Date-16/1/18

Problem1:Find the value of definite integral using Trapezoidal rule

M-File:

function fun7=func\_Trap(a,b,n)

x=zeros(n+1,1);

h=(b-a)/n;

for j=1:1:n

x(j)=a+j.\*h;

end

s1=0;s2=0;

for j=1:1:n

y=x(j);z=x(j+1);

s1=s1+func\_f(y);

s2=s2+func\_f(z);

end

fun7=(h./2).\*(s1+s2);

OUTPUT:

func\_Trap(0,1,1500)

ans = 0.7852

Date-16/1/18

Problem2:Find the value of definite integral using Simpson 1/3rd  rule

M-File:

function fun7=func\_simpson(a,b,n)

h=(b-a)/n;

x=a+h;

sum=func\_f(a)+func\_f(b)+4\*func\_f(x);

for i=3:2:n-1

x1=a+i\*h;

x2=a+(i-1)\*h;

sum=sum+4\*func\_f(x1)+2\*func\_f(x2);

end

fun7=(h/3)\*sum;

OUTPUT

func\_simpson(0,1,1500)

ans =0.7854

Date-16/1/18

Problem3:Find the value of definite integral using waddle’s rule

M-File:

function fun7=func\_waddles(a,b,n)

x=zeros(n+1,1);

s1=0;

s2=0;

s=0;

h=(b-a)/n;

m=n/6;

if n%6==0

for i=1:1:m

s=s+((3\*h/10)\*(func\_f(a)+func\_f(a+2\*h)+5\*func\_f(a+h)+6\*func\_f(a+3\*h)+func\_f(a+4\*h)+5\*func\_f(a+5\*h)+func\_f(a+6\*h)));

a=a+6\*h;

end

end

fun7=s;

OUTPUT

>> func\_waddles(0,1,1500)

ans =0.7854

Date-6/3/18

Problem4:Find the value of differential equation dy/dx=y provided y(0)=1 and n=10 using Euler’s method.

M-File:

function fun7=func\_euler(x0,xf,y0,n)

x=zeros(1,n+1);

y=zeros(1,n+1);

z=zeros(1,n+1);

x(1)=x0;

y(1)=y0;

h=(xf-x0)/n;

z(1)=exp(x(1));

for i=2:1:n+1

x(i)=x(i-1)+h;

y(i)=y(i-1)+h\*dydx(x(i-1),y(i-1));

z(i)=exp(x(i));

end

plot(x,y);

hold on;

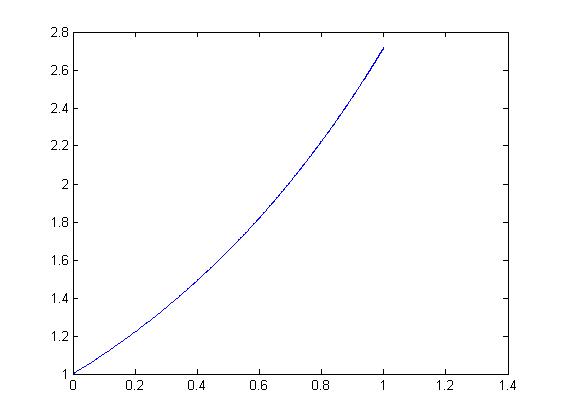
plot(x,z);

fun7=y(n+1);

OUTPUT

>> func\_euler(0,1,1,500)

ans = 2.7156



Date-6/3/18

Problem5:Find the value of differential equation dy/dx=y provided y(0)=1 and n=10 using Runge Kutta’s method.

M-File:

function fun7=func\_RungeKutta(x,y,n,f)

h=(f-x)/n;

fprintf('X Y\n');

for i=1:1:n

k1=h.\*func\_g(x,y);

k2=h.\*func\_g(x+h/2.0,y+k1/2.0);

k3=h.\*func\_g(x+h/2.0,y+k2/2.0);

k4=h.\*func\_g(x+h,y+k3);

k=(k1+(k2+k3).\*2.0+k4)/6.0;

x=x+h;

y=y+k;

fprintf('%f\t%f\n',x,y);

end

OUTPUT

>> func\_RungeKutta(0,1,10,1)

X Y

0.100000 1.105171

0.200000 1.221403

0.300000 1.349858

0.400000 1.491824

0.500000 1.648721

0.600000 1.822118

0.700000 2.013752

0.800000 2.225540

0.900000 2.459601

1.000000 2.718280

Date-13/3/18

Problem7:Interpolate the value of the function corresponding to x=4 using laggrange’s interpolation formula.

M-File:

function fun7=func\_lagrange(xf,n)

x=zeros(n+1,100);

y=zeros(n+1,100);

for i=1:1:n

x(i)=input('\nEnter the set of x:');

y(i)=input('\nEnter the set of y:');

end

fprintf('X Y\n');

for i=1:1:n

fprintf('%f\t%f\n',x(i),y(i));

end

Y=0.0;

fprintf(' ');

for i=1:1:n

fprintf('X(%d)| ',i);

end

fprintf('\n');

for i=1:1:n

sum=y(i);

fprintf('X(%d)| ',i);

for j=1:1:n

if (i~=j)

sum=sum\*(xf-x(j))./(x(i)-x(j));

fprintf('%f ',(x(i)-x(j)));

else

fprintf(' ');

end

end

fprintf('\n');

Y=Y+sum;

end

fprintf('\nHence for X= %d value for Y=%d',xf,Y);

OUTPUT

>> func\_lagrange(4,5)

X Y

2.000000 10.000000

3.000000 15.000000

5.000000 25.000000

8.000000 40.000000

12.000000 60.000000

X-X(1)| X-X(2)| X-X(3)| X-X(4)| X-X(5)|

X-X(1)| -1.000000 -3.000000 -6.000000 -10.000000

X-X(2)| 1.000000 -2.000000 -5.000000 -9.000000

X-X(3)| 3.000000 2.000000 -3.000000 -7.000000

X-X(4)| 6.000000 5.000000 3.000000 -4.000000

X-X(5)| 10.000000 9.000000 7.000000 4.000000

Hence for X= 4 value for Y=20

Date-13/3/18

Problem8:Interpolate the value of the function usin Newton’s Forward interpolation formula.

M-File:

function fun7=func\_newtonforward(xf,n)

a=zeros(n+1,100);

b=zeros(n+1);

sum1=zeros(n+1,100);

for i=1:1:n

b(i)=input('\nEnter the set of x:');

a(1,i)=input('\nEnter the set of y:');

end

fprintf('X | Y\n');

for i=1:1:n

fprintf('%.2f |%f \n',b(i),a(1,i));

end

y0=a(1,1);

fprintf('\nThe difference table is:\n');

for i=1:1:n-1

fprintf('dely[%d]: ',i);

for j=1:1:n-i

a(i+1,j)=a(i,j+1)-a(i,j);

fprintf('%f ',a(i+1,j));

end

fprintf('\n');

end

u=(xf-b(1))/(b(2)-b(1));

sum2=0;

for i=1:1:n-1

sum=1;

for j=0:1:i-1

sum=sum\*(u-j);

end

sum1(i)=(sum\*a(i+1,1))/factorial(i);

sum2=sum2+sum1(i);

end

y=y0+sum2;

fprintf('\nHence for X= %.2f value of Y=%f',xf,y);

OUTPUT

>> func\_newtonforward(2.02,6)

X | Y

2.00 | 0.301030

2.20 | 0.342420

2.40 | 0.380210

2.60 | 0.414970

2.80 | 0.447160

3.00 | 0.477210

The difference table is:

dely[1]: 0.041390 0.037790 0.034760 0.032190 0.030050

dely[2]: -0.003600 -0.003030 -0.002570 -0.002140

dely[3]: 0.000570 0.000460 0.000430

dely[4]: -0.000110 -0.000030

dely[5]: 0.000080

Hence for X= 2.02 value of Y=0.305351

Date-13/3/18

Problem9:Interpolate the value of the function usin Newton’s Backward interpolation formula.

M-File:

function fun7=func\_newtonbackward(xf,n)

a=zeros(n+1,100);

b=zeros(n+1);

sum1=zeros(n+1,100);

for i=1:1:n

b(i)=input('\nEnter the set of x:');

a(1,i)=input('\nEnter the set of y:');

end

fprintf('X | Y\n');

for i=1:1:n

fprintf('%.2f |%f \n',b(i),a(1,i));

end

fprintf('\nThe difference table is:\n');

for i=1:1:n-1

fprintf('dely[%d]: ',i);

for j=1:1:n-i

a(i+1,j)=a(i,j+1)-a(i,j);

fprintf('%f ',a(i+1,j));

end

fprintf('\n');

end

for i=1:1:n+1

if b(i)<xf && xf<b(i+1)

break;

end

end

k=1;

k=i+1;

u=(xf-b(k))/(b(2)-b(1));

sum2=0;

for i=1:1:n-1

sum=1;

for j=0:1:i-1

sum=sum\*(u+j);

end

sum1(i)=(sum\*a(i+1,n-i))/factorial(i);

sum2=sum2+sum1(i);

end

y=a(1,k)+sum2;

fprintf('\nHence for X= %.2f value of Y=%f',xf,y);

OUTPUT

>> func\_newtonbackward(0.37,5)

X | Y

0.00 | 1.000000

0.10 | 1.221400

0.20 | 1.491800

0.30 | 1.822100

0.40 | 2.225500

The difference table is:

dely[1]: 0.221400 0.270400 0.330300 0.403400

dely[2]: 0.049000 0.059900 0.073100

dely[3]: 0.010900 0.013200

dely[4]: 0.002300

Hence for X= 0.37 value of Y=2.095927

Date-20/3/18

Problem10:Interpolate the value of the function usin Newton’s Divided differenc Formula.

M-File:

function fun7=func\_divided\_difference(xf)

n=input('\nEnter the no of points :');

x=zeros(n+1,1);

y=zeros(n+1);

yd=zeros(n+1,100);

for i=1:1:n

x(i)=input('\nEnter the set of x:');

y(i)=input('\nEnter the set of y:');

end

fprintf('X | Y\n');

for i=1:1:n

fprintf('%.2f |%f \n',x(i),y(i));

end

for i=1:1:n-1

yd(i,1)=(y(i+1)-y(i))/(x(i+1)-x(i));

end

for j=2:1:n-1

for i=1:1:n-j

yd(i,j)=(yd(i+1,j-1)-yd(i,j-1))/(x(i+j)-x(i));

end

end

fprintf('\n x | f(x) |');

for i=1:1:n-2

fprintf('f[x0');

for j=1:1:i

fprintf(',x%d',j);

end

fprintf('] |');

end

for i=1:1:n

fprintf('\n%10.2f |%10.2f',x(i),y(i));

for j=1:1:n-2

if j>n-i

break;

end

fprintf('|%10.2f',yd(i,j));

end

end

sum=y(1);

for i=1:1:n-1

m=1;

for j=1:1:i

m=m\*(xf-x(j));

end

sum=sum+m\*yd(1,i);

end

fun7=sum;

end

OUPUT

>> func\_divided\_difference(0.3)

Enter the no of points :5

X | Y

0.00 | 1.000000

1.00 | 3.000000

3.00 | 49.000000

4.00 | 129.000000

7.00 | 813.000000

x | f(x) | f[x0,x1] | f[x0,x1,x2] | f[x0,x1,x2,x3] |

0.00 | 1.00 | 2.00 | 7.00 | 3.00

1.00 | 3.00 | 23.00 | 19.00 | 3.00

3.00 | 49.00 | 80.00 | 37.00

4.00 | 129.00 | 228.00

7.00 | 813.00

ans = 1.8310

Date-27/3/18

Problem11:To find the roots of a linear equation using gauss elimination without using pivoting

M-File:

function fun7=func\_sol\_linear\_eqn(A,b)

Ab=[A b];

[m n]=size(Ab);

x=zeros(m,1);

for k=1:1:m-1

for i=k+1:1:m

m1=Ab(i,k)./Ab(k,k);

for j=1:1:n

Ab(i,j)=Ab(i,j)-m1\*Ab(k,j);

end

end

end

Ab

A=Ab([1:m],[1:m]);

b=Ab([1]);

x(m)=Ab(m,n)./Ab(m,m);

for i=m-1:-1:1

m2=1./Ab(i,i);

sum=0;

for j=m:-1:i

if(i==j)

sum=sum+0;

else

sum=sum+Ab(i,j).\*x(j);

end

end

x(i)=m2.\*(Ab(i,n)-sum);

end

Aug=Ab;

fun7=x;

OUTPUT :

>> func\_sol\_linear\_eqn([1 2 1;2 21 1;-1 2 0],[1 1 2]')

Ab =

1.0000 2.0000 1.0000 1.0000

0 17.0000 -1.0000 -1.0000

0 0 1.2353 3.2353

ans =

-1.8095

0.0952

2.6190