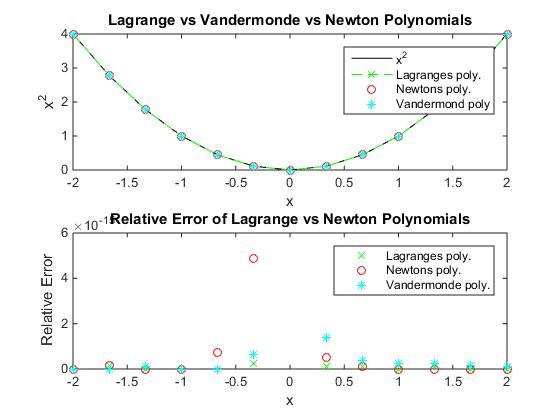
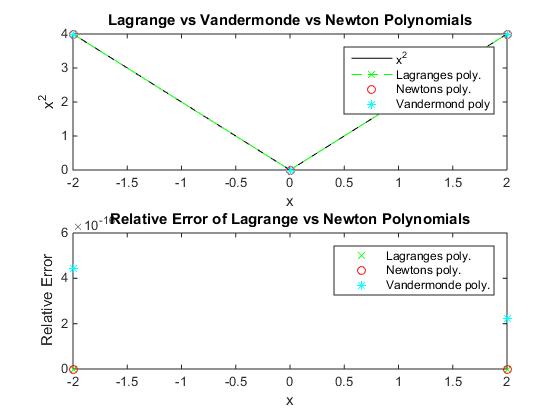
Andrew Rozniakowski

5/5/2016

448 Final

1. To test my Newtons method for solving nonlinear 2x2 systems of equations I used 2 test systems. To be able to test my order and convergence rates I wrote a program in Matlab that can be found in the files turned in, named “Rate\_Order\_Convg\_mod”. This program takes the error from my Newtons method and uses polyfit to determine the order and convergence rates. The first system I tested my program on was f = x^2 + y^2 – 10 and g = 2\*x + y -1. My program converged to the solution (-1,3) with an order of 1.938 for both x and y and a rate of .256 for x and .134 for y. For my second test I used the system f = x^2 – 2\*y^2 + 2 and g = y\*x + 2. In this result my program returned the solution (-1.41, 1.41). In this situation I received the order 1.918 and a rate of .299 for x and an order of 1.97 and a rate of .47 for y. In both my test cases I was able to get an order very close to 2, which makes me believe my Newtons program is able to converge to the correct roots at a quadratic rate.
2. To compare the Vandermonde, Newton and Lagrange methods of interpolation I chose to plot the function x^2 using each of them. For each method I used an order 4 polynomial. In the first case I 13 equally spaced nodes and got the result below.

We can see from the top graph that all 3 of our methods gives us a pretty good representation of our function x^2. By looking at the error in the second graph we can see which is better. We get the least amount of error from our Lagrange polynomial while Newtons is the worst. At some points our Vandermonde method is worse than Newtons but for the most part its better. Next we look at only 3 points.



Looking at only 3 points we once again see all three of our methods give us a good representation of x^2. We can see from the error graph that our Vandermonde method gives us the most error. All the error comes at the end points, where we can see both our Newton and Lagrange method gives us next to no error, while our Vandermonde is giving us some. Looking at our two different test cases vs. each other, we had less error when modeling fewer point, however the methods that gave us the most error changed. It seems as though the Vandermonde method struggles the most with fewer points and Newtons struggles with more points. Newtons and Lagrange are also better at finding the end points than the middle points.

1. The problem with numerical differentiation is the round off error. This can accumulate throughout the process and cause our differentiation to be off. As an example I chose to use the function sin(x) to approximate the derivative cos(x) numerically. Using the program I wrote, which used the two point formula, my error was between .0429 and .296\*10^-9 depending on the value of h. Next I wrote a method for numerically solving the derivative using Richardsons extrapolation. In this case my error was between .0009 and .111\*10^-11. As we can see from our results Richardson’s method is much better but we still end up with some error.