Homework/Programming 1. (Due Aug. 30)

Math. 609-600

This homework involves developing a simple matlab (or python) script for implementing the bisection method for computing the roots of a fourth order polynomial. This week's recitation went over a matlab script for implementing Newton's iteration.

The mathematical algorithm corresponding to the bisection method is described in Section 17.2 of the 609d lecture notes and your code should be based on it.

We consider the polynomial

$$P(x) = 48x^4 + 68x^3 - 28x^2 - 27x + 9. \tag{0.1}$$

This polynomial can be implemented in MATLAB as an "anonymous" matlab function:

```
fun=@(x) 9+x*(-27+x*(-28+x*(68+x*48))).
```

so that $fun(x)$ evaluates $P(x)$. This definition should appear as the first statement in your script.

The above form of the polynomial $P(x)$ is more efficient than the natural presentation in (0.1) because it contains only 4 multiplications while (0.1) appears to take 10.

Note that the bisection routine depends on n,a,b,fun where n is the desired number of steps (or iterations), a and b are the initial interval endpoints and fun is the function. The algorithm does not make any sense unless $f(a) * f(b) < 0$ so whatever values you start with must satisfy this condition.

The values of a and b are to be set before the loop executing the iteration. fa:=fun(a) and fb:=fun(b) are also computed before the iteration. As in this week's recitation example, you must reuse the storage locations for a, b, fa and fb inside the iteration loop. To do this, inside the loop, define mid=(a+b)/2, fmid=fun(fmid). If fmid=0 then mid is the root (stop and print). Otherwise, the sign of fa*fmid dictates how a,b,fa, and fb have to be updated. Note that you will need only one fun evaluation per iterative step (inside the loop). It is important not to use arrays to store a,b,fa,fb,mid or fmid.

With good graphical information, it is easy to find intervals which bracket the 4 roots of $P(x)$, namely, [-2,-1], [-1,0], [0,.4], [.4,1].

Problem 1. *Write a matlab or python script implementing the bisection algorithm. Using the anonymous function "fun" and the intervals bracketing the roots given above run the bisection iteration with $n = 20$ and $n = 30$ iterations for each root. The approximation for the root after n iterations is given by*

$$x^*_{app,n} = (b + a)/2$$

and satisfies

$$|x^* - x^*_{app,n}| \leq \epsilon_n := (b - a)/2^n.$$

For each root and n, report $x^*_{app,n}$ and ϵ_n.

Problem 2. *Redefine fun so that it computes*

$$fun(x) = sin(x)/x - .5$$

From calculus, you know that the limit of $fun(x)$ as $x \to 0$ is $.5$ so there is a root of $fun(x) = 0$ in the interval $[0, \pi]$. However, to avoid division by zero, we will use the interval $[.001, \pi]$. Use you bisection code with this interval to approximate a root of $fun(x) = 0$ using $n = 5, 10, 15, 20$ and report your results.

Remark: I do not know how to solve the above problem without iteration.

Hand in the results for the above problems a copy of your code.