Math 451 PROJECT 1 **Due at the start of class on Monday, April 21st**

Introduction

The purpose of this project is to compare results of bisection method with those of Newton's method, and to see some problems that can come up with Newton's method. You will need codes for both of those methods. You should have developed them on your own in Assignments 4 and 6, but I am providing basic codes here so that you have something to use even if you had trouble with one or both of those assignments. *Neither code has any output, so you will need to add lines for the desired output(s), and you will need to do some other modifications as well.*

Newton's Method

```
clear all;
f= @(x) function;
df= @(x) derivative of function;
x(1)= x_1;
k=1;
while stopping condition and safety
   x(k+1)=x(k)-f(x(k))/df(x(k));
   k=k+1;
end
```

Bisection Method

```
clear all;
f= @(x) function;
a(1)= a_1;
b(1)= b_1;
k=1;
while stopping condition and safety
   m=(a(k)+b(k))/2;
   if f(a(k))*f(m)>0
      a(k+1)=m;
      b(k+1)=b(k);
   else
      a(k+1)=a(k);
      b(k+1)=m;
   end
   k=k+1;
end
```

You will create a script that will include the above codes multiple times. You do not need to comment any of the code unless you wish to, but you will eventually publish your results as a Word document (or some other type if you wish) using the *Publish* command, and then you will add discussion of your results. There is a sample at the class web page of the format your final document should be in. The following describes what should be in the various sections of your paper.

Rate of Convergence

Suppose that we use two methods of approximating a root, and Method A gives a satisfactory approximation after 7 iterations and Method B after 12 iterations. Then we will say that Method A converges $\frac{12}{7} \approx 1.7$ times faster than Method B. In this portion of your paper you will compare the rates of convergence of the bisection method and Newton's method. Here are some things you need to do:

- Note that when we use Newton's method we create a single sequence of numbers that approaches the desired root, but with the bisection method we create two sequences. Modify the above code for the bisection method (or use your own) in order to create a third sequence m_k of midpoints of the intervals $[a_k, b_k]$. You will need to add a line to do it once before entering the while loop, and you'll move the one in the while loop to compute m_{k+1} right after a_{k+1} and b_{k+1} have been computed. Test that your changes work by using a simple stopping condition like `k<6`.
- Throughout the project you are going to use a new stopping condition. Note that we are trying to solve $f(x) = 0$, so we'll stop when $f(x_k)$ (for Newton's method) or $f(m_k)$ (for the bisection method) is sufficiently close to zero.
- We will be using the function $f(x) = x^2 - 4$, which has two roots that we know. Use the bisection method with a first interval of $[1, 4]$ and Newton's method with $x_1 = 4$ to approximate the positive root of the equation to within 10^{-8} using the stopping condition just described. Add a `format long` command in the appropriate place so that we can see the sequence values to enough decimal places.

Save your file as something. Later you will combine its code with the code for the following investigations to create a single script file that performs all the calculations for the entire project.

Error

When we know the actual value that we are trying to approximate, we can calculate the error of our approximations. (Many numerical methods have ways in which we can determine the maximum error we can expect even though we don't know the actual value of what we are approximating!) There are two commonly used ways to measure error, **absolute error** and **relative error**:

$$absolute\ error = |approximate\ value - actual\ value| \qquad relative\ error = \frac{|approximate\ value - actual\ value|}{|actual\ value|}$$

In your investigation of the errors using the bisection method and Newton's method you will use the absolute error. You should create a new script file for this part of the project, and combine it with your other script files later. Here is what you need to do:

- Run eight iterations each of the bisection method and Newton's method for the same function $f(x) = x^2 - 4$, with the same starting interval and starting value as you did earlier in your investigation of the rate of convergence. To do this you can just change your stopping condition or you can change your loops to `for` loops. Modify your script file to compute a sequence e_k of absolute errors for each method. (Remember that you can, and should, name each sequence with a more suggestive name that indicates what it is.) Plot the two sequences *of errors* on the same graph, and include a legend that tells which is which.

- As expected, the sequence of errors for Newton's method gets small much faster than the sequence for the bisection method. For Newton's method it turns out that $e_{k+1} \approx ce_k^2$ for some constant c. Solving this for c gives us a sequence of constants $c_k = \dfrac{e_{k+1}}{e_k^2}$. Modify your code to create this sequence of constants and output it *after* the plot of errors.

- Repeat the above process for the bisection method, with the relationship $e_{k+1} \approx ce_k$. You can output this sequence before or after the sequence for Newton's method but, again, output it after the plot.

Instability of Newton's Method

If a function has a root in an interval, the bisection method will find it. (If there is more than one root in the starting interval, it may or may not find the particular root we want.) As you will see, Newton's method is not so predictable. In this part of the project you will be working with the function

$$f(x) = x^3 - 3x^2 + x + 3.$$

Hopefully you know enough about the appearance of the graph of a third degree polynomial to recognize that this function must have at least one root. You should create yet another new file to do the following.

- Attempt to use Newton's method with $x_1 = 1$ to find a root of the function. Modify your code to give an output that adequately shows what happens. Then create a plot that has the function and the first and second tangent lines for Newton's method all on the same grid. **Use an appropriate scale to "show some shape" in the graph of the function and to focus on the area of interest, including the root**. Again, label the axes and include a legend to distinguish which of the three graphs is which. Use the result of the code and the plot to support the discussion that you will add later. **You may need to revisit the Plotting Activity to see how to do some of this.**

- Try starting values of $x_1 = 0.44$ and $x_1 = 0.45$ and see what happens. You might try to create a graph with tangent lines that shows why this happens.

- Based on your graph, speculate on a range of starting points x_1 for which the sequence generated by Newton's method should converge to the root, and test a few values to see if your conjecture seems to hold.

The Final Product

You now have the "data" for creating your report.

- Look at the sample format I posted at the class web page to see how your report should be organized, using the *MATLAB Publish* tool.

- Combine your script files, in the order of the topics given above and in the sample format, into a single script file. Make sure it runs.

- Use double percents at each point in your script where a new section begins, and add the titles to the sections. Add the main title and other information shown in the sample. Add a conclusion section at the end.

- Use single percents to add breaks where you will wish to insert discussion later. You might just add a line of comment in each of those places, in the manner I did in the sample. After you publish you can go back and change those comments as you wish, so don't worry too much about what they say initially.

- Publish to Word (or some other format if you wish). Then go back and flesh out the paper with ample discussion in each section.

- DO NOT print your paper immediately (unless you are up against the deadline, of course). Take a day off from it, and then go back and read it carefully to see if it does what it is supposed to and if all of your discussion still makes sense to you! Look at it in the context of the grading scheme outlined below. Once you are satisfied, print it out and bring it to class on Monday, and e-mail me an electronic version as well. **You may print it in black and white rather than color if you wish. I'll check your legends and color coding on the electronic copy.**

Grading Criteria

- **Timeliness:** (*5 points*) Papers turned in by the due date and time will receive five points for this, those turned in by 4 PM that day will receive zero points, and *those turned in after that will receive at most half credit for the entire paper.*

- **Organization:** (*5 points*) Your paper should follow the format given in the sample. There should be breaks before outputs and plots, with a description before each of them of what they are. Discussion should not occur as commenting in the script file, but rather as text added to the published file. Spacing should be used where appropriate.

- **Appropriate Outputs:** (*5 points*) All sequences and plots asked for should be included, and there should be no extra outputs.

- **Graphics:** (*5 points*) All graphs should have both axes labeled, a title, and a legend if there is more than one plot on the same grid. Use gridlines and solid line axes when plotting graphs of functions and tangent lines.

- **Quality of Writing:** (*5 points*) Punctuation, spelling and grammar should be correct. Sentences should have good structure, and discussion should flow in a logical manner.

- **Adequacy and Correctness of Discussions:** (*5 points* $\times$ *4*) The discussion in each section should adequately address all issues investigated in that section. Correct terminology should be used, and results should be interpreted correctly.