2. MATRIX RELATED

d) Matrices (more properties + orthogonal matrix)

November 27, 2021

1 AIM

Here, we will explore more properties of inputted matrices. We will also discuss orthogonal matrices.

2

We will input a matrix and find some of its properties.

[37]:
```
# Support functions...
from numpy import matrix, zeros, sum, product, trace, min, max, sort
def inputPositiveInteger(prompt):
    while True:
        try:
            i = input(prompt)
            if i == "x": return 0
            i = int(i)
            if i <= 0: i = 1/0
            return i
        except:
            print("Invalid integer, please re-enter.")
def floatInput(prompt):
    while True:
        try:
            i = float(input(prompt))
            return i
        except:
            print("Invalid number, please re-enter.")

def matrixInput(nRow, nCol):
    print("\nEnter row by row, each element in the row separated by comma...")
    A, i = zeros((nRow, nCol)), 0
    while i < nRow:
        row = input("R{0}: ".format(i + 1)).split(",")
        if "x" in row: break # To stop inputting anymore
        if len(row) != nCol:
            print("ERROR: You must only enter", nCol, "per row")
            continue
```

```
        for j in range(0, nCol):
            try:
                A[i][j] = float(row[j])
            except:
                print("ERROR: Non-numeric inputs.")
                j = -1
                break
        if j != -1: i = i + 1
    return A
```

[13]:
```
# NOTE:
# For a multi-dimensional array or matrix...
# - sum from numpy can add all the elements
# - product from numpy can multiply all the elements
# - min and max from numpy can find the minimum and maximum values

# Matrix input
nRow = inputPositiveInteger("Rows\t: ")
nCol = inputPositiveInteger("Columns\t: ")
M = matrixInput(nRow, nCol)
print("The matrix...\n{0}\n".format(M))
print("Sum =", sum(M))
print("Product =",product(M))
print("Trace =", trace(M))
print("Minimum =", min(M))
print("Maximum =", max(M))
print("\nThe matrix with every row sorted...\n{0}".format(sort(M)))
# NOTE
# Even though we pass the matrix, the sort function only sorts each row, since␣
 ↪it only sorts 1D arrays.
```

```
Rows    : 2
Columns : 3

Enter row by row, each element in the row separated by comma…
R1: 1,2,3
R2: 1,5,2
The matrix…
[[1. 2. 3.]
 [1. 5. 2.]]

Sum = 14.0
Product = 60.0
Trace = 6.0
Minimum = 1.0
Maximum = 5.0
```

```
The matrix with every row sorted…
[[1. 2. 3.]
 [1. 2. 5.]]
```

3

We will generate one random matrix, and one random skew-matrix. We will find various properties of these matrices.

[96]:
```
# Some functions...
def randomMatrix(n, m):
    A = zeros((n, m))
    for i in range(0, n):
        for j in range(0, m):
            A[i][j] = random.randint(1, 25)
    return A
def randomSkewMatrix(n, m):
    A = zeros((n, m))
    # Creating upper triangle...
    for i in range(0, n):
        A[i][i] = 0
        for j in range(i + 1, m):
            A[i][j] = random.randint(1, 25)
    for i in range(0, n):
        for j in range(0, i):
            A[i][j] = -A[j][i]
    return A
```

[130]:
```
from numpy import linalg, matrix, identity, tril, random, round
# tril returns the lower triangular matrix for an array or matrix.
# The 1st argument is the array or matrix.
# The 2nd arguemnt specifies whether zeros should be at (k = -1) or above (k =␣
 ↪0) the diagonal.

# Matrix input
# n = inputPositiveInteger("n (rows or columns in the square matrix): ")
# M = matrixInput(n, n)
M = matrix([[4, 3, 2], [1, 4, 1], [3, 10, 4]])
print("Matrix:\n{0}".format(M))
# Proving properties of eigenvalues and eigenvectors...
# 1. For a nxn matrix, the number of eigen values is n.
print("------------------------")
print("PROPERTY 1:")
print("Eigen values:")
eigenValues = linalg.eigvals(M)
for e in eigenValues: print(e)
print("\nNumber of eigen values:", len(eigenValues))
```

```
# 2. The sum of eigen values is equal to the sum of the diagonal elements of␣
 ↪matrix.
print("------------------------")
print("PROPERTY 2:")
print("Sum of eigen values:", round(sum(eigenValues), 3))
print("Sum of diagonal values:", trace(M))
# 3. The product of eigenvalues is equal to the determinant of the matrix.
print("------------------------")
print("PROPERTY 3:")
print("Product of eigen values:", round(product(eigenValues), 3))
print("Determinant of matrix of:", (linalg.det(M)))
# 4. The eigen value for an identity matrix is 1.
print("------------------------")
print("PROPERTY 4:")
I = identity(3)
print("Identity matrix:\n{}".format(I))
print("Eigen values:")
tmp = linalg.eigvals(I)
for e in tmp: print(e)
# 5. The eigen value of a triangular matrix is same as the diagonal elements of␣
 ↪a matrix.
print("------------------------")
print("PROPERTY 5:")
# Random matrix...
A = matrix(randomMatrix(5, 5))
# Lower triangular matrix...
T = tril(A, 0)
print("Lower triangular matrix:\n{}".format(T))
print("Eigen values:")
tmp = linalg.eigvals(T)
for e in tmp: print(e)
# 6. For a skew symmetric matrix, the eigen values are imaginary.
print("------------------------")
print("PROPERTY 6:")
A = matrix(randomSkewMatrix(3, 3))
print("Matrix:\n{0}".format(A))
print("Eigen values:")
tmp = linalg.eigvals(A)
for e in tmp: print(e)
# 7. For orthogonal matrix the values of eigen values are 1 or -1.
print("------------------------")
print("PROPERTY 7:")
A = matrix([[1, 2, 2], [2, 1, -2], [-2, 2, -1]]) / 3
print("Orthogonal matrix:\n{0}".format(A))
print("Eigen values:")
tmp = linalg.eigvals(A)
for e in tmp: print(round(e, 3))
```

```
# 8. For idempotent matrix the eigenvalues are 0 and 1.
print("------------------------")
print("PROPERTY 8:")
A = matrix([[2, -2, -4], [-1, 3, 4], [1, -2, -3]])
print("Idempotent matrix:\n{0}".format(A))
print("Eigen values:")
tmp = linalg.eigvals(A)
for e in tmp: print(round(e, 3))
```

```
Matrix:
[[ 4  3  2]
 [ 1  4  1]
 [ 3 10  4]]
------------------------
PROPERTY 1:
Eigen values:
8.982056720677654
2.1289177050273396
0.8890255742950103

Number of eigen values: 3
------------------------
PROPERTY 2:
Sum of eigen values: 12.0
Sum of diagonal values: 12
------------------------
PROPERTY 3:
Product of eigen values: 17.0
Determinant of matrix of: 17.0
------------------------
PROPERTY 4:
Identity matrix:
[[1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]]
Eigen values:
1.0
1.0
1.0
------------------------
PROPERTY 5:
Lower triangular matrix:
[[ 5.  0.  0.  0.  0.]
 [21. 23.  0.  0.  0.]
 [ 9. 12.  2.  0.  0.]
 [24. 20. 19. 14.  0.]
 [13.  6. 13.  6.  9.]]
```

```
Eigen values:
9.0
14.0
2.0
23.0
5.0
------------------------
PROPERTY 6:
Matrix:
[[  0.   8.  22.]
 [ -8.   0.  10.]
 [-22. -10.   0.]]
Eigen values:
(-6.661338147750939e-16+25.455844122715714j)
(-6.661338147750939e-16-25.455844122715714j)
(6.676925621055143e-16+0j)
------------------------
PROPERTY 7:
Orthogonal matrix:
[[ 0.33333333  0.66666667  0.66666667]
 [ 0.66666667  0.33333333 -0.66666667]
 [-0.66666667  0.66666667 -0.33333333]]
Eigen values:
(1+0j)
(-0.333+0.943j)
(-0.333-0.943j)
------------------------
PROPERTY 8:
Idempotent matrix:
[[ 2 -2 -4]
 [-1  3  4]
 [ 1 -2 -3]]
Eigen values:
0.0
1.0
1.0
```

3.1 NOTES

3.1.1 Orthogonal matrices

When we say two vectors are orthogonal, we mean that they are perpendicular or form a right angle.

In linear algebra, an orthogonal matrix, or orthonormal matrix, is a real square matrix whose columns and rows are orthonormal vectors.

A square matrix with real numbers or elements is said to be an orthogonal matrix, if its transpose is equal to its inverse matrix. Or we can say, when the product of a square matrix and its transpose

gives an identity matrix, then the square matrix is known as an orthogonal matrix.