Nacombined

**Experiment 6: Power Method**

**Name -** Panshul Saxena

**Batch -** 2CS10

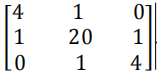
**Roll No -** 102196006

2.

Determine the largest eigen-value and the corresponding eigen-vector of the following

matrices using the power method. Use *x*0 = [1,1,1]T and ϵ = 10-3:

(a)



Use x0 = [1,1,1]T and ϵ = 10-3

**Code-**

clc

A=input("matrix A : \n");

x=input("vector x : \n");

tolerance=input("tolerance : \n");

iterations=input("iterations : \n");

k = zeros(1,100);

i = 1;

while i < iterations

y=A\*x;

n=norm(y,inf);

x=y/n;

k(i)=n;

if (i > 1)

if (abs(k(i) - k(i-1)) < tolerance )

break;

end

end

i = i+1;

end

if i > iterations

disp("Max iterations reached \n ");

end

fprintf("No of iterations : ");

disp(iterations);

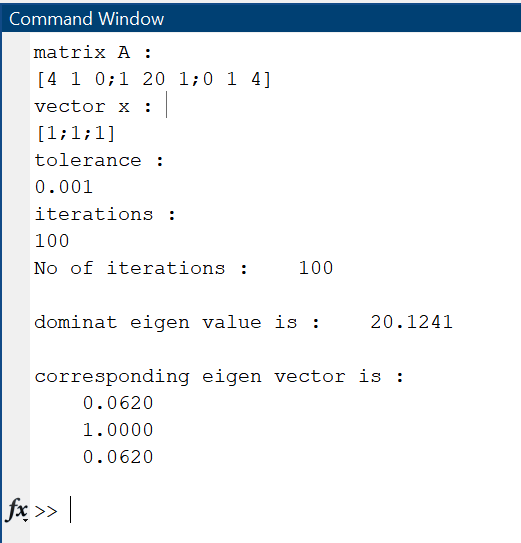
fprintf("dominat eigen value is : ");

disp(k(i));

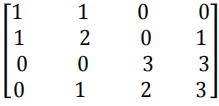
fprintf("corresponding eigen vector is : \n");

disp(x);

**Output-**



(b)



Use x0 = [1,1,0,1]T and ϵ = 10-3

**Code-**

clc

A=input("matrix A : \n");

x=input("vector x : \n");

tolerance=input("tolerance : \n");

iterations=input("iterations : \n");

k = zeros(1,100);

i = 1;

while i < iterations

y=A\*x;

n=norm(y,inf);

x=y/n;

k(i)=n;

if (i > 1)

if (abs(k(i) - k(i-1)) < tolerance )

break;

end

end

i = i+1;

end

if i > iterations

disp("Max iterations reached \n ");

end

fprintf("No of iterations : ");

disp(iterations);

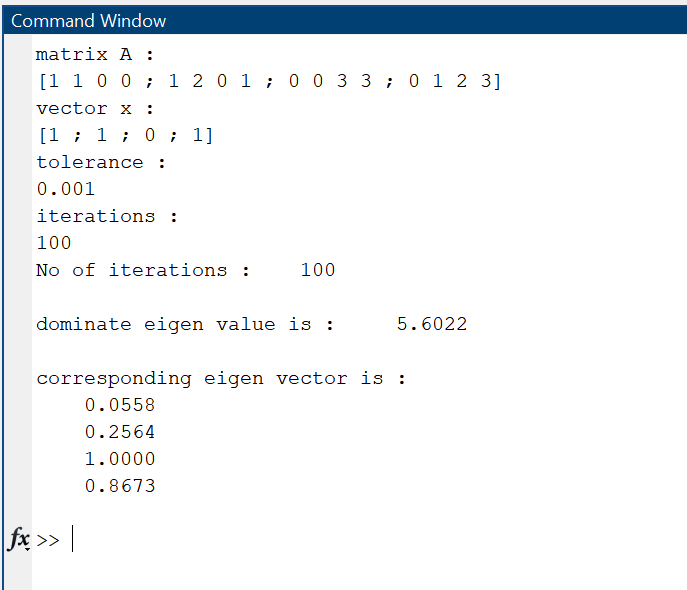
fprintf("dominate eigen value is : ");

disp(k(i));

fprintf("corresponding eigen vector is : \n");

disp(x);

**Output-**

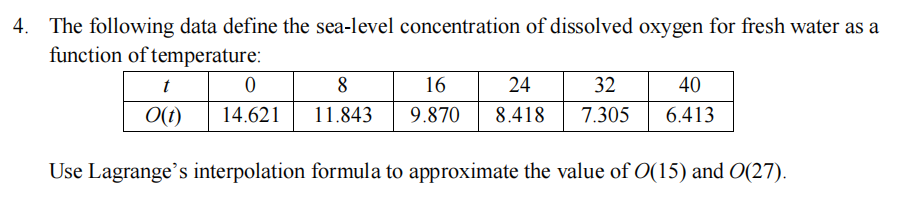


**Experiment 6: Power Method and Lagrange Interpolation**

**Name -** Panshul Saxena

**Batch -** 2CS10

**Roll No -** 102196006



Code

clc

clear

x = input("enter x \n");

y = input("enter y \n");

n = length(x);

xp = [15,27];

for k=1:length(xp)

sum = 0;

for i=1:n

pr=1;

for j=1:n

if j~=i

pr=pr.\*(xp- x(j))/(x(i)-x(j));

end

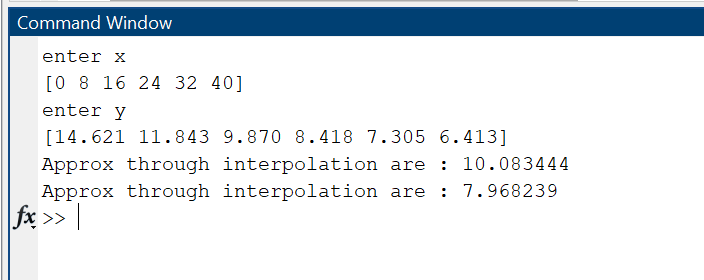
end

sum = sum + y(i) \* pr;

end

end

fprintf("Approx through interpolation are : %f\n",sum);



5.

Generate eight equally-spaced points from the function *f*(*x*) = sin2*x* from *x* = 0 to 2π. Use

Lagrange interpolation to approximate *f*(*0.5*), *f*(*3.5*), *f*(*5.5*) and *f*(*6.0*).

Code

clc

clear

x=linspace(0,(2\*pi),8)

y=sin(x).\*sin(x)

n=length(x)

xp = input("Enter xp \n");

for k=1:length(xp)

sum=0;

for i=1:n

pr=1;

for j=1:n

if j~=i

pr=pr.\*(xp- x(j))/(x(i)-x(j));

end

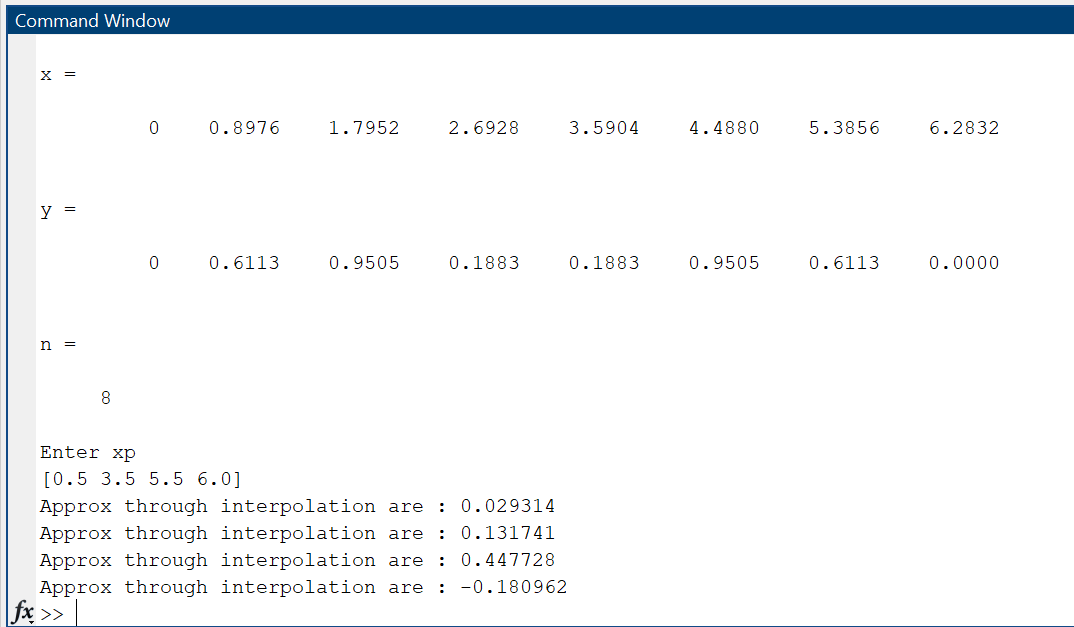
end

sum = sum + y(i) \* pr;

end

end

fprintf("Approx through interpolation are : %f\n",sum);

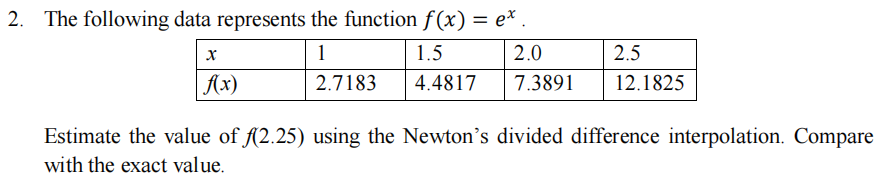


**Experiment 7: Newton’s Divided Difference Interpolation**

**Name -** Panshul Saxena

**Batch -** 2CS10

**Roll No -** 102196006



clc

clear

n=4;

x = [ 1, 1.5, 2, 2.5 ];

y = [2.7183, 4.4817, 7.3891, 12.1825];

x0 = 2.25;

fun= ones(1,4);

for i = 1:n

fun(i,1) = y(i);

end

for i = 2:n

for j = 2:i

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

end

end

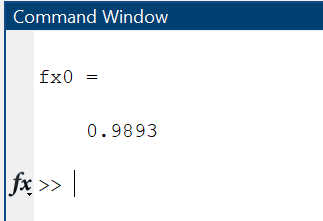
fx0 = fun(n,n);

for i = n-1 :1

fx0 = fx0\*(x0-x(i)) + fun(i,i);

end

display(fx0)



3.

Approximate *f*(0.43) by using Newton’s divided difference interpolation, construct the

interpolating polynomials for the following data.

*f*(0) = 1, *f*(0.25) = 1.64872, *f*(0.5) = 2.71828, *f*(0.75) = 4.4816.

clc

clear

n = 4;

x = [ 0, 0.25, 0.5, 0.75 ];

y = [1, 1.64872, 2.71828, 4.4816];

x0 = 0.43;

Def= ones(1,4);

for i = 1:n

Def(i,1) = y(i);

end

for i = 2:n

for j = 2:i

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

end

end

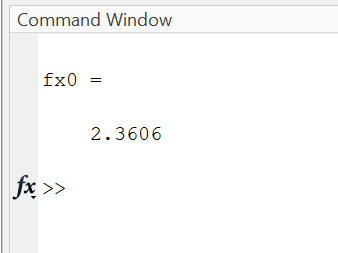
fx0 = Def(n,n);

for i = n-1:-1:1

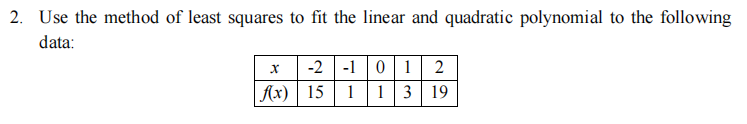
fx0 = fx0\*(x0-x(i)) + Def(i,i);

end

display(fx0)



Experiment 8: Least Square Approximation



clc

clear

X=[-2 -1 0 1 2];

Y=[15 1 1 3 19];

F=input("Enter 1 for linear fit, 2 for quadratic and so on : ");

n=length(X);

N=F+1;

A= zeros(N,N);

for i=1:N

for j=1:N

A(i,j)= sum(X.^(i+j-2));

end

end

B= zeros(N,1);

for K=1:N

B(K)= sum((X.^(K-1)).\*Y);

end

Z=A\B;

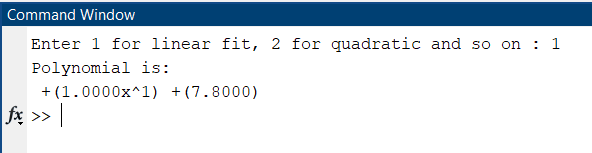
disp("Polynomial is: ")

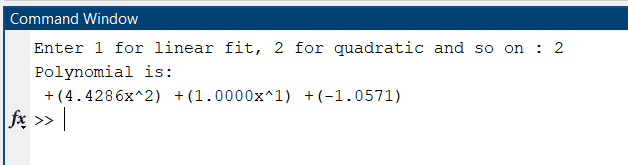
for k=N:-1:2

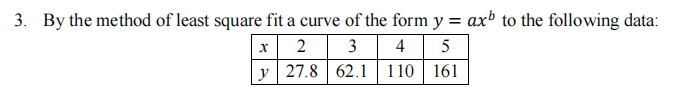
fprintf(" +(%.4fx^%d)",Z(k),k-1)

end

fprintf(" +(%.4f)\n",Z(1))







clc;

n= input("Enter the number of observations : ");

x= input("Enter the value of x : ");

y= input("Enter the value of f(x) : ");

sum1=0;

sum2=0;

sum3=0;

sum4=0;

for i=1:n

sum1=sum1+x(i);

end

for i=1:n

sum2=sum2+log10(y(i));

end

for i=1:n

sum3=sum3+(x(i)\*log10(y(i)));

end

for i=1:n

sum4=sum4+(x(i))^2;

end

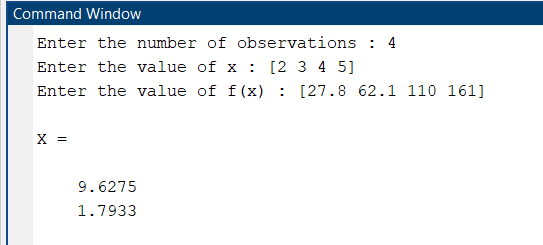
A=[n sum1; sum1 sum4];

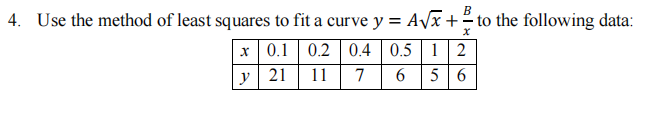
B=[sum2; sum3];

X=A\B;

X(1,1)= 10^(X(1,1));

X(2,1)= 10^(X(2,1))





clc;

clear all;

x=[0.1 0.2 0.4 0.5 1 2];

y=[21 11 7 6 5 6];

n=length(x);

sumyrtx=0; sumx=0; sumxinv=0; sumxrtinv=0; sumxsqrinv=0;sumyx=0;

for i=1:n

sumyrtx=sumyrtx+(y(i)\*(x(i).^(0.5)));

sumx=sumx+x(i);

sumxinv=sumxinv+(1/x(i));

sumxrtinv=sumxrtinv+(1/(x(i).^(0.5)));

sumxsqrinv=sumxsqrinv+(1/(x(i)\*x(i)));

sumyx=sumyx+(y(i)/x(i));

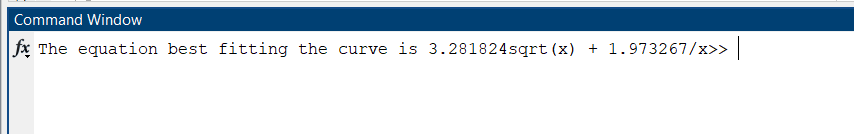
end

matt1=[sumx sumxrtinv; sumxrtinv sumxsqrinv];

matt2=[sumyrtx; sumyx];

val=matt1\matt2;

fprintf('The equation best fitting the curve is %fsqrt(x) + %f/x' ,val(1), val(2));



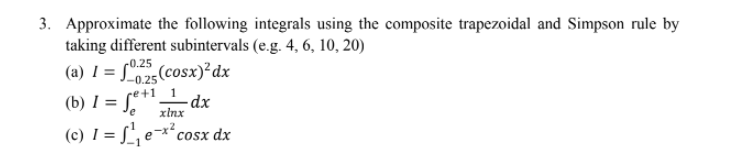
**Experiment 9:Numerical Quadrature**

**Name -** Panshul Saxena

**Batch -** 2CS10

**Roll No -** 102196006

**Trapezoidal rule** –



**a)**

f=@(x) (cos(x))^2;

a=-0.25;

b=0.25;

n=4;

h=(b-a)/n;

sum=0;

for i=1:n-1

x=a+h\*i;

sum=sum+2\*f(x);

end

sum=sum+f(a)+f(b);

ans=(sum\*h)/2;

disp(ans);



f=@(x) 1/(x\*(log(x)));

a=exp(1);

b=exp(1)+1;

n=4;

h=(b-a)/n;

sum=0;

for i=1:n-1

x=a+h\*i;

sum=sum+2\*f(x);

end

sum=sum+f(a)+f(b);

ans=(sum\*h)/2;

disp(ans);



f=@(x) exp(-power(x,2))\*cos(x);

a=-1;

b=1;

n=4;

h=(b-a)/n;

sum=0;

for i=1:n-1

x=a+h\*i;

sum=sum+2\*f(x);

end

sum=sum+f(a)+f(b);

ans=(sum\*h)/2;

disp(ans);



**Simpson rule** -

**3) a)**

f=@(x) (cos(x))^2;

a=-0.25;

b=0.25;

n=4;

h=(b-a)/n;

sum=0;

for i=1:n-1

x=a+h\*i;

if rem(i,2)==0

sum=sum+2\*f(x);

else

sum=sum+4\*f(x);

end

end

sum=sum+f(a)+f(b);

ans=(sum\*h)/3;

disp(ans);



**b)**

f=@(x) 1/(x\*(log(x)));

a=exp(1);

b=exp(1)+1;

n=4;

h=(b-a)/n;

sum=0;

for i=1:n-1

x=a+h\*i;

if rem(i,2)==0

sum=sum+2\*f(x);

else

sum=sum+4\*f(x);

end

end

sum=sum+f(a)+f(b);

ans=(sum\*h)/3;

disp(ans);



f=@(x) exp(-power(x,2))\*cos(x);

a=-1;b=1;

n=4;

h=(b-a)/n;

sum=0;

for i=1:n-1

x=a+h\*i;

if rem(i,2)==0

sum=sum+2\*f(x);

else

sum=sum+4\*f(x);

end

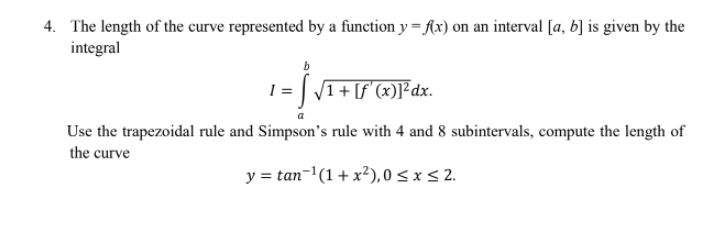
end

sum=sum+f(a)+f(b);

ans=(sum\*h)/3;

disp(ans);



****

1. **Trapezoidal rule -**

With 4 intervals -

g=@(x)(atan(1+x^2));

f=@(x) sqrt(1+((2\*x)/((x^2 + 1)^2 + 1))^2);

a=0;

b=2;

n=4;

h=(b-a)/n;

sum=0;

for i=1:n-1

x=a+h\*i;

sum=sum+2\*f(x);

end

sum=sum+f(a)+f(b);

ans=(sum\*h)/2;

disp(ans);



With 8 intervals-

g=@(x)(atan(1+x^2));

f=@(x) sqrt(1+((2\*x)/((x^2 + 1)^2 + 1))^2);

a=0;

b=2;

n=8;

h=(b-a)/n;

sum=0;

for i=1:n-1

x=a+h\*i;

sum=sum+2\*f(x);

end

sum=sum+f(a)+f(b);

ans=(sum\*h)/2;

disp(ans);



**II. Simpson’s rule -**

With 4 intervals-

g=@(x)(atan(1+x^2));

f=@(x) sqrt(1+((2\*x)/((x^2 + 1)^2 + 1))^2);

a=0;

b=2;

n=4;

h=(b-a)/n;

sum=0;

for i=1:n-1

x=a+h\*i;

if rem(i,2)==0

sum=sum+2\*f(x);

else

sum=sum+4\*f(x);

end

end

sum=sum+f(a)+f(b);

ans=(sum\*h)/3;

disp(ans);



With 8 intervals –

g=@(x)(atan(1+x^2));

f=@(x) sqrt(1+((2\*x)/((x^2 + 1)^2 + 1))^2);

a=0;

b=2;

n=8;

h=(b-a)/n;

sum=0;

for i=1:n-1

x=a+h\*i;

if rem(i,2)==0

sum=sum+2\*f(x);

else

sum=sum+4\*f(x);

end

end

sum=sum+f(a)+f(b);

ans=(sum\*h)/3;

disp(ans);

