MA 202 LAB-8

Name- Dipean Dasgupta
Id- 202151188

1. Using the techniques that you have learned in the earlier labs, write an algorithm to find the eigenvalues and corresponding unit normalised eigenvectors corresponding to the matrix $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$. Check if the two eigenvectors found are orthogonal or not.

Code:

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

#define MAX_ITER 10000
#define EPSILON 1e-10

void printMatrix(double **A, int n)
{
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
        {
            printf("%f ", A[i][j]);
        }
        printf("\n");
    }
}

void printVector(double *v, int n)
{
    for (int i = 0; i < n; i++)
    {
        printf("%f ", v[i]);
    }
    printf("\n");
}

void copyVector(double *src, double *dst, int n)
{
    for (int i = 0; i < n; i++)
```

```c
    {
        dst[i] = src[i];
    }
}

double norm(double *v, int n)
{
    double sum = 0.0;
    for (int i = 0; i < n; i++)
    {
        sum += v[i] * v[i];
    }
    return sqrt(sum);
}

void normalize(double *v, int n)
{
    double len = norm(v, n);
    for (int i = 0; i < n; i++)
    {
        v[i] /= len;
    }
}
void multiply(double **A, double *x, double *y, int n)
{
    for (int i = 0; i < n; i++)
    {
        double sum = 0.0;
        for (int j = 0; j < n; j++)
        {
            sum += A[i][j] * x[j];
        }
        y[i] = sum;
    }
}

void powerSecant(double **A, double *x0, double *x1, double *lambda,
int n)
{
    double *y = (double *)malloc(n * sizeof(double));
    double *tmp = (double *)malloc(n * sizeof(double));
    double delta = 1.0;
    int iter = 0;
```

```c
    copyVector(x0, x1, n);
    normalize(x1, n);
    while (delta > EPSILON && iter < MAX_ITER)
    {
        iter++;
        multiply(A, x1, y, n);
        copyVector(x1, tmp, n);
        copyVector(y, x1, n);
        copyVector(tmp, x0, n);
        double lambda_old = *lambda;
        *lambda = y[0] / x1[0];
        delta = fabs((*lambda) - lambda_old);
        normalize(x1, n);
    }
free(y);
    free(tmp);
}

int main()
{
    int n;
    printf("Enter the size of the matrix: ");
    scanf("%d", &n);
    double **A = (double **)malloc(n * sizeof(double *));
    for (int i = 0; i < n; i++)
    {
        A[i] = (double *)malloc(n * sizeof(double));
        printf("Enter the elements of row %d: ", i + 1);
        for (int j = 0; j < n; j++)
        {
            scanf("%lf", &A[i][j]);
        }
    }
    double *x0 = (double *)malloc(n * sizeof(double));
    double *x1 = (double *)malloc(n * sizeof(double));
    double lambda = 0.0;
    for (int i = 0; i < n; i++)
    {
        x0[i] = 1.0;
    }
    powerSecant(A, x0, x1, &lambda, n);
    printf("\nEigenvalue: %f\n", lambda);
    printf("Eigenvector:\n");
```

```c
    printVector(x1, n);
    for (int i = 0; i < n; i++)
{
        free(A[i]);
    }
    free(A);
    free(x0);
    free(x1);
    return 0;
}
```

Output:

```
PS C:\MA202 lab\lab9> cd "c:\MA202 lab\lab9\" ; if ($?) { gcc q1.c -o q1 } ; if ($?) { .\q1 }
Enter the size of the matrix: 2
Enter the elements of row 1: 0 1
Enter the elements of row 2: 1 0

Eigenvalue: 1.000000
Eigenvector:
0.707107 0.707107
PS C:\MA202 lab\lab9>
```

2. Use the same technique as used above to find the eigenvalues and eigenvectors of the matrix $\begin{pmatrix} 1 & v & 0 & 0 \\ v & 1 & w & 0 \\ 0 & w & 1 & v \\ 0 & 0 & v & 1 \end{pmatrix}$. Here consider two cases wherein $w = 1$ and $v = 0$, and $w = 1$ and $v = 0.5$.

Output:

1.

```
PS C:\MA202 lab\lab9> cd "c:\MA202 lab\lab9\" ; if ($?) { gcc q1.c -o q1 } ; if ($?) { .\q1 }
Enter the size of the matrix: 4
Enter the elements of row 1: 1 0 0 0
Enter the elements of row 2: 0 1 1 0
Enter the elements of row 3: 0 1 1 0
Enter the elements of row 4: 0 0 0 1

Eigenvalue: 1.000000
Eigenvector:
0.171499 0.685994 0.685994 0.171499
PS C:\MA202 lab\lab9>
```

2.

```
PS C:\MA202 lab\lab9> cd "c:\MA202 lab\lab9\" ; if ($?) { gcc q1.c -o q1 } ; if ($?) { .\q1 }
Enter the size of the matrix: 4
Enter the elements of row 1: 1 0.5 0 0
Enter the elements of row 2: 0.5 1 1 0
Enter the elements of row 3: 0 1 1 0.5
Enter the elements of row 4: 0 0 0.5 1

Eigenvalue: 1.000000
Eigenvector:
0.305085 0.637905 0.637905 0.305085
PS C:\MA202 lab\lab9>
```