5. SETS OF VECTORS FROM VECTOR SPACE

a) Linear span

November 27, 2021

1 AIM

Span of a set of vectors S is the set of all vectors that can be expressed as a linear combination of the elements of S. Our aim in this record is to use linear equation system solving methods to find out whether a vector belongs to the linear span of a set of vectors, or equivalently, if a vector can be expressed as a linear combination of a set of vectors.

2 Does a given vector belong to the span of a set of vectors?

To see if a vector belongs in the span of a set of vectors, we must see if any linear combination of the set of vectors equals the given vector.

[25]:
```
import numpy as np
from sympy import *
print("\nGiven vector u:")
u = np.matrix([4, 7, 1])
print(u)

print("\nMatrix of vectors {v1, v2, v3} (vectors are the columns):")
v1 = np.matrix([[5], [-2], [7]])
v2 = np.matrix([[6], [-8], [3]])
v3 = np.matrix([[4], [7], [-1]])
V = np.hstack([v1, v2, v3])
print(V)

print("\nCoefficients of v1, v2 and v3 so that their linear combination is u:")
print(np.linalg.solve(V, u.T))
```

```
Given vector u:
[[4 7 1]]

Matrix of vectors {v1, v2, v3} (vectors are the columns):
[[ 5  6  4]
 [-2 -8  7]
 [ 7  3 -1]]
```

```
Coefficients of v1, v2 and v3 so that their linear combination is u:
[[ 0.35491607]
 [-0.20623501]
 [ 0.86570743]]
```

The presence solutions means that there is some linear combination of the set of vectors V = $\{v_1, v_2, v_3\}$ that results in the given vector i.e. $(4, 7, 1)$. Hence, the given vector is in the linear span of V.

3 Is a given vector a linear combination of a set of vectors?

[26]:
```python
print("\nGiven vector y")
y = np.matrix([6, 4, 3])
print(y)

x1 = np.matrix([[1], [2], [1]])
x2 = np.matrix([[3], [1], [2]])
x3 = np.matrix([[3], [2], [1]])

print("\nMatrix of vectors {x1, x2, x3}:")
X = np.hstack([x1, x2, x3])
print(X)

print("\nCoefficients of x1, x2 and x3 so that their linear combination is y:")
print(np.linalg.solve(X, y.T))
```

```
Given vector y
[[6 4 3]]

Matrix of vectors {x1, x2, x3}:
[[1 3 3]
 [2 1 2]
 [1 2 1]]

Coefficients of x1, x2 and x3 so that their linear combination is y:
[[0.5       ]
 [0.66666667]
 [1.16666667]]
```

3.0.1 Expressing the above fact

Hence, we have that $(6, 4, 3) = \frac{1}{2}(1, 2, 1) + \frac{2}{3}(3, 1, 2) + \frac{7}{6}(3, 2, 1)$

(NOTE: 0.666666... = 2/3 and 1.1666666... = 7/6)

4 CONCLUSION

A linear combination of a set of vectors can be expressed as a system of equations, where the nth equation represents the sum of the scalar multiples of the elements in the nth position in each vector. So, for a linear combination $c_1 a + c_2 b = c_1(a_1, a_2) + c_2(b_1, b_2)$, equation 1's left hand side would be $c_1 a_1 + c_2 b_1$, and equation 2's left hand side would be $c_1 a_2 + c_2 b_2$. The right hand side is the nth element of the given vector, for which you should check if it lies in the linear span or not.

5. SETS OF VECTORS FROM VECTOR SPACE

b) Linear independence (method 1)

November 27, 2021

1 INTRODUCTION

If a set of vectors is linearly independent, then the only way their linear combination will equal zero is if all their scalar coefficients are zero. There are other ways to check if a set of vectors is linearly independent, based on different results. Here,I will look at two of them, and demonstrate the insufficiency of the first method…

2 Using system of linear equations

We cannot use the methods for solving equations for finding the independence of a set of vectors, since the 'solve' function in the 'linalg' module in the NumPy library will only return the trivial solution (i.e. all scalars are zero) if we use the linear equation system solving methods as follows…

[2]:
```python
import numpy as np
v1 = np.matrix([[5], [-2], [7]])
v2 = np.matrix([[6], [-8], [3]])
v3 = np.matrix([[4], [7], [-1]])

print("\nMatrix of vectors {v1, v2, v3}:")
V = np.hstack([v1, v2, v3])
print(V)

print("\nVector of zeros:")
c = np.zeros((3, 1))
print(c)

print("\nCoefficients for v1, v2 and v3 if their linear combination is zero:")
print(np.linalg.solve(V, c))
```

```
Matrix of vectors {v1, v2, v3}:
[[ 5  6  4]
 [-2 -8  7]
 [ 7  3 -1]]

Vector of zeros:
[[0.]
```

```
 [0.]
 [0.]]
```

```
Coefficients for v1, v2 and v3 if their linear combination is zero:
[[ 0.]
 [-0.]
 [ 0.]]
```

The given coefficients i.e. scalars multiplied to the vectors in the given set of vectors will always be given as 0, since this is the simplest solution, and this is all the solve function looks for.

3 Checking the determinant of the vector matrix

So, to check for linear independence, we use a result. Note that this result is applicable only to vector spaces that are defined over the real number field. The result is as follows…

The set of vectors S from the vector space V defined over the field of real numbers is linearly independent if and only if $|S|$ 0, where S is the matrix constructed from the set of vectors S.

[18]:
```
import numpy as np
v1 = np.matrix([1, -2, 7])
v2 = np.matrix([6, -2, 3])
v3 = np.matrix([3, 1, -2])

print("\nMatrix of vectors V = {v1, v2, v3}:")
V = np.vstack([v1, v2, v3])
print(V)

print("\nDeterminant of V = 0?:", np.linalg.det(V) == 0)
```

```
Matrix of vectors V = {v1, v2, v3}:
[[ 1 -2  7]
 [ 6 -2  3]
 [ 3  1 -2]]

Determinant of V = 0?: False
```

Hence, we have that the set of vectors $\{(1, -2, 7), (6, -2, 3), (3, 1, -2)\}$ is linearly independent.

5. SETS OF VECTORS FROM VECTOR SPACE

c) Linear independence (method 2)

November 27, 2021

1 AIM

One result states that a set of vectors is linearly dependent if and only if a vector in the set can be expressed as a linear combination of the other vectors. Using this result, we can identify whether the set of vectors in linearly independent or not.

Support functions...

[1]:
```
from sympy import Matrix, zeros, pprint
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
```

Main…

[8]:
```
M = []
print("\nINPUT VECTORS (1 ROW, 3 COLUMNS)")
for i in range(0, 4):
    M.append(matrixInput(1, 3))
print("\nYOUR VECTORS")
for i in range(0, 4):
    print(M[i])
N = Matrix(M)
print("\nMATRIX OF YOUR VECTORS")
pprint(N)
r = N.rank()
print("Rank: ", r)
if(r < 4): print("At least one vector is a linear combination of the others.")
else: print("None of these vectors are linear combinations each other.")
```

```
INPUT VECTORS (1 ROW, 3 COLUMNS)

Enter row by row, each element in the row separated by comma…
R1: 1,2,3

Enter row by row, each element in the row separated by comma…
R1: 2,5,3

Enter row by row, each element in the row separated by comma…
R1: 2,7,4

Enter row by row, each element in the row separated by comma…
R1: 3,8,9

YOUR VECTORS
Matrix([[1.00000000000000, 2.00000000000000, 3.00000000000000]])
Matrix([[2.00000000000000, 5.00000000000000, 3.00000000000000]])
Matrix([[2.00000000000000, 7.00000000000000, 4.00000000000000]])
Matrix([[3.00000000000000, 8.00000000000000, 9.00000000000000]])

MATRIX OF YOUR VECTORS
 1.0  2.0  3.0
```

```
 2.0  5.0  3.0

 2.0  7.0  4.0

 3.0  8.0  9.0
Rank:  3
At least one vector is a linear combination of the others.
```

2 CONCLUSION

To get the rank of a matrix, you need to reduce to row echelon form. To reduce to row echelon form, you need to do row operations, and row operations involve linear combination of vectors. Now, if one of the rows becomes 0 in the process, that means the vector that was in that row was equal to some linear combination of the other vectors (which is why it was cancelled out). So, since there are 4 rows, if the rank is less than 4, we know that one or more rows have been cancelled out, hence at least one vector is a linear combination of the others. Since we have that if a vector in a set of vectors can be expressed as a linear combination of the other vectors, then the set of vectors is linearly dependent, we have that the above set of vectors is linearly dependent.

5. SETS OF VECTORS FROM VECTOR SPACE

d) Plotting linear transformations

November 27, 2021

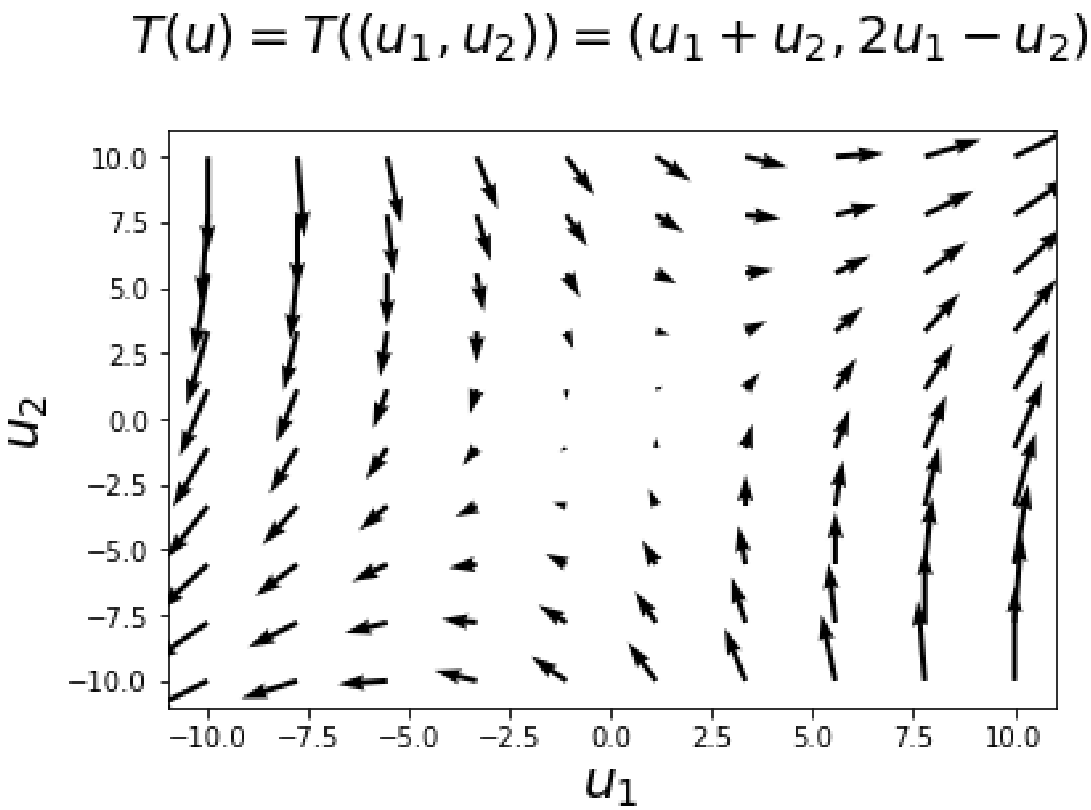
1 AIM

Linear transformation is a homomrphism between two vector spaces. It is a one-to-one function from a domain to a range, and hence, can be plotted. This is the focus of this record.

2 EXAMPLE

Let U and V be two vector spaces of two dimensional vectors defined over the field of real numbers. Consider the transformation $T : U - - > V$ such that $T(u) = T((u_1, u_2)) = (u_1 + u_2, 2u_1 - u_2)$

[47]:
```
import matplotlib.pyplot as plt
import numpy as np
u1, u2 = np.meshgrid(np.linspace(-10, 10, 10), np.linspace(-10, 10, 10))
v1, v2 = u1 + u2, 2*u1 - u2
plt.quiver(u1, u2, v1, v2, alpha = 1)
plt.title("\n$T(u)=T((u_1,u_2))=(u_1+u_2,2u_1-u_2)$\n", size = 20)
plt.xlabel('$u_1$', size = 20)
plt.ylabel('$u_2$', size = 20)
None
```



3 CONCLUSION

Here, each set of coordinates (u_1, u_2) is a vector from the vector space U, and the vector originating from this point on the graph is the transformed vector.