Inverse Scattering

Brad Nelson
3/9/2012
Math 126 Final Project

This paper investigates the ability to determine the boundary of an unknown object scattering waves using three different scenarios. In the first, we assume all far-field data is known from the scattering of a single incident wave. In the second, we assume that there are several nodes that emit waves and receive information back about the scattered wave. In the third, there is only one node which can emit waves with different wavenumbers and measure the scattered wave.

I. INTRODUCTION

Inverse scattering problems basically fall into one of two categories: optimizing a component for a particular use (such as designing an airplane wing to provide maximum lift), or determining the properties of an object using scattering of waves [1]. The inverse scattering problem investigated in this paper is of the second kind, and effectively asks if one can determine the shape of an object that scatters waves based on the scattered waves measured at some distance from the object. This technique is important for its use in modern technologies such as radar, sonar, medical imaging, geological exploration, or any sort of non-invasive imaging problem [2]. The particular problem investigated is the scattering of waves governed by the Helmholtz equation with Dirichlet boundary conditions

$$\begin{cases} \Delta u + k^2 u = 0 \\ u|_{\partial\Omega} = 0 \\ \lim_{r\to\infty}(\frac{\partial u}{\partial r} - iku^s) = 0 \end{cases}$$

The final condition is the Sommerfield radiation condition, which ensures that the wave is properly behaved at far distances. One of the defining features of the inverse scattering problem is that it is improperly posed: a correct solution may not be attainable with provided information [1].

The method used in this investigation uses multiple iterations of solving the forward problem with different trial shapes. This method was first proposed in the 1980s by several groups of researchers [2]. This solution requires a numerical forward solver, which determines the far-field data for a given boundary, which is then compared with the measured far-field data. This process is performed many times for different boundaries, and is fed into an optimization loop to try to find a minimum difference.

II. METHODS

Throughout this project, objects were assumed to be smooth deformations of a sphere. That is, their boundary could be described using a truncated Fourier series, which is 2π periodic. The object was described using $2n+1$ Fourier coefficients, $(a_0, a_1, \dots a_n, b_1, \dots, b_n)$, and the radius of the object from the origin was

$$R(\theta) = \sum_{i=0}^{n} a_i \cos(i\theta) + \sum_{i=1}^{n} b_i \sin(i\theta)$$

In creating objects to generate data, a mean radius of 1 was assumed ($a_0 = 1$). In all three scenarios data was collected at a radius of 4 from the origin. The incident wave hitting the object was assumed to be a plane wave, which is a decent approximation of a point source wave if the source is far-off.

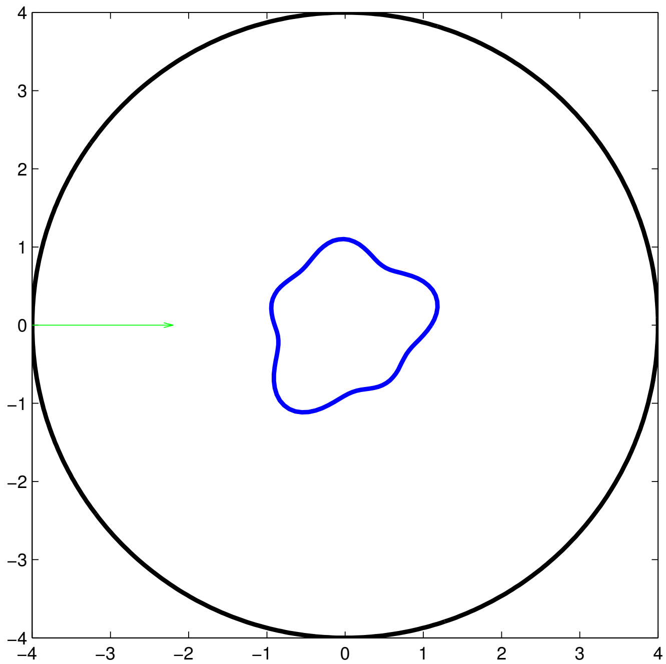
The forward solver used the Boundary Integral Method. The Helmholtz Double-Layer kernel was used, and 50 periodic trapezoid nodes were used along the boundary of the object. This method can potentially produce a (spuriously) singular kernel depending on the wavelength used and shape of the object, but the probability of this result is very small, which allowed this implementation to work well enough for the purposes of the following numerical experiments.

The experiments were run in MATLAB. The following method was used to produce and process data:

1. A known coefficient vector a_i was used to generate data at the desired points d_i.
2. A function was created that took an arbitrary coefficient vector a as an argument, and compared the resulting data d to d_i by taking its 2-norm. $f(a) = ||d - d_i||_2$

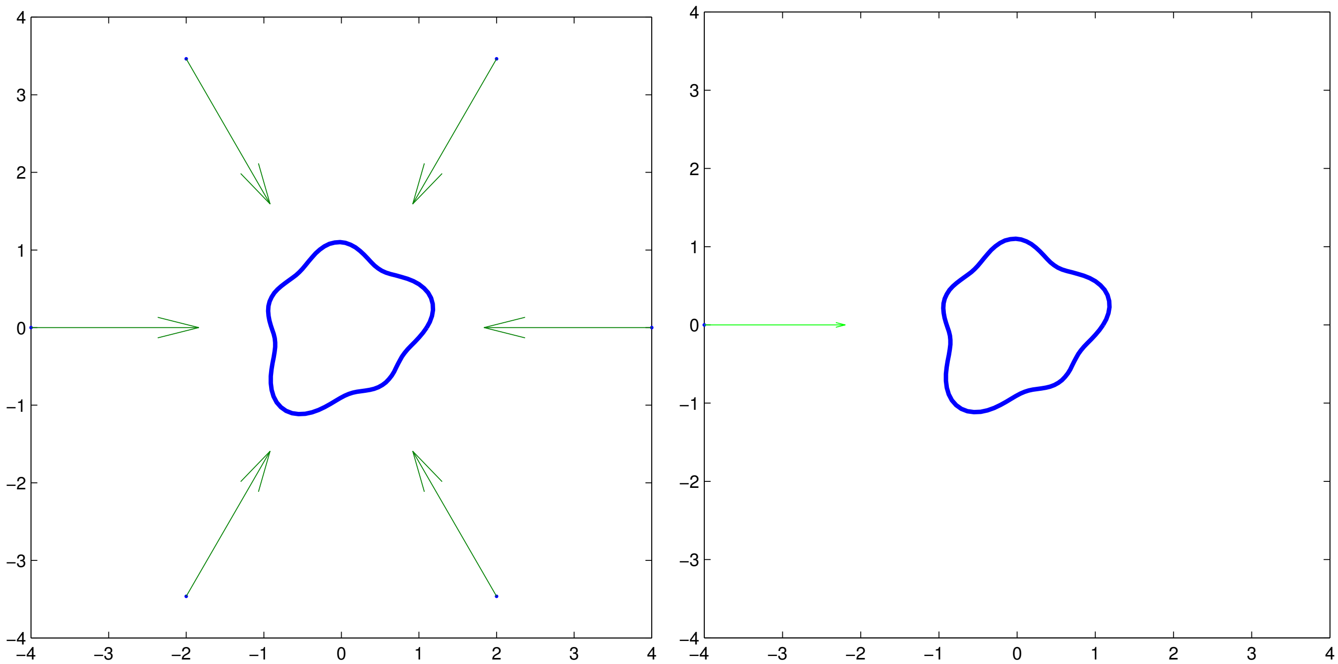
3. The function $f(a)$ was minimized by starting with an initial guess a_g and using the MATLAB optimization package function `fminunc`, returning a_f.

Clearly, the goal is to find $a_f = a_i$, in which case $f(a_f) = 0$. The default choice of a_0 was $(1, 0, 0, \dots)$, which is just a circle. One danger of using `fminunc` is that it could potentially settle on a non-global minimum. Also to note is the fact that the solution vector a_f can not be any longer than a_0, which can be problematic, as discussed later. In general for a_0 $n = 5$, so there were 11 degrees of freedom. In all the following experiments, `fminunc` was stopped after $200 \times \texttt{numel}(a_0)$.



Far-Field data collection set-up: Incident wave indicated by arrow. Data is collected along the circle of radius 4. The unknown object is in the center of the figure.

For set-up in which the full far-field data is known, only the intensity of the scattered wave was used in the data. 100 equally spaced points were used to sample the far-field data. In this set-up, a the wavenumber k was set to 5.



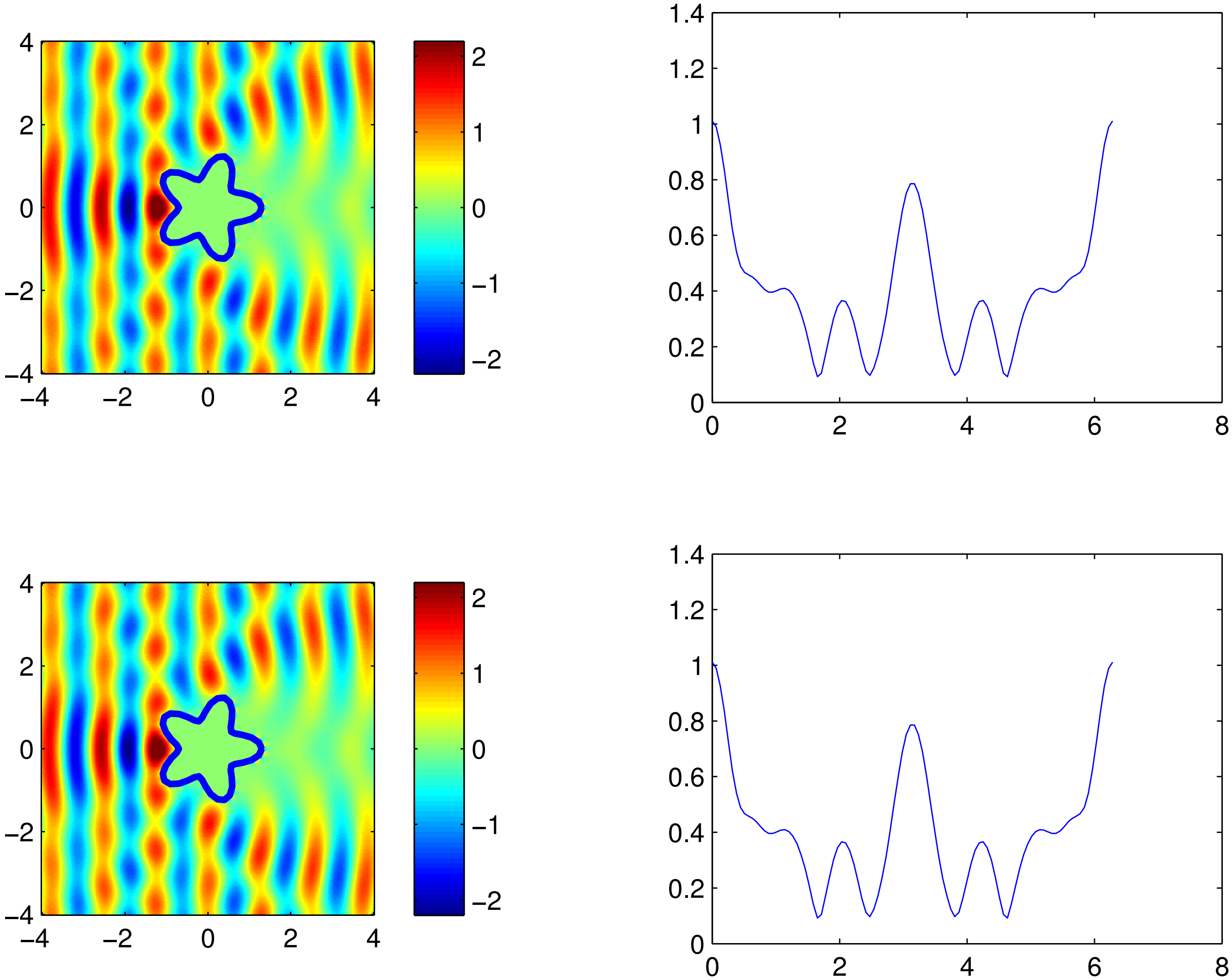
Left: Multiple Node data collection set-up: Incident waves indicated by arrows. Data for each wave is only collected at the node that it originated from. Right: Single Node data collection set-up: Incident wave indicated by arrow. Several incident waves are emitted at different wavenumbers, and the scattered wave is measured at the node.

For both the multiple and single node set-ups, phase as well as intensity information was used for the scattered wave data (if only intensity information was used, the results were not very good). The multiple node set-up used a wavenumber k of 5.

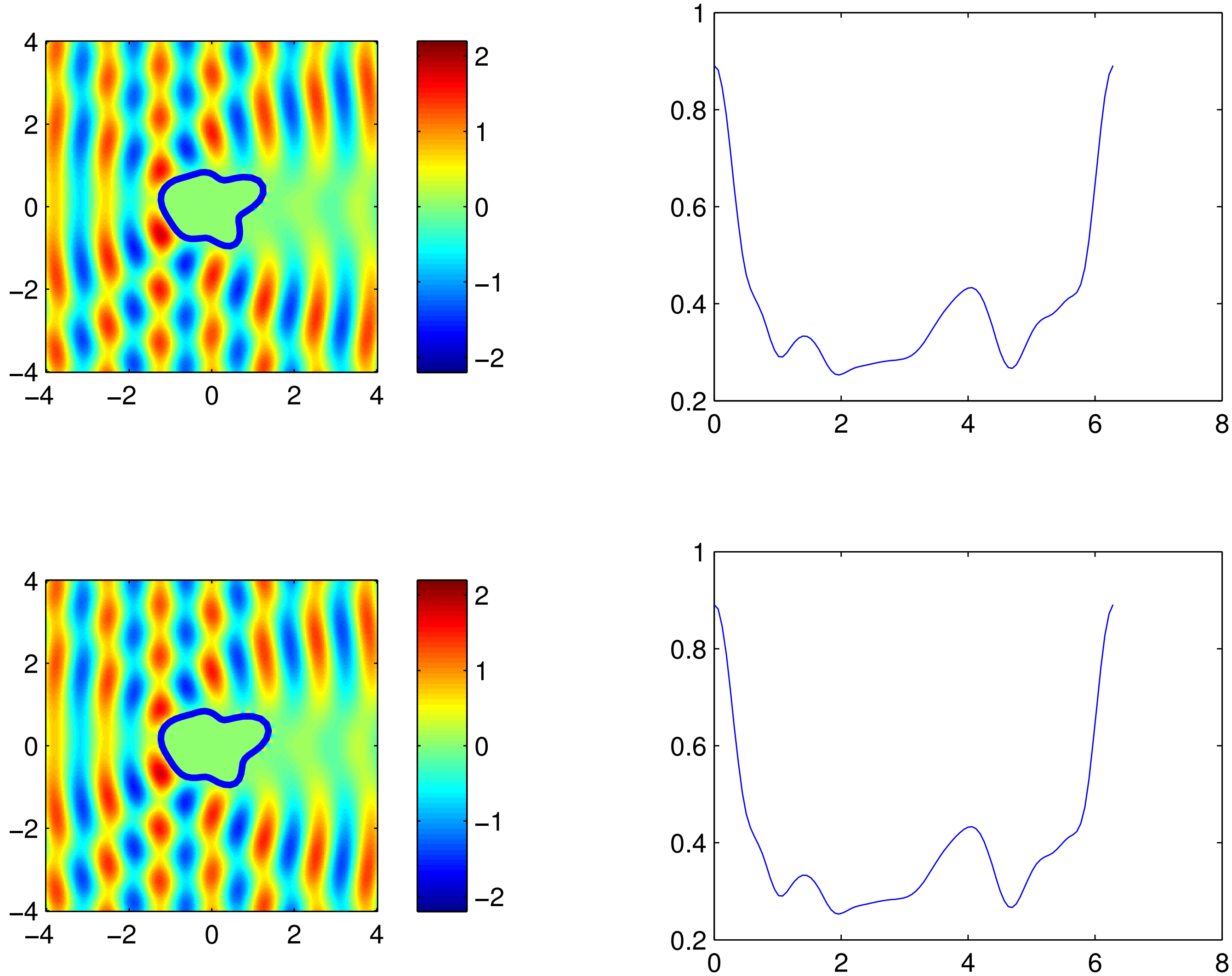
III. RESULTS

A. Full Far-Field Data

In general, this model performed well. $f(a_f) < 0.1$ in most cases, which was very good considering there were 100 data points. This method returned a_fs that were very close to a_i, which can be see visually in the following examples:



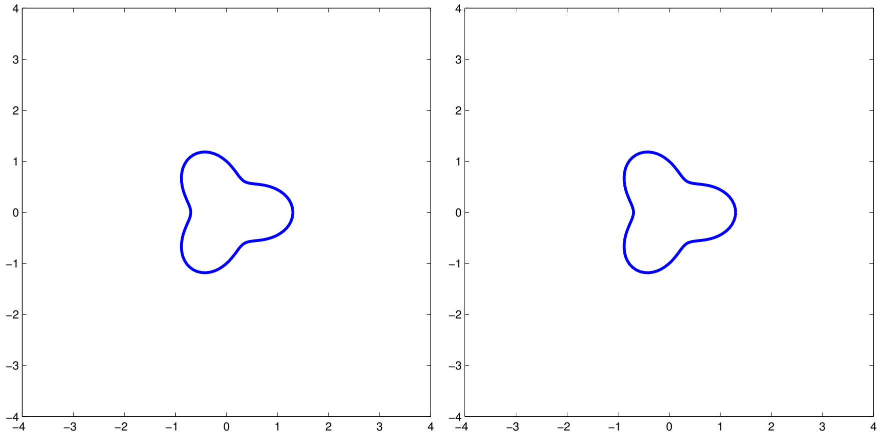
The reconstruction of the shape with $a_0 = 1, a_5 = 0.3$. Bottom left: original shape. Bottom right: original far-field data, d_i. Top left: reconstructed shape. Top right: far-field data of reconstructed shape. $f(a_f) = 0.0229196$



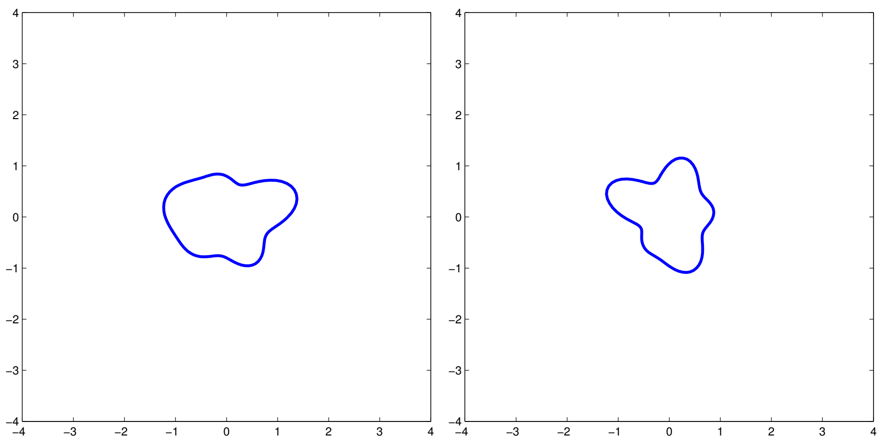
The reconstruction of the shape with $a_0 = 1, a_2 = 0.2, b_3 = b_4 = b_5 = 0.1$. Bottom left: original shape. Bottom right: original far-field data, d_i. Top left: reconstructed shape. Top right: far-field data of reconstructed shape. $f(a_f) = 0.0340535$

B. Multiple Nodes

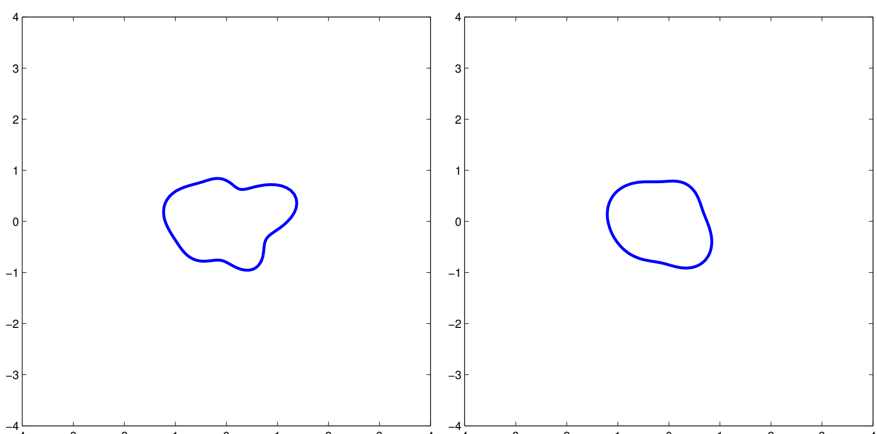
In general, this model performed rather poorly. It was able to figure out the basic shape of the 3rd Fourier mode of the circle, but did not do well with a more random shape. An attempt to increase the accuracy of the solver by increasing the number of nodes from 6 to 50 did not do much more than smooth out the shape. Overall, this method seemed prone to finding non-global minima, although it did at times find the correct shape.



Left: actual shape ($a_0 = 1, a_3 = 0.3$). Right: reconstructed shape using 6 nodes. $f(a_f) = 1.80354 \times 10^{-6}$



Left: actual shape ($a_0 = 1, a_2 = 0.2, b_3 = b_4 = b_5 = 0.1$) Right: reconstructed shape using 6 nodes. $f(a_f) = 0.130949$

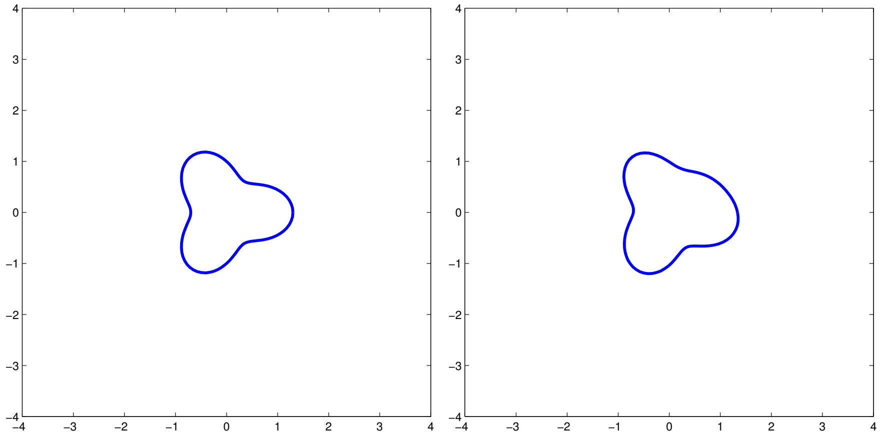


Left: actual shape ($a_0 = 1, a_2 = 0.2, b_3 = b_4 = b_5 = 0.1$) Right: reconstructed shape using 50 nodes. $f(a_f) = 1.71822$

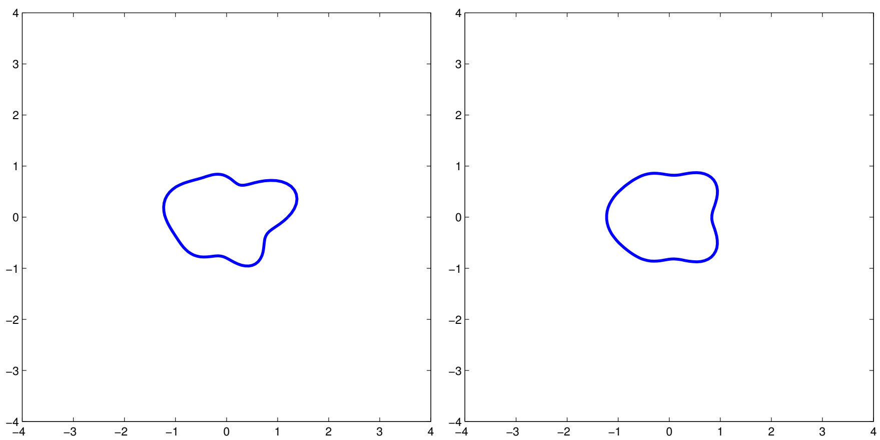
C. Single Node

This solver was performed fairly well considering the amount of data it received. As seen in the first example, it did not do as good a job with the trefoil shape as the previous two methods, but did a better job on the more random shape than the multiple node scenario. One thing to note is that logarithmically spaced k generally performed better than linearly spaced k. However, this could be due to the choice of interval for k, which was $[1, 10]$ in all the below examples, rather than a property of the method.

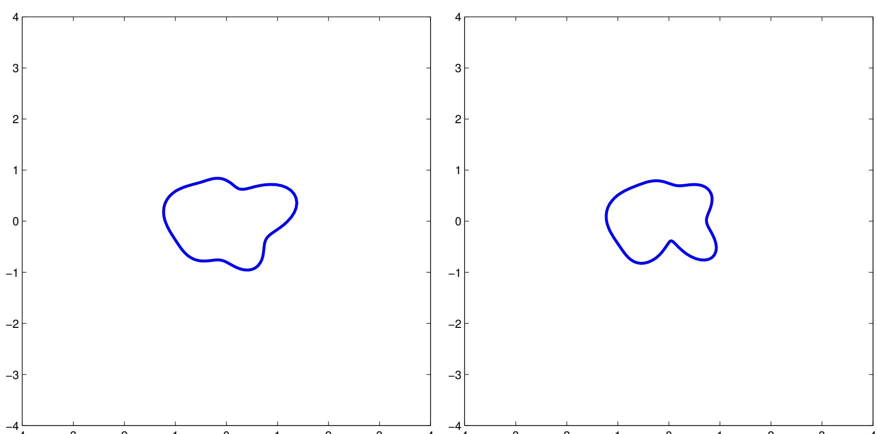
One strength of this method, even if it was not as precise as using the full far-field data, was that it was able to reconstruct the near face of the object quite well. As seen in the examples below, the face of the object that is closest to the wave source/receiver matches the true shape much more closely than the rest of the object. This can especially be seen in the final figure in which 20 different wavelengths are used: the near side of the object looks like it is well-determined, while the far side of the object is not as close a match.



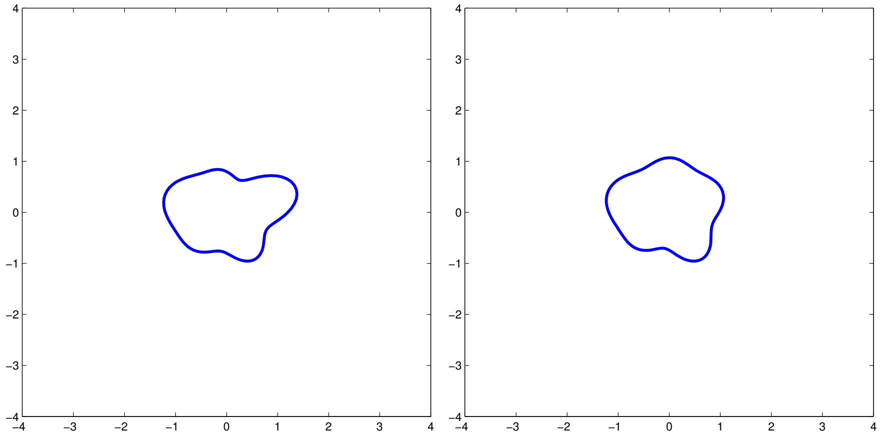
Left: actual shape ($a_0 = 1, a_3 = 0.3$) Right: reconstructed shape using 5 log spaced k between 1 and 10. $f(a_f) = 0.00163156$



Left: actual shape ($a_0 = 1, a_2 = 0.2, b_3 = b_4 = b_5 = 0.1$) Right: reconstructed shape using 5 log spaced k between 1 and 10. $f(a_f) = 0.0272783$



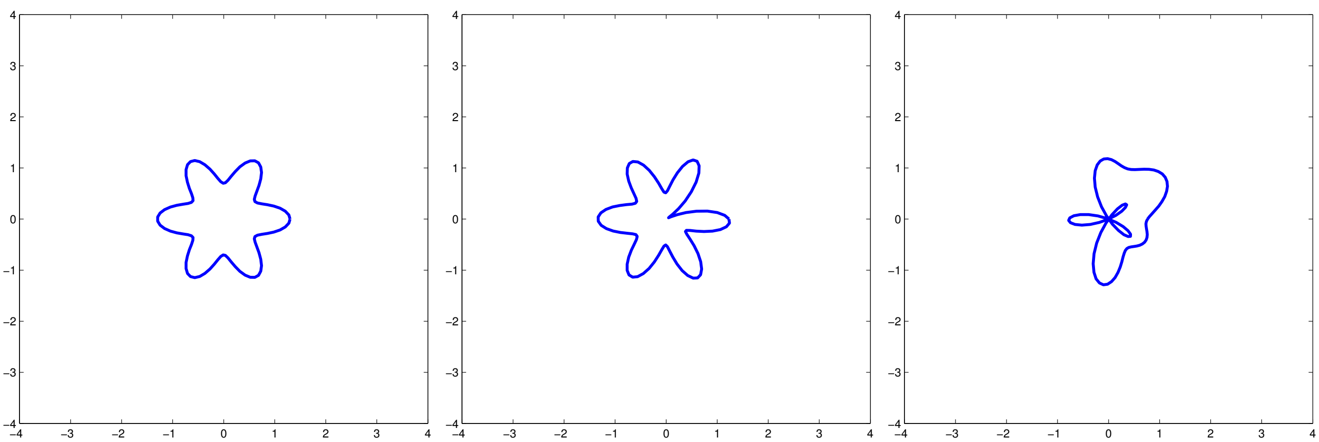
Left: actual shape ($a_0 = 1, a_2 = 0.2, b_3 = b_4 = b_5 = 0.1$) Right: reconstructed shape using 5 linearly spaced k between 1 and 10. $f(a_f) = 0.00609546$



Left: actual shape ($a_0 = 1, a_2 = 0.2, b_3 = b_4 = b_5 = 0.1$) Right: reconstructed shape using 20 log spaced k between 1 and 10. $f(a_f) = 0.0146112$

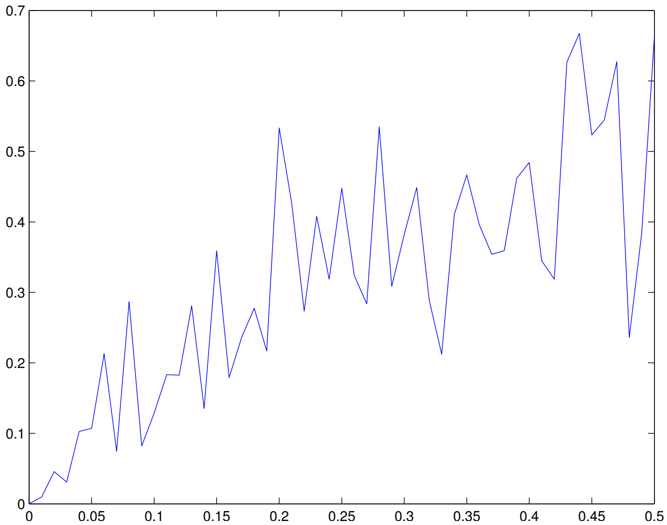
D. Extensions

To test the ability of a solver to cope with realistic problems, two tests were performed. The first tested to see what would happen if the solver was not allowed to use enough coefficients to fully describe the object. The second test added noise to the data to see how closely the coefficients matched. In both cases, the full far-field data was used, as it displayed the most accurate performance to begin with.



Left: actual shape ($a_0 = 1, a_6 = 0.3$) Center: reconstructed shape allowing coefficients up to a_6, b_6. $f(a_f) = 0.0863037$. Right: reconstructed shape allowing coefficients up to a_5, b_5. $f(a_f) = 0.454935$.

As seen above, the solver was able to come up with a reasonable approximation if enough coefficients were allowed ($n = 6$). However, when not enough coefficients were allowed ($n = 5$), the solver came up with a pretty bad answer. Note how the second reconstructed shape has a negative radius, which is not physically realistic. However, if a positive radius were enforced, the result would probably match the far field data even more poorly, as the above result is a minimum.



$f(a_f)$ after introducing noise to the data generated by a_i with $a_0 = 1, a_3 = 0.3$. Noise is multiplicative, uniformly distributed between 0 and the upper limit, which varies from 0 to 50% on this plot.

To test the ability of the solver to cope with noise, random noise was added to the far-field data. The noise was uniformly distributed within an allowed range (0-50% of the data value in the above figure), and was multiplicative. As seen above, the coefficients matched fairly well for small amounts of noise (as expected), but diverged as the noise increased. One interesting aspect of the noise is that certain random noise sets has less effect than others on the final result, as seen by the jagged nature of the line.

IV. CONCLUSIONS

Overall, knowing the full far-field data was the best scenario tested in this project. It was consistently able to find a good approximation to the actual shape. Using a single node that emitted waves at different wavelengths was able to determine the near side of the unknown object fairly well, while the far side remained a bit more variable. Using multiple nodes at the same wavelength was the worst-performing scenario, which was not remediated by adding more nodes.

It was found that not using enough coefficients to describe the unknown object can cause the solver to arrive at a drastically different-looking shape than the actual shape. This indicates that care must be taken to include enough Fourier coefficients in the reconstruction of a shape, or the results will be inaccurate. Adding noise to the data gave the expected results: just a little noise did not affect the coefficient vectors to a large degree, but the solver had more and more trouble at approximating the actual shape as the noise increased. One possible method to improve the model's ability to cope with noise could be to fit a Fourier series to the data. Since the far-field data takes on a fairly smooth shape (as it is a wave's cross section), this could improve results by removing any jaggedness that random noise adds.

Probably the best way to improve the accuracy of the results would be to start with a better guess than a circle. This could randomly choosing a certain number of different sets of coefficients and see which is closer to start with. A more sophisticated method could involve categorizing different far-field patterns so that sets of coefficients that produce similar patterns are all tried if data falls into a certain category. This could be done either with manual input or some implementation of a machine learning algorithm.

Acknowledgements

Thanks to Professor Alex Barnett for his helpful suggestions concerning this subject, as well as for all the guidance he provided throughout Math 126.

[1] Ghosh Roy, D.N., Couchman, L.S. *Inverse Problems and Inverse Scattering of Plane Waves* Academic Press: San Diego. 2002.

[2] Colton, D., Kress, R. *Inverse Acoustic and Electromagnetic Scattering Theory*. Springer-Verlag: Berlin. 1992.