Software Project Lab-01

Project Name : Leverage Score Sampling

Presented by :
Mahir Faisal
Roll- 1316

Supervised by :
Prof.Dr.Mohammad Shoyaib

Overview

Tasks that I have already completed of my project.

- Input a matrix from file
- Calculate transpose of the matrix
- Multiply two matrix
- Finding coefficients for the polynomial equation from the given matrix
- Calculating determinant of the matrix
- Calculating the inverse of a matrix
- Solving polynomial equation
- Finding eigenvalues of a matrix
- Linear equation solving using Gauss Jordan elimination

Transpose matrix calculation

Transpose matrix of M is M^T such that all rows of the matrix M convert to column of M^T & all column convert to rows

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}^T = \begin{bmatrix} a_{11} & a_{21} & a_{31} \\ a_{12} & a_{22} & a_{32} \\ a_{13} & a_{23} & a_{33} \end{bmatrix}$$

Like,

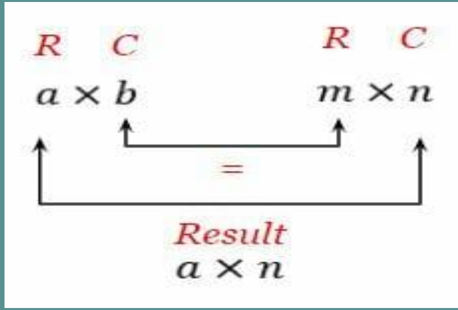
$$M = \begin{bmatrix} 5 & -8 & 9 \\ -6 & 7 & 3 \\ 10 & 8 & 12 \end{bmatrix}$$

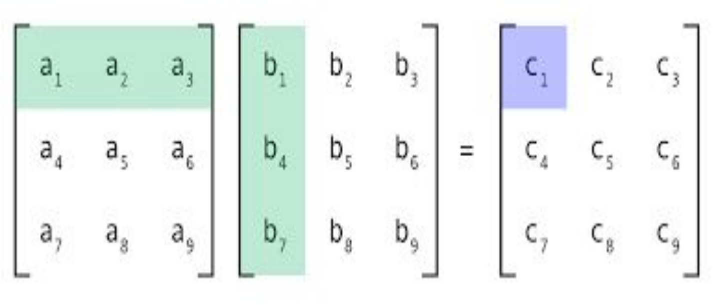
So,

$$M^T = \begin{bmatrix} 5 & -6 & 10 \\ -8 & 7 & 8 \\ 9 & 3 & 12 \end{bmatrix}$$

Matrix Multiplication

If we consider two matrix A(a×b) & B(m×n), then the criteria for multiplication of AB





Let,

$$A = \begin{bmatrix} 5 & -8 & 9 \\ -6 & 7 & 3 \\ 10 & 8 & 12 \end{bmatrix}$$

$$AB = \begin{bmatrix} (5\times5)+(-8 \times -8)+(9\times9) & \dots & \dots \\ (-6\times-6)+(7\times7)+(3\times3) & \dots & \dots \\ (10\times10)+(8\times8)+(12\times12) & \dots & \dots \end{bmatrix}$$

$$B = \begin{bmatrix} 5 & -6 & 10 \\ -8 & 7 & 8 \\ 9 & 3 & 12 \end{bmatrix}$$

$$AB = \begin{bmatrix} 170 & -59 & 94 \\ -59 & 94 & 32 \\ 94 & 32 & 308 \end{bmatrix}$$

Process of finding Eigenvalues & corresponding Eigenvectors

First of all we have to create a polynomial equation of Eigenvalues λ_i from the characteristic equation $|w-\lambda I| = 0$ of the data matrix M

After solving characteristic equation, we find a polynomial equation, such that

$$C_1\lambda^{n}+C_2\lambda^{n-1}+\dots\ \dots+C_{n-1}\lambda + C_{n}$$

We will use Bairstow algorithm to find n Eigenvalues. According to Bairstow algorithm we have to input the coefficients, that means $C_1, C_2, \dots \dots C_{n}$

Thus we will use Faddeev–LeVerrier algorithm to find those coefficients from the data matrix M

That means the outputs of the Faddeev–LeVerrier algorithm are $C_1, C_2, \dots \dots C_{n}$

Faddeev–LeVerrier algorithm

Initialize a identity matrix, $A_1 = \begin{bmatrix} 1 & 0 & 0 & \dots & \dots \\ 0 & 1 & 0 & \dots & \dots \\ 0 & 0 & 1 & \dots & \dots \\ . & . & . & \ddots & . \\ . & . & . & \dots & 1 \end{bmatrix}$

Then we calculate,

$$A_m = MA_{m-1} + C_{n-m+1}I$$

$$C_{\square-m} = -\frac{1}{m}\ tr(\, AM_m \,)$$

Where,

M is the given matrix

n is the dimension of the matrix M

C are the coefficients of the polynomial equation

I is an identity matrix

$tr(\, AM_m \,)$ = trace of the matrix AM_m

Let $M = \begin{bmatrix} 3 & 1 & 5 \\ 3 & 3 & 1 \\ 4 & 6 & 4 \end{bmatrix}$ $\quad C_3 = 0$

$A_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

$MA_1 = \begin{bmatrix} 3 & 1 & 5 \\ 3 & 3 & 1 \\ 4 & 6 & 4 \end{bmatrix}$

$$C_2 = -\frac{1}{1}(\, 3+3+4)$$

$$= -10$$

$$A_2 = \begin{bmatrix} -7 & 1 & 5 \\ 3 & -7 & 1 \\ 4 & 6 & -6 \end{bmatrix} \qquad MA_2 = \begin{bmatrix} 2 & 26 & -14 \\ -8 & -12 & 12 \\ 6 & -14 & 2 \end{bmatrix}$$

$$C_1 = -\frac{1}{2}(-8) = 4$$

$$A_3 = \begin{bmatrix} 6 & 26 & -14 \\ -8 & -8 & 12 \\ 6 & -14 & 6 \end{bmatrix} \qquad MA_3 = \begin{bmatrix} 40 & 0 & 0 \\ 0 & 40 & 0 \\ 0 & 0 & 40 \end{bmatrix}$$

$$C_0 = -\frac{1}{3}(120) = -40$$

Furthermore, $A_4 = MA_3 + C_0I = 0$ which confirms the above calculations

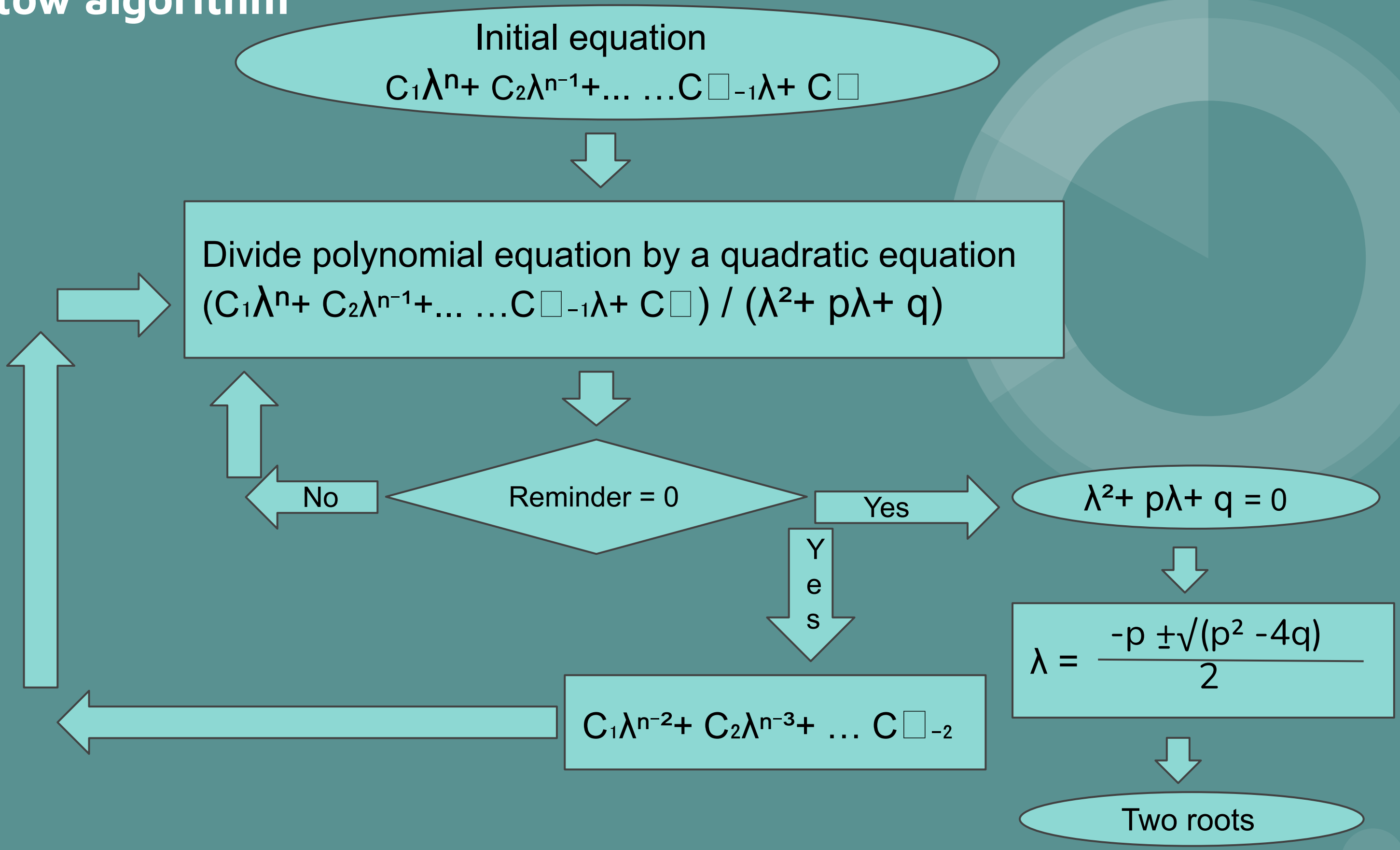
It's very glad to know that $(-1)^3.C_0 = 40$ is the determinant of the given matrix M

It's also an interesting part that the inverse of the given matrix is $(-1/C_0).A_3$

So, $M^{-1} = (-1/(-40)).A_3$

$$M^{-1} = \begin{bmatrix} 3/20 & 13/20 & -7/20 \\ -1/5 & -1/5 & 3/10 \\ 3/20 & -7/20 & 3/20 \end{bmatrix}$$

Bairstow algorithm



Bairstow algorithm (continued)

The quadratic equation $x^2 + px + q = 0$ starts with a initial guess p & q, then it will change iteratively

a is the array of coefficients, found from Faddeev–LeVerrier algorithm

Initial p & q are, $p = (a_{\square-1}/a_{\square})$, $q = (a_{\square-2}/a_{\square})$, if(p = 0) p=0.1, if(q=0) q=0.1

b is the array such that...

$$b_{\square} = a_{\square}$$

$$b_{\square-1} = a_{\square-1} + b_{\square} * p$$

Then, for i = (n-2) to 0

$$b_i = a_i + (b_{i+1} * p) + (b_{i+2} * q)$$

Another array c such that...

$$c_{\square} = b_{\square}$$

$$c_{\square-1} = b_{\square-1} + c_{\square} * p$$ and so on

Iterative approach for changing p,q

$$\triangle p = \frac{b_0 c_3 - b_1 c_2}{c_2 c_2 - c_1 c_3}$$

$$\triangle q = \frac{b_1 c_1 - b_0 c_2}{c_2 c_2 - c_1 c_3}$$

$p' = p$, $q' = q$

$p = p + \triangle p$

$q = q + \triangle q$

Bairstow algorithm (continued)

```
Root finding
    last = false ; phi = 1.0e-12 ;
    if( n =1 ) root =( a□-1 / a□ ) ;
    else if(n=2) last = true ;
        call-> printRoot( a□, a□-1, a□-2 );
    else  while(1)
        cal_col(a,b) ;     cal_col(b,c) ;
        cal_p_q() ;
        ratio_p = △p/p' ;   ratio_q = △q/q';
        if(abs(b0<=phi) & abs(b1<=phi) or
        abs(ratio_p<=phi) or abs(ratio_q<=phi))
        call-> printRoot(1,p,q) ;
        if(n=4) last = true ;
        call-> printRoot(b□,b□-1,b□-2) ; break ;
        if(n=3) root = b□-1/ b□ ; break ;
        call->  reduce_equation() ;
```

```
reduce_equation()
    reform the array a, such that
        ai = bi+2 ;  n = (n-2) ;  for i=0 to (n-1) ;

remove_error(double v)
    int i = v;
    if (abs(i - v)<= phi) v = (double) i ;
```

Bairstow algorithm (continued)

printRoot(double x, double y, double z)

```
if( !last )
   y = -1*remove_error(y) ;
   z = -1*remove_error(z) ;
else
   y = remove_error(y) ;
   z = remove_error(z) ;

determine = (y*y) - 4*x*z ;
if(determine>0)
   determine = sqrt( determine) ;

   root_1 = remove_error(((-y) - determine) / (2*x)) ;
   root_2 = remove_error(((-y) + determine) / (2*x)) ;
```

Gauss Jordan Elimination

Consider a Augmented Matrix Ab

for i=0 to Row

$L = Ab_{ii}$;

for j = 0 to Col+1

$Ab_{i\square} = Ab_{i\square} / L$;

for k=0 to Col+1

if(k!= i) $M = -Ab_{\square i}$;

$Ab_{\square\square} = Ab_{\square\square} + M * Ab_{i\square}$;

So, the value of all variable of the linear equation are...

for i=0 to Row

$Ab_{i\beta}$; where β = number of column

Let, the linear equation

$$\begin{aligned} x - 2y &= -4 \\ -5y + z &= -9 \\ 4x - 3z &= -10 \end{aligned}$$

Augmented Matrix

$$Ab = \left[\begin{array}{ccc|c} 1 & -2 & 0 & -4 \\ 0 & -5 & 1 & -9 \\ 4 & 0 & -3 & -10 \end{array}\right]$$

After applying Gauss Jordan...

$$Ab = \left[\begin{array}{ccc|c} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & 6 \end{array}\right]$$

So, variables...

$x = 2$

$y = 3$

$z = 6$

Singular Value Decomposition

Here SVD is used for **Sampling a data which is considered as a matrix**

Let, the data matrix is M (m□n), such that...

$$M = U \sum V^{T}$$

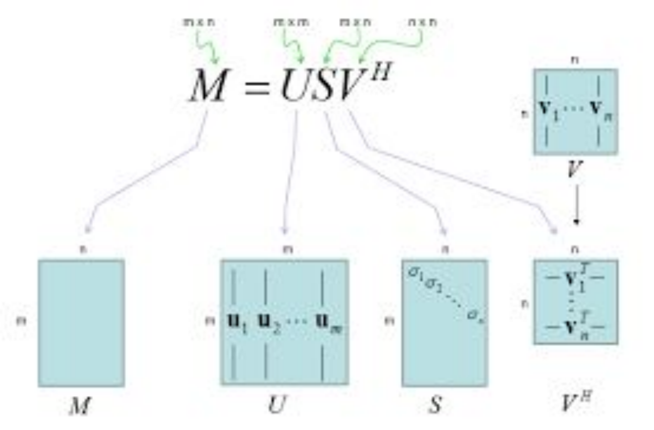
For square matrix m = n

Where,

U is a mxm orthonormal eigenvectors of $M M^{T}$

∑ is a m□n diagonal matrix of the singular values Which are the square roots of the eigenvalues of $M^{T}M$

V^{T} is a n□n orthonormal eigenvectors of $M^{T}M$



Procedure of computing SVD of the given matrix M

First of all we need to compute the left singular vector by finding Eigenvalues of w= $M M^T$

Solving characteristic equation $|w-\lambda I| = 0$

We find m eigenvalues & eigenvectors U_i for the corresponding eigenvalues

Now, create the left singular vector U from the normalize vectors U_i such that …

$$U = \begin{bmatrix} U_1 & U_2 & \dots & \dots & \dots & U_m \end{bmatrix}$$

Now, we compute the right Eigenvector by finding eigenvalues of $w = M^T M$

Solving characteristic equation $|w - \lambda I| = 0$

We find n eigenvalues & eigenvectors V_i for the corresponding eigenvalues

Now, create the right singular vector V from the normalize vectors V_i such that ...

$$V = \begin{bmatrix} V_1 & V_2 & \dots & \dots & \dots & V_n \end{bmatrix}$$

Then, calculate V^T

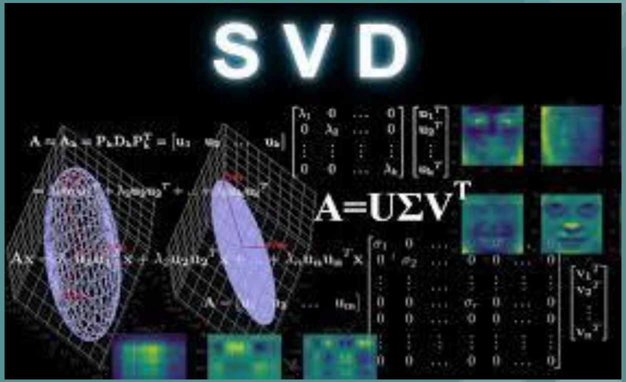
$$V^T = \begin{bmatrix} V_1^T & V_2^T & \dots & \dots & \dots & V_n^T \end{bmatrix}$$

For calculating diagonal matrix Σ, we need n singular values which are the square Root of the eigenvalues of $M^T M$

If the eigenvalues of $M^T M$ are $\lambda_1, \lambda_2 \ldots \ldots \lambda_{\square}$, then the singular values are $\sigma_1, \sigma_2 \ldots \ldots, \sigma_{\square}$

Where $\sigma_i = \sqrt{\lambda_i}$

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & 0 & \ldots \\ 0 & \sigma_2 & 0 & \ldots \\ . & . & . & \ldots \\ . & . & . & \ldots \\ . & . & . & \sigma_{\square} \end{bmatrix}$$



Overview(again)

Tasks that will be done at the next step

- ➢ Echelon form of a Matrix
- ➢ Eigenvector calculation
- ➢ Singular value Decomposition(**SVD**)
- ➢ Probability from **SVD**
- ➢ **Pseudo inverse of the matrix using SVD**
- ➢ **Leverage Score Sampling**
- ➢ Probability Estimation based on **Leverage Score Sampling**

Challenges Faced

- Passing a 2D array to a function & finding result
- Mathematical notation conversion to code
- Finding coefficients for polynomial(characteristic) equation
- Defining Regular inverse of the matrix
- Calculating eigenvalues from the polynomial equation
- Separate real & complex eigenvalues
- Solving eigenvectors for corresponding eigenvalues
- Define orthonormal vectors
- Handling big line of code for the first time

Lines of code

775+

Thank You