EE407002 Numerical Analysis

Project. Polynomial Roots Finder.

Due: June 7, 2017

Given a polynomial of degree n as the following

$$P_n(x) = a_n x^n + a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_1 x + a_0 \tag{1.1}$$

with all coefficients real, $a_i \in \mathbb{R}$, $i = 0, \cdots, n$. It is known that there are n roots, x_i, $i = 1, \cdots n$, for this polynomial, however, not all roots are real. It is also known that if a root $x_k = z \in \mathbb{C}$ then its complex conjugate $\overline{z}$ is also a root of the polynomial, $P(\overline{z}) = 0$.

For this project, you are required to write a program that can find all roots given a polynomial. Three different methods have been discussed in the class.

1. Newton-Horner method, Algorithm (7.2.5). However, to be able to handle complex roots, all the arithmetic operations should be modified to be able to handle complex numbers.
2. Lin's quadric method. Using Equations (7.2.25) and (7.2.26), one can find a quadratic factor of $P_n(x)$. Finding the roots of the quadratic factor is straightforward. Once those two roots are found, the deflation process can be continued and the quadratic method repeated to find all roots.
3. Bairstow's method, Algorithm (7.3.4). As discussed, this is also a Newton's method applied to find a quadratic factor of $P_n(x)$.

Of course, there are other methods available and not discussed in our class. You are free to use any method (or combination of different methods) as long as it can solve all polynomials, $P_n(x)$, automatically. As such, your program should be able to set up initial guess, x_0, to enable convergent Newton iterations, if it is needed.

As we have done for the whole semester, you need to turn in a `C++` program and a report. All source and header files for your program need to be turned in. For example, if any of the `VEC` functions is used, then `VEC.h` and `VEC.cpp` files should be turned in. The source file containing the `main` function should be named `proj.cpp`.

The report, name `proja.pdf`, should document the following:

1. The goal of the program.
2. The approach (or approaches) you take.
   Your algorithm should be described clearly. Since the initial guess can play an important role for the iterative process, you should document clearly how are they set up.
3. Compared to those known methods, why is your approach better?
   Quantitative comparisons are usually more persuasive than qualitative descriptions.

4. Conclusions.

To facilitate your development, a set of polynomials have also been included. They are `ta1.dat` to `tf5.dat`, totally 30 polynomials. The first line of these files is the degree, n, of the polynomial, followed by n lines of the coefficients and x^k terms. Your program should be able to handle this kind of polynomial inputs and writes out its roots, real and complex. No specific order is required for the root output.

Notes.

1. Submit your files on EE workstations. Please use the following command to submit.

   ```
   $ ~ee407002/bin/submit hw16 proja.pdf proj.cpp MAT.h MAT.cpp VEC.h VEC.cpp
   ```

   where `hw16` indicates that it is for the project assignment.

2. Your report should be clearly written such that I can understand it. The writing, including English grammar, is part of the grading criteria.