Programming Project PART 1
CS 323 - Numerical Analysis

1 Introduction

This project will be divided in two parts, the first part (this handout) contains the first algorithms that you must implement from the ones we have covered in class. The second part will be assigned after we have covered in class the remaining algorithms that you will need to implement.

For this project you will write matlab code for several of the algorithms covered during this course. You will be given a collection of test cases for each one of your programs, be aware that some might contain "pathological" examples and you must modify your algorithms to take care of those situations.

The test cases will be posted on Sakai.

2 Individual work

For this project you will work individually. You must turn in your own work and all your programs must run.

Please start working as soon as possible. Try your programs with your own test cases if test cases for that particular algorithm have not yet been posted on Sakai.

3 About the programs

3.1 Programming Language

The programs must be written in Matlab (m files). When a function is required as input, they must take the function as a string. and then use the *feval* matlab function to evaluate the function. This means that in the case of algorithms that require the computation of a derivative $f'(x)$, you should compute the derivative by hand and give it as input to the program.

3.2 Style

All your programs must be correctly indented and must include comments (use % in matlab) describing the major parts of the algorithm. If you use any other variables that are not the usual ones described in the lecture notes, please explain in your comments the meaning of each one of those variables.

4 Algorithms

The following is the list of all the algorithms that you must implement:

1. **Horner:** Given the coefficients of a polynomial $a_0, a_1, \ldots a_n$, and a real number x_0, find $P(x_0)$ and $P'(x_0)$.
   Sample input representing $P(x) = 2 + 3x - x^2 + 2x^3, \quad x_0 = 3.5$:

   ```
   3
   2
   3
   -1
   2
   3.5
   ```

   the first number is the degree of the polynomial, the coefficients are in the order $a_0, a_1, \ldots, a_n$, the las number is x_0.

2. **Newton Raphson with Horner:** Given the coefficients of a polynomial $a_0, a_1, \ldots a_n$, and an initial guess $x_0 \in \mathbb{R}$, find **one root** of the polynomial. Input format is the same as above.
   IMPORTANT: NO CREDIT WILL BE GIVEN IS THE METHOD DOES NOT USE HORNER'S METHOD

3. **Cramer's Rule:** Given a matrix A representing a system of equations, and a vector b of independent terms, use Gaussian Elimination to find all the determinants needed to solve the system. The program must output the value of each of the $n + 1$ determinants as well as the solution $x_1, x_2, \ldots, x_n$.
   Sample input representing the system

   $$\begin{pmatrix} 3 & 4 & 2 \\ -1 & 3 & -4 \\ 2 & 2 & 5 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 5 \\ 2 \\ -6 \end{pmatrix}$$

   is:

   ```
   3
   3
   4
   2
   -1
   3
   -4
   2
   2
   5
   ```

```
5
2
-6
```

The expected output should be:

```
determinant A = 41
determinant A1= 215
determinant A2= -53
determinant A3= -114
x1 =  5.24390
x2 = -1.29268
x3 = -2.78049
```

4. **Neville's Method:** Given a set of points $(x_0, y_0), \ldots (x_n, y_n)$, and a real number x_0, use Neville's Algorithm to find $P_n(x_0)$ where $P_n(x)$ is the unique polynomial that goes through all of the given points.
   Sample input representing the points $(-1, 1), (0, 0), (1, 1)$ and $x_0 = 0.5$:

   ```
   2
   -1
   1
   0
   0
   1
   1
   ```

   where the first number is n.

   The expected output should be:

   ```
   P(x_0)=0.25
   ```