HW-2

1. Taylor's polynomial approximating a function is defined as follows

$$P_n(x) = f(a) + \sum_{j=1}^{n} \frac{f^{(j)}(a)(x-a)^j}{j!}$$

$$R_n(x) = \frac{(x-a)^{n+1}}{(n+1)!} f^{(j)}(\mu), \qquad \alpha \le x \le \beta\,, and\ a \le \mu \le x.$$

$$f(x) = P_n(x) + R_n(x).$$

   i. The tangent line at the point x_0 is the first degree Taylor's polynomial $P_1(x) = f(x_0) + f'(x_0)(x - x_0)$ that has as a root the next point $x_1, i.e. P_1(x_1) = 0,$. First Derive Newton's method

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, \qquad n = 0, 1, 2, \ldots$$

   using the above assumption and then show that the error is given by:

$$|x_{n+1} - p| = |x_n - p|^2\, M_n, \quad M_n = |f''(\sigma_n)|/|2f'(x_n)|$$

   ii. Use the Newton program attached to solve the following equations

   a. $x^2 - 2 = 0$. This method derives the square root of 2! Use the above error to show that the method converges for every $x_0 \neq 0$. Verify your results by running the program. Modify the program so that you print the results in a file as you did in your first HW-1.

   b. $1 - e^x = 0$. Find the solution using Newton's program. Does it converge for every initial starting point?

2. Find all solutions of the $f(x) = x^2 - \sin(x) - 0.5$ using bisection and newton's method. Present the results for all steps along with error in each step. For error use either the bracket for bisection, the evaluation of the function , and the consecutive steps difference for Newton's. Use the stopping criterion $eps \le 10^{-8}$. Provide an explanation of the errors and the number of steps taken by each method using our theoretical results for the error behavior for each method(e.g. the error for bisection reduces by half in each step.).
$|p - x_n| \le |b - a|/2^{n+1}$ for bisection and $|p - x_n| \cong |p - x_{n-1}|^2\ |f''(x_{n-1})/(2f'(x_{n-1})\ |$for Newton's.

```
function [root] = newtongraphics(x0,error_bd,max_iterate)
   syms x real;
   syms z real;
   %Input function
   f = x^2-2;
   %Taylor's polynomial of degree 1, Tangent line.
   p=z^2-2+2*z*(x-z);


format short e
error = 1;
it_count = 0;
iteration=[];
```

HW-2

```
while abs(error) > error_bd && it_count <= max_iterate
   grid
   ezplot(f,[0,2])
   hold on
   ezplot(subs(p,'z',x0),[0,2])
   grid
   fx =  subs(f,'x',x0);
   dfx = subs(diff(f,x,1),'x',x0);
   if dfx == 0
      disp('The derivative is zero.  Stop')
      return
   end
   x1 = x0 - fx/dfx;
   error = abs(x1 - x0);
%   Internal print of newton method. Tap the carriage
%   return key to continue the computation.
disp('it_count x0 fx dfx error')
   iteration = [iteration
      it_count x0 fx dfx error];
   disp(double(iteration))


   pause
   x0 = x1;
   it_count = it_count + 1;
end

if it_count > max_iterate
   disp('The number of iterates calculated exceeded')
   disp('max_iterate.  An accurate root was not')
   disp('calculated.')
else
   root =iteration;

end
```

```
function root=bisect(a0,b0,ep,max_iterate,index_f)
%
% function bisect(a0,b0,ep,max_iterate,index_f)
%
% This is the bisection method for solving an equation f(x)=0.
%
% The function f is defined below by the user. The function f is
% to be continuous on the interval [a0,b0], and it is to be of
% opposite signs at a0 and b0.  The quantity ep is the error
% tolerance.  The routine guarantees this as an error bound
% provided: (1) the restrictions on the initial interval are
% correct, and (2) ep is not too small when the machine epsilon
% is taken into account.  Most of these conditions are not
% checked in the program!  The parameter max_iterate is an upper
% limit on the number of iterates to be computed.
%
% For the given function f(x), an example of a calling sequence
% might be the following:
%    root = bisect(1,1.5,1.0E-6,10,1)
% The parameter index_f specifies the function to be used.
%
% The following will print out for each iteration the values of
%      count, a, b, c, f(c), (b-a)/2
% with c the current iterate and (b-a)/2 the error bound for c.
% The variable count is the index of the current interate.  Tap
% the carriage return key to continue with the iteration.
```

HW-2

```
if a0 >= b0
    disp('a0 < b0 is not true.  Stop!')
    return
end

format short e
a = a0; b = b0;
fa = f(a,index_f); fb = f(b,index_f);

if sign(fa)*sign(fb) > 0
    disp('f(a0) and f(b0) are of the same sign.  Stop!')
    return
end

c = (a+b)/2;
it_count = 0;
while b-c > ep && it_count < max_iterate
    it_count = it_count + 1;
    fc = f(c,index_f);
%   Internal print of bisection method. Tap the carriage
%   return key to continue the computation.
    iteration = [it_count a b c fc b-c]
    if sign(fb)*sign(fc) <= 0
        a = c;
        fa = fc;
    else
        b = c;
        fb = fc;
    end
    c = (a+b)/2;
    pause
end

format long
root = c
format short e
error_bound = b-c
format short
it_count

%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function value = f(x,index)

% function to define equation for rootfinding problem.

switch index
case 1
    value = x.^6 - x - 1;
case 2
    value = x - exp(-x);
case 3
    value= x^2-1;
case 4
    value=x-1-.5*sin(x);
case 5
    value=x-3-2*sin(x);
end
```

3. Write Matlab program that implements the fixed point iteration method $x_{(k+1)}=g(x_k)$ k=1,2,… with a stopping criterion, the $|x_k-x_{(k-1)}|<tol$, and a max number of iterations kmax. Then solve the equation $f(x)=x^2-3x+2$ by using the following iteration functions.
   a. Newton's method for $g(x)=x-f(x)/f'(x)$
   b. $g(x)=(x^2+2)/3$
   c. $g(x)=\sqrt{3x-2}$

HW-2

d. $g(x)=3-2/x$
e. $g(x)=(x^2-2)/(2x-3)$

Select the initial points by using ezplot to first plot f(x) and identify the regions for the root.

Do these methods converge, if YES or NOT provide an explanation. If they converge identify the interval of convergence, i.e. any initial starting point in that region will converge to a root.

What are the rates of convergence for each method? If they converge?

4. We would like to solve $e^{-x}= x$, determine the rate of convergence for Newton's method for points that are farther to the root and points near the root. Will newton's method converge for any starting point? Can you propose a method that is faster than Newton's method?
5. Show that $x_{n+1} = \frac{x_n(x_n^2+3a)}{3x_n^2+a}$ is a third order method for computing $\sqrt{a}$. Implement the above method by modifying Newton's program in sakai. Compare newton's method bisection method and the above third order method for the computation of $\sqrt{5}$ within $eps = 10^{-12}$. Write conclusions of your comparison.