MATH 3940 Problem Set 3 Solutions - Matlab

Question 1: (d) M-file for the fixed point method is

```
function [k,p,err,P] =fixpt (g, p0, tol,max1)
P(1)=p0
for k=2:max1
   P(k)=feval(g,P(k-1));
   err=abs(P(k)-P(k-1));
   relerr=err/abs(P(k));
   p=P(k);
   if (err<tol) | (relerr<tol),
      k=k;
      break;
   end
  end
   P=P';
```

M- file for the function is

```
function y=g413(x)
y=(x^2+5*x-12)/4;
```

```
>> [k p err P]=fixpt('g413',-3.5,10^-5,40)
P =  -3.5000
k =   37
p =  -4.0000
err =  3.2000e-005
P =  -3.5000
      -4.3125
       ……..
      ……..
     -4.0000
     -4.0000
```

The convergence is achieved in 36 iterations.

(e) This is not asked in the question, it is just to show that fixed point converges with p0=-0.25

```
>> [k p err P]=fixpt('g413',-0.25,10^-5,40)

k =    39

p =   -4.0000

err =  3.1016e-005

P =   -0.2500

       -3.2969

       ………….

       ………..

     -3.9999

     -4.0000
```

The convergence is achieved in 38 iterations.

Question 2: (a) It is easy to use roots command in Matlab for polynomials.

```
>> p=[1 1 -3 -3]; % the coefficients of the polynomial

>> roots(p)

ans =     1.7321

           -1.7321

           -1.0000
```

(b) (i) M file for the function is

```
function y = g5116(x)
y = ((3+3*x-x^2)/x)^(1/2);
```

```
>> [k p err P]=fixpt('g5116',1,10^-5,25)

k =    23

p =    1.7320

err =   1.3480e-05

P =    1.0000

        2.2361

       ………..

       ………..

       1.7321
```

1.7320

Convergence to 1.732 is achieved in 22 iterations.

(ii) M file for the function is function y = g5216(x)

y = -1+(3*x+3)/(x^2);

[k p err P]=fixpt('g5216',1,10^-5,25)

k = 25

p = 4.8961

err = 5.3990

P = 1.0000

5.0000

…………

…………

24.6873

-0.8736

-0.5029

4.8961

The iterations do not converge.

(iii) M file for the function is function y = g5316(x)

y = (x^3+x^2-x-3)/2;

>> [k p err P]=fixpt('g5316',1,10^-5,25)

k = 3

p = -1

err = 0

P = 1

-1

-1

Convergence to -1 is achieved in 2 iterations.

<u>Question 4</u>: M-File for the function is function y = fq3(x)

y = x-2^(-x);

(a) >> fzero('fq3',0)

ans = 0.6412

(b) M-file for the bisection method is:

```
function [c, k, err, yc]=bisect(f,a,b,tol,maxite)
ya=feval(f,a);
yb=feval(f,b)
for k=1:maxite
   c=(a+b)/2;
   yc=feval(f,c);
   if yc==0
      a=c;
      b=c;
   elseif yb*yc>0
      b=c;
      yb=yc;
   else
      a=c;
      ya=yc;
   end
   if b-a <tol   break,
    end
end
c=(a+b)/2;
err=b-a;
yc=feval(f,c);
```

>> [c, k, err, yc]=bisect('fq3',0,1,10^-5,20)

c = 0.6412

k = 17

err = 7.6294e-006

yc = 2.3101e-008

(c) M-file for the method of false position is:

```
function [c,k,err,yc]=regula(f,a,b,tol,epsilon,maxite)
ya=feval(f,a);
yb=feval(f,b);
for k=1:maxite
    dx=yb*(b-a)/(yb-ya);
    c=b-dx;
   ac=c-a;
    yc=feval(f,c);
    if yc==0, break;
    elseif yb*yc>0
        b=c;
        yb=yc;
    else
        a=c;
        ya=yc;
    end
    dx=min(abs(dx),ac);
     if (abs(yc)<epsilon)|(abs(dx)<tol)
        break,     end
end
c;
err=abs(b-a)/2;
yc=feval(f,c);
>> [c,k,err,yc]=regula('fq3',0,1,10^(-5),10^(-10),20)
c =    0.6412
k =     5
err =    0.3206
yc = 1.0914e-006
```

(d) M-file for the Secant method is:

```
function [p1, err, k, y]=secant(f,p0,p1,tol,epsilon,maxite)
for k=1 :maxite
        p2=p1-feval(f,p1)*(p1-p0)/(feval(f,p1)-feval(f,p0));
        err=abs(p2-p1);
        relerr=err/abs(p2);
        p0=p1;
        p1=p2;
        y=feval(f,p1);
        if (err<tol) | (relerr<tol) | (abs(y)<epsilon), break, end
end
>> [p1, err, k, y]=secant('fq3',0,1,10^(-5),10^(-10),20)
p1 =   0.6412
err =  2.5153e-006
k =    4
y =  3.5986e-010
```

(e) M-file for the Newton's method is:

```
function [p, err, k, y]=newton(f, df, p0, tol, epsilon, max1)
for k=1 :max1
        p1=p0-feval(f,p0)/feval(df,p0);
        err=abs(p1-p0) ;
        relerr=err/abs(p1);
        p0=p1;
        y=feval(f,p0);
        if (err<tol) | (relerr<tol) | (abs(y)<epsilon) ,
        break,
       end
end
```

p=p1;

M- files for the function and its derivative are:

function y = fq3(x)

y = x-2^(-x);

function y = dfq3(x)

y = 1+(2^(-x))* log (2);

>> [p, err, k, y]=newton('fq3', 'dfq3', 1,10^(-5), 10^(-10), 20)

p =   0.6412

err = 1.6710e-005

k =   3

y = -4.3008e-011

(f) We see that bisection method converges in 17 iterations, method of false position converges in 5 iterations, secant method converges in 4 iterations, and Newton method converges in 3 iterations, this means that Newton method is most successful here. We know that the convergence rate of bisection method and method of false position is linear, secant method is nearly quadratic, and Newton is quadratic. That is why Newton method converges faster than any other method.

Question 5: (c) M-File for the function and its derivative are:

function y = fq4(x)

y = x*cos(x)-x;

function y = dfq4(x)

y = cos(x)-x*sin(x)-1;

Using the program of Newton's method from the previous question, we obtain

>> [p, err, k, y]=newton('fq4', 'dfq4', 1,10^(-5), 10^(-7), 15)

p =   0.0049

err =   0.0024

k =   13

y = -5.8105e-008