

Matrix Operations

Suppose we have a matrix A and a matrix B both $m \times n$. Then, if we add A and B we have

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} + \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & & b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ b_{m1} & b_{m2} & \cdots & b_{mn} \end{bmatrix} = \begin{bmatrix} a_{11} + b_{11} & a_{12} + b_{12} & \cdots & a_{1n} + b_{1n} \\ a_{21} + b_{21} & a_{22} + b_{22} & & a_{2n} + b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} + b_{m1} & a_{m2} + b_{m2} & \cdots & a_{mn} + b_{mn} \end{bmatrix}$$

Similarly, if we subtract B from A we have

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} - \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & & b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ b_{m1} & b_{m2} & \cdots & b_{mn} \end{bmatrix} = \begin{bmatrix} a_{11} - b_{11} & a_{12} - b_{12} & \cdots & a_{1n} - b_{1n} \\ a_{21} - b_{21} & a_{22} - b_{22} & & a_{2n} - b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} - b_{m1} & a_{m2} - b_{m2} & \cdots & a_{mn} - b_{mn} \end{bmatrix}$$

Below is the code for matrix addition and subtraction.

```
//Matrix addition between 2 matrices
vector<vector<double>> matrixAdd(vector<vector<double>> A,
    vector<vector<double>> B, int n, int m)
{
    vector<vector<double>> ans(n, vector<double>(m));
    for (int j=0; j<=m-1; j++)
    {
        for (int i=0; i<=n-1; i++)
            ans[i][j] = A[i][j] + B[i][j];
    }

    return ans;
}

//Matrix subtraction between 2 matrices
vector<vector<double>> matrixSubtract(vector<vector<double>> A,
    vector<vector<double>> B, int n, int m)
{
    vector<vector<double>> ans(n, vector<double>(m));
```

```

for (int j=0; j<=m-1; j++)
{
for (int i=0; i<=n-1; i++)
ans[i][j] = A[i][j] - B[i][j];
}

return ans;
}

```

Next, we consider the distance between two vectors \vec{u} and \vec{v} . This is given by

$$\|\vec{u} - \vec{v}\| = \sqrt{\sum_{i=1}^n |u_i - v_i|^2}$$

Here is the code.

```

//Distance between two vectors
double vectorDistance(vector<double> A, vector<double> B, int n)
{
double sum = 0.0;
double ans;

for (int i=0; i<=n-1; i++)
sum += pow(B[i] - A[i], 2);

ans = sqrt(sum);

return ans;
}

```

Suppose we have A and B , where A is $n \times p$ and B is $p \times m$.

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1p} \\ a_{21} & a_{22} & & a_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{np} \end{bmatrix} \quad B = \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1m} \\ b_{21} & b_{22} & & b_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{p1} & b_{p2} & \cdots & b_{pm} \end{bmatrix}$$

Then the product AB is $n \times m$ and its entries are $(AB)_{i,j} = \sum_{k=1}^p A_{ik}B_{kj}$ where $1 \leq i \leq n$ is the index of the row and $1 \leq j \leq m$ is the index for the column. Below is the code for matrix multiplication.

```

//Matrix multiplication between 2 matrices
//A is n x p and B is p x m
vector<vector<double> > matrixMult(vector<vector<double> > A,
    vector<vector<double> > B, int n, int m, int p)
{
vector<vector<double> > ans(n, vector<double> (m));
for (int j=0; j<=m-1; j++)
{
for (int i=0; i<=n-1; i++)
{
ans[i][j] = 0;
    for (int k=0; k<=p-1; k++)
        ans[i][j] += A[i][k] * B[k][j];
}
}

return ans;
}

```

Suppose we have matrix A of size $m \times n$ and a vector \vec{v} of size $n \times 1$.

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \quad \vec{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

$$A\vec{v} = \begin{bmatrix} a_{11}v_1 + a_{12}v_2 + \cdots + a_{1n}v_n \\ a_{21}v_1 + a_{22}v_2 + \cdots + a_{2n}v_n \\ \vdots \\ a_{m1}v_1 + a_{m2}v_2 + \cdots + a_{mn}v_n \end{bmatrix}$$

The code for matrix-vector multiplication is give below.

```

//Multiplication between matrix and vector
//A is n x m and B is m x 1
vector<double> matVecMult(vector<vector<double> > A, vector<double> B, int n, int m)
{
vector<double> ans(n);
for (int j=0; j<=m-1; j++)
{

```

```

for (int i=0; i<=n-1; i++)
{
ans[i] = 0;
    for (int k=0; k<=m-1; k++)
        ans[i] += A[i][k] * B[k];
}
}

return ans;
}

```

Now we want to compute the determinant of a matrix A . If $A=[a]$ is 1×1 then $\det A = a$. If A is 2×2 so that

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

then $\det A = ad - cb$. For larger matrices, we need to introduce the concept of the *minor* of a matrix. The *minor* matrix M_{ij} is the determinant of the $(n - 1) \times (n - 1)$ submatrix of A constructed by deleting the i th row and the j th column of A . Below is the algorithm for obtaining M_{ij} .

```

//Minor of a matrix
vector<vector<double> > minor(vector<vector<double> > A, int deleteRow, int deleteCol)
{
int n = A.size();
vector<vector<double> > M((n-1), vector<double> (n-1));
int col = 0;

for (int j = 0; j <= n-1; j++)
{
if (j != deleteCol)
{
int row = 0;
for (int i = 0; i <= n-1; i++)
{
if (i != deleteRow)
{
M[row][col] = A[i][j];
row++;
}
}
col++;
}
}
}

```

```

}
}

return M;
}

```

So the determinant of A which is of size $n \times n$ is given by

$$\det A = \sum_{j=1}^n a_{ij} A_{ij} = \sum_{j=1}^n (-1)^{i+j} a_{ij} M_{ij} \quad \text{for } i = 1, \dots, n$$

Here is the code.

```

//Determinant of a matrix
double det(vector<vector<double> > A)
{
int n = A.size();
double determinant = 0;

if (n==1)
determinant = A[0][0];
else if (n==2)
determinant = A[0][0]*A[1][1] - A[0][1]*A[1][0];
else
{
for (int j = 0; j <= n-1; j++)
determinant += pow(-1.0, j)*A[0][j]*det(minor(A, 0, j));
}

return determinant;
}

```

Sometimes it is also useful to find the *transpose* of a matrix. Suppose A is $n \times m$. Then its transpose, denoted A^T will be $m \times n$ where the (i, j) element of A^T is the (j, i) element of A . Below is the algorithm.

```

//Transpose of matrix A
vector<vector<double> > transpose(vector<vector<double> > A, int n, int m)
{
vector<vector<double> > ans(n, vector<double> (m));
for (int j=0; j<=m-1; j++)
{
for (int i=0; i<=n-1; i++)

```

```
ans[i][j] = A[j][i];  
}  
  
return ans;  
}
```