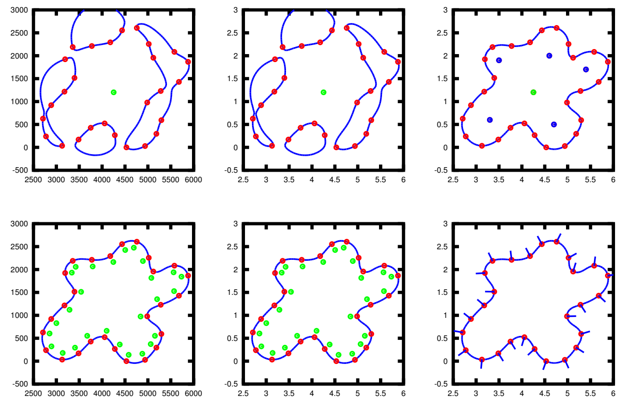


Figure 16: Testing Turk-O'Brien on Shen et al. [10] dataset. Top row: interior constraints. Bottom row: pseudo-normal constraints.

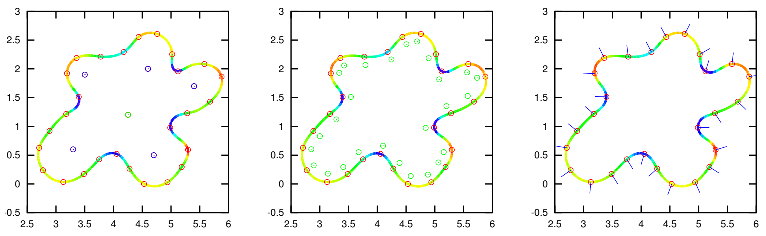


Figure 17: Curvature plots Turk-O'Brien's scheme with pseudo-normals on Shen data set. Color to curvature correspondence: red: $+5$; green: 0; blue: -5. Max/min curvatures: left $-10.9/5.2$; middle $-7.9/4.0$; right $-6.8/4.6$.

In the second row, the first two images show using pseudo-normal constraints at each point;[5] the third image moves the normal constraints to a distance of 0.01 from the corresponding point on the curve (with the normals in this picture have been drawn as line segments). Again we see that the curve is not perpendicular to the normal constraints (suggesting that Shen et al.s' term "pseudo-normal constraints" is more appropriate). More important: in the bottom row of images, we do not see the ringing artifacts that occur with Shen et al.'s method with this type of normal constraint.

Figure 17 shows curvature plots of the three curves created for the Shen et al. dataset.[6] The left curve in this figure is the curvature plot for the upper right curve of Figure 16; the middle one matches the bottom middle of Figure 16; and the right curve corresponds to the bottom right of Figure 16. In these signed curvature figures, green corresponds to 0 curvature, with red and blue corresponding the positive and negative curvatures respectively. From the middle and right plots of this figure, we see that the distance of the normal constraint points has some influence on the curvature of the curve, although it is difficult to determine any precise effects from the figure.

2.4 Conclusions and Discussion

The Turk-O'Brien method has several nice properties: it is easy to implement, it has affine invariance, and even though it is a global scheme, it appears to behave reasonably locally. However, there are several concerns about the scheme. First, when using interior/exterior points it is unclear how to set the weights, which (from the examples) clearly have a large impact on the shape of the curves. Second, the pseudo-normals are not interpolated, but only coarse approximated. This may or may not be a problem depending on the application. Regardless, pseudo-normals do seem to do a reasonable job in specifying the inside of the surface. And third, the most serious problem is the appearance of unwanted sheets. It is unclear how common they are. While they appeared in four examples we tested, all four examples had a fair amount of symmetry. However, if these extra

[5]The normal at point c_i was set to be perpendicular to the line segment $\overline{c_{i-1}c_{i+1}}$.

[6]The curvature was approximated at a point P on F by sampling at a distance 0.001 the points above, below, to the left, and to the right of P. Two more zeros on F were found on this square of four points, and the curvature was estimated using the circle passing through these three points on F

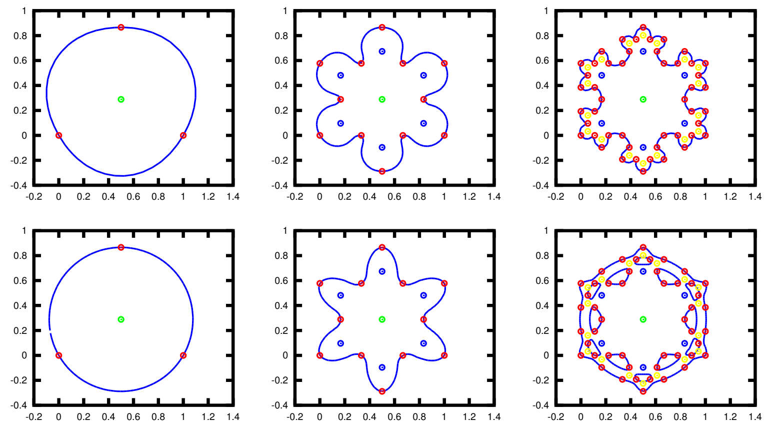


Figure 18: Interpolating Koch snowflake data. The red points are to be interpolated (i.e., the implicit should take value 0); at the green point, the implicit is 1; at blue, 1/3; and at yellow, 1/9. Top row: Turk-O'Brien's method. Bottom row: The Least.

sheets are at all common, then they likely make the scheme unusable. A fourth concern is that their method requires inverting a system of linear equations whose size is proportional the the amount of data to interpolate. While this currently limits their method to a few thousand points (or for interactive use, a few hundred points), even with a few thousand points one should considering approximating the data rather than interpolating it. Of these four concerns, the extra sheets is the only serious one.

Turk and O'Brien claim their method is intended for surfaces. While the examples shown here are for curves, we expect all the concerns we have to carry over to surfaces, which are typically much harder than curves.

In the bigger picture, it is less clear how to incorporate Turk-O'Brien's work into general modeling systems. While their method can be used to model freeform objects, hierarchical modeling such as found in the BlobTree [15] is used because precise objects such as cylinders are often wanted, and sharp edges are a common necessity. While the Turk-O'Brien method could possibly be used in the BlobTree, the elements in the BlobTree have finite support, unlike the infinite support of the Turk-O'Brien surfaces. Likewise, the C^∞ surfaces of Turk-O'Brien likely would have limited use in a Computer-Aided Design (CAD) system since flat faces, precision, and sharp edges are critical for CAD design. Further, engineering design relies on cylinders and similar objects because they are useful (and machinable) real world components.

2.5 The Least

The polynomial Least of de Boor and Ron [2] could potentially provide polynomials to use in place of radial basis functions for constructing an interpolatory implicit function. If it works well, there would be several advantages of using it over radial basis functions:

- The interpolant would be an algebraic function; this means that root finding using Sturm sequences, etc., could be used.
- A variation on the Least [5] constructs Newton and Lagrange bases to interpolate the data. This would result in fewer basis functions to evaluate, since the basis functions associated with the data points would be given weight 0. This would also reduce the size of the matrix to invert.
- As a minor advantage, if the data to interpolate lies on a circle, the Least will reproduce this circle.

Figure 18 shows some preliminary tests comparing the results of Turk-O'Brien's method to using the Least on three Koch snowflake data sets. On the first two data sets, the Least produces surface similar to Turk-O'Brien; on the third data set, the Least result has multiple connected components and is far inferior to that of Turk-O'Brien.

Clearly, if the Least is to be used to construct interpolatory algebraic surfaces, appropriate constraints are required to reproduce the desired topology. Even if this is can be done, using radial basis functions might still be a better choice than the Least for several reasons. First, the degree of the interpolatory surface constructed with the Least increases as the cube root of the amount of data to interpolate. And second, the Least construction is more complicated than using the $|x - x_i|^3$ radial basis functions of Turk-O'Brien.

3 Shen

Shen [9] developed multiple data fitting methods in his Ph.D. thesis. For this project, we will look at the curve variant of the method he describes in chapters 3 and 4 of his thesis.

The idea of Shen is to use Moving Least Squares to compute an implicit that interpolates a set of points. In particular, for a set of points $\mathbf{x}_i$ and values v_i, we use an error function

$$R(\mathbf{x}) = \sum_{i=1}^{N} w(||\mathbf{x} - \mathbf{x}_i||)[v_i - f(\mathbf{x_i})]^2, \tag{2}$$

where $f(\mathbf{x_i})$ is a polynomials, where in the case of interpolation we want $f(\mathbf{x}_i) = v_i$, and where $w(r)$ is a weight function. Shen uses

$$w(r) = \frac{1}{r^2 + \epsilon^2},$$

where ϵ is a user defined parameter. When $\epsilon = 0$, the resulting implicit interpolates the data, and when $\epsilon \neq 0$, the implicit approximates the data.

Although Shen derives formulas for f of degree one, in practice he uses f of degree zero. For f of degree zero, we have

$$R(\mathbf{x}) = \sum_{i=1}^{N} w(||\mathbf{x} - \mathbf{x}_i||)[v_i - c_0]^2,$$

since $f(\mathbf{x}) = c_0$. Taking the derivative of R with respect to c_0 and setting this equal to 0 gives

$$c_0 = \frac{\sum w(\mathbf{x} - \mathbf{x}_i)v_i}{\sum w(\mathbf{x} - \mathbf{x}_i)}. \tag{3}$$

Thus, in Shen's Moving Least Squares formulation, the coefficients of f are themselves functions. Further, for constant f, we merely get a radial basis function weighting of the data, which is Sheppard's method [11] (using constant basis functions with moving least squares being Sheppard's method has been noted elsewhere [14]).

I implemented two versions of Shen's scheme, one using constant f and one using a linear f. I tested each version of f with both the pseudo-normals of Turk-O'Brien and with the true-normals of Shen, giving four variations on Shen's scheme. The primary data set I used was one from Shen's dissertation, but I also give some tests using the Koch snowflake data sets. Most of the tests were for the interpolating version of Shen's scheme, although I also give a few tests of the approximating scheme.

In Section 3.1, I give the results of testing Shen's scheme with constant f and pseudo-normals. In Section 3.2, I give the results of testing Shen's scheme with linear f and true-normals. In Section 3.3, I tested the Shen's scheme with linear f and pseudo-normals, and in Section 3.4 I test the scheme with linear f using true-normals.

In Section 3.5, I give a comparison of the four Shen variations. Then in Sections 3.6 and 3.7, I explore a few serious problems with the curves created by his schemes. In Appendix A, I show the results of running Shen's schemes on the Koch snowflake.

In all cases, the results were mostly unsatisfying: in addition to creating curves with poor shapes at a large scale, there were often severe, small scale artifacts.

3.1 Pseudo-normals

In implementing Shen's scheme, there is the usual problem of needing non-zero constraints to avoid the trivial solution. Shen tries the "normal constraints" of Turk-O'Brien (which Shen calls *pseudo-normal constraints*) and a second normal scheme discussed later. Unlike Turk-O'Brien, Shen's scheme appears to need both positive and negative normal constraints (in the few test cases I ran with only positive normal constraints, the resulting implicit was non-negative everywhere, with value 0 at the $\mathbf{x}_i$).

Running this method on the hand digitized Shen data set, we get the results seen in Figure 19. Note the sensitivity of the interpolatory curve to the distance of the normal constraint points to the data points: as the normal constraint points move further away, the curve becomes smoother. However, even the smoothest case here is worse than the result Shen got in his dissertation; I are unsure why there is the discrepancy in our results.

Further note in boxed area in Figure 19, lower right, that the curve appears to not pass through the red data point. An enlargement of that appears appears in Figure 20, left. In this figure, we see that the curve does not pass through the red data point. In Figure 20, right, I have coloured this region to indicate the unsigned value of the implicit surface (the colours were the default colour map used by Octave), with blue indicating 0. From this figure, we see that the curve passes close to this data point, and that there is an isolated 0 point at the data point.[7]

These examples indicate problems with Shen's method (or at least, with my implementation of Shen's method): the constructed curve is lumpy and potentially some of the data points are isolated zeros rather than being on the implicit curve. Shen

[7]Note that it is also possible that a thin arm extends from the main curve and forms a loop that passes through this "isolate" zero.

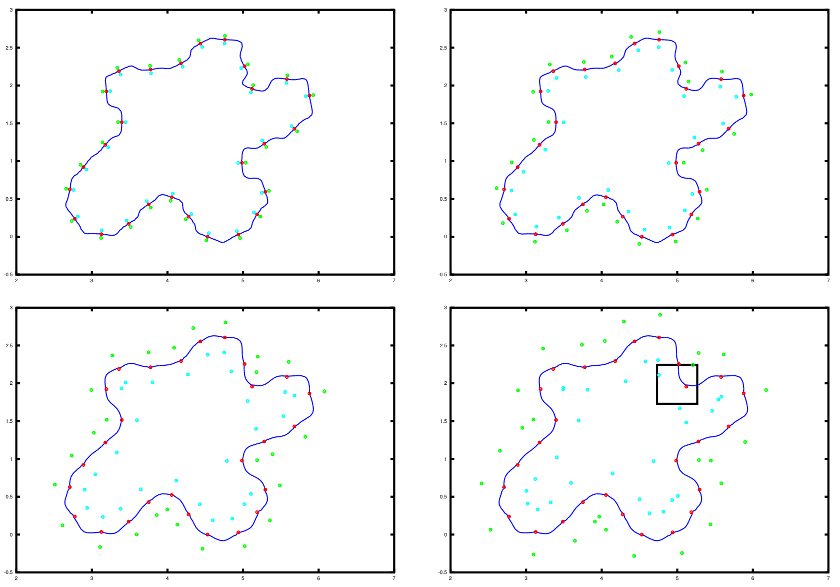


Figure 19: Shen’s scheme, constant with pseudo-normals. The red points are the points to interpolate; the green and blue points are the normal constraint points.

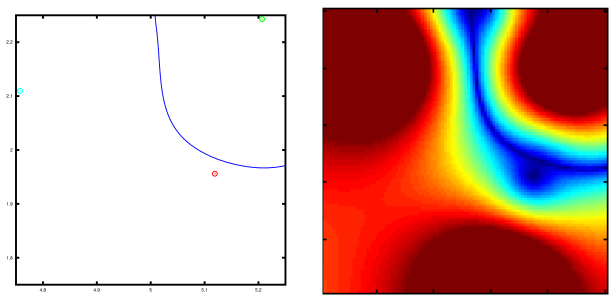


Figure 20: Shen’s scheme, constant with pseudo-normals. Left: enlargement of Shen curve in Figure 19, bottom right, showing that curve does not interpolate one of the data points; right, colour map indicating the value of the implicit function constructed in this area.

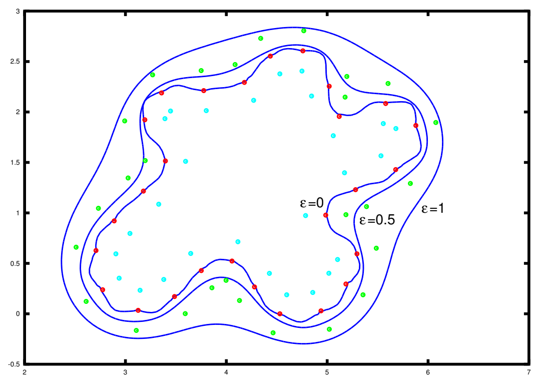


Figure 21: Shen's constant method with pseudo-normals. Using a non-zero value of ϵ results in approximating curves.

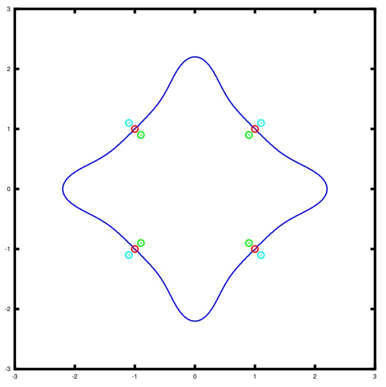


Figure 22: Shen's constant method with pseudo-normals.

noted the first of these two problems, and proposed an alternative normal scheme, which I present in the next section. However, I present two more results for my implementation of Shen's scheme.

Figure 21 shows the effects of varying the value of ϵ. In this figure, the inner curve have an epsilon value of 0, the middle curve has $\epsilon = 0.5$ and the outer curve has $\epsilon = 1.0$. As noted by Shen, using a non-zero value of ϵ results in the implicit only approximating the data, and in the implicit expanding beyond the data points. To get an approximation that better approximates the data points, Shen suggests setting their values to interpolate to be non-zero (either all positive or all negative, depending on your sign convention); however, beyond this he provides no guidance for setting ϵ.

As a final example of Shen's method, I used my implementation to fit a curve to a square of points. The result is shown in Figure 22, where we see that the surface bulges out way beyond the boundary of the square of points. There was only a minimal change in the curve when we moved the pseudo-normal points closer and further way from the data points.

3.2 True Normals

To improve the quality of the curves/surfaces produced by the method in the previous section, Shen introduced the idea of what he calls *true normals*. For each normal $\mathbf{n}_i$ associate with point $\mathbf{x}_i$ and value v_i, the linear function

$$\begin{aligned} S(\mathbf{x}) &= v_i + (\mathbf{x} - \mathbf{x}_i) \cdot n_i \\ &= \phi_{0i} + \phi_{1i}x^1 + \phi_{2i}x^2, \end{aligned} \tag{4}$$

where (x^1, x^2) are the 2D coordinates of $\mathbf{x}$. Then rather than interpolate the value v_i at point $\mathbf{x}_i$, Shen interpolates the functions S_i. For a point without an associated normal, Shen notes that using an S function with the zero vector in place of the normal reduces to interpolating v_i at the point. Since (3) is linear v_i, we can achieve this interpolation of S_i be applying Shen's method of the previous section to the ϕ_{0i}, to the ϕ_{1i} and to the ϕ_{2i} independently.

Using (3) to compute the ϕ's and then compute the curve gives poor results, as shown in Figure 23, left. However, looking at Shen's derivation in his dissertation, in (3.18) he uses W^2, while standard weighted least squares uses W (see, for example, [7,

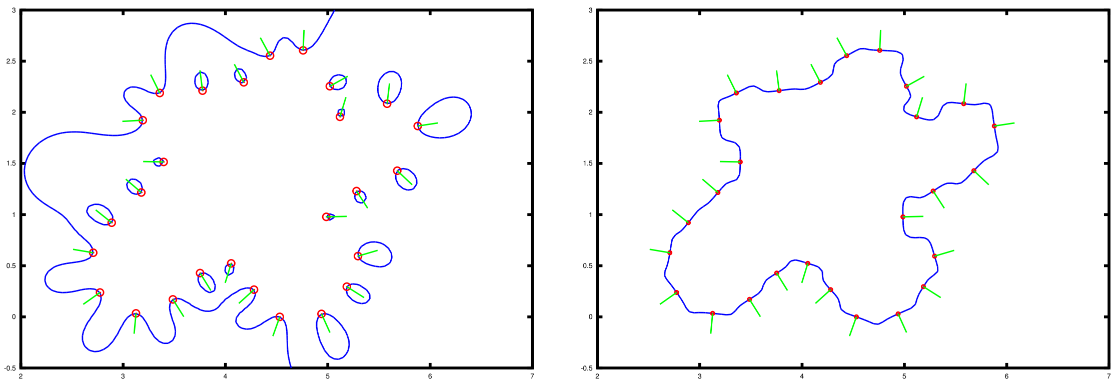


Figure 23: Shen's scheme, constant with true-normals. Left: Using $w(r) = 1/r^2$ to compute the curve with Shen's method. Right: Using $w(r) = 1/(r^2)^2$. Points to interpolate shown in red; normals shown in green.

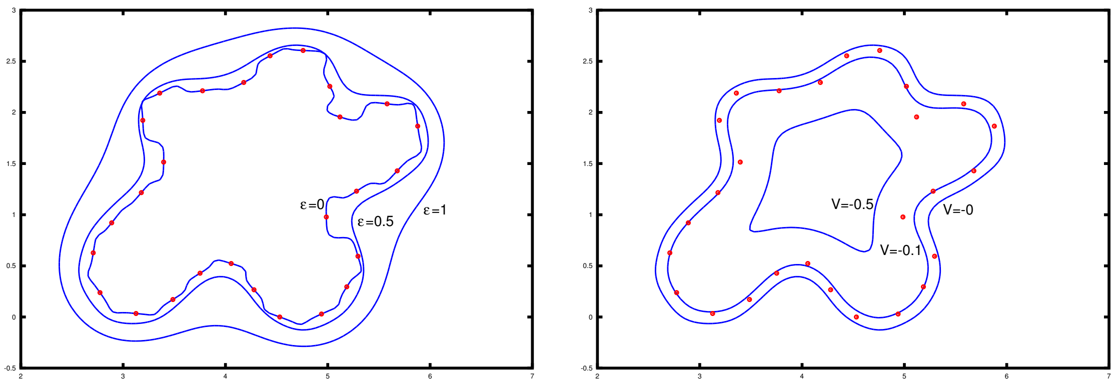


Figure 24: Shen's scheme, constant with true-normals. Left: adjusting the ϵ parameter. Right: using $\epsilon = 0.5$, adjusting V_i to get the approximation curve to lie closer to data points.

14]). The effect of using W^2 here is the same as using W with a weight function $w(r) = 1/(r^2 + \epsilon^2)^2$. Using $w(r) = 1/(r^2 + \epsilon^2)^2$ with (3.18) gives much better results as shown in Figure 23, right.

Again, the curve we get in Figure 23, right is similar to the one in Shen's dissertation, but slightly different. Comparing Figure 23, right to the curves in Figure 19, we see that much of the lumpiness of the curves have gone away, but has been replaced by flat regions on the curves.[8]

Adjusting the ϵ parameter, we get curves that approximate the data as shown in Figure 24. Increasing ϵ also causes the curves to "grow bigger", so that they completely enclose the data and move further away from the data as ϵ increases. To bring the approximations closer to the data, Shen suggests setting the v_i (the values to interpolate) to non-zero values (i.e., either all positive or all negative, depending on your sign choice). Figure 24, right, shows some examples created by trial-and-error guess work.

3.3 Linear Shen With Pseudo-Normals

Although Shen derives equations for a linear polynomial in the weighted least squares, he only used a constant polynomial. Given the lumpiness of his results with the pseudo-normals and the flatness with the true normals, I implemented the linear version of Shen's method.

In particular, the R function changes to

$$R(\mathbf{x}) = \sum_{i=1}^{N} w(||\mathbf{x} - \mathbf{x}_i||)[v_i - (c_0 + c_1^1 x^1 + c_1^2 x^2)]^2.$$

[8]The corresponding figure in Shen's dissertation corresponding looks better than Figure 23, right, but Shen's curve also has flat spots near the vertices.

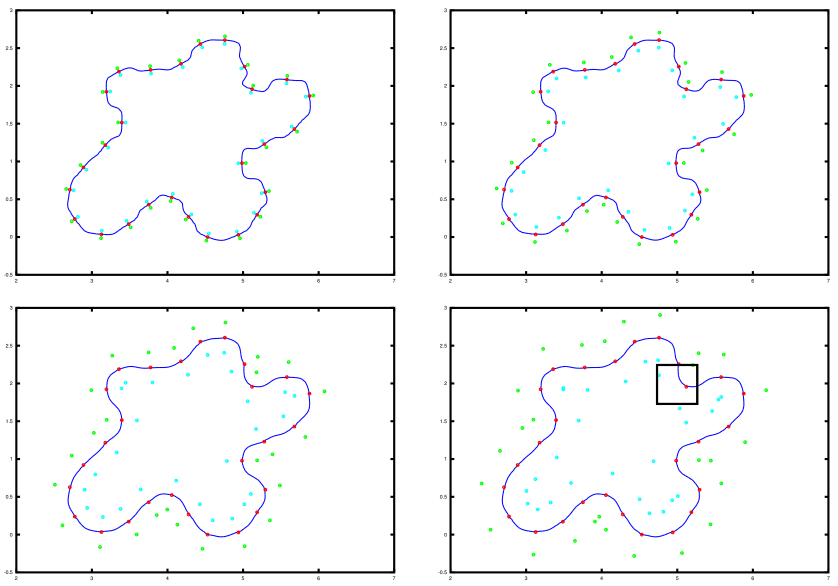


Figure 25: Shen's scheme, linear with pseudo-normals. The red points are the points to interpolate; the green and blue points are the pseudo-normal constraint points.

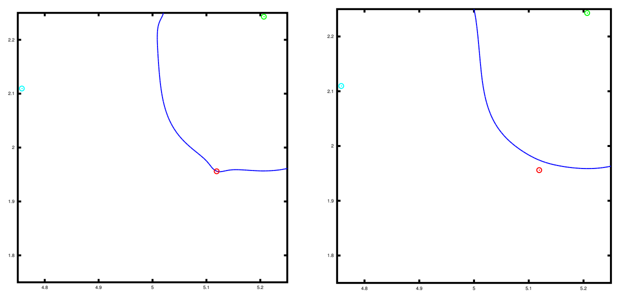


Figure 26: Shen's scheme, linear with pseudo-normals. Left: enlargement of linear Shen curve showing bump on curve at point of interpolation; right, linear Shen curve constructed with $\epsilon = 0.05$.

Taking the derivatives with respect to c_0, c_1^1 and c_2^1 gives the following matrix equations:

$$\begin{bmatrix} \sum_{i=1}^N w(||\mathbf{x}-\mathbf{x}_i||) & \sum_{i=1}^N w(||\mathbf{x}-\mathbf{x}_i||)x_i^1 & \sum_{i=1}^N w(||\mathbf{x}-\mathbf{x}_i||)x_i^2 \\ \sum_{i=1}^N w(||\mathbf{x}-\mathbf{x}_i||)x_i^1 & \sum_{i=1}^N w(||\mathbf{x}-\mathbf{x}_i||)(x_i^1)^2 & \sum_{i=1}^N w(||\mathbf{x}-\mathbf{x}_i||)x_i^2x_i^1 \\ \sum_{i=1}^N w(||\mathbf{x}-\mathbf{x}_i||)x_i^2 & \sum_{i=1}^N w(||\mathbf{x}-\mathbf{x}_i||)x_i^1x_i^2 & \sum_{i=1}^N w(||\mathbf{x}-\mathbf{x}_i||)(x_i^2)^2 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1^1 \\ c_1^2 \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^N w(||\mathbf{x}-\mathbf{x}_i||)v_i \\ \sum_{i=1}^N w(||\mathbf{x}-\mathbf{x}_i||)v_ix_i^1 \\ \sum_{i=1}^N w(||\mathbf{x}-\mathbf{x}_i||)v_ix_i^2 \end{bmatrix}$$

or in matrix form

$$\mathbf{W}\mathbf{c} = \mathbf{V}.$$

Inverting the 3×3 matrix W allows us to solve for $\mathbf{c}$:

$$\mathbf{c} = \mathbf{W}^{-1}\mathbf{V}. \tag{5}$$

We then evaluate the linear polynomials $f(\mathbf{x}) = c_0 + c_1^1x^1 + c_1^2x^2$. Note that c_0, c_1^1 and c_1^2 are functions of $\mathbf{x}$.

I implemented this in Octave and constructed curves for the Shen data set using the same pseudo-normals as in Figure 19. The results of this fit appear in Figure 25. At first glance, the results appear quite good: the curves (especially the bottom row) are much smoother than those constructed by the constant Shen method using either pseudo-normals are true-normals. However, closer inspection of the region in the square highlighted in Figure 25, lower right (the same area as in Figure 19, lower right) revealed an unwanted bump on the curve. Further, these unwanted bumps appear to occur at most if not all of the data points. A check of the Octave condition number (`cond`) of M on a 100×100 grid of samples in the region of Figure 26 shows that the maximum condition number of M was on the order of 10^7, which seems reasonable. On setting ϵ to 0.05, we

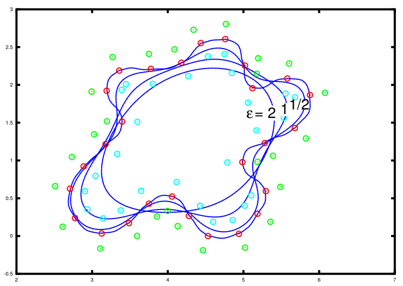


Figure 27: Shen's scheme, linear with pseudo-normals. The result of varying ϵ with the linear Shen reconstruction. The unlabeled curve has $\epsilon = 0.5$.

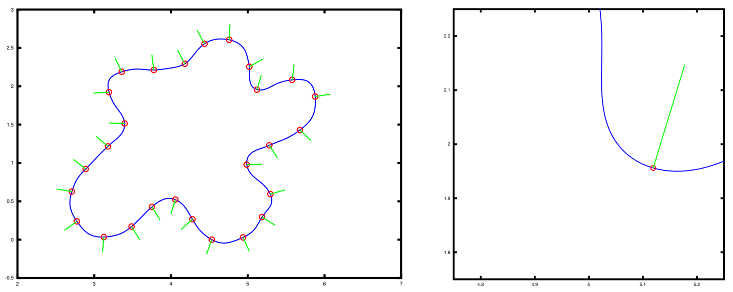


Figure 28: Linear version of Shen's method using true normals.

find that the bump is smoothed away (Figure 26) (the bump was still present for $\epsilon = 0.01$). This suggests that, for $\epsilon = 0$, as we get near one of the points to interpolate, the basis function for that point dominates the other basis functions in a way that distorts the curve, resulting in the bump.

A further surprise came when we set ϵ to other non-zero values. Figure 27 shows the result of setting ϵ to 0, 0.5, 1, and 2. As can be seen in the figure, when we increase ϵ, the linear version of Shen's method shrinks, unlike the constant method which grows as we increase ϵ.

3.4 Linear with True Normals

I implemented the linear version of Shen's method using true normals by applying (5) to each of ϕ_{0i}, ϕ_{1i} and $\phi 2i$ of (4). The result on the Shen data set is seen in Figure 28. Compared to the constant versions of Shen's method (Figures 19, 23, right), this is a great improvement. Compared to the linear method using pseudo-normals (Figure 25), the results are a bit mixed. The true normal version is much better than the pseudo-normal version when the pseudo-normal points are close to the data points, but when the pseudo-normals are further away, then it is unclear which is better. However, zooming in (Figure 28, right) we see that the artifact in the linear, pseudo-normal version is not present in the true normal version.

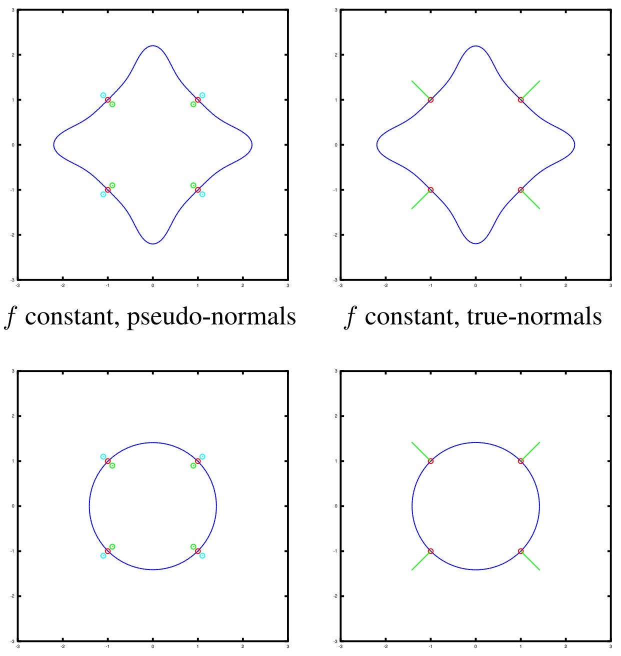
3.5 Comparison of Four Shen Variations

For a direct comparison of the four variations of Shen's method that I tested, I used the square of four points at $(\pm 1, \pm 1)$, with pseudo-normal points at a distance 0.1 away on the diagonals, and normals pointing away from the origin. Figure 29 shows the four results.[9] Clearly, the linear methods produce better shapes than the constant methods, although zooming in on the data points of the linear method with pseudo-normals shows an artifact similar to that of Figure 26.

I timed the four variations of Shen's method. The marching squares grid was sampled on a 101×101 grid over the ± 3 square. Times for the methods are given in Table 1. The time shown in this table is the time to do the marching squares sampling (but not the subsequent traversal and linear interpolation of the grid), as well as (for the true normal schemes) the time to compute the ϕs.

For costs, note that the constant, pseudo-normal scheme needs to compute two sums (computing the weighting function once per each pair of terms in the two sums) on $3N$ values. The constant, true-normal scheme calls the the constant, pseudo-normal scheme three times on N values.

[9]Figure 29, upper left, is the same as Figure 22.



f constant, pseudo-normals f constant, true-normals

f linear, pseudo-normals f linear, true-normals

Figure 29: Comparison of Shen variations on square of points.

	Pseudo	**True**
Constant	8	11
Linear	29	38

Table 1: Comparison of timings in seconds. Run on a Core2 Duo 2.20GHz Intel CPU with 2.0GB of RAM.

The linear schemes compute 9 sums (they would compute 12 sums, but symmetry in the W reduces this to 9), computing the weighting function once for each set of 9 corresponding terms in the sums. Again, the linear scheme with pseudo-normals has $3N$ terms in the sums, while the linear scheme with true-normals calls the constant scheme three times but with only N terms in the sums. Note also that the true normals schemes square the weight function

The timings where done using Octave, and because of Octave overhead these relative timings should be considered only as a general guideline.[10]

I also ran the four Shen methods on four of the Koch snowflake data sets (K_0, K_1, K_2, and K_3). The primary purpose of these tests (shown in Appendix A) were to check that Shen's methods did not create disconnected components on these data sets as the Turk-O'Brien method did (see Section 3.8). While all four methods interpolated each data set as a single curve (with the exception of one method that interpolated the data point with each point as its own components as already detailed above in Figure 23), a few additional interesting issues were revealed. In particular, Figure 30, top, shows a section of the curve created by the the linear method with true-normals on K_2. As we can see in this figure, the curve overshoots the data. For the true-normal methods of Shen, I used $w(r) = 1/r^4$. As a test on this data set, I also tried using $w(r) = 1/r^2$ with the linear method with true-normals. The result is seen in Figure 30.

There are two things of note in these figures. First, neither curve is particularly good, but the curve created with $w(r) = 1/r^2$ looks significantly better than the curve created with $w(r) = 1/r^4$. In Shen's dissertation, he glosses over the choice of the distance function; this figure (as well as some other figures in this report) suggest that the distant function has a large, important impact on the shape of the curve and that it needs further study. The second item of note is that from these figures, it is apparent that Shen's true-normal methods appear to not interpolate the specified normals. This question (of whether or not Shen's true normal scheme interpolates the normals) is explored more in the next two sections.

[10]Indeed, using one of Octave's matrix features, I could reduce the cost of the constant method with true normals from 11 seconds to 5 seconds, making it faster than the constant method with pseudo-normals, which the linear method was calling as a subroutine!

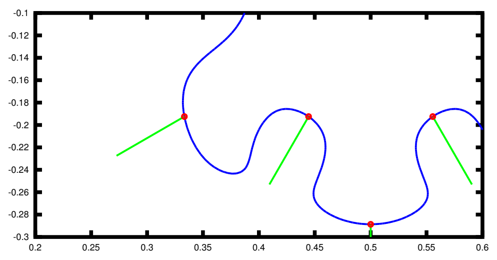
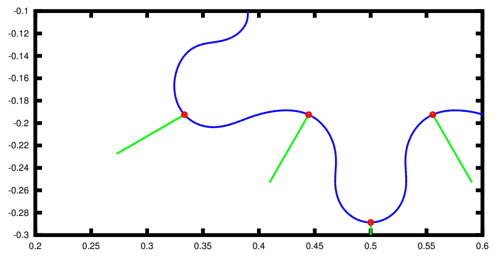



Figure 30: Linear Shen with true-normals on K_2 with different w functions. Top: $w(r) = 1/r^4$. Bottom: $w(r) = 1/r^2$.

3.6 An uneasy wiggle

Zooming in on one section of the curve created on Shen's data set by Shen's constant method using pseudo-normals revealed a problem. The zooms are shown in Figure 31. In the lower right image, we see that there is a small wiggle in the curve. The main concern with a feature this small is that a numerical computation of the normal with a small enough step size will result in a normal that does not look perpendicular to the curve at a larger scale.

3.7 Numerical issues, locality of interpolation of normals

While Shen calls his normal scheme "true-normals", he never proves that the curve interpolates the normal. Looking at some of the normals in Figure 28, the normals often appear to not be perpendicular to the curve (the curve in this figure was created with the linear variation of Shen's scheme using true-normals). Since the normal possibly is interpolated at a smaller scale, I zoomed in on one of the normals. A sequence of these zooms is shown in Figure 32.

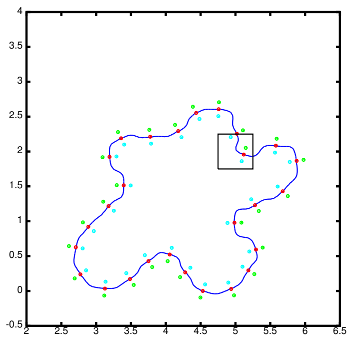
The bottom right figure suggests that something more problematic is occurring on the curve. Figure 33, left, shows a zoom of the region in the black box of this figure. From this figure, it is clear that something is failing numerically. Octave reported a near singular matrix and it became clear that the problem was with the 3×3 system of equations of Section 3.3. Looking at this system, if the point of evaluation became too close to one of the data points (a distance of about 0.002 in this example), then the corresponding term in the sums in the matrix entries dominated, making the matrix near singular.

To test the numerics of the inverse a bit further, three methods were used to solve this system of equations in Octave: `inverse`, Octave's back-slash operator, and LU-decomposition with pivoting. Figure 33, left, was generated using the `inverse` function of Octave to solve the linear system of Section 3.3. Using Octave's back-slash operating to solve this system gave the figure in the middle of Figure 33. While much better, this figure appears to answer our earlier question: looking near the data point, the curve appears to interpolate the normal, although on a very small scale.[11] Figure 33, right, shows the result of using LU-decomposition to solve the 3×3 system of equations; surprisingly, LU-decomposition was only a small improvement over using the inverse, and it was not as good as using the Octave back-slash operation.

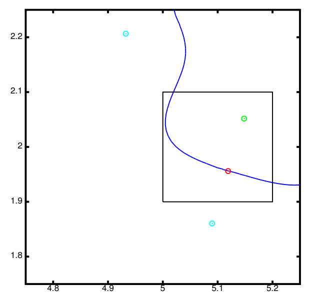
3.8 True Normals and Turk-O'Brien

We can also use the true normal scheme with the Turk-O'Brien scheme. The idea is that rather than interpolate v_i in (1), we interpolate the ϕ_i's of the normal functionals in (4). In essence, we solve the Turk-O'Brien system three times: once on the ϕ_{0i}, once on the ϕ_{1i}, and once on the ϕ_{2i}, giving three sets of W_i (for the curve case; for the surface case, we will have four sets of ϕs and four sets of W_is).

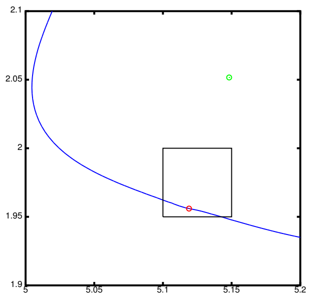
[11] While this intuitively makes sense, and the figure seems to support this assertion, given the numerical issues in the neighborhood of this point, it is hard to draw a definite conclusion.



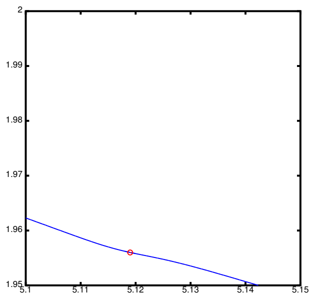
(a)



(b)

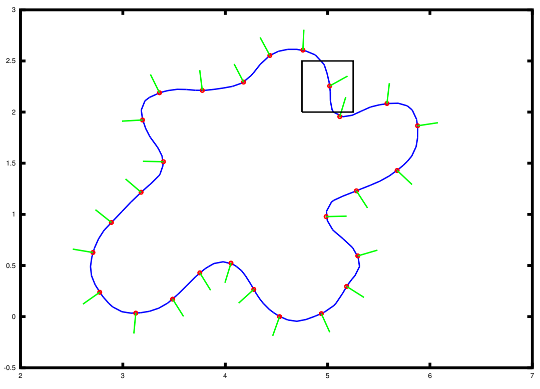


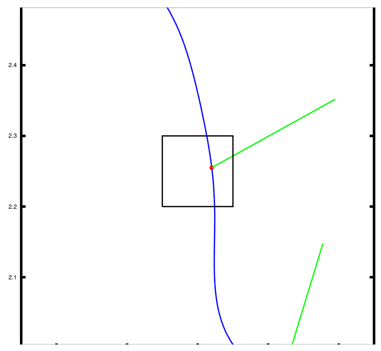
(c)

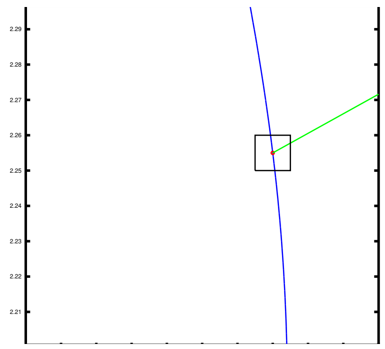


(d)

Figure 31: Shen's scheme, constant with pseudo-normals. A sequence of zooms on one data point to check whether the normal is perpendicular to the curve. The black box in one figure indicates the region shown in the next figure.


(a)


(b)


(c)

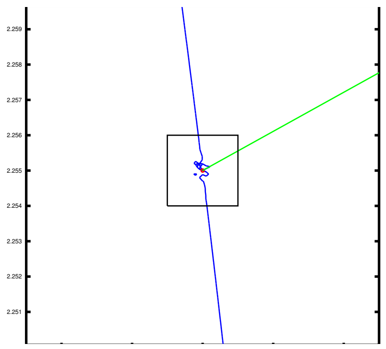

(d)

Figure 32: Shen's scheme, linear with true-normals. A sequence of zooms on one data point to check whether the normal is perpendicular to the curve. The black box in one figure indicates the region shown in the next figure. See also Figure 33, which is a zoom of the region in the black box of the lower right figure.

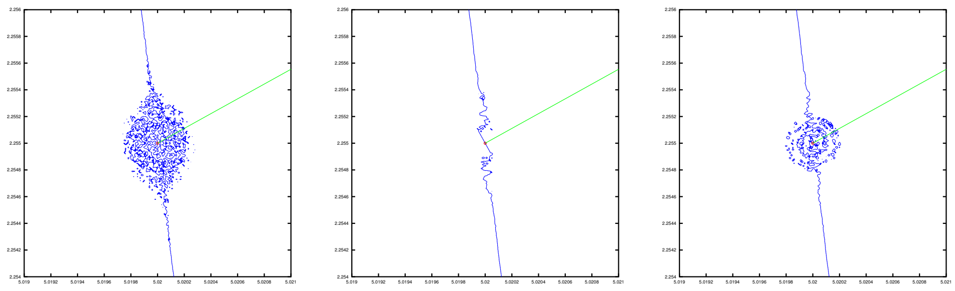


Figure 33: Shen's scheme, linear with true-normals. A close-up of the region in the black box of the bottom right figure of Figure 32. Left: using `inverse` to solve the 3×3 system of equations; middle: using the back-slash operator; right: using LU-decomposition.

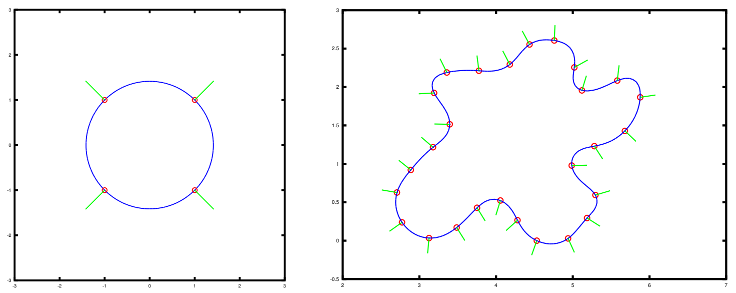


Figure 34: Turk-O'Brien using True Normals. Left: four points. Right: Shen data.

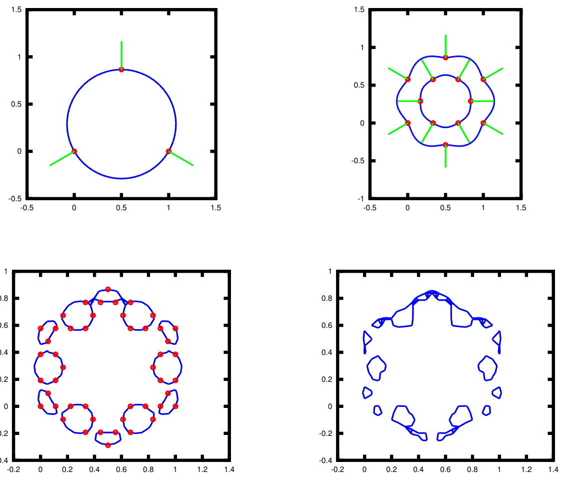


Figure 35: Turk-O'Brien using True Normals on Koch snowflake data sets.

The matrix used to solve for the three sets of W_i is identical, and needs only be constructed and inverted once. This has a huge computational advantage for the Turk-O'Brien scheme, since it halves the size of the matrix we need to invert. On the other hand, the cost of evaluating the resulting curve/surface will increase, since we need to evaluate it once for each set of W_is; again, some of the computations here can be reused, since the weight functions are identical for each set of W_is.

However, the results are not encouraging. Figure 34 shows the result of applying this true normal scheme with the Turk-O'Brien scheme. On the four point data, this method results in a good approximation to a circle; however, for Shen's data, while the shape of the curve is okay, the shape it not as good as the curves created using the pseudo-normals (Figure 14).

Worse, however, are the results on the Koch snowflake data (Figure 35). While the method constructs a good approximation of a circle for the initial Koch snowflake triangle, refining one or more times leads to a disconnected curve. Additional tests were run with Shen's methods to test if this was a problem with the data (it wasn't—see Appendix A for details).

3.9 Conclusions

Of the four version of Shen's method tested in this report, the linear method with true normals is the only one that yields curves with acceptable shape. The true normal method seems a much better technique for creating non-zero data to allow for non-trivial solutions to the implicit data fitting problem. In particular, the use of pseudo-normals is awkward in that you need to determine how far away to place the pseudo-normal points, and regardless, the resulting curve does not have this normal at the interpolated points (it is unclear that the curve interpolates the true normals either, but even if true normal scheme curve does not interpolate the normals, it appears to come closer to interpolating them than the pseudo-normal scheme).

4 Conclusions

This technical report has investigated the curves constructed by two interpolatory implicit function schemes: one by Turk and O'Brien [12, 13] and one by Shen [9]. Both methods require that some of the data to interpolate be non-zero since otherwise the trivial solution $f(P) = 0$ is obtained. Two main methods for specifying this non-zero data (pseudo-normals and true-normals) for specifying this data were investigated, as well as (for the Turk-O'Brien scheme) the approach of specifying just a few, non-zero values. Two variations of Shen's scheme were investigated: the constant scheme that Shen used in his dissertation and the linear scheme that he derived but appears not to have used in his dissertation.

Of the two schemes, the Turk-O'Brien scheme produced far better curves with less effort. While using their positive and negative constraints was difficult (where to place the positive/negative points? what value should they get?), using their pseudo-normal placement of positive or negative constraints was straightforward and produced reasonably good results. The main weakness of their scheme is that, for N points to interpolate with N pseudo-normal points, their scheme requires inverting a $2N \times 2N$ matrix. Inversion of this matrix is expensive, and seems to limit the data they can interpolate to a few thousand points. This restriction (of being able to interpolate only a few thousand points) does not seem serious, since even with that amount of data you are likely to want to approximate the data rather than interpolate it, and should use a different approach anyway. It was also unfortunate that the Turk-O'Brien scheme did not work with the true-normal scheme of Shen, since that would reduce the size of the matrix to invert and possibly double the amount of points that could be interpolated. Regardless, the Turk-O'Brien scheme was easy to implement and worked well on all but the sparsest data sets.

Shen's scheme, on the other hand, was fraught with problems. Aside from it being difficult to determine exactly what his scheme was from his dissertation, his scheme as implemented in his dissertation (using a constant f in (2)) gave very poor results. Using the linear scheme gave noticeable improvements, although the resulting curves using pseudo-normals still had serious shape issues. Shen's linear scheme with true-normals gave the best results of these Shen's variations, but serious shape artifacts still occurred on some of the data sets. Likewise, while Shen was somewhat blasé about the particular radial basis function used, this appears to be a critical shape parameter in his scheme.

In this report, I only tested these methods on curve data. Both methods, however, were primarily intended for constructing surfaces. Generally, when generalizing to surfaces, any artifacts you see in the curve method appear in the surface generalization (but worse), with additional surface artifacts often occurring. Thus, while the Turk-O'Brien method might produce reasonable surfaces, it seems unlikely that Shen's method(s) would. While Shen showed in his dissertation a few examples of his method being used to interpolate surface data, there was too little analysis to determine the quality of these surfaces. It should be noted, though, that the Shen scheme investigated in this report is not the primary focus of his dissertation, and thus the lack of analysis of it is not too surprising.

Another issue to consider is that while Turk and O'Brien are dismissive of "parametric modeling approaches" (presumably Bézier/B-spline schemes) because of the complications of patch stitching [13], the non-local approach of their implicit schemes results in (for modestly large data sets) surfaces that are very expensive to evaluate, essentially linear in the amount of data. Bézier/B-spline schemes (and subdivision schemes, although subdivision schemes are not parametric methods) have local evaluation algorithms, resulting in far lower evaluation times. Further, most parametric modeling scheme allow for modeling surfaces with sharp corners and flat surfaces, something that is desirable in most modeling tasks, neither of which the Turk-O'Brien scheme appears capable of (for example, what does the bottom of the bunny in Figure 6 of their paper look like?). While hierarchical methods and other techniques can be used to address at least some of these issues with these implicit schemes, using such additional techniques complicates the interpolatory implicitly modeling method, making any gains over parametric schemes less clear.

Mathematically, the papers describing these two schemes are lacking. In particular, there is no mention of three big issues: extra sheets, disconnected components, and topology/connectivity. While the only cases of a disconnected component I encountered occurred either because I was trying to make it happen (Figure 4, bottom), or when using interior/exterior (other than pseudo-normal) constraints (Figure 6 top-left, Figure 8, top-center, Figure 9, Figure 16), or because I was testing variations on normals schemes or weight functions (i.e., I was trying variations on the schemes that differed from those in the papers) (Figure 35), several times I ran into unexpected extra sheets with the Turk-O'Brien scheme (Figure 9 top-right, Figure 15, plus others not shown in this report).

Thus, a big unanswered question is what constraints must the data meet to ensure that you do not get extra sheets? Likewise, what is needed to guarantee that you have a single, connected component? Related to this is the question of topology: the curve data is specified as a polyline; the surface data as a polyhedron. What guarantees are there that the resulting curve/surface has the same topology/connectivity as the polyline/polyhedron? While some of these questions may be answered in the radial basis function literature, none of them were mentioned in the Turk-O'Brien papers or in Shen's dissertation. Shen does describe a second scheme in his dissertation that interpolates polylines/polyhedrons [10, 9]. However, while you can interpolate a polyline with this other method of Shen, and you can approximate a polyline with it, you can not use it to interpolate the points of a polyline with a smooth curve.

While useful, the tests run in this technical report are just a start. Before seriously considering either scheme, additional testing would be required. But since the tests in this report were for curve data while the scheme were intended for surface creation, a study of surfaces created with these methods in one obvious area of future study, as is a study of the many possible variations of Shen's method (would a quadratic f improve the surface quality, or would it run into more severe numerical issues when inverting the corresponding matrix? what is the best w function?). Likewise, determining the mathematical properties described in the previous paragraphs would be useful. Finding something similar to Shen's true-normals that works with the Turk-O'Brien method would also be useful, since it would allow for the Turk-O'Brien method to work on larger data sets.

Perhaps more interesting would be to determine what constraints would be required to make the Polynomial Least variation of the Turk-O'Brien method (Section 2.5) create single, connected components. Such a method would allow for the creation of relatively low degree interpolatory algebraic patches that presumably would inherit the many nice properties of the Polynomial Least [2].

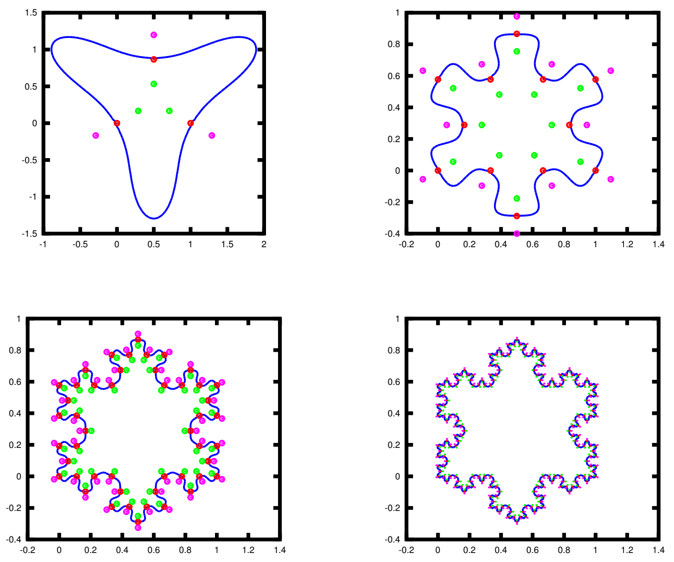


Figure 36: Shen's constant method with pseudo-normals on Koch snowflake data.

	Constant		Linear	
	Pseudo-Normals	True-Normals	Pseudo-Normals	True-Normals
K_0	7 (100)	4 (100)	22 (100)	32 (100)
K_1	25 (100)	11 (100)	79 (100)	91 (100)
K_2	216 (150)	157 (200)	684 (150)	1304 (200)
K_3	856 (150)	605 (200)	2712 (150)	5059 (200)

Table 2: Numbers in parentheses is marching squares size.

A Additional Test of Shen's Method: Koch Snowflake

Additional tests were run on Shen's methods. In particular, the four Shen variations were tested on the Koch snowflake. The primary reason for the tests was to ensure that the true normal scheme with Shen's method did not have the problem that the Turk-O'Brien scheme did on these data sets (see Section 3.8).

The results are shown in Figure 36, 37, 38 and 39. Note that some artifacts in these figures are due to under sampling in marching squares (in particular, in the lower right subfigures of Figure 37 and Figure 39). However, other artifacts are in the curves; in particular, in the upper right of Figure 36, the curve appears to not interpolate some of the data points. Looking closer at one such point reveals the curve does not interpolate these points (Figure 40, left), a problem that was noted in Section 3.1. On the other hand, Figure 40, right, shows that while Shen's linear method with pseudo-normals has regions of high curvature, in this example it interpolates the data points without the bumps seen in Figure 26, left.

Regardless, these figures show that all four variations of Shen's method handle these four Koch snowflake data sets without the problems that arose for Turk-O'Brien's method with true normals.

Additionally, Table 2 gives the timings for these sixteen examples. For the constant method with true-normals, I made use of one of the more efficient Octave matrix operations, and thus the faster than expected performance. Different grid sizes were used for the examples, with the number in parentheses indicating the grid size in one dimension (e.g., a number of '100' indicates a grid size of 100 $\times$ 100 cells). The timings should be scaled to adjust for the grid size when comparing methods or data sets that used different grid sizes.

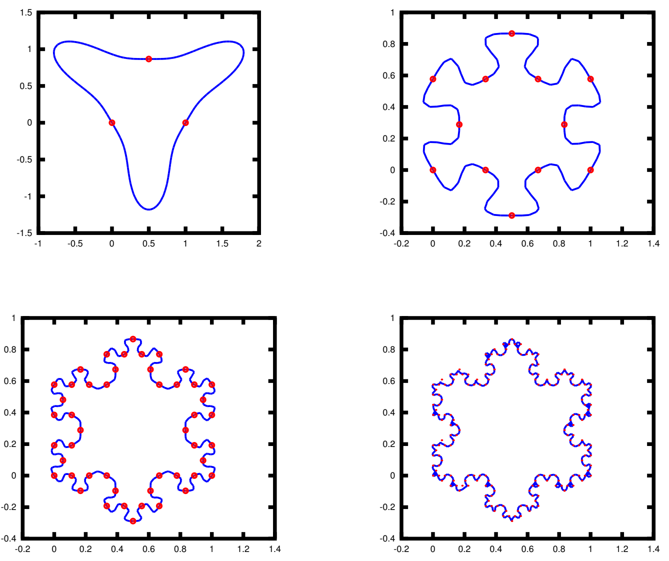


Figure 37: Shen’s constant method with true-normals on Koch snowflake data.

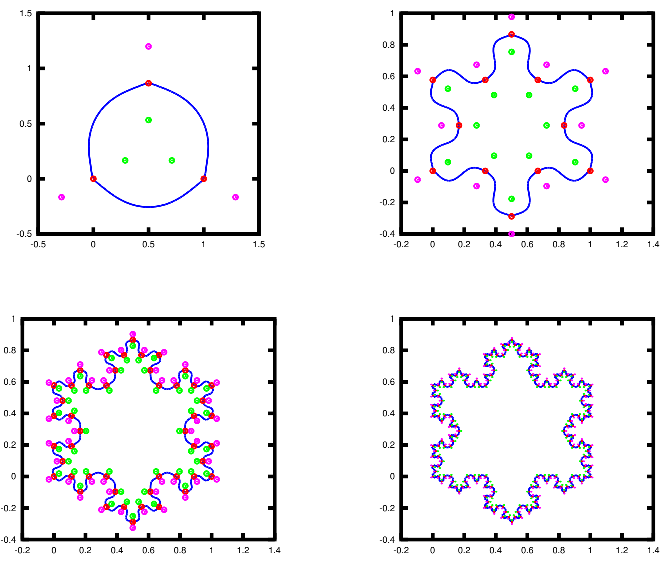


Figure 38: Shen’s linear method with pseudo-normals on Koch snowflake data.

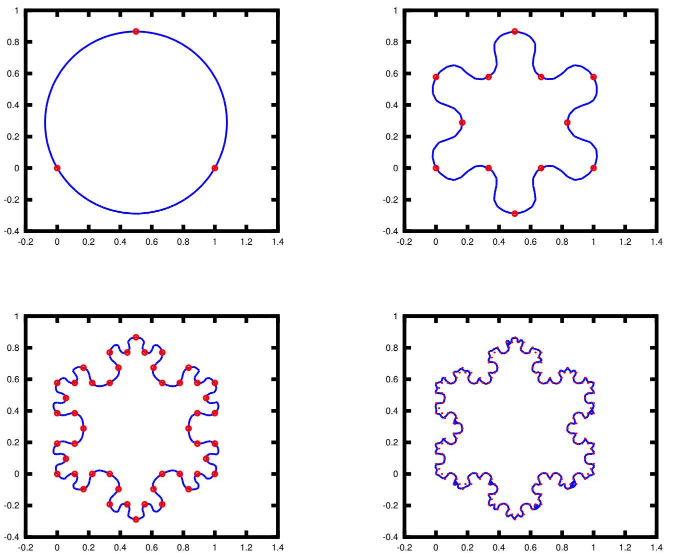


Figure 39: Shen’s linear method with true-normals on Koch snowflake data.

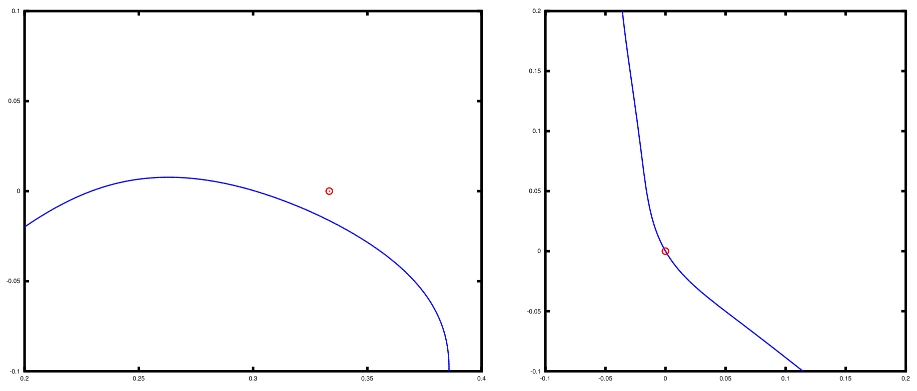


Figure 40: Left: close-up of Figure 36, upper right. Right: close-up of Figure 38, upper right. These close-ups were made with a finer marching squares grid than used in the full figures.

B Data

Here are the data sets used in this technical report.

B.1 Shen

The Shen data set (digitized by hand) is

```
(4.759, 2.606), (5.020, 2.255), (5.119, 1.956), (5.580, 2.084),
(5.879, 1.866), (5.679, 1.429), (5.281, 1.230), (4.986, 0.978),
(5.295, 0.594), (5.186, 0.295), (4.939, 0.029), (4.531, 0.000),
(4.279, 0.266), (4.056, 0.523), (3.752, 0.428), (3.486, 0.171),
(3.126, 0.034), (2.774, 0.238), (2.708, 0.627), (2.888, 0.921),
(3.178, 1.216), (3.396, 1.515), (3.192, 1.923), (3.358, 2.189),
(3.776, 2.212), (4.179, 2.293), (4.436, 2.554)
```

References

[1] Khodakhast Bibak, Chun Liu, Hamideh Vosoughpour, Grace Yao, Zanab AlMeraj, Alex Pytel, William Cowan, and Stephen Mann. Implicit surfaces seminar, spring 2012. Technical Report CS-2013-08, Cheriton School of Computer Science, University of Waterloo, September 2013.

[2] Carl de Boor and Amos Ron. The least solution for the polynomial interpolation problem. *Math. Z.*, pages 347–378, 1992.

[3] Huong Quynh Dinh, Greg Turk, and Greg Slabaugh. Reconstructing surfaces by volumetric regularization using radial basis functions. *IEEE Transactions on Patterna Analysis and Machine Intelligence*, 24(10):1358–1371, 2002.

[4] John W. Eaton, David Bateman, and Sren Hauberg. *GNU Octave Manual Version 3*. Network Theory Limited, 2008.

[5] Kirk Haller and Stephen Mann. Error sensitive multivariate polynomial interpolation. in preparation.

[6] Xiaogang Jin, Hanqiu Sun, and Qunsheng Peng. Subdivision interpolating implicit surfaces. *Computers and Graphics*, 27:763–772, 2003.

[7] P. Lancaster and K. Salkauskas. Surfaces generated by moving least squares methods. *Mathematics of Computation*, 37(155):141–158, July 1981.

[8] Benoit B. Mandelbrot. *The Fractal Geometry of Nature*. Times Books, 1 edition, 1982.

[9] Chen Shen. *Building Interpolating and Approximating Implicit Surfaces Using Moving Least Squares*. PhD thesis, University of California, Berkeley, 2007. UCB/EECS-2007-14.

[10] Chen Shen, James F. O'Brien, and Jonathan R. Shewchuk. Interpolating and approximating implicit surfaces from polygon soup. In *Proceedings of ACM SIGGRAPH 2004*, pages 896–904. ACM Press, August 2004.

[11] D. Sheppard. A two dimension interpolation function for irregularly spaced data. In *Proceedings of ACM National conference*, pages 517–524. ACM, 1968.

[12] Greg Turk and James F. O'Brien. Shape transformation using variational implicit functions. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '99, pages 335–342, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[13] Greg Turk and James F. O'Brien. Modelling with implicit surfaces that interpolate. *ACM Trans. Graph.*, 21(4):855–873, 2002.

[14] Holger Wendland. *Scattered Data Approximation*. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 2005.

[15] Brian Wyvill, Andrew Guy, and Eric Galin. The blobtree. warping, blending and boolean operations in an implicit surface modelling system. Technical report, Calgary, 1999.