Assignment1

In this assignment, our aim is to find the roots of the giving function(which only contains 1 variable). Basically, we have two kinds of methods to solve this, one is Bisection method and the other one is Newton-Raphson method. Here are the functions that I'm going to solve.

a)$\sqrt{5}$;

b) y=$(5e^{-x}-1)/(1-e^{-2x})$;
We need to define them as functions:

a) $g(x)=\sqrt{5}-x$;

b) $f(x)=(5e^{-x}-1)/(1-e^{-2x})$;

1. Bisection method

For the bisection method, we firstly guess a range [a,b] where the root could exist in. Next, we verify it by defining f(a)*f(b)<0 or not. If it is, we print "cannot find root between this range", otherwise, we define c as the midpoint of a and b. Then, check whether f(a)*f(c)<0. If it is, that proves the root is in [a,c], and we make b equals to c. If not, the root lies in [c,b], we define a as c. After finite iterations, we can find the root until a approaches to b.
Here is the source:

```
function root=bisection(f,a,b,e)
 c=0;
   if f(a)*f(b)>0
          disp ('cannot find any roots')
       else
          while abs(b-a)>e
              c=(b+a)/2;
              if f(a)*f(c)>0
                  a=c;
              elseif f(a)*f(c)<0
                  b=c;
              elseif f(c)==0
                  a=c;
                  b=c;
              end
          end
          root=c;
fprintf (1,'%g\n',c)
       end
   end
```

for question 1, we call that function by typing "bisection(@g,1,3,1e-5)". Here comes the result:
>> bisection(@g,1,3,1e-5)

2.23606

ans =

2.2361

For question 2, we type “bisection(@f,1,4,1e-5)”

```
>> bisection(@f,1,4,1e-5)
1.60944

ans =

    1.6094
```

2. Newton-Raphson method

In Newton’s method, we firstly input a point (a,f(x)), then using Taylor’s expansion:

$$f(x_{i+1}) = f(x_i) + \Delta x \frac{df(x_i)}{dx}$$

We have to solve the 1st order derivative by approximation. So, we define dx as the tolerance which is an input variable and dy equals to f(dx). After that, we iterate subroutines above to approach f(x)<tol.

```
function findzero=newton(f,a,e,n)
%a  first point(x)
for i=1:n
    x=a;
    dx=e;
    y=f(x);
    dy=f(x+dx)-f(x);
    k=((y+dy)-y)/((x+dx)-x);
    if abs(y)<e
        break
    elseif k*y<0
        a=a+abs(y/k);
    elseif k*y>0
       a=a-abs(y/k);
    end
    if a<=0
        disp ('cannot find root,please change a')
        break
    end
end
 fprintf (1,'%g\n',a)
end
```

For question1, we call this function by typing “newton(@g,1,1e-5,1000)”, the last variable is

iteration time.
>> newton(@g,1,1e-5,1000)
2.23607
For question2, here is the result:
>> newton(@f,1,1e-5,1000)
1.60944

3. Comparison

By comparing two methods, we can absolutely see the answer is same on the condition of inputting considerably small accuracy. But if we input slightly bigger accuracy, the results are different.
>> bisection(@f,1,3,1e-3)
1.61035;
>> newton(@f,1,1e-3,1000)
1.60909;
As we can see, eps equals 1e-3 and we solve the same function (question2 for example). It seems Newton-Raphson method is more accurate than Bisection method. And we testify it by question1.
>> bisection(@g,1,3,1e-3)
2.23535;
>> newton(@g,1,1e-3,1000)
2.23607;
Obviously, Newton's method seems better than bisection method as we didn't input enough accuracy. So we'd better solve a one-variable problem by using Newton-Raphson method.