PPS

Programming for Problem Solving

Mini Project

AAROHI MATHUR

RA2111019010071

Problem Statement

Write a program in C to find the determinant of a matrix or to find the eigen values of a matrix.

Analysis:

* Accept the choice by user.
* Accept the matrix by user.
* Calculate the determinant by Gauss elimination method.
* Calculate the Eigen values and Eigen vector by power method.
* Display the result.

FDT-Function Description Table

|  |  |  |  |
| --- | --- | --- | --- |
| Function Name | Return Type | Purpose | Parameter List |
| main | int | To take user’s inputs | - |
| fabs | int | To find the absolute value | Z[1] |
| Getch() | void | To get the input | - |

Algorithm for determinant of a matrix

The Gaussian elimination is normally used to solve the system of linear equations. We convert the system into the matrix form, reduce the augmented matrix into its upper triangular form or into its row echelon form.  
Suppose that we are given with a 3×3 matrix A. We will reduce this matrix into an upper triangular matrix U using elementary operations.  
We can interchange two rows of the matrix A we can multiply any row of the matrix A with a scalar and we can add a multiple of a row to another for reducing the matrix A into an upper triangular matrix U.  
After finding the upper triangular matrix U, we calculate its determinant by multiplying the diagonal elements U11, U22 and U33.  
Since the matrix A is equivalent to the upper triangular matrix U, the determinant of both the matrices are equal. Therefore, the determinant of the matrix A is equal to the product of the diagonal elements of the upper triangular matrix U. That is, detA=U11U22U33.  
This is true for any matrix.

In this program, we first convert given square matrix to upper triangular matrix using Gauss Elimination Technique then determinant is calculated simply by multiplying principle diagonal elements.

Input Matrix:

a b c

d e f

g h i

Is Changed To:

a b c

0 e' f'

0 0 i''

Then Determinant Is:

a \* e' \* i''

Note: Here e' represents value of e is

Changed once and i'' represents value of

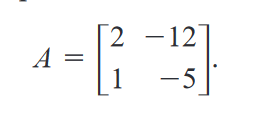
i is changed twice during row operation.

Algorithm for Eigen values and Eigen vector of a matrix.

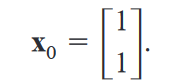
The Power Method is used to find a dominant eigenvalue (one having the largest absolute value), if one exists, and a corresponding [eigenvector](https://www.sciencedirect.com/topics/mathematics/eigenvector).

To apply the Power Method to a [square matrix](https://www.sciencedirect.com/topics/mathematics/square-matrix) **A**, begin with an initial guess **u**0 for the eigenvector of the dominant eigenvalue. Then, for i ≥ 1, calculate **u**i = **Au**i−1/||**Au**i−1||, until consecutive vectors **u**i are either identical or opposite. If **u**k denotes the last vector calculated in this process, then **u**k is an [approximate eigenvector](https://www.sciencedirect.com/topics/mathematics/approximate-eigenvector) of **A**, and ||**Au**k|| is the absolute value of the dominant eigenvalue for **A**.

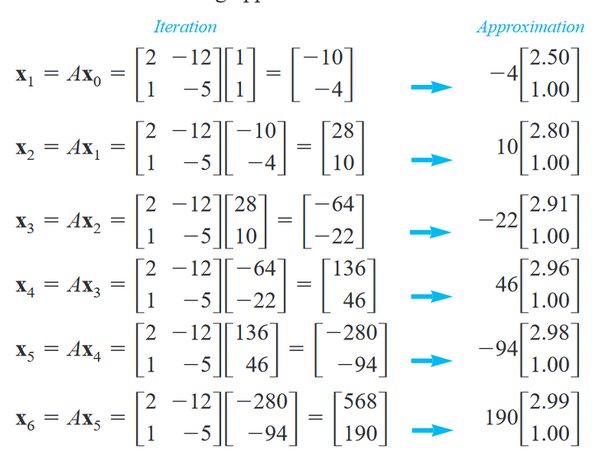
Let us assume the following matrix A.



Let us make an initial approximation for the Eigen Vector X.



So now we have A and x0, we then determine x1 = A \* x0. Now x2 = A \* x1, x3 = A \* x2 and so on…



We can see a repetition of the X vector between the 5th and the 6th iteration. This is the dominant Eigen vector. Now that we have a converged Eigen vector, we divide the components of the last iteration by the previous iteration.

568 / -280 = -2.02 (and) 190 / -94 = -2.02. Approximately this equals 2. Thus the corresponding Eigen value = -2. Thus the dominant Eigen value = -2.

Source Code

#include<stdio.h>

#include<math.h>

#include<stdlib.h>

#include<conio.h>

#define SIZE 10

int main()

{

int choice;

printf("\t\t\t\tWELCOME TO AAROHI'S MATRIX CALCULATOR\t\t\t\t\n");

printf("What do you want to calculate?\n");

printf("1.Find the determinant of a matrix\n2.Find the highest eigen value and eigen vector of a matrix\n");

scanf("%d",&choice);

if(choice==1)

{

float a[SIZE][SIZE], x[SIZE], ratio, det=1;

int i,j,k,n;

/\* Inputs \*/

/\* 1. Reading number of unknowns \*/

printf("Enter Order of Matrix: ");

scanf("%d", &n);

/\* 2. Reading Matrix \*/

printf("\nEnter Coefficients of Matrix: \n");

for(i=0;i< n;i++)

{

for(j=0;j< n;j++)

{

printf("a[%d][%d]=",i,j);

scanf("%f", &a[i][j]);

}

}

/\* Here we are using Gauss Elimination

Technique for transforming matrix to

upper triangular matrix \*/

/\* Applying Gauss Elimination \*/

for(i=0;i< n;i++)

{

if(a[i][i] == 0.0)

{

printf("Mathematical Error!");

exit(0);

}

for(j=i+1;j< n;j++)

{

ratio = a[j][i]/a[i][i];

for(k=0;k< n;k++)

{

a[j][k] = a[j][k] - ratio\*a[i][k];

}

}

}

/\* Displaying upper triangular matrix \*/

/\* Not required, just for the sake of understanding \*/

/\* By analyzing upper triangular matrix you

will get what's going on :) \*/

printf("\nUpper Triangular Matrix: \n");

for(i=0;i< n;i++)

{

for(j=0;j< n;j++)

{

printf("%0.2f\t",a[i][j]);

}

printf("\n");

}

/\* Finding determinant by multiplying

elements in principal diagonal elements \*/

for(i=0;i< n;i++)

{

det = det \* a[i][i];

}

printf("\n\nDeterminant of given matrix is: %0.3f", det);

}

else

{

int i,j,n;

float A[40][40],x[40],z[40],e[40],zmax,emax;

printf("\nEnter the order of matrix:");

scanf("%d",&n);

printf("\nEnter matrix elements row-wise\n");

for(i=1; i<=n; i++)

{

for(j=1; j<=n; j++)

{

printf("A[%d][%d]=", i,j);

scanf("%f",&A[i][j]);

}

}

printf("\nEnter the assumed column vector\n");

for(i=1; i<=n; i++)

{

printf("X[%d]=",i);

scanf("%f",&x[i]);

}

do

{

for(i=1; i<=n; i++)

{

z[i]=0;

for(j=1; j<=n; j++)

{

z[i]=z[i]+A[i][j]\*x[j];

}

}

zmax=fabs(z[1]);

for(i=2; i<=n; i++)

{

if((fabs(z[i]))>zmax)

zmax=fabs(z[i]);

}

for(i=1; i<=n; i++)

{

z[i]=z[i]/zmax;

}

for(i=1; i<=n; i++)

{

e[i]=0;

e[i]=fabs((fabs(z[i]))-(fabs(x[i])));

}

emax=e[1];

for(i=2; i<=n; i++)

{

if(e[i]>emax)

emax=e[i];

}

for(i=1; i<=n; i++)

{

x[i]=z[i];

}

}

while(emax>0.001);

printf("\n The required eigen value is %f",zmax);

printf("\n\nThe required eigen vector is :\n");

for(i=1; i<=n; i++)

{

printf("%f\t",z[i]);

}

getch();

}

}

VDT-Variable Description Table

Sample Input Output

|  |  |  |  |
| --- | --- | --- | --- |
| Variable Name | Data Type | Purpose | Scope |
| choice | int | To store the user’s choice | Program |
| A[][] | float | To store the matrix | Program |
| X[] | float | To store the assumed column matrix | Program |
| ratio | float | To store the ratio of each term of the matrix | Program |
| det | float | To store the value of determinant | Program |
| i,j,k | int | To run the loops | Program |
| z[] | float | To store the augumented matrix | Program |
| e[] | float | To store the difference of the 2 matrix | Program |
| zmax | float | To store the dominant eigen value | Program |
| emax | float | To main the minimum error percentage | Program |

Sample Input Output

