Jeffrey Gay & Marc Thomson
APPM 3310 - 003
May 1st, 2014

A Matrix Approach to Data Approximation

Abstract:

This paper approaches the problem of fitting a curve to a data using different matrices. For small or simple data sets this is a very simple problem, but it quickly becomes difficult when the data set is large or more complicated. This leads to many solutions of varying difficulty, from linear systems to high level polynomial or even more complicated, non-polynomial systems. One of the ways to fit higher level polynomials to data is to use a Vandermonde matrix. A Vandermonde matrix (A) incorporates a set of x-values, evaluated over polynomial. This forms the linear system:

$$A\,\vec{c} = \vec{y}$$

Where A is the Vandermonde matrix, $\vec{c}$ is a column vector of the coefficients of the polynomial, and $\vec{y}$ is the vector of the y-values of the data. In the curve fitting problem, solving for $\vec{c}$ is a strait forward process. For 2-dimensional data sets this results in a good approximation. However, when the data set is 3-dimensional, the approximation becomes much worse.

To find a solution to this, both periodic functions and radial basis functions (RBF) are considered in the making of A. After analysis, the polynomial and sinusoidal approximations proved mediocre at approximating oddly shaped data in 2-D, while the RBF creates an accurate model that would not be useful for interpolation. However, only the RBF system can perfectly model exceptionally large and complicated data sets in 3-D. Such data sets often arise in landscape analysis and image processing, both of which are enhanced by RBF systems.

Introduction:

Often it is a necessary to fit a curve or surface to a set of points. This is curve can be used for interpolation, data analysis, or visualization of trends in the data. For small, simple data sets this is an easy task. However, modern data sets are becoming larger and more complicated as measurement technology improves. The most common way to do curve fitting is to do a form of regression by minimizing the squared distance between each point and the proposed curve. These proposed curves are normally polynomials, but can be any continuous function. One way of fitting higher level polynomials to data is to use a Vandermonde matrix to solve for the coefficients of the polynomial. We will be investigating the use of Vandermonde matrices as well as variations on the Vandermonde matrix in order to create curves that fit various data sets.

Vandermonde matrices are a family of matrices that are defined by using the x values of a set of coordinates (vector $\vec{x}$). The rows of the Vandermonde matrix are formed by evaluating each value in $\vec{x}$ for a given n^{th} degree polynomial. Given a set of m points, (x_1, y_1), (x_2, y_2), ..., (x_m, y_m), the $\vec{x}$ vector and associated Vandermonde matrix (A) are defined as[1]:

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} \qquad A = \begin{bmatrix} {x_1}^0 & {x_1}^1 & {x_1}^2 & \dots & {x_1}^n \\ {x_2}^0 & {x_2}^1 & {x_2}^2 & \dots & {x_2}^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ {x_m}^0 & {x_m}^1 & {x_m}^2 & \dots & {x_m}^n \end{bmatrix}$$

In an attempt to form better data approximations, two variations on the Vandermonde matrix were considered. The first was to construct a matrix similar to a Vandermonde matrix, replacing each row with a non-polynomial function that is evaluated at each point. The sine and cosine functions were chosen to be these non-polynomial functions as they are periodic, yet still relatively simple functions so as to not increase the computation time too drastically. The variation is shown below as *A'*, uses the same $\vec{x}$ as above. It should be mentioned that n is not necessarily the same for the Vandermonde matrix and the periodic variation matrix.

$$A' = \begin{bmatrix} {x_1}^0 & \sin(x_1) & \cos(x_1) & \sin(2x_1) & \cos(2x_1) & \dots & \sin(nx_1) & \cos(nx_1) \\ {x_2}^0 & \sin(x_2) & \cos(x_2) & \sin(2x_2) & \cos(2x_2) & \dots & \sin(nx_2) & \cos(nx_2) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ {x_m}^0 & \sin(x_m) & \cos(x_m) & \sin(2x_m) & \cos(2x_m) & \dots & \sin(nx_m) & \cos(nx_m) \end{bmatrix}$$

The second variation is to use radial basis functions (RBF) to create a matrix (G) that can be used instead of the Vandermonde matrix. A RBF is some function that depends on the distance (r) from the origin or a selected point. There are many different RBF's that are commonly used; some of the most common being the Gaussian function (left) and the multiquadric function (right), both shown below.

$$\phi_{Gauss}(r) = e^{-(\varepsilon r)^2} \qquad \phi_{Multiquadric}(r) = \sqrt{1 + (\varepsilon r)^2}$$

This would mean that a normal distribution, assuming the Gauss RBF is used, is centered about a center point, C, and the function is evaluated at a given point, x. In these functions, ε is the shape parameter which modifies the width of the radial function and r is the distance from the center point to the evaluation point as defined by the Euclidian norm, as seen in the equation below.[1]

$$r = \|x - C\|$$

Model development/Mathematical Formulation:

Vandermonde matrices are used in curve fitting because they can be used to determine the coefficients of an nth degree polynomial. The starting equation of this analysis is

$$A\,\vec{c} = \vec{y}$$

Where A is the associated Vandermonde matrix for a set of $\vec{x}$ values, $\vec{c}$ is a column vector of the coefficients of the polynomial, and $\vec{y}$ is the vector of the y-values not in the column space of A. Therefore, this equation has no solution. The vector $\vec{c}$ is then changed to find the minimum

[1] All background information on RBF found in "Solving PDEs with radial basis functions" [3]

error, which is the difference between the two sides of the equation. Using an equation whose derivation is excluded from this paper, it is determined that [2]:

$$A^T A\vec{c}^* = A^T\vec{y}$$

Where $\vec{c}^*$ is the optimal vector of coefficients to minimize the error. Solving for $\vec{c}^*$ results in the following equation:

$$\vec{c}^* = (A^T A)^{-1}A^T\vec{y}$$

This solution for the Vandermonde matrix can be directly applied to the periodic variation matrix. In this variation, the coefficients are applied to the sine and cosine terms respectively. Symbolically these curves would be written as follows:

$$y(x) = {c^*}_1 + {c^*}_2 x + {c^*}_3 x^2 + \cdots + {c^*}_n x^n$$

$$y(x) = {c^*}_1 + {c^*}_2 \sin(x) + {c^*}_3 \cos(x) + {c^*}_4 \cos(2x) + {c^*}_5 \cos(2x) + \cdots + {c^*}_n \cos(nx)$$

RBF

Given the same data set of solve for a curve with radial basis functions, a matrix G_{mxm} is constructed as follows:

$$G = [g_{ij}] = [\phi(r_{ij})] = [\phi(\|x_i - x_j\|)]$$

Where r_{ij} is the distance between two values of $\vec{x}$ and ϕ(r) is the chosen RBF. The starting equation using RBF is:

$$G\vec{\lambda} = \vec{f}$$

Where $\vec{\lambda}$ is a vector of weighting coefficient and $\vec{f}$ is the y values. Because $\vec{f}$ is within the column space of G, there is a solution for $\vec{\lambda}$. Essentially, a RBF approximation of *n* data points will define to function as a sum of *n* radial basis functions centered about each point, with coefficients $\vec{\lambda}$. The symbolic form of this curve can be written as follows[3]:

$$y(x) = \sum_{k=1}^{n} \lambda_k \phi(\|x_k - x\|)$$

3-D

In order to accommodate 3-Dimensional data, slight modifications had to be made to each approximating method. For the Vandermonde method, the size of the matrix had to be increased to account for the additional variable. An additional n columns were added for $[y^1, y^2, \ldots, y^n]$ to the Vandermonde matrix. The periodic variation must go through a similar process, the size of the matrix must be increase for the variable. This must be increased by 2n for the $[\sin(y), \cos(y), \ldots, \sin(ny), \cos(ny)]$ terms. For both of these methods, the solution was changed to use $\vec{z}$, a vector of z values, instead of $\vec{y}$:

$$\vec{c}^* = (A^T A)^{-1} A^T \vec{z}$$

The RBF method had to be modified very little to accommodate the extra variable. The only change was in r_{ij}. Instead of only taking in one value, $\vec{x}$ became the vector $(x, y)^T$. This works because the input of RBF is a scalar value, r.

Numerical Work and Examples

Example Data

To test the effectiveness and applications of different methods of curve and surface fitting, several example data sets were tested. These sets were generated manually with the intention that they show a pattern without being on a simple curve or surface. Figure 1 shows a 2-Dimensional data set, while figure 2 shows a 3-Dimensional set.

2-D Data

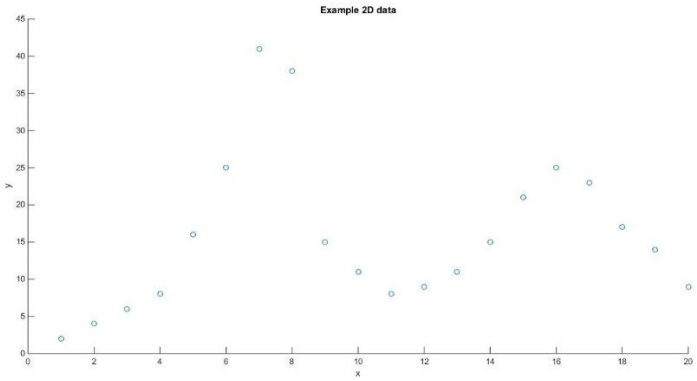

Figure 1: Sample Data

3-D Data

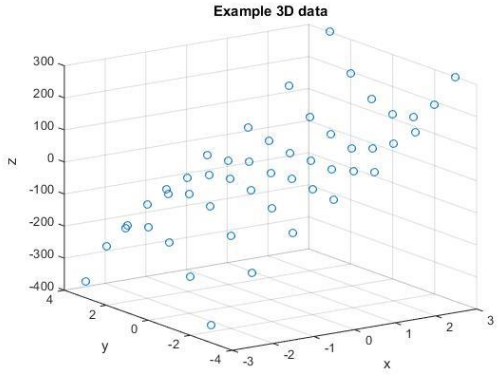

Figure 2: 3D Sample Data

Polynomial Approximation

The first solution is a simple application of Vandermonde matrices to create a least-squares polynomial approximation of the data. As shown in figure 3, even using a 5th degree

polynomial creates a poor model, with R^2 of only 0.37. The 3-D version of this solution, with degree 4 for both variables, is marginally better than its 2-D counterpart. It produces an R^2 of 0.83 for the example data seen in figure 4. Due to difficulties in iterating cross terms, this model does not include terms containing both x and y (ex: xy^2).

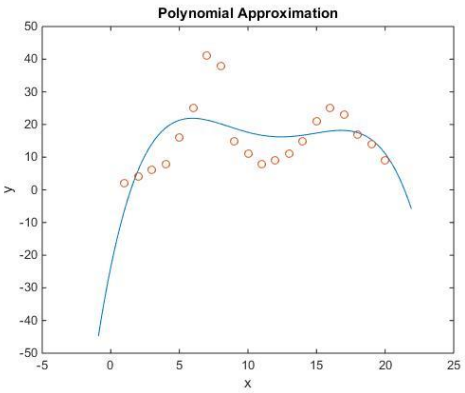

Figure 3: 5th degree polynomial approximation.
R^2: 0.3678
Elapsed time: 0.064 seconds

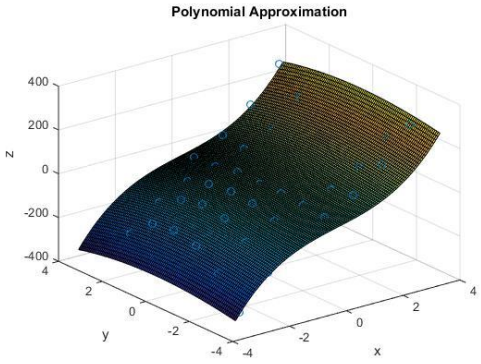

Figure 4: 4th degree polynomial surface
R^2: 0.8307
Elapsed time: 0.077 seconds

Manually adding cross terms to the 3-D polynomial model vastly increases the accuracy of its approximation. Figure 5, below, shows this approximation including all cross terms with degrees of two or less, and yields an R^2 of 0.97. However, because the matrix is coded manually, it would be difficult to build for larger sets of data that require higher degree polynomials to approximate the data.

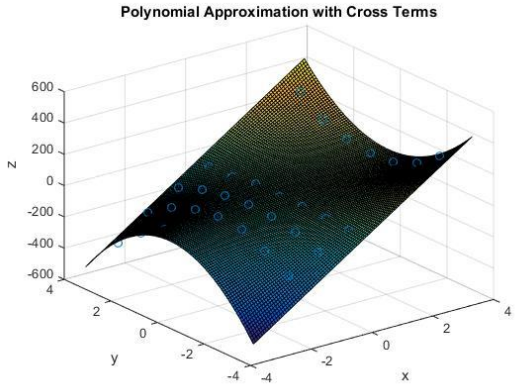

Figure 5: 2nd degree polynomial with cross terms
R^2: 0.9724
Elapsed time: 0.017 seconds

Sinusoidal Approximation

The sinusoidal approximation performs better than the polynomial approximation, producing an R^2 of 0.9370 for the 2-D data, figure 6, using a fourth degree model (the sinusoidal terms go up to sin(4x)). Figure 7 shows that the 3-D sinusoidal (3rd order) approximation performs roughly as well as the 3-D polynomial approximation, with an R^2 of 0.8308. This R^2 is

nearly identical to the polynomial approximation, indicating that this model is also inadequate for large sets of data. This is again, due to the lack of cross terms.

R2: 0.9370
Elapsed time: 0.059 seconds

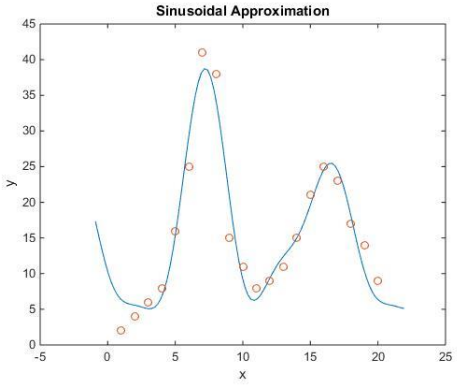

Figure 7: 4th order sinusoid

R2: 0.8308
Elapsed time: 0.77 seconds

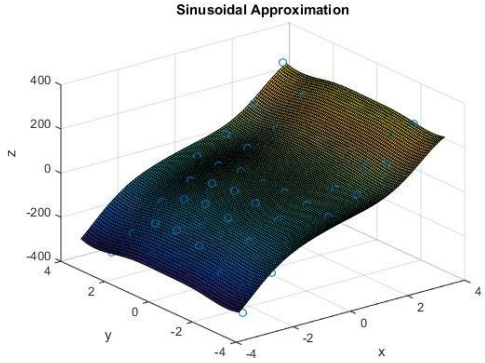

Figure 6: 3rd order sinusoid

Radial Basis Function (RBF) Approximation

Using the RBF network, a much more accurate model is constructed. The result of this approximation is an analytical function that passes through every input point, shown in figure 8. However, data in 2-D often has error in outliers, so the RBF approximation would not be useful in most cases. The 3-D RBF approximation, figure 9, does the same as its 2-D counterpart, fitting every point to the surface. However, there are far more situations in 3-D in which fitting a surface through every input point is useful. Its success indicates that ability of these approximations to model in 3-D.

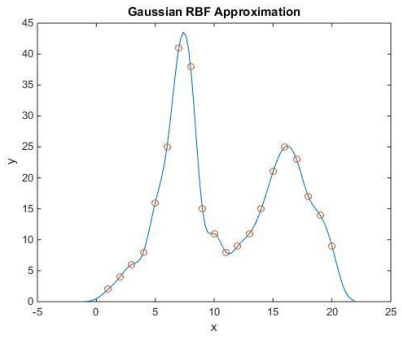

Figure 9: Gaussian RBF curve approximation

R^2*: 1*
Elapsed time is 0.057 seconds

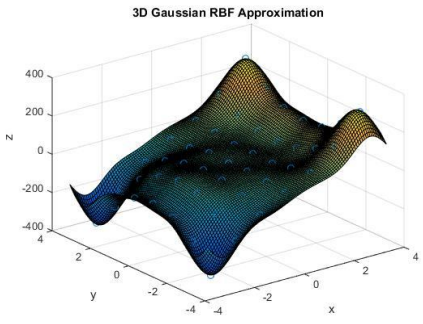

Figure 8: Gaussian RBF surface approximation

R^2*: 1*
Elapsed time is 0.12 seconds.

Using 3-D RBFs to Manipulate Images

Original Image

One of the more interesting applications of 3-D RBF networks is in image processing. To explore this possibility, the grayscale clown in figure 10 will be examined. This image is built into MatLab and can be called easily. The intensity of the image can be used in place of the z-value of each point.

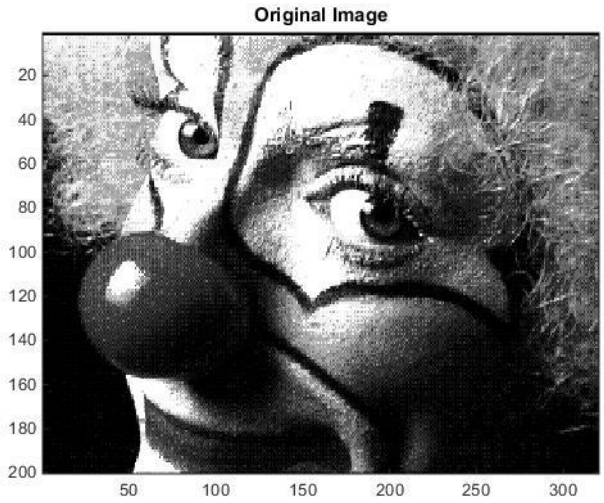

Figure 10: Stock test image

Approximating Images from a Sample of Points

RBF networks can be used to approximate an image from a smaller sample of points. This can be used to compress an image or to deal with an image with broken pixels. In this case, the image was compressed to one sixteenth of its original size. The results, figure 11, are obviously blurrier and less clear than the original image, but do quite well considering the small percentage of points being sampled.

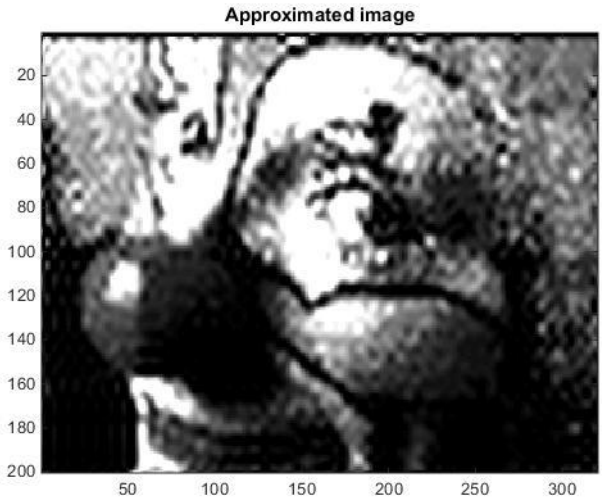

Figure 11: Reconstructed image

Elapsed time is 8.709339 seconds.
Number of points processed=4000
Percentage of points processed=6.2500

Enhancing a Zoomed Image

RBFs can also smooth an image during zooming. For moderate resolution images, zooming will reduce resolution and bring out apparent pixelated, as seen in figure 12 below. Figure 13 shows that by using RBF's, the pixilation can be blended, leading to a sharper, higher resolution zoomed image. Because of the large number of points required in this case (6396), the

program was relatively slow (about 15 seconds.) However, given the immense size of this matrix, the slow speed is understandable.

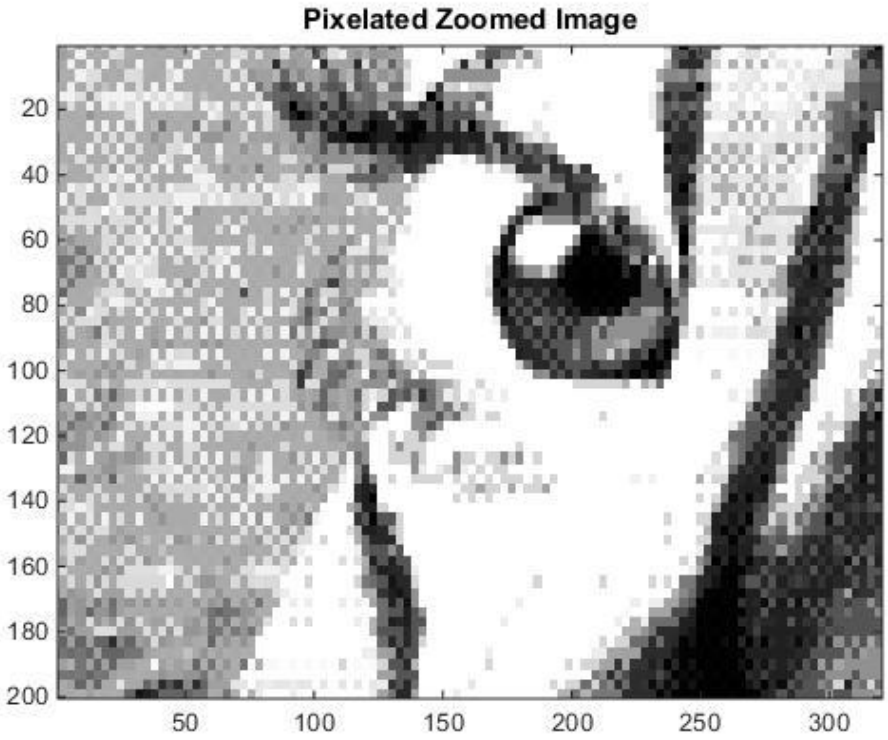

Figure 12: Image at 9x zoom, stock

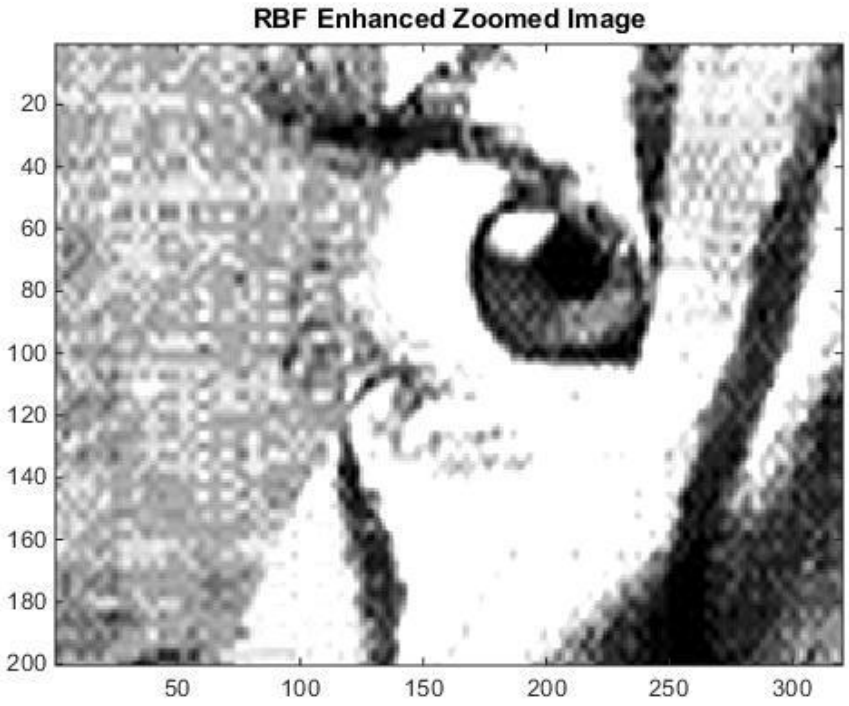

Figure 13: Image at 9x zoom, post enhancement

Elapsed time is 15.437256 seconds.
Number of points processed=6363

Building a Topo Map from a Sample of Points

3-D RBF approximations can easily be extended to landscape approximation. For example, take 100 points obtained from the United States Geological Survey (USGS) for a small region of Chautauqua Park in Boulder, Colorado. Using RBF approximations, a topo map of the region around Green Mountain can be created, seen in figure 14. The elevation is in feet, with the equipotential lines representing changes of 125 feet. For this example, the multiquadric RBF was used in place of the Gaussian RBF because it produces better results.

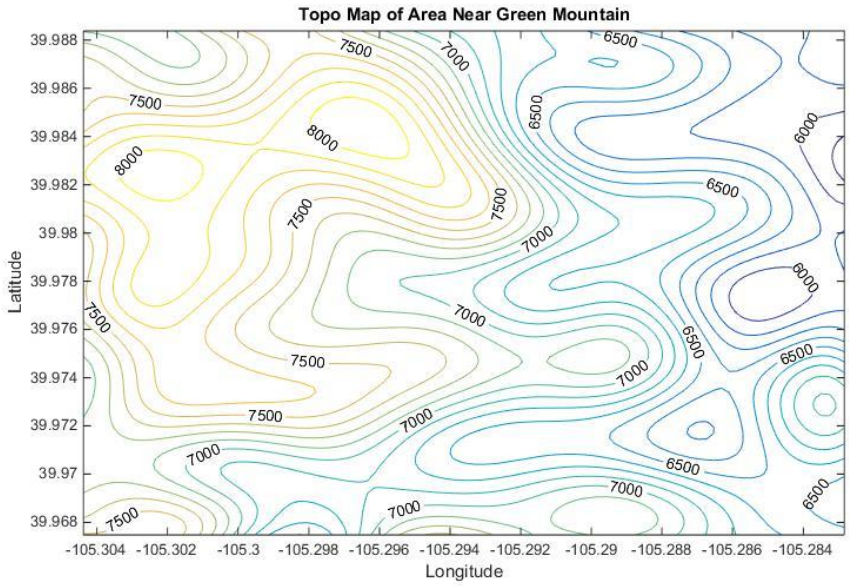

Figure 14: Constructed topo map of Green Mt. area.

Elapsed time is 0.196920 seconds.

Discussion/Conclusions

For the 2-D data, only the polynomial approximation failed to create a good curve. Nevertheless, it is expected that either the polynomial or periodic method would create a good approximation of a given data set, while the other would not. This is because data often either trends toward being periodic or non-periodic. Polynomial approximations are useful in the non-periodic case, while sinusoidal approximations are useful in the periodic case. The RBF approximation had a "perfect" fit to the data, an R^2 value of 1, but an RBF forces the curve to go through each point. Considering this, RBF could be useful for interpolation within the data set as it does create a very good approximation. However, the RBF approximation is very vulnerable to outliers and random errors in the data. This is due to the fact that RBF forces the curve through every point, meaning that every point has a high influence on the final function. Another downside to using the RBF approximation is inaccuracies around the edges of the data. Depending on which RBF is used, the curve would approach zero or $\pm\infty$ as the function gets farther from range of data. Because of this, its approximation should only ever be used for interpolation inside the range of data.

For the 3-D approximations, the curve produce through RBF was clearly better. This is mainly due to the lack of cross terms (terms that include both x and y). For polynomials of degree less than three, it is manageable to manually build the Vandermonde matrix, but for degree of three or more it is unreasonable as there are n^2 cross terms. This cross terms issue limits both the polynomial and periodic approximation in how close the curve will fit. Cross

terms do not present an issue for the RBF approximation because it combines x and y into one variable, distance. Because of this it creates the closest approximation to the data, regardless of what the original surface is.

This has multiple large implications. As in the 2-D model, if there is an outlier in the data it will affect the RBF model much more than the other two. This limits the usefulness of the RBF for interpolation in data sets that might have significant random error. The fact that it can reliably create smooth surfaces regardless of how complex or large the data is means that it can be applied to pictures. As seen in figure 11 it can recreate an image from a small selection of points. This can be thought of as uncompressing a low quality image. This also means that a user can "zoom" into a particular area of a picture and smooth the sharp lines of the image while preserving the color of the original image. This "smoothing" can also be used to map terrain. It is may not be possible to create perfect topography maps of a landscape due to terrain or other hazards. Using a small number of points, RBF is able to create a fairly accurate map of the area. Another possible application is enhancing full sized images to take an older, low quality image and increase the resolution. Though a grayscale image was used in this paper, RGB color values could be used in place of grayscale intensity, allowing RBF's to be used for color images.

This research opens the door for a wide range of additional exploration, it would be interesting to investigate other RBF's and see the differences in the resulting curves. Another aspect that could use modified is the shape parameter, ε. This parameter controls the width of the RBF, which needs to be balanced with the data. For the purpose of this report, ε was adjusted manually based on the quality of the resulting approximation. If the data has a large range and few points, the RBF should be wide to account for this; whereas if the data is dense, the RBF should be narrower. The next step in the RBF approximations is to automate the value of ε.

The most important conclusion for this research is that the method of modeling data is highly dependent on the application. For interpolation of 2-D sets of data, Vandermonde matrices with polynomial or sinusoidal terms might be a better solution than RBF's. Often, this is the case in 3-D, when the data sets are subject to random error. However, 3-D RBF's show a great degree of promise for handling large, complex systems like images or landscapes.

References

[1] Weisstein, Eric W. "Vandermonde Matrix." From MathWorld--A Wolfram Web Resource. http://mathworld.wolfram.com/VandermondeMatrix.html

[2] Olver, Peter J., and Chehrzad Shakiban. Applied Linear Algebra. Upper Saddle River, NJ: Prentice Hall, 2006. Print.

[3] Fornberg, B., & Flyer, N. (2015). Solving PDEs with radial basis functions. Cambridge University Press, 1-44. Retrieved April 21, 2015, from Cambridge University Press.

[4] Navidi, W. (2011). *Statistics for engineers and scientists* (3rd ed.). New York: McGraw-Hill.

[5] (n.d.). Retrieved April 1, 2015, from http://www.mathworks.com